



HAL
open science

Validation, synthèse et paramétrage des cartes cognitives

Aymeric Le Dorze

► **To cite this version:**

Aymeric Le Dorze. Validation, synthèse et paramétrage des cartes cognitives. Modélisation et simulation. Université d'Angers, 2013. Français. NNT : . tel-00956983

HAL Id: tel-00956983

<https://theses.hal.science/tel-00956983v1>

Submitted on 7 Mar 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de Doctorat

Aymeric LE DORZE

*Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université d'Angers
sous le label de l'Université de Nantes Angers Le Mans*

École doctorale : Sciences et technologies de l'information, et mathématiques

Discipline : Informatique et applications, section CNU 27

Unité de recherche : Laboratoire d'Études et de Recherche de l'Université d'Angers (LERIA)

Soutenue le 04/11/2013

Thèse n° : ED 503-1369

Validation, synthèse et paramétrage des cartes cognitives

JURY

Président :	M. Christophe ROCHE , Professeur, Université de Savoie
Rapporteurs :	M. Patrice BUCHE , HDR, Montpellier SupAgro M. Juliette DIBIE-BARTHÉLEMY , Professeur, AgroParisTech
Examineurs :	M. David GENEST , Maître de conférences, Université d'Angers M. Laurent GARCIA , Maître de conférences, Université d'Angers
Invité :	M. Philippe LERAY , Professeur, École Polytechnique de l'Université de Nantes
Directeur de thèse :	M. Stéphane LOISEAU , Professeur, Université d'Angers

Remerciements

Je tiens tout d'abord à vivement remercier Patrice BUCHE, ingénieur à Montpellier SupAgro, et Juliette DIBIE-BARTHÉLEMY, professeur à AgroParisTech, pour avoir accepté d'être rapporteurs de cette thèse.

Je tiens également à remercier cordialement Christophe ROCHE, professeur à l'Université de Savoie, qui m'a fait l'honneur de présider le jury de thèse.

Je remercie par ailleurs Philippe LERAY, professeur de l'École Polytechnique de l'Université de Nantes, qui a aimablement partagé ses connaissances dans son domaine d'expertise.

J'adresse mes plus sincères remerciements à mon directeur de thèse Stéphane LOISEAU, professeur à l'Université d'Angers et à mes co-encadrants, David GENEST et Laurent GARCIA, maîtres de conférences à l'Université d'Angers, pour leur disponibilité et leurs conseils qui m'ont grandement aidé.

Je remercie également les membres du LERIA et du département informatique de l'Université d'Angers, pour les bons moments passés en leur compagnie et pour leur soutien moral, et notamment :

- Béatrice DUVAL qui a bien voulu accepter de relire certains de mes travaux ;
- Lionel CHAUVIN pour ses conseils et son soutien lors de ma première année de thèse ;
- Caner CANDAN pour son soutien lors des années suivantes.

Merci enfin à mes amis et à ma famille qui m'ont toujours soutenu et encouragé au cours de ces trois années.

Mes pensées s'adressent également à Pascal NICOLAS pour m'avoir donné le goût de la recherche.

Sommaire

Introduction	7
1 Cartes sémantiques	13
1.1 Carte mentale	14
1.2 Carte conceptuelle	20
1.3 Graphe conceptuel	25
1.4 Carte cognitive	31
1.5 Comparaison	35
2 Carte cognitive	43
2.1 Définition du calcul de l'influence propagée	44
2.2 Carte cognitive taxonomique	48
2.3 Vue d'une carte cognitive taxonomique	52
2.4 Paramétrage	57
3 Validation d'une carte cognitive	67
3.1 Vérification	68
3.2 Test	72
3.3 Paramétrage	77
3.4 Conflit	84
4 Synthèse de cartes cognitives	97
4.1 Cartes attribuées et préférences	99
4.2 Opérateurs de synthèse	103
4.3 Vue d'une carte synthétisée	111
4.4 Personnalisation	118
5 Discussion sur les valeurs	129
5.1 Interprétation des valeurs	130
5.2 Carte cognitive probabiliste	146
5.3 Sémantique des concepts et des influences	164
6 Système	169
6.1 Cartes cognitives	170
6.2 Calcul d'influences	175
6.3 Validation	181
6.4 Synthèse	184

6.5 Entrées/Sorties et interface graphique	190
6.6 Cas d'utilisation	196
Conclusion	207
Annexes	211
A Validation de cartes du monde de la pêche	211
B Preuve du théorème 1	216
Table des matières	227
Table des figures	231
Bibliographie	235
Publications personnelles	247

Introduction

Problématique

Notre travail s'inscrit dans la représentation des connaissances et la formalisation du raisonnement, liées aux problématiques de l'Intelligence Artificielle (IA). De nombreux modèles d'IA et de psychologie cognitive proposent une visualisation, souvent partielle, des connaissances fournies sous forme visuelle à base de graphe. On appelle *cartes sémantiques* des modèles développés pour représenter visuellement des informations, notamment les relations [Tricot, 2006]. Ces cartes sémantiques permettent de communiquer, de modéliser et d'interroger efficacement des informations au travers de représentations graphiques. Les modèles s'intéressent souvent à la communication entre plusieurs personnes au sein d'un groupe ; ils s'attachent ainsi plus particulièrement à la visualisation des connaissances représentées.

Ces modèles de cartes sont très vastes et regroupent des modèles développés dans de nombreux domaines allant des sciences humaines jusqu'à l'informatique et les mathématiques en passant par les sciences de la vie. En IA, on peut citer des modèles de conception tels ceux utilisés dans KADS¹ (UML, arbres de tâches...), des modèles de résolution tels ceux issus des *réseaux sémantiques* [Lehmann, 1992] (graphes conceptuels [Sowa, 1976; Sowa, 1984], notation graphique des logiques de description [Baader *et al.*, 2003]) ou des réseaux de causalité tels les *réseaux bayésiens* [Pearl, 1988; Pearl, 2009]. En psychologie cognitive, on peut citer les modèles conçus pour aider à la mémorisation et à l'apprentissage tels les *cartes mentales* [Buzan et Buzan, 2003] et les *cartes conceptuelles* [Novak et Cañas, 2006]. Les *cartes cognitives* [Axelrod, 1976] sont aussi souvent mises en avant en psychologie cognitive pour décrire des relations d'influence. Elles intègrent des capacités de raisonnement assez évoluées et des travaux portant sur celles-ci ont été réalisés dans la communauté de l'IA : les cartes cognitives peuvent être considérées comme un type de réseau sémantique issu de la psychologie cognitive. Nous nous intéressons à la représentation formelle des cartes cognitives et à leur application potentielle au raisonnement. Les modèles trouvant leur origine dans le domaine des Sciences Humaines, tels que les cartes mentales et les cartes conceptuelles, présentent souvent un aspect peu formel contrairement à ceux issus du monde de l'informatique, tels les graphes conceptuels. Les cartes cognitives, bien qu'elles aient été utilisées à l'origine dans le cadre des sciences politiques, ont l'avantage de représenter formellement des connaissances tout en étant facilement utilisables

¹<http://www.commonkads.uva.nl/>

par tout un chacun.

Une *carte cognitive* permet de représenter des influences entre différents concepts sous la forme d'un graphe. À chaque sommet est associé un *concept* qui représente une notion du monde réel. Chaque arc entre ces sommets représente une *influence* à laquelle est attribuée une *valeur* qui permet de la quantifier. Cette valeur appartient à un ensemble de valeurs numériques ou symboliques prédéfini. La structure graphique d'une carte cognitive et les valeurs associées aux arcs permettent le calcul de l'*influence propagée*. L'influence propagée d'un concept sur un autre représente l'influence globale de ce concept sur l'autre dans le cadre du système représenté par la carte cognitive. Cette opération s'effectue en trois temps. Tout d'abord, on liste les différents chemins qui permettent de lier entre eux les concepts dans le graphe. Ensuite, on agrège, pour chaque chemin, les valeurs des influences le composant pour calculer l'*influence propagée sur ce chemin*. Enfin, on agrège les influences propagées sur les différents chemins pour obtenir la valeur de l'influence propagée d'un concept sur un autre. Ce calcul n'est pas formellement défini pour l'ensemble des cartes cognitives : il est propre à chaque formalisme et dépend notamment du type des valeurs portées par les influence ainsi que de la sémantique associée à l'ensemble de valeurs. De par ce mécanisme, les cartes cognitives sont un excellent outil pour l'aide à la prise de décision. En effet, le calcul de l'influence propagée d'un concept sur un autre permet d'avoir un aperçu des conséquences de cette décision en analysant l'impact d'un concept sur un système. Elles ont été utilisées dans de nombreux domaines d'études tels que la biologie [Tolman, 1948], la sociologie [Poignonec, 2006], l'écologie [Celik *et al.*, 2005] ou encore la politique [Levi et Tetlock, 1980].

Si une carte cognitive est relativement facile à manipuler par un utilisateur ou un groupe d'utilisateurs qui chercheraient à prendre une décision ou simplement étudier un système, il est nettement plus difficile de la construire. En effet, une carte cognitive représente souvent les savoirs d'un ou plusieurs experts dans un domaine particulier. Nous appelons de tels experts constructeurs de cartes des *designers*. Or, lors de la construction d'une carte, des divergences entre ces designers peuvent apparaître qui entraînent des contradictions.

L'objectif de cette thèse est triple. Il s'agit tout d'abord de s'assurer de la validité de la carte cognitive lors de sa construction. Il s'agit ensuite de proposer des mécanismes permettant de synthétiser et d'utiliser une carte cognitive à plusieurs. Il s'agit enfin de discuter la notion de valeur portée par les influences ainsi que la sémantique qui lui est associée et de proposer des critères pour choisir quel type de valeur utiliser.

Critiques et propositions

La *validation* d'une carte cognitive permet de s'assurer que l'information déduite de la carte ne comporte pas d'ambiguïté. Lors du calcul de l'influence propagée d'un concept sur un autre, il arrive fréquemment que les influences propagées sur les différents chemins permettant de lier les deux concepts n'ont pas la même valeur. Le calcul de l'influence propagée nécessite d'agréger ces valeurs. Or, certaines valeurs peuvent représenter des notions opposées, ce qui aboutit à une agréga-

tion non déterminée qui produit une valeur ambiguë. Avec certains formalismes de calcul, il est impossible d'obtenir un consensus, représenté par une valeur intermédiaire. Certains formalismes y parviennent mais la valeur obtenue ne reflète pas nécessairement l'hétérogénéité des valeurs. Savoir qu'il y a une ambiguïté entre les différents chemins nécessaires au calcul de l'influence d'un concept sur un autre pourrait néanmoins être intéressant. La validation permet de repérer de telles ambiguïtés et de les mettre en avant dès la construction de la carte. Ainsi, un designer peut corriger sa carte de manière à ce qu'elle ne soit pas ambiguë et un utilisateur de cette même carte est assisté dans l'étude du système représenté par la mise en évidence de ces ambiguïtés. Le designer doit pouvoir définir différents critères de qualité que la carte doit valider, chacun d'entre eux décrivant une contrainte bien définie et ce afin de relever les différentes ambiguïtés.

Une *carte synthétisée* permet de regrouper les informations issues d'un ensemble de cartes, en privilégiant certaines informations par rapport à d'autres. En effet, lors d'une séance de brainstorming entre différents intervenants, il est très difficile de construire une unique carte cognitive. Les différentes informations que chaque intervenant cherche à introduire dans la carte peuvent s'avérer contradictoires les unes les autres. Il est bien plus facile pour chacun d'entre eux d'être le designer de sa propre carte cognitive, représentant ses connaissances et ses croyances. La difficulté consiste alors à réunir ces différentes cartes en une carte unique, facile à manipuler. Pour cela, on définit un mécanisme automatique pour construire une *carte synthétisée*. Ce mécanisme s'appuie sur une taxonomie de concepts permettant d'exprimer des liens de spécialisation entre les différents concepts utilisés par les designers. Cette taxonomie est fournie aux designers avant même la construction de leur carte afin qu'elle leur serve de vocabulaire. Il est cependant toujours possible pour eux de l'enrichir en ajoutant de nouveaux concepts et de nouveaux liens de spécialisation à condition que les différents designers s'accordent sur cette modification. La construction se fait étape par étape, chacune de ces étapes étant définie de telle manière qu'elle réponde aux besoins des designers. Ainsi, un ensemble de designers peut produire une carte synthétisant les différentes informations qu'ils ont apportées, tout en étant facilement utilisable.

L'opération de calcul de l'influence propagée d'un concept sur un autre est fortement dépendante de l'ensemble des valeurs que peuvent prendre les influences et de la manière de les interpréter. Nous proposons différentes manières d'adapter les outils introduits en fonction de cela. Nos outils sont définis dans un premier temps pour un ensemble de valeurs donné et pour un opérateur de calcul spécifique. Nous proposons ensuite une manière de les adapter pour d'autres ensembles de valeurs. Ainsi, nous parlons de *paramétrage* lors d'une discussion sur l'adaptation de nos outils pour d'autres ensembles de valeurs. Nous parlons de *personnalisation* lors d'une discussion sur l'adaptation de nos outils pour d'autres opérateurs de calcul de l'influence propagée. Par ailleurs, l'opération de calcul de l'influence propagée peut parfois être considérée comme peu formelle. Elle est en effet généralement basée sur l'intuition plutôt que sur un formalisme bien défini. Nous proposons par conséquent différentes manières de définir cet opérateur selon la sémantique associée aux valeurs. Nous fournissons ainsi un nouvel opérateur basé sur la notion de probabilité fondé sur un travail comparatif des cartes cognitives et des réseaux

bayésiens.

Mise en œuvre des contributions

Nous nous basons essentiellement sur le modèle des cartes cognitives défini par Lionel Chauvin dans sa thèse [Chauvin, 2009] car il généralise différents modèles de cartes cognitives issus d'autres travaux. Nous définissons une carte cognitive sous la forme d'un graphe dont les sommets sont des concepts et les arcs des influences. Chaque carte est définie sur un ensemble de valeurs représentant les différentes valeurs pouvant être associées à chaque influence. Une carte cognitive est également définie sur une opération de calcul de l'influence propagée de n'importe quel concept de la carte sur n'importe quel autre. L. Chauvin introduit aussi la notion de *carte cognitive ontologique* qui permet d'associer une carte cognitive à une *ontologie*. Une ontologie est définie comme étant une relation de spécialisation entre différents concepts. Ce modèle permet de calculer l'*influence ontologique* d'un concept de l'ontologie sur un autre. Celle-ci est calculée en agrégeant les influences propagées des différents concepts de la carte regroupés par ceux de l'ontologie. La notion d'ontologie est équivalente à la notion de taxonomie dont nous aurons besoin pour la synthèse de cartes cognitives². Nous rappellerons également les définitions des notions de *vue* et d'*échelle*, notamment les échelles basées sur des *profils partagés*, à laquelle nous avons participé. Les vues et les échelles permettent de construire des vues partielles ou généralisées d'une carte cognitive.

À notre connaissance, il n'y a pas eu de travaux dédiés spécifiquement à la validation des cartes cognitives. Pour valider une carte, nous avons besoin de nous assurer que celle-ci respecte certains critères qui assurent qu'on ne déduira pas de l'information ambiguë. Nous définissons pour cela différents *critères de qualité* pour valider une carte. Chaque critère de qualité définit formellement des contraintes que doit respecter une carte cognitive. Nous différencions deux types de critères de qualité. D'une part, les critères de *vérification* ne nécessitent pas d'information externe : ils valident une carte cognitive en considérant uniquement l'information qui y est présente. On étudie ici seulement la validation de la cohérence interne d'une carte. D'autre part, les critères de *test* nécessitent, eux, une information externe qui est comparée à celle représentée dans la carte cognitive : ils spécifient un jeu de tests que doit valider une carte cognitive. Un tel jeu de test est appelé une *spécification*. Celle-ci est composée d'un ensemble de *contraintes*. Chaque contrainte représente une influence attendue entre deux concepts. On utilise la taxonomie pour définir une contrainte valable pour un ensemble de concepts.

Pour aider à la *correction* d'une carte cognitive pour laquelle des critères de qualité ne sont pas respectés, nous définissons, pour chaque critère de qualité, l'ensemble des conflits associé. Un conflit est un ensemble de composants d'un système qui

²Nous estimons que le terme de « taxonomie » est plus adapté pour représenter cette notion que celui d'« ontologie ». En effet, les connaissances ontologiques ne se limitent généralement pas à des liens de spécialisation entre des objets. D'autres informations sont également regroupées sous ce terme telles que la notion de classe ou celle de relation entre les objets. C'est pourquoi, si nous reprenons pour l'essentiel les définitions de L. Chauvin, nous remplaçons le terme d'« ontologie » par celui de « taxonomie ».

suffisent à provoquer l'invalidation d'un certain critère de qualité [Reiter, 1987]. L'idée est de présenter à l'expert chargé de corriger le système seulement une partie des composants du système dont on sait qu'elle contient une erreur. Ainsi, corriger seulement un composant du conflit peut potentiellement corriger ce conflit et donc valider le critère. Dans notre cas, un *conflit* est une partie des informations d'une carte cognitive qui invalide le critère de qualité. Grâce à l'ensemble des conflits d'une carte selon un critère de qualité, on est en mesure de calculer un *diagnostic*. Ce diagnostic indique quels composants de la carte modifier pour que celle-ci soit conforme avec le critère de qualité considéré, c'est-à-dire qu'elle le valide. Un diagnostic corrige donc chacun des conflits. Cette double démarche (validation/correction) a été souvent utilisée pour valider des systèmes à base de connaissance [Beauvieux et Dague, 1990; Console *et al.*, 1993; Loiseau, 1997].

À notre connaissance, le seul travail portant sur la synthèse de cartes cognitives est celui de Denis Poignonec [Poignonec, 2006]. Ce travail n'est pas formalisé car il s'agit essentiellement d'un travail de sociologie appliquée. À noter également le travail de Gwen Christiansen qui effectue une comparaison de différentes cartes construites par des pêcheurs dans le but d'en réaliser une synthèse. Pour y parvenir, on représente les cartes sous forme de connaissances ontologiques dans le but d'étudier leurs similitudes et leurs divergences. Toutefois, avec ces deux approches, les solutions mises en œuvre sont assez peu paramétrables et considèrent que les informations apportées par différents designers ont la même importance. Or, la synthèse de cartes cognitives nécessite de regrouper les informations similaires et de gérer le cas des informations contradictoires d'un designer à l'autre. Pour ce dernier point, nous définissons des *préférences* sur les designers des différentes cartes. Ces préférences [Kaci, 2009] sont un ordre qui va nous permettre d'indiquer que les informations représentées par le designer le plus haut dans l'ordre sont préférées à celles représentées par les designers les plus bas. Ainsi, dès que deux informations apportées par deux designers différents sont contradictoires, on privilégie celle apportée par le designer le plus haut dans l'ordre. Pour exprimer les liens entre les concepts utilisés par les différents designers, nous utilisons une taxonomie. Nous pouvons ainsi exprimer le fait qu'un concept utilisé par un designer regroupe un ou plusieurs concepts utilisés par d'autres designers. La synthèse des cartes s'effectue en trois étapes successives. Tout d'abord, la *synthèse des concepts* détermine quels sont les concepts qui seront présents dans la carte synthétisée grâce à la taxonomie de concepts ainsi qu'aux préférences sur les designers. Ensuite, la *synthèse des influences* détermine quelles sont les influences qui seront présentes entre les concepts synthétisés dans la carte synthétisée grâce aux préférences sur les designers. Enfin, la *synthèse des valeurs* détermine quelle est la valeur de chaque influence synthétisée. Pour cela, on utilise également les préférences sur les designers ayant fourni les différentes valeurs d'une influence synthétisée. Ces notions de préférences nous permettent en outre d'améliorer la définition des profils partagés.

Le paramétrage et la personnalisation des différents mécanismes introduits dans ce mémoire de thèse nécessitent une analyse sémantique des valeurs. Les travaux portant ce sur point [Louçã, 2000] sont essentiellement des discussions sur les différents types de valeurs et d'agrégation de ces valeurs par les calculs. Nous proposons de comparer le modèle des cartes cognitives à d'autres modèles de représentation

graphique des connaissances figurant des liens valués entre des concepts. Nous introduisons ainsi des modèles dont les valeurs représentent des probabilités et d'autres associés à la logique floue [Zadeh, 1965]. Dans ce contexte, nous définissons un opérateur permettant le calcul de l'influence propagée d'un concept sur un autre dans une carte cognitive tel que les valeurs de cette carte sont apparentées à des probabilités. Pour prouver que cet opérateur est adapté à la sémantique associée aux valeurs, nous proposons une manière de construire un réseau bayésien représentant une carte cognitive, sous certaines contraintes que nous mettons en évidence. Nous démontrons ensuite que l'opérateur défini est équivalent au mécanisme d'inférence d'un tel réseau bayésien.

Annnonce du plan

Ce mémoire de thèse est composé de six chapitres. Au cours de ces chapitres, nous expliquons comment les différentes notions introduites s'adaptent à différents ensembles de valeurs et à différents opérateurs de calcul de l'influence propagée.

Les chapitres 1 et 2 constituent respectivement un état de l'art des cartes sémantiques et des cartes cognitives. Dans le chapitre 1, nous présentons les modèles des cartes mentales, des cartes conceptuelles, des graphes conceptuels et des cartes cognitives. Nous comparons ces différents modèles, en listons les avantages et les inconvénients. Le chapitre 2 présente plus en détail le modèle des cartes cognitives, notamment l'opération de calcul de l'influence propagée d'un concept sur un autre. Ce chapitre définit également les notions de taxonomie, d'influence taxonomique, de vue, d'échelle, de profil et de profils partagés.

Les chapitres 3 et 4 constituent le cœur de notre thèse et peuvent être lus indépendamment l'un de l'autre. Le chapitre 3 présente la validation des cartes cognitives. Nous introduisons différents critères de vérification et de test. Nous définissons, pour chaque critère de qualité, la notion de conflit qui est composée d'influences de la carte provoquant une invalidité. Le chapitre 4 présente la synthèse de cartes cognitives. Nous introduisons les notions de préférences sur les designers. Nous définissons également les trois mécanismes de synthèse qui composent les différentes étapes de la synthèse de cartes cognitives.

Le chapitre 5 discute de la sémantique associée aux différents ensembles de valeurs sur lesquels une carte cognitive peut être définie et de la manière d'adapter les opérateurs de calcul de l'influence propagée d'un concept sur un autre en conséquence. Nous présentons également un nouvel opérateur de calcul de l'influence propagée lorsque la sémantique des valeurs d'une carte s'apparente à des probabilités. Pour cela, nous faisons un rapprochement entre le modèle des cartes cognitives et celui des réseaux bayésiens en proposant une méthode de traduction d'une carte cognitive en un réseau bayésien, sous certaines conditions.

Enfin, le chapitre 6 présente l'outil logiciel développé au cours de ce travail de thèse. Il met en œuvre les différents mécanismes décrits dans ce manuscrit.



Cartes sémantiques

Introduction

Ce chapitre propose d'offrir une analyse comparative des principaux modèles visuels de représentation des connaissances, que nous appelons cartes sémantiques. Notre étude porte sur les modèles des cartes mentales [Buzan et Buzan, 2003], des cartes conceptuelles [Novak et Cañas, 2006], des cartes cognitives [Axelrod, 1976] et des graphes conceptuels [Sowa, 1976; Sowa, 1984]. Ainsi, nous allons nous intéresser à des modèles dont les objectifs sont différents, certains portant une attention particulière à l'aspect visualisation alors que d'autres sont plus focalisés sur le raisonnement.

Pour chacun des modèles que nous nous proposons d'étudier, nous suivrons la présentation suivante :

1. une introduction au modèle dans lequel nous le définissons de manière informelle ;
2. une définition plus formelle du modèle ;
3. un exemple d'utilisation commun à chacun des modèles ; une référence à des logiciels qui permettent de le manipuler ;
4. une discussion autour du modèle dans lequel nous évoquons les problèmes soulevés par son utilisation, en particulier en terme de raisonnement.

Nous terminons ce chapitre par une étude comparative des différents modèles qui reprend les éléments introduits dans les sections précédentes et qui propose une comparaison tant du point de vue de la représentation des connaissances et de la visualisation que du raisonnement.

La section 1.1 présente le modèle des cartes mentales. La section 1.2 présente le modèle des cartes conceptuelles. La section 1.3 présente le modèle des graphes

conceptuels. La section 1.4 présente le modèle des cartes cognitives. Enfin, la section 1.5 compare les différents modèles de cartes sémantiques présentés, sur différents critères.

1.1 Carte mentale

Les *cartes mentales* ont pour but de décrire une idée principale de manière précise et structurée au moyen de concepts liés à cette idée. Les concepts sont également des idées et sont généralement représentés par un texte bref ou une image. Pour présenter ce modèle, nous en faisons tout d'abord une présentation dans la section 1.1.1 avant de le définir formellement dans la section 1.1.2. Nous montrons ensuite l'application de ce modèle sur un exemple dans la section 1.1.3 et enfin nous discutons de ces points forts et de ces points faibles dans la section 1.1.4.

1.1.1 Présentation

Une carte mentale permet de décrire un concept principal en le liant à d'autres concepts pouvant à leur tour être décrits par de nouveaux concepts jusqu'à atteindre le niveau de détail souhaité. Les cartes mentales ont été utilisées depuis des siècles mais les cartes mentales modernes, également appelées *cartes heuristiques*, ont été formalisées par Tony Buzan [Buzan et Buzan, 2003]. Différents styles d'écriture peuvent être utilisés afin de structurer les concepts : la couleur, la casse et la taille du texte par exemple. On peut également ajouter d'autres éléments que du texte pour diversifier la carte, comme des images ou des liens. Les cartes mentales se construisent rapidement car elles utilisent essentiellement des mots correspondant aux concepts et des traits. Elles se remplissent aisément autour du concept principal au fur et à mesure que le cerveau intègre les idées. Des concepts proches dans leurs relations avec un concept apparaîtront également visuellement proches de ce concept dans la carte mentale. Une relation entre deux concepts est soit représentée directement par un lien entre ces deux concepts, soit par un nouveau concept décrivant précisément la nature de ce lien, lié lui aussi aux deux concepts. Elles sont faciles à mémoriser par leur présentation visuelle, l'usage de couleurs, de dessins, d'images ou de textes.

T. Buzan propose 10 règles pour construire une bonne carte mentale [Buzan, 2003] :

- commencer avec une image au milieu de la page en utilisant au minimum 3 couleurs ;
- utiliser des images ou des symboles partout dans la carte mentale ;
- choisir des mots-clés et les écrire en lettres capitales ou minuscules ;
- écrire chaque mot ou image seul sur sa propre ligne ;
- connecter les lignes, en partant de l'image centrale : les lignes proches du centre étant plus épaisses, celle en périphérie plus fines ;

- donner la même longueur aux lignes que le mot/l'image ;
- utiliser des couleurs pour l'aide à la mémoire d'évocation et stimuler la créativité ;
- développer son propre style de carte mentale ;
- surligner et montrer les associations dans la carte mentale ;
- garder la carte mentale claire en utilisant une hiérarchie radiale, un ordre numérique ou des cadres sur les branches.

Un premier usage de ces cartes est la prise de notes aisée : la rapidité de construction des cartes mentales est avérée. Un deuxième usage est l'aide au brainstorming en organisant les idées échangées par plusieurs individus lors d'une séance, les cartes mentales permettant de noter et de hiérarchiser les idées au fur et à mesure qu'elles sont énoncées par les intervenants. Un troisième usage est l'amélioration des moteurs de recherche en représentant les méta-informations de documents par une carte mentale. Un quatrième usage des cartes mentales est l'enseignement. Une étude a montré que l'utilisation des cartes mentales offrait une meilleure faculté de mémorisation pour des élèves, comparée à des méthodes d'apprentissage libre, mais diminuait en contrepartie leur motivation [Farrand *et al.*, 2002].

1.1.2 Représentation

Une carte mentale se présente sous la forme d'une arborescence dont la racine est située au centre. Les *concepts* sont les étiquettes des sommets de cette arborescence. Le sommet racine contient le concept principal de la carte. Ce sommet est connecté par des arêtes à d'autres sommets représentant des concepts plus spécifiques qui décrivent le concept principal.

La proposition d'éditeurs informatiques de cartes ou de partage de cartes entre plusieurs utilisateurs nécessite une véritable formalisation de celles-ci, que nous esquissons dans ce qui suit. La plupart des modèles de cartes mentales sont une arborescence dont les sommets et les arêtes sont étiquetés. La nature des étiquettes varie d'un modèle de carte mentale à l'autre. Ainsi, un sommet peut être étiqueté par un concept représenté par un simple mot ou une phrase entière, voire par une image. Une arête peut quant à elle être étiquetée par une couleur ou par une épaisseur de trait.

Nous proposons ici une formalisation des cartes mentales généralisant la plupart des modèles. Pour un modèle de carte mentale, un *concept* peut être un bref texte descriptif ou une image. Un *attribut* permet de décrire un aspect d'un sommet ou d'une arête. Ainsi, *Couleur* et *Épaisseur de trait* sont des attributs possibles pour les sommets et les arêtes ; *Police* est un attribut possible pour un sommet. Un attribut peut prendre différentes valeurs, selon la notion qu'il représente. *Rouge* ou *Noir* peuvent ainsi être des valeurs pour *Couleur*. On regroupe ces attributs en deux ensembles : un *ensemble des descriptions des sommets* et un *ensemble des descriptions des arêtes*. Certains attributs étant à la fois applicables aux sommets

et aux arêtes, ces deux ensembles ne sont pas nécessairement disjoints. On appelle *ensemble des descriptions* l'ensemble qui contient toute expression qui peut être utilisée pour étiqueter les sommets et les arêtes de l'arborescence. C'est donc l'union de l'ensemble des descriptions des sommets de l'ensemble des descriptions des arêtes.

Une carte mentale est une arborescence définie sur un ensemble des descriptions des sommets et sur un ensemble des descriptions des arêtes et pour laquelle à chaque sommet et à chaque arête est associée une étiquette sous la forme d'un sous-ensemble de l'ensemble des descriptions.

Définition 1.1 (Carte mentale)

Soit C un ensemble de concepts. Soient $X_V = X_{V_1}, \dots, X_{V_m}$ un ensemble des descriptions des sommets de m attributs et $X_A = X_{A_1}, \dots, X_{A_n}$ un ensemble des descriptions des arêtes de n attributs.

Une carte mentale définie sur C , X_V et X_A est une arborescence étiquetée $MM = (V, A, \text{label})$ où :

- V est l'ensemble des sommets de l'arborescence ;
- A est l'ensemble des arêtes de l'arborescence ;
- label est une application d'étiquetage qui associe à tout sommet et à toute arête une étiquette :
 - $\forall v \in V, \text{label}$ est définie sur $C \times X_{V_1} \times \dots \times X_{V_m}$;
 - $\forall a \in A, \text{label}$ est définie sur $X_{A_1} \times \dots \times X_{A_n}$.

On note $r \in S$ la racine de l'arborescence.

La plupart des modèles de cartes mentales sont des cas particuliers de cette définition. D'autres cartes offrent plus de libertés : par exemple, certaines cartes permettent de lier deux concepts qui ne sont pas sur la même branche en ajoutant une arête d'un sommet représentant un des concepts vers un sommet représentant l'autre. Pour caractériser de telles cartes, il faut étendre la définition précédente avec un ensemble d'arcs entre les sommets du graphe par exemple.

1.1.3 Édition et exemple illustratif

Il existe de nombreux logiciels permettant de créer et de manipuler des cartes mentales : des logiciels payants comme iMindMap¹, MindVisualizer² et MindManager³ et des gratuits comme FreeMind⁴, VYM⁵ ou VUE⁶. Ce dernier n'est pas dédié uniquement aux cartes mentales, mais à un plus grand nombre de représentations graphiques de connaissances. Il existe également des éditeurs de cartes mentales

¹<http://www.imindmap.com/>

²<http://www.innovationgear.com/index.php>

³<http://www.mindjet.com/>

⁴http://freemind.sourceforge.net/wiki/index.php/Main_Page

⁵<http://www.insilmaril.de/vym/>

⁶<http://vue.tccs.tufts.edu/>

en ligne, comme Mind42⁷ ou WiseMapping⁸. FreeMind, l'un des plus utilisés, gère, entre autres, l'épaisseur et la couleur des arêtes entre différents concepts, la manière de les tracer, ainsi que la taille ou la couleur du texte décrivant les concepts, la couleur de fond, l'ajout d'images, le type d'encadrement du concept, sa disposition. . . FreeMind permet également d'ajouter des liens aux concepts qui pointent vers des ressources externes ou internes à la carte et de lier graphiquement deux sommets quelconques de la carte par un arc.

Pour comparer les différents modèles de cartes sémantiques, nous figurons, tout au long de ce chapitre, le même exemple représenté par chacun des modèles. Cet exemple est une description d'Internet, son histoire et son utilisation. Les différentes cartes doivent faire figurer les informations suivantes :

- Internet a été conçu pour avoir un réseau standard, c'est-à-dire capable de faire communiquer entre elles un grand nombre de machines, et acentralisé ;
- Internet est basé sur ARPANET ;
- le terme "Internet" a été officialisé en 1982 ;
- Internet permet d'échanger du contenu ;
- ce contenu peut être soit une idée, soit une donnée ;
- ce contenu peut également être du texte, de l'audio ou de la vidéo ;
- Internet procure un anonymat relatif ;
- Internet permet de chiffrer des informations ;
- la transmission de cette information se fait par paquets qui sont routés au cœur du réseau ;
- cette transmission peut avoir lieu en mode connecté ou non connecté ;
- différents standards régissent Internet et ces standards sont normés par des organismes, comme l'IETF ou le W3C ;
- le W3C est chargé de définir les normes du Web ;
- les standards d'Internet sont organisés au sein d'une pile basée sur le modèle OSI.
- le protocole TCP transporte les données en mode connecté tandis le protocole UDP les transporte en mode non connecté.

La figure 1.1 est une carte mentale que nous avons générée avec le logiciel FreeMind décrivant différentes informations relatives à Internet. Cette carte est définie entre autres sur les concepts *Internet*, *Naissance*, *Transmission*, *W3C*, *Web*, qui font donc partie de l'ensemble des concepts. La présence du cadre est un attribut

⁷<http://mind42.com/>

⁸<http://www.wisemapping.com/c/home.htm>

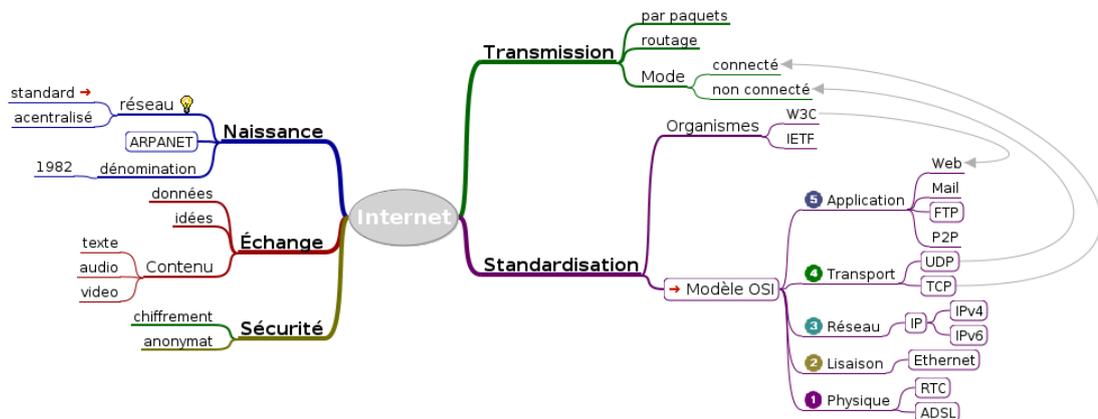


FIGURE 1.1 – Carte mentale décrivant Internet.

pour les sommets et peut prendre la valeur *Bulle* qui correspond à un sommet encadré : les concepts qui étiquettent les sommets apparaissent alors directement sur le sommet, comme *Ethernet*, *IP* ou *RTC*, plutôt que sur l'arête qui mène au sommet, comme *Naissance*, *réseau* ou *W3C*. L'épaisseur du trait est un attribut pour les arêtes et peut prendre les valeurs 1, 2, 4 ou 8 qui correspondent chacune au nombre de pixels utilisés pour représenter cette épaisseur.

Pour décrire Internet, on s'intéresse à 5 concepts composant les 5 branches principales de la carte : sa naissance, l'échange d'information qu'il permet, la sécurité liée au réseau, le mode de transmission de l'information et enfin la standardisation d'Internet qui s'est effectuée grâce à différentes normes régissant Internet. La carte est découpée en deux parties : la partie gauche concerne les informations qui pourraient intéresser une personne lambda, avec des connaissances en informatique ou non ; la partie droite concerne quant à elle les informations qui s'adressent uniquement à un informaticien. Sur le concept *Modèle OSI*, un lien hypertexte vers l'article de Wikipédia permet d'obtenir plus d'informations sur ce qu'est le modèle OSI⁹ tandis qu'un lien interne sur le concept *standard* renvoie vers le concept *Standardisation*. Par convention, les concepts encadrés sont des concepts représentant des normes. L'arc entre les sommets étiquetés par les concepts *W3C* et *Web* signifie que le W3C norme le Web et les arcs partant des sommets étiquetés par les concepts représentant les protocoles TCP et UDP les lient à leurs modes de transport respectifs.

1.1.4 Discussion

Les cartes mentales ont été imaginées pour être manipulables par des êtres humains. Par conséquent, il faut les adapter si on veut que les machines soient capables de raisonner sur les connaissances qu'elles représentent. Lorsqu'elles sont réalisées à la main sur une feuille de papier, les cartes mentales offrent énormément de liberté : l'utilisateur s'impose des conventions que lui seul utilisera. Par conséquent, il est difficile de réaliser un logiciel dans lequel on peut construire toutes

⁹http://fr.wikipedia.org/wiki/Mod%C3%A8le_OSI

les cartes mentales, car il est difficile de lister l'intégralité des représentations possibles, chaque utilisateur inventant ses propres conventions. Lorsqu'un utilisateur applique une convention à un sommet ou une arête, cette convention lui sert à donner une sémantique particulière à ce sommet ou à cette arête. Si l'utilisateur ne communique pas ses conventions sémantiques, certaines informations risquent d'échapper aux lecteurs potentiels de la carte. Si on veut que la carte soit modifiable par plusieurs concepteurs, il faut alors soit communiquer l'intégralité des conventions, soit convenir à l'avance de règles qui les définiront et qu'il faudra respecter.

Dans l'exemple de carte mentale de la figure 1.1, quelques informations sont peu claires ou mal définies. Il aurait été possible de les représenter, mais la carte aurait à nouveau perdu en concision. Il faut choisir entre richesse du contenu de l'information et concision de la représentation de cette information. Prenons par exemple la sous-branche issue du concept *Mode* de la branche *Transmission*. Il est difficile de savoir ce qu'on entend par « mode » avant d'avoir lu les sous-concepts qui décrivent cette notion. On comprend alors que la notion de « mode » fait référence au mode de transmission de l'information, représentée par les concepts *connecté* et *non connecté*. Autre exemple, la date 1982 se réfère à la date de la création officielle du terme Internet. Cette relation a été résumée par l'insertion d'un concept identifié par le simple mot *dénomination*. On aurait pu être plus précis, mais l'étiquette aurait alors été très longue, trop pour être facilement mémorisable. Comme cette information est finalement assez imprécise, seul le créateur de la carte peut savoir à quoi cette date fait exactement référence, rendant plus difficile la compréhension par autrui. Les sommets des cartes mentales peuvent donc représenter des concepts ou des relations.

Certaines relations ne sont pas identifiées dans la carte mentale et sont uniquement représentées par un lien entre deux concepts, comme par exemple celle entre les concepts *Internet* et *ARPANET*. On aurait pu ajouter cette relation dans la carte mentale sous forme d'un nouveau sommet, mais la carte aurait perdu en lisibilité. Le concept *Contenu* sert à énumérer les types de contenu que peuvent être les *données* et les *idées*. Ceux-ci peuvent tous deux être du *texte*, de l'*audio* ou de la *vidéo*. Or, une carte mentale étant généralement un arbre, il était impossible de tracer à la fois un lien de *données* à *texte* et de *idées* à *texte*.

Prenons maintenant la sous-branche *standard* située à gauche de la carte et la branche *Standardisation*. Ces deux notions sont fortement liées, il aurait été possible que la branche *Standardisation* ne soit que le prolongement de la sous-branche *standard*. Cependant, la sous-branche *standard* est située à l'intérieur de la branche *Naissance*. Si le prolongement avait été fait, les standards d'Internet auraient été classés dans la partie *Naissance* d'Internet, ce qui pourrait laisser penser que les standards n'ont existé qu'au cours des premières années de l'existence d'Internet, au moment de sa conception. Pour lever cette possible ambiguïté, il est donc préférable de séparer cette branche en deux, pour bien différencier d'une part le besoin de standardisation qui a amené à la création d'Internet et d'autre part les standards qui ont défini et définissent encore Internet.

Les concepts *réseau* et *Mode* sont chacun en relation avec deux autres sous-concepts. Les propositions représentées sont d'une part que « le réseau est à la

fois standard et acentralisé » et d'autre part que « la transmission se fait en mode connecté ou non connecté ». Le type des relations n'est donc pas le même : pour le concept *réseau*, on a affaire à une relation de type *et* et pour le concept *Mode* à une relation de type *ou exclusif*. Or, rien ne permet de déduire cette différence sémantique à partir de la carte mentale seule.

Les cartes mentales sont centrées sur la représentation des connaissances. Elles ne sont vouées qu'à présenter l'information. Par conséquent, elles ne permettent pas de raisonner sur ces connaissances : aucune information non présente sur la carte ne sera déduite. Les informations d'une carte peuvent toutefois être manipulées par d'autres systèmes. Il est par exemple possible de calculer le degré de relation entre deux documents en fonction de leur emplacement dans une carte mentale : plus les documents sont proches dans la carte mentale, plus les notions qu'ils évoquent sont approchées, et plus les documents sont liés [Beel et Gipp, 2010]. Avant la rédaction d'un article par exemple, on réalise une carte mentale organisant les idées qui sont développées dans cet article, en plaçant également les documents qui réfèrent certaines informations. Le calcul du degré de relation entre deux documents fonctionne alors sur une idée simple : plus les documents sont proches dans la carte mentale, plus les notions qu'ils évoquent sont proches, et plus les documents sont liés.

1.2 Carte conceptuelle

Les *cartes conceptuelles* ont pour but de répondre à une question centrale. Pour cela, des concepts, identifiés par de brefs textes représentant une notion particulière, sont liés entre eux par des relations, elles aussi décrites par de brefs textes les identifiant. Pour présenter ce modèle, nous en faisons tout d'abord une présentation dans la section 1.2.1 avant de le définir formellement dans la section 1.2.2. Nous montrons ensuite l'application de ce modèle sur un exemple dans la section 1.2.3 et enfin nous discutons de ces points forts et de ces points faibles dans la section 1.2.4.

1.2.1 Présentation

Les cartes conceptuelles ont été conçues pour la représentation et l'organisation de l'information dans le but d'en faciliter l'apprentissage. Elles ont été inventées par le biologiste Joseph D. Novak en 1972 [Novak et Cañas, 2006]. Elles sont particulièrement adaptées pour répondre à une question centrale amenant une réponse complexe en la décomposant en plusieurs sous-questions plus spécifiques, à leur tour expliquées. Ceci permet d'apprendre facilement des connaissances à une tierce personne via l'utilisation de cette carte, ce qui en fait un excellent outil d'aide à l'apprentissage. Les cartes conceptuelles ont aussi une utilisation similaire aux cartes mentales : elles peuvent être utilisées pour la prise de notes ou bien pour un travail de groupe. Elles diffèrent cependant par leurs modélisations et par leurs méthodes de construction. Les cartes conceptuelles sont beaucoup plus formelles que les cartes mentales : elles amènent à une réflexion stricte sur la manière d'organiser les connaissances.

Pour construire une carte conceptuelle, la première étape consiste à se poser une question centrale à laquelle la carte devra répondre. La seconde étape consiste à élaborer une liste des concepts qui seront utilisés dans la carte, sans s'occuper de leurs relations, ni de leur emplacement final dans la carte. La troisième étape consiste à réaliser une première ébauche de la carte en plaçant les concepts en fonction de leurs relations avec les autres concepts. Ainsi, dans un contexte d'apprentissage, pour que les étudiants puissent apprendre efficacement, ceux-ci doivent construire eux-mêmes la carte conceptuelle en partant d'une ébauche de carte réalisée par le professeur. La quatrième étape consiste à construire la carte en liant les concepts entre eux par des flèches via leurs relations et des mots de liaison, et en rajoutant éventuellement des concepts supplémentaires. Des ressources telles que du texte, des images, des pages Web ou bien d'autres cartes conceptuelles peuvent être liées aux concepts de la carte. Lier des concepts à d'autres cartes conceptuelles permet de découper une grande carte conceptuelle en plusieurs cartes conceptuelles plus petites et plus faciles à comprendre. On construit alors une macro-carte conceptuelle qui représentera le sommaire des connaissances et dont les concepts seront liés à des micro-cartes qui décriront chacune une partie des connaissances.

1.2.2 Représentation

Les cartes conceptuelles permettent de représenter des concepts et leurs relations. Un *concept* est représenté par un texte, généralement encadré ou entouré, et correspond à un individu, comme *Tour Eiffel*, à un ensemble d'individus, comme *Voiture*, ou à un événement, comme *Augmentation des impôts*. Un mot ou un groupe de mots de liaison lie deux concepts entre eux en exprimant la *relation* qu'ils partagent. Cette relation a un sens représenté par une flèche partant du concept « sujet » vers le concept « objet ». Un même concept ou une même relation peuvent apparaître plusieurs fois dans une même carte. Une *proposition* est une information à propos d'un concept. Une proposition contient deux concepts ou plus liés par une ou des relations pour former une assertion ayant un sens. La réponse à la question centrale amène généralement à la création d'un concept plus général que les autres, représentant la réponse à cette question. En effet, bien que ces questions puissent être très diverses (« Qu'est-ce que... ? », « Comment... ? », « Pourquoi... ? »), elles concernent bien souvent une notion principale, qui fera office de concept général. La carte est le plus souvent organisée de manière hiérarchique avec les concepts les plus généraux en haut et les concepts les plus spécifiques en bas.

Une carte conceptuelle est un graphe dans lequel des arcs lient des sommets étiquetés par des concepts à des sommets étiquetés par des relations.

Définition 1.2 (Carte conceptuelle)

Soient C un ensemble de concepts et R un ensemble de relations.

Une carte conceptuelle définie sur C et R est un graphe biparti orienté étiqueté $CM = (V_C, V_R, A, \text{label})$ où :

- V_C est l'ensemble des sommets concept ;

- V_R est l'ensemble des sommets relation ;
- $A \subseteq (V_C \times V_R) \cup (V_R \times V_C)$ est l'ensemble des arcs ;
- label est une application d'étiquetage qui associe à tout sommet une étiquette :
 - $\forall v \in V_C, \text{label}(v) \in C$;
 - $\forall v \in V_R, \text{label}(v) \in R$.

1.2.3 Édition et exemple illustratif

Le principal logiciel utilisé pour manipuler les cartes conceptuelles est CmapTools¹⁰, développé par l'IHMC¹¹. Notons qu'avec CmapTools, on peut modifier les cartes conceptuelles en spécifiant l'épaisseur et la couleur des arcs, la manière de les tracer, ainsi que la taille ou la couleur des étiquettes décrivant les sommets ainsi que la couleur de fond, l'ajout d'images, le type d'encadrement du sommet, sa disposition, avec une gestion d'attributs du même type que celle des cartes mentales.

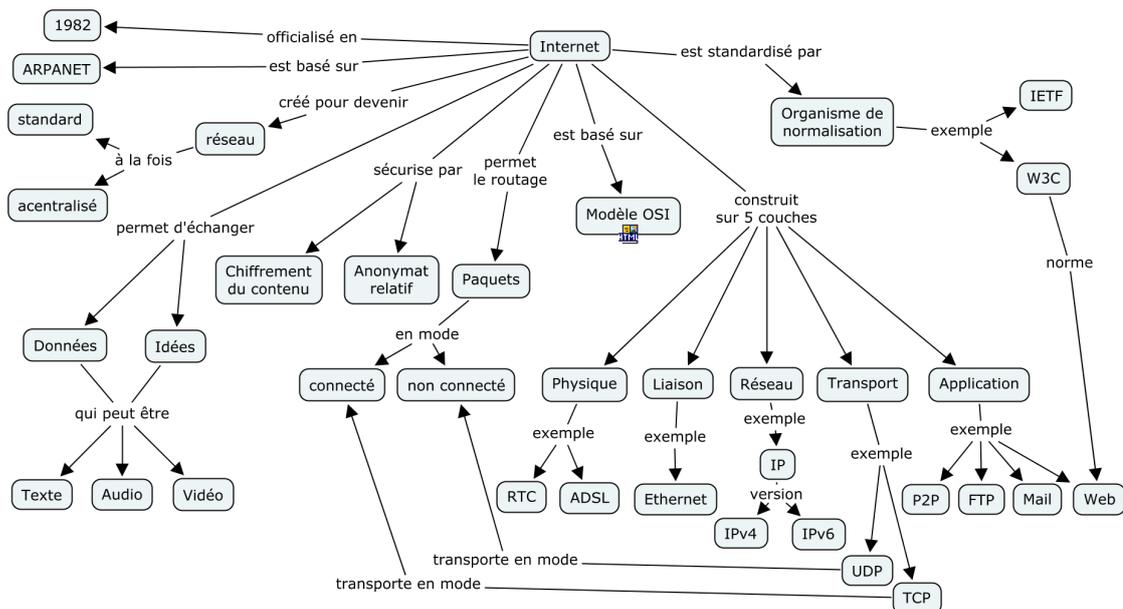


FIGURE 1.2 – Carte conceptuelle décrivant Internet.

La carte conceptuelle de la figure 1.2 a été générée avec le logiciel CmapTools et elle décrit elle aussi différentes informations relatives à Internet. Elle répond à la question « Qu'est-ce qu'Internet ? ». On peut y lire la proposition : « Internet permet le routage de paquets en mode non connecté ». Sur le concept *Modèle OSI*, un lien hypertexte pointe vers l'article de Wikipédia.

¹⁰<http://cmap.ihmc.us/>

¹¹Florida Institute for Human & Machine Cognition

1.2.4 Discussion

L'ordonnancement des couches sur lequel est basée l'architecture d'Internet n'apparaît pas dans la carte conceptuelle de la figure 1.2. Dans la carte conceptuelle, le lien entre le concept *Internet* et chacun des concepts représentant les couches est déjà étiqueté par une relation, il est donc impossible de l'étiqueter encore avec des numéros. Seul l'ordre de lecture (de gauche à droite) permet d'établir un ordre. Il aurait toutefois été possible d'indiquer explicitement l'ordre des concepts en utilisant une relation de type *précède* entre les concepts, mais la carte aurait perdu en concision. On aurait pu également réifier la relation, comme présenté dans la carte conceptuelle de la figure 1.3.

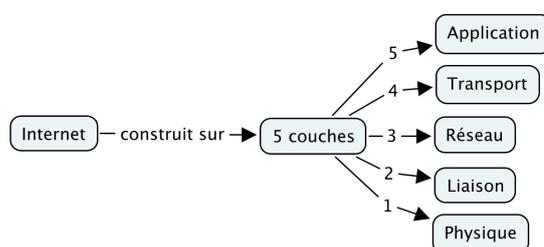


FIGURE 1.3 – Ordonnancement pour une carte conceptuelle.

Malgré leur aspect relativement formel, les cartes conceptuelles laissent une grande liberté à l'utilisateur, ce qui peut entraîner confusions et incohérences dans la carte. Ainsi, les propositions étant découpées en concepts reliés par des relations, il est parfois difficile de savoir où commence et où s'arrête une proposition. Par exemple, on modélise dans la carte conceptuelle de la figure 1.4 les deux propositions suivantes :

- les États-Unis ont acheté la Louisiane à la France ;
- la Nouvelle-Orléans est la capitale de la Louisiane.

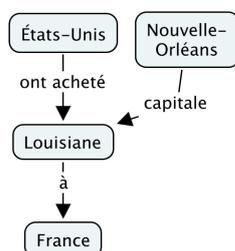


FIGURE 1.4 – Confusion sur l'arité de la relation.

Cette carte peut entraîner une confusion dans les propositions : en lisant naïvement cette carte, on peut lire la proposition « la Nouvelle-Orléans est la capitale de la Louisiane à la France ». Seul le non-sens de cette proposition permet d'éviter cette erreur.

Un autre problème se pose dû au regroupement habituel des relations de même étiquette : par souci de clarté, les relations de même étiquette ayant le même

concept sujet ou le même concept objet sont fusionnées. Ce procédé peut apporter des confusions sur l'association entre les concepts sujets et objets. Ainsi, dans la carte conceptuelle de la figure 1.2, les relations *exemple* et *transporte en mode* apparaissent plusieurs fois avec différents sujets et objets et les fusionner aurait apporté une confusion sur les couples de concepts liés par ces relations. Dans la

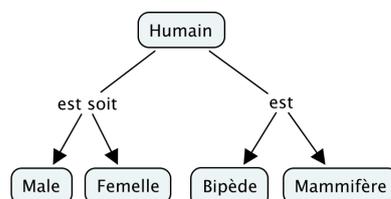


FIGURE 1.5 – Confusion entre relation binaire et n-aire.

carte conceptuelle de la figure 1.5, la relation *est soit* est une relation n-aire entre un concept sujet et $n - 1$ concepts objet. La relation *est* est une relation binaire entre un concept sujet et un concept objet. Cependant, il est impossible de déduire l'arité de la relation *est* uniquement à partir de la carte, puisqu'elle apparaît identique à la relation *est soit* alors qu'elles sont d'arités différentes. On pourrait lever la confusion en représentant explicitement les deux relations *est*, mais dans un souci de cohérence, il faudrait en faire de même pour toutes les relations binaires de la carte.

Certains sommets concept peuvent être étiquetés par le même concept s'ils représentent deux idées très proches. Cependant, un même concept peut étiqueter deux sommets différents, tout en représentant la même idée. La carte conceptuelle

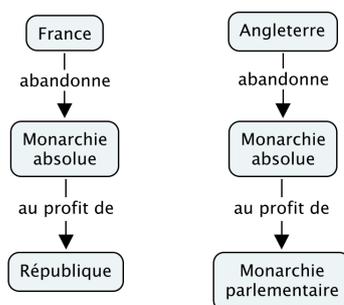


FIGURE 1.6 – Un même concept apparaît deux fois.

de la figure 1.6 contient 2 propositions :

- la France abandonne la monarchie absolue au profit de la république ;
- l'Angleterre abandonne la monarchie absolue au profit de la monarchie parlementaire.

Dans ces deux propositions, la notion de *Monarchie absolue* est la même. Cependant, cette notion apparaît deux fois sous la forme de concepts dans la carte conceptuelle afin d'éviter une confusion. En effet, si on fusionnait les deux concepts *Monarchie absolue*, nous obtiendrions alors la carte conceptuelle de la figure 1.7.

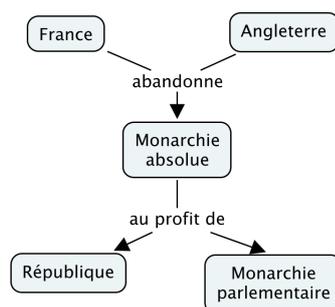


FIGURE 1.7 – Confusion entre les relations.

Il serait impossible de savoir vers quel régime politique a tendu chacun des deux pays. Il est cependant important d'insister sur le fait que ces deux concepts représentent effectivement la même notion. Une solution possible est de réifier ces relations. Les étiquettes de ces relations réifiées peuvent être par exemple les événements politiques qui ont causé l'abandon des monarchies, comme présenté sur la figure 1.8.

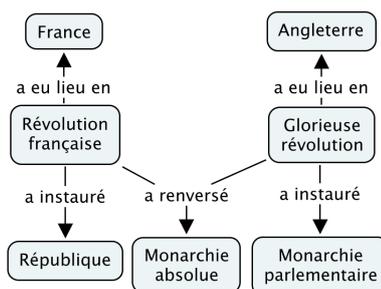


FIGURE 1.8 – Réification des relations.

Enfin, l'une des principales caractéristiques des cartes conceptuelles est qu'elles ne sont pas conçues pour être interprétées par des machines. En effet, il n'y a pas de vocabulaire prédéfini pour les concepts ou les relations et par conséquent, les informations de la carte ne sont pas assez précises et formelles pour être interprétées afin de déduire des connaissances. Une carte conceptuelle dont les propositions sont obligatoirement formelles et qui peut donc être interprétée par un ordinateur devient un réseau sémantique [Cañas et Novak, 2009; Lehmann, 1992].

1.3 Graphe conceptuel

Les *graphes conceptuels* servent à représenter graphiquement et à raisonner sur des connaissances portant sur des concepts et des relations. Les concepts et relations utilisés dans les graphes sont issus d'une ontologie construite préalablement. Pour présenter ce modèle, nous en faisons tout d'abord une présentation dans la section 1.3.1 avant de le définir formellement dans la section 1.3.2. Nous montrons ensuite l'application de ce modèle sur un exemple dans la section 1.3.3 et enfin nous discutons de ces points forts et de ces points faibles dans la section 1.3.4.

1.3.1 Présentation

Les graphes conceptuels servent à représenter graphiquement des connaissances de manière à ce qu'on puisse raisonner sur ces connaissances, c'est-à-dire être capable de les interroger et en déduire de nouvelles. Les graphes conceptuels font partie de la famille des réseaux sémantiques [Lehmann, 1992] et ont été inventés par John F. Sowa [Sowa, 1976; Sowa, 1984]. Le modèle des graphes conceptuels distingue deux types de connaissances : des connaissances ontologiques et des connaissances factuelles. Les connaissances ontologiques sont représentées au moyen d'un support qui définit un ensemble de types de concepts, un ensemble de types de relations et un ensemble d'individus. Les graphes conceptuels représentent les connaissances factuelles, sous la forme de relations entre des concepts. Ces concepts peuvent être identifiés à l'aide des individus du support, et sont étiquetés par un type de concept. De la même manière, les relations entre concepts sont étiquetées par un type de relation.

Les graphes conceptuels ne permettent pas simplement de visualiser des connaissances, mais aussi de raisonner grâce à des opérations, dont la projection qui permet d'interroger un graphe-fait au moyen d'un graphe-requête. La projection permet de savoir si l'information représentée par un graphe-requête existe dans le graphe-fait, ou bien d'obtenir les concepts qui vérifient la requête. Pour cela, on cherche à savoir si le graphe conceptuel-requête se projette dans le graphe conceptuel-fait. Les graphes conceptuels ont une sémantique logique, c'est-à-dire qu'ils sont exprimables en formules de la logique du premier ordre [Chein et Mugnier, 1992]. Ils sont utilisés en informatique, en intelligence artificielle et dans les sciences cognitives.

1.3.2 Représentation

Les graphes conceptuels sont définis sur un *support*, chargé de représenter des connaissances ontologiques, telles que les liens de spécialisation entre les types de concepts et ceux entre les types de relations.

Définition 1.3 (Support)

Un support est un quintuplet $S = (C, R, \sigma, I, \tau)$ [Mugnier et Chein, 1996] où :

- T_C est l'ensemble partiellement ordonné des types de concepts où \top est le plus grand élément ;
- $T_R = T_{R_{i_1}} \cup \dots \cup T_{R_{i_p}}$ est l'ensemble partitionné des types de relations où chaque $T_{R_{i_j}}$ est l'ensemble partiellement ordonné des types de relations d'arité $i_j > 0$ où \top_{i_j} est le plus grand élément ;
- $\sigma : T_{R_{i_j}} \rightarrow (T_C)^{i_j}$ est une application appelée signature qui associe à tout type de relation le type maximal de chacun de ses arguments : pour tout $t_r \in T_{R_{i_j}}$, elle associe un i_j -uplet $\sigma(t_r) \in (T_C)^{i_j}$ qui vérifie $\forall t_{r_1}, t_{r_2} \in T_{R_{i_j}}, t_{r_1} < t_{r_2} \Rightarrow \sigma(t_{r_1}) \leq \sigma(t_{r_2})$, où l'ordre considéré des signatures est l'ordre produit sur

$(T_C)^{i_j^{12}}$; on note $\sigma_i(t_r)$ le i -ième argument de $\sigma(t_r)$;

- I est l'ensemble dénombrable non nécessairement fini des marqueurs individuels ; il existe un marqueur générique, noté $*$; l'ensemble $I \cup \{*\}$ est muni de l'ordre suivant : $*$ est plus grand que tous les marqueurs individuels de I , qui sont deux à deux incomparables ;
- $\tau : I \rightarrow T_C$ est une application qui associe à tout marqueur m un type de concept t .

Les graphes conceptuels représentent les relations qu'entretiennent des *individus* ; chaque individu étant typé par un type de concept et représenté par un marqueur individuel ou par un marqueur générique. Un concept est donc l'association d'un individu et d'un type de concept.

Définition 1.4 (Graphe conceptuel)

Un graphe conceptuel est un multigraphe $G = (C, R, A, \text{label})$ non orienté, biparti, non nécessairement connexe défini sur un support $S = (T_C, T_R, \sigma, I, \tau)$ [Mugnier et Chein, 1996] où :

- C est l'ensemble des sommets concept ;
- R est l'ensemble des sommets relation ;
- $A \subseteq (C \times R) \cup (R \times C)$ est l'ensemble des arêtes ;
- label est une application d'étiquetage qui associe à tout sommet et à toute arête une étiquette :
 - $\forall c \in C, \text{label}(c) \in T_C \times (I \cup \{*\})$ est un couple $(\text{type}(c), \text{marqueur}(c))$ avec $\text{type}(c) : C \rightarrow T_C$ et $\text{marqueur}(c) : C \rightarrow I \cup \{*\}$; si $\text{marqueur}(c) = *$, alors le concept c est appelé concept générique et si $\text{marqueur}(c) \in I$, le concept est appelé concept individuel,
 - $\forall r \in R, \text{label}(r) \in T_R$ est le type du sommet relation,
 - $\forall a \in A, \text{label}(a) \in \mathbb{N}$;
- label obéit aux contraintes fixées par σ et τ :
 - l'ensemble des arêtes adjacentes à tout sommet relation r est totalement ordonné, ce qu'on représente en étiquetant les arêtes de 1 à l'arité de r ; on note $G_i(r)$ le i -ième voisin de r dans G ,
 - $\forall c \in C, \text{marqueur}(c) \in I \Rightarrow \tau(\text{marqueur}(c)) \leq \text{type}(c)$,
 - $\forall r \in R, \text{type}(G_i(r)) \leq \sigma_i(\text{type}(r))$.

¹²Le type associé au k -ième argument de t_{r_1} est inférieur ou égal au type associé au k -ième argument de t_{r_2} .

Le marqueur générique $*$ est donc utilisé dans un graphe quand le créateur du graphe veut représenter des connaissances sur un individu mais qu'il n'est pas capable d'exprimer de quel individu particulier du support il s'agit.

Une fois que des connaissances factuelles ont été exprimées sous la forme d'un graphe, ce graphe peut être interrogé à l'aide d'une requête exprimée elle aussi sous la forme d'un graphe conceptuel. Cette interrogation se fait à l'aide de l'opération de *projection* [Chein et Mugnier, 1992]. La projection d'un graphe-requête H dans un graphe-fait G permet de demander si l'information représentée par H est incluse dans l'information représentée par G . Plus précisément, la projection est un homomorphisme de graphe qui associe à tout sommet de H un sommet de G dont le type est égal ou plus spécialisé, et qui préserve les arêtes et leurs étiquettes. Dans le cas des sommets concept, un sommet étiqueté par un marqueur de I ne peut avoir pour image qu'un sommet portant le même marqueur, alors qu'un sommet étiqueté par $*$ peut avoir comme image un sommet quelconque. Ceci permet d'exprimer, dans une requête, une recherche sur des individus précis ou des individus quelconques.

Définition 1.5 (Projection)

Une projection d'un graphe conceptuel $H = (C_H, R_H, A_H, \text{label}_H)$ dans un graphe conceptuel $G = (C_G, R_G, A_G, \text{label}_G)$, tous deux définis sur un même support, est un couple d'applications $\Pi = (f, g)$, avec $f : C_H \Rightarrow C_G$ et $g : R_H \rightarrow R_G$, tel que :

- Π peut spécialiser les étiquettes des sommets :
 - $\forall c \in C_H, \text{label}_G(f(c)) \leq \text{label}_H(c)$,
 - $\forall r \in R_H, \text{label}_G(g(r)) \leq \text{label}_H(c)$;
- Π préserve les arêtes et les étiquettes : pour tout arête $rc \in A_H$, $g(r)f(c)$ est une arête de A_G ; de plus, si $c = H_i(r)$, alors $f(c) = G_i(h(r))$.

1.3.3 Édition et exemple illustratif

Il existe quelques logiciels permettant de créer et d'interroger des graphes conceptuels, comme CharGer¹³, développé par l'Université d'Alabama. Cogitant¹⁴ est une bibliothèque permettant modéliser et raisonner sur des graphes conceptuels. Il existe une interface graphique permettant de créer des graphes conceptuels et qui les interroge grâce à Cogitant, CoGui¹⁵.

La figure 1.9 est un graphe conceptuel représentant des informations sur Internet. Pour simplifier la visualisation, les arêtes étiquetées par 1 sont représentées par une flèche partant du sommet concept vers le sommet relation et les autres arêtes par une flèche partant du sommet relation vers le sommet concept. On ne précise pas la numérotation pour les arêtes étiquetées par 1 et, dans le cas des relations binaires, pour les arêtes étiquetées par 2.

Ce graphe conceptuel est construit sur le support dont les types de concepts et les types de relations sont représentés dans la figure 1.10. Le type de concept

¹³<http://sourceforge.net/projects/charger/>

¹⁴<http://cogitant.sourceforge.net/>

¹⁵<http://www.lirmm.fr/cogui/>

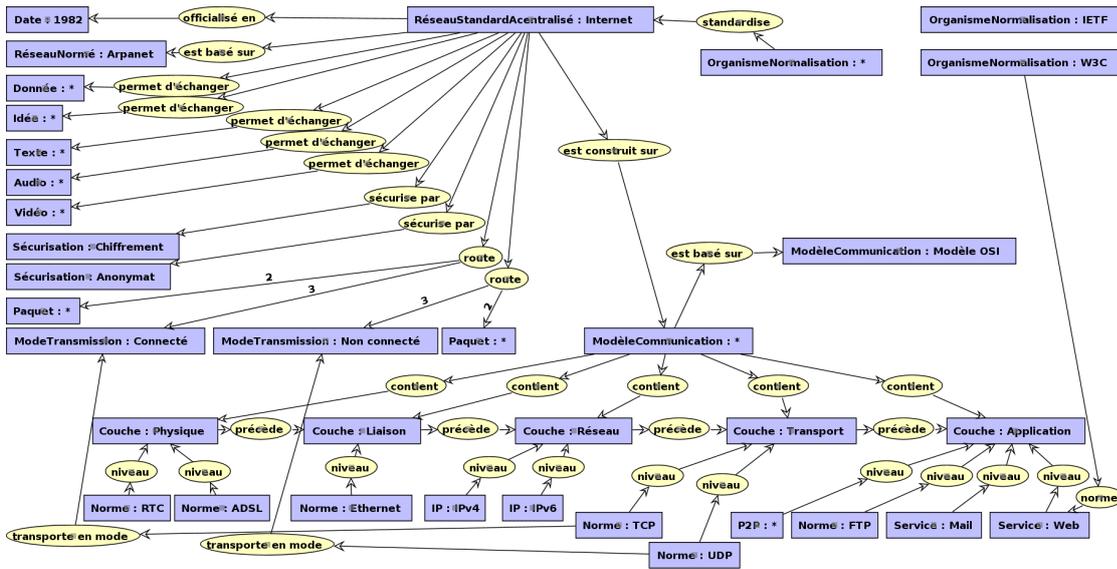


FIGURE 1.9 – Graphe conceptuel représentant diverses informations sur Internet.

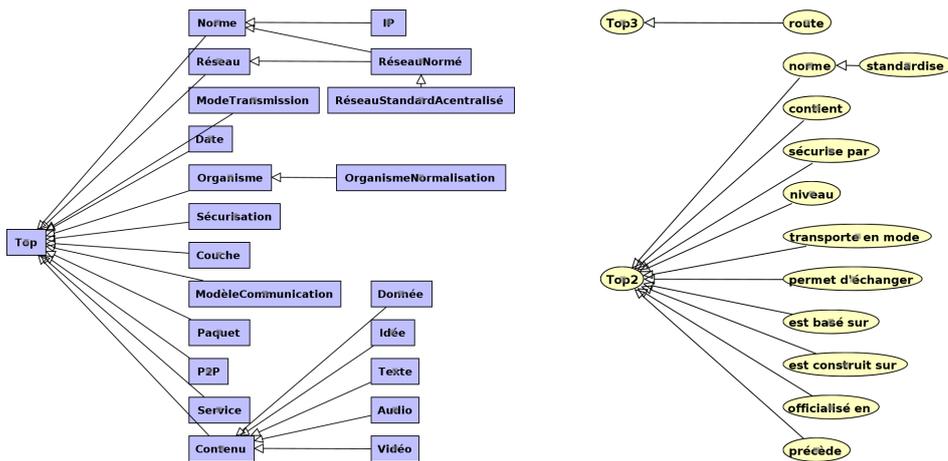


FIGURE 1.10 – Ensemble des types de concepts et de types de relations du graphe conceptuel de la figure 1.9.

Top sur cette figure est complètement équivalent au type de concept T de la définition 1.3. Les types de relations *Top2* et *Top3* correspondent respectivement aux types de relations T_2 et T_3 . Dans la hiérarchie de concepts du support, on peut lire que $IP \leq Norme$ et donc que les protocoles IP sont des normes. Nous avons choisi de créer un type IP pour regrouper les deux versions principales d'IP, IPv4 et IPv6, qui sont, elles, représentées par des individus.

La figure 1.11 représente les signatures des relations. Par souci de concision, celles ayant pour une signature du type $\sigma(r) = (T, T)$ ou $\sigma(r) = (T, T, T)$ ne sont pas représentées. L'ensemble des marqueurs individuels est représenté dans le tableau de la figure 1.12.

Le graphe de la figure 1.13 est un graphe-requête qui peut être interprété de la façon suivante : « Au niveau de quelle couche est la norme IPv4 et quelle est la couche précédente et quelle est la couche suivante ? ». L'opération de projec-

Type de relation (\mathbf{r})	$\sigma_1(\mathbf{r})$		$\sigma_2(\mathbf{r})$
officialisé en	T		Date
sécurise par	T		Sécurisation
niveau	T		Couche
transporte en mode	Norme		ModeTransmission
norme	Organisme	Normalisation	T
standardise	Organisme	Normalisation	T

Type de relation (\mathbf{r})	$\sigma_1(\mathbf{r})$	$\sigma_2(\mathbf{r})$	$\sigma_3(\mathbf{r})$
route	Réseau	T	ModeTransmission

FIGURE 1.11 – Signatures des types de relations.

Marqueur	Type	Marqueur	Type
Internet	RéseauStandardAcentralisé	Physique	Couche
1982	Date	Liaison	Couche
Chiffrement	Sécurisation	Réseau	Couche
Anonymat	Sécurisation	Transport	Couche
Connecté	ModeTransmission	Application	Couche
Non connecté	ModeTransmission	IPv4	IP
Modèle OSI	ModèleCommunication	IPv6	IP
RTC	Norme	UDP	Norme
ADSL	Norme	TCP	Norme
Ethernet	Norme	FTP	Norme
IETF	OrganismeNormalisation	Mail	Service
W3C	OrganismeNormalisation	Web	Service

FIGURE 1.12 – Marqueurs individuels.

tion retourne un résultat positif pour le graphe de la figure 1.9, présenté dans la figure 1.14. Le sommet *Norme:IPv4* a pour image *IP:IPv4*, puisque $IP \leq Norme$. Le sommet *Couche:** qui indique la couche au niveau de laquelle se trouve *IPv4* a pour image *Couche:Réseau*. Le sommet *Couche:** qui indique la couche précédente a pour image *Couche:Liaison*. Le sommet *Couche:** qui indique la couche suivante a pour image *Couche:Transport*. Ainsi, on en conclut que la norme IPv4 est au niveau de la couche Réseau et que la couche précédente est la couche Liaison et la suivante la couche Transport.

1.3.4 Discussion

L'aspect visuel des graphes conceptuels n'inclut pas des attributs du modèle des cartes mentales, comme la couleur ou l'épaisseur des traits qui aident à l'apprentissage et la mémorisation. En effet, la priorité est donnée à l'information représentée par les graphes conceptuels, plutôt qu'à l'aspect graphique de ces informations. Il n'est donc pas non plus possible d'ajouter des liens internes ou externes pour mieux décrire un concept.

Un graphe conceptuel nécessite de définir beaucoup d'informations : là où les autres modèles se limitent à un graphe de connaissances factuelles, les graphes

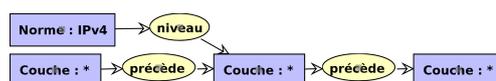


FIGURE 1.13 – Graphe-requête pour le graphe conceptuel de la figure 1.9.



FIGURE 1.14 – Graphe-image issu du graphe-requête de la figure 1.13 pour le graphe conceptuel de la figure 1.9.

conceptuels nécessitent de définir en plus un support. Cela rend les graphes conceptuels longs et parfois difficiles à construire, l'accent étant donné sur la précision de l'information.

Il n'existe pas de moyen clair et concis de représenter un ordre dans un graphe, si ce n'est en explicitant clairement la relation d'ordre entre différents individus. Ainsi, le graphe conceptuel de la figure 1.9 utilise une relation *précède* pour ordonner les couches du modèle de communication, ce qui charge l'aspect visuel du graphe conceptuel. De façon plus générale, les graphes conceptuels ne cherchent pas à avoir un aspect visuel qui permette une mémorisation facile.

Il existe différentes extensions du modèle des graphes conceptuels. Ainsi, la sémantique claire du modèle a permis d'exprimer des règles Datalog [Baget *et al.*, 2009]. Cependant, dans le modèle original, il n'est pas possible de manipuler des valeurs littérales, d'où la création d'un concept *Date* et d'un individu *1982* dans la figure 1.9. Par ailleurs, la numérotation des arêtes permet de préciser le rôle de chacun des individus qui interviennent dans une relation. Un remplacement de ces nombres par un texte descriptif du rôle de l'individu concerné dans la relation offrirait plus de clarté.

Il n'est pas non plus possible de représenter un type abstrait : dans notre exemple, un contenu est soit une donnée, soit une idée. Or, avec le modèle et le support définis ici, un individu peut être instance de *Contenu* sans être instance ni de *Donnée*, ni de *Idée*.

Il faut noter que, malgré ces problèmes, l'un des intérêts majeurs des graphes conceptuels est qu'ils permettent le raisonnement sur des connaissances non ambiguës.

1.4 Carte cognitive

Les *cartes cognitives* ont pour but de représenter des influences entre des concepts. Les concepts sont généralement des événements et sont décrits par des textes brefs et liés entre eux par des influences d'un concept-cause vers un concept-effet auxquelles une valeur est associée. Pour présenter ce modèle, nous en faisons tout d'abord une présentation dans la section 1.4.1 avant de le définir formellement dans la section 1.4.2. Nous montrons ensuite l'application de ce modèle sur un exemple dans la section 1.4.3 et enfin nous discutons de ces points forts et de ces points faibles dans la section 1.4.4.

1.4.1 Présentation

Les cartes cognitives représentent des influences entre des concepts. De par leur structure graphique et la notion d'influence, elles peuvent être rapprochées des réseaux d'influence [Pearl, 2005]. Elles peuvent par conséquent être également rapprochées des réseaux bayésiens [Kjaerulff et Madsen, 2008]. Des mécanismes permettant de calculer l'influence d'un concept sur un autre sont fournis. Elles aident un ou plusieurs utilisateurs à prendre une décision et comprendre les conséquences de celle-ci. Le terme de carte cognitive a été inventé par Edward C. Tolman [Tolman, 1948] qui étudiait la manière dont des rats sortaient d'un labyrinthe. Il utilise le terme de carte cognitive pour désigner une représentation de l'espace par les rats sous la forme d'une carte stockée dans leur cerveau. Le terme a été repris par Robert Axelrod [Axelrod, 1976] qui en fait un modèle d'influence clair. Il les a modélisées pour le domaine des sciences politiques comme étant les croyances d'un individu dans un domaine particulier. Une carte cognitive ne représente donc plus un environnement spatial mais à la fois le contenu de la pensée d'un individu et sa représentation graphique. James Doyle et David Ford [Doyle et Ford, 1999] ont séparé ces deux notions et le terme de carte cognitive désigne désormais la représentation graphique qui n'a pas nécessairement pour but de représenter la pensée. Les cartes cognitives sont parfois nommées *cartes causales* ou *graphes causaux*. Elles sont utilisées dans de nombreux domaines tels que la biologie [Tolman, 1948], la sociologie [Poignonec, 2006], l'écologie [Celik *et al.*, 2005], la politique [Levi et Tetlock, 1980].

Les informations représentées par une carte cognitive étant exclusivement des influences entre des concepts, elles sont moins expressives que les autres modèles de cartes sémantiques précédemment présentés. Certaines informations ne sont donc pas représentables par ce modèle. Dans notre exemple, nous ne pouvons donc représenter les informations décrits dans les exemples des sections 1.1.3, 1.2.3 et 1.3.3 puisque celles-ci ne sont pas des influences. Nous construisons donc un exemple spécialement pour ce modèle figurant des influences entre des concepts.

1.4.2 Représentation

Une carte cognitive permet de modéliser les influences de concepts sur d'autres concepts. Un *concept* est représenté par un texte bref le caractérisant. L'*influence* est une relation causale d'un concept-cause sur un concept-effet représentée par une flèche. La *valeur de l'influence* est représentée par un symbole étiquetant cette influence. On appelle *ensemble de valeurs* l'ensemble des valeurs que peut prendre une influence dans une carte cognitive. Selon les modèles de carte cognitive, cet ensemble de valeurs peut être un ensemble de valeurs symboliques. Parmi cette catégorie d'ensembles de valeurs, on retrouve l'ensemble $\{+, -\}$ défini par R. Axelrod [Axelrod, 1976] ainsi que des ensembles de valeurs symboliques munis d'un ordre total comme $\{nul < faible < moyen < fort\}$ [Dickerson et Bart, 1994; Zhou *et al.*, 2003]. Cet ensemble de valeurs peut également être un intervalle de valeurs numériques comme $[-1; 1]$ [Kosko, 1986; Satur et Liu, 1999].

Une carte cognitive est un graphe défini sur un ensemble de concepts représentant les concepts et sur un ensemble de valeurs d'influence et dans lequel les

sommets représentant les concepts sont liés entre eux par des arcs représentant les influences. Une carte cognitive figure donc trois différents types d'informations : des concepts, des influences et des valeurs.

Définition 1.6 (Carte cognitive)

Soit C un ensemble de concepts. Soit I un ensemble de valeurs.

Une carte cognitive définie sur I est un graphe orienté étiqueté $CM = (C, A, \text{label})$ où :

- les concepts de C sont les sommets du graphe ;
- $A \subseteq C \times C$ est l'ensemble des arcs appelés influences ;
- $\text{label} : A \rightarrow I$ est une application d'étiquetage qui associe à toute influence une valeur d'influence.

Dans le cas où $I = \{+, -\}$, une valeur d'influence de $+$ signifie qu'un concept influence positivement un autre concept ; une valeur d'influence de $-$ signifie qu'un concept influence négativement un autre concept.

Un chemin d'influence d'un concept vers un autre correspond à un chemin dans le graphe représentant la carte cognitive de ce concept vers l'autre. Le premier concept est appelé concept-source et le second concept-destination. La méthode de calcul de l'influence propagée d'un concept sur un autre est laissée au libre choix du concepteur de la carte cognitive. Généralement, celui-ci se décompose en trois étapes successives. La première consiste à lister l'intégralité des chemins d'influence permettant de lier le concept-source au concept-destination. Sur chacun de ces chemins, on calcule l'influence propagée sur ce chemin. L'influence propagée sur un chemin d'influence est la valeur de l'influence du concept-source sur le concept-destination sur ce chemin. Elle est calculée en cumulant les valeurs des influences composant ce chemin. On calcule enfin l'influence propagée d'un concept sur un autre. L'influence propagée est la valeur de l'influence globale du concept-source sur le concept-destination sur l'ensemble des chemins d'influence permettant de lier ces deux concepts. Elle est calculée en combinant les valeurs des influences propagées sur tous les chemins menant du concept-source au concept-destination.

Ainsi, Axelrod [Axelrod, 1976] a proposé, pour une carte définie sur l'ensemble de valeurs $\{+, -\}$, une méthode de calcul pour l'influence propagée dans un chemin d'influence basée sur la définition d'un opérateur \wedge , ainsi qu'une méthode de calcul pour l'influence propagée d'un concept sur un autre basée sur la définition d'un opérateur \vee . Pour ce dernier opérateur, il ajoute deux valeurs supplémentaires à l'ensemble de valeurs $\{+, -\}$:

- 0 qui représente une valeur nulle : le concept-source n'a aucune influence sur le concept-destination ;
- ? qui représente une valeur ambiguë, c'est-à-dire dont on ne peut décider si l'influence du concept-source sur le concept-destination est positive ou négative.

1.4.3 Édition et exemple illustratif

Un logiciel capable de représenter des cartes conceptuelles est généralement capable de représenter des cartes cognitives, à condition de remplacer les étiquettes des arcs par des valeurs d'influence. Cependant, de tels logiciels sont incapables de calculer les influences propagées entre deux concepts. Or, le calcul de ces influences est l'intérêt principal des cartes cognitives. SCCO¹⁶ est dédié spécifiquement à la manipulation de cartes cognitives et permet de calculer des influences propagées entre deux concepts.

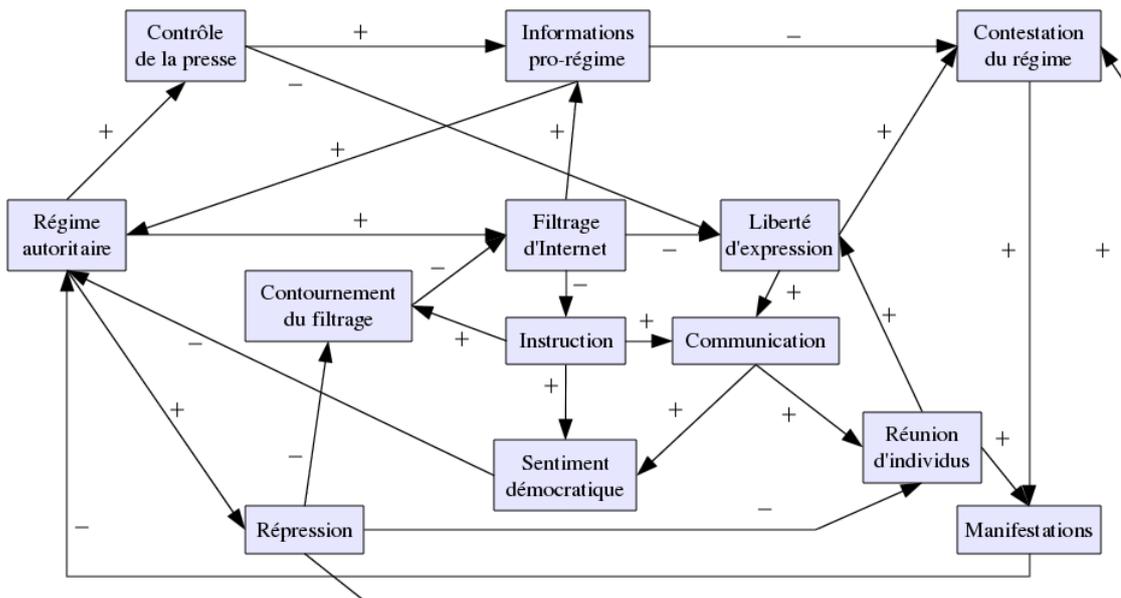


FIGURE 1.15 – Carte cognitive représentant l'influence du filtrage d'Internet sur le régime politique en place.

La carte cognitive de la figure 1.15 est définie sur l'ensemble de valeurs $I = \{+, -\}$ et représente l'influence du filtrage d'Internet ainsi que celle d'autres facteurs sur la contestation du régime autoritaire d'un État. Si on considère les concepts *Régime autoritaire* et *Filtrage d'Internet*, l'influence évaluée par $+$ partant de *Régime autoritaire* et allant vers *Filtrage d'Internet* signifie qu'un régime autoritaire influence favorablement le filtrage d'Internet. Autrement dit, un régime autoritaire filtre souvent Internet. De la même manière, si on considère les concepts *Filtrage d'Internet* et *Liberté d'expression*, l'influence évaluée par $-$ partant de *Filtrage d'Internet* et allant vers *Liberté d'expression* signifie que le filtrage d'Internet influence défavorablement la liberté d'expression. Autrement dit, le fait qu'Internet soit filtré limite la liberté d'expression.

En utilisant la définition des influences propagées d'Axelrod, on cherche à savoir comment un *Régime autoritaire* influence les *Informations pro-régime*. Avant tout chose, on doit lister les différents chemins qui permettent de lier *Régime autoritaire* à *Informations pro-régime*. Ces chemins sont au nombre de 3. Pour connaître l'influence propagée sur le chemin d'influence *Régime autoritaire* \rightarrow *Contrôle de*

¹⁶<http://forge.info.univ-angers.fr/~lionelc/CCdeGCjava/>

la presse \rightarrow Informations pro-régime, on utilise un opérateur \wedge , défini par Axelrod. Sur ce chemin, on note que le concept *Régime autoritaire* influence positivement *Contrôle de la presse* qui influence à son tour positivement *Informations pro-régime*. Intuitivement, on en déduit que sur ce chemin, *Régime autoritaire* influence positivement *Informations pro-régime* et que la valeur de l'influence propagée sur ce chemin est $+$. On applique le même raisonnement sur les autres chemins en calculant également une influence positive. Pour connaître l'influence propagée de *Régime autoritaire* sur *Informations pro-régime*, on doit agréger les influences propagées sur les différents chemins. Puisque sur les 3 chemins d'influence différents permettant de lier *Régime autoritaire* à *Informations pro-régime*, le premier concept influence positivement le second, on en déduit intuitivement que globalement, *Régime autoritaire* influence positivement *Informations pro-régime*. La valeur de l'influence propagée de *Régime autoritaire* sur *Informations pro-régime* est donc $+$.

1.4.4 Discussion

Le point fort du modèle des cartes cognitives est que des raisonnements y sont possibles : le calcul de l'influence d'un concept sur un autre. Cependant, la définition de ce calcul n'est pas unifiée. Selon la sémantique associée aux valeurs et la manière d'obtenir les informations, les opérateurs permettant ce calcul doivent être adaptés. En conséquence, l'influence propagée d'un concept sur un autre calculée ne représente pas toujours une valeur réelle si les opérateurs utilisés n'obéissent pas à un formalisme particulier. Néanmoins, un calcul même non formel permet généralement de se donner une idée de la valeur de l'influence.

Un problème causé par la définition de l'influence propagée par R. Axelrod est qu'elle aboutit souvent à une valeur ambiguë pour les influences propagées entre deux concepts. Par exemple, si la quasi-totalité des chemins permettant de lier deux concepts portent la valeur $+$ mais qu'un seul d'entre eux porte la valeur $-$, la valeur d'influence d'un concept sur l'autre sera ambiguë (?). Il est donc naturel de vouloir une représentation plus riche de l'influence, de manière à déterminer si une valeur d'influence ambiguë est plus proche de $+$ ou de $-$. On peut trouver préférable d'utiliser des valeurs numériques, telles que celles de l'intervalle $[-1; 1]$ ou symboliques telles que celles de $\{nul < faible < moyen < fort\}$. La difficulté est alors de définir des opérateurs adéquats capables de combiner ces valeurs. Si on utilise par exemple des valeurs numériques, selon ce que représentent ces valeurs, les opérateurs seront différents. On utilise généralement un produit ou une moyenne [Chauvin *et al.*, 2011] mais si elles représentent des possibilités [Dubois et Prade, 1988] ou des valeurs floues [Zadeh, 1965], on utilisera plutôt un min ou un max [Kosko, 1986].

1.5 Comparaison

Pour comparer les différents modèles de cartes sémantiques, nous présentons tout d'abord un tableau comparant les modèles de cartes selon différents critères dans la section 1.5.1. Cette comparaison concerne différents critères qui sont détaillés dans

cette section. Étant donné la particularité des informations d'une carte cognitive, nous proposons une manière de représenter la notion d'influence pour chacun des autres modèles de cartes sémantiques dans la section 1.5.2.

1.5.1 Tableau de comparaison

Le tableau de la figure 1.16 compare les différents modèles de cartes sémantiques sur plusieurs points. Il discute de l'importance que chaque modèle accorde à ces critères en leur attribuant une valeur parmi *faible*, *moyen*, *fort* et *très fort*.

Pour expliquer les différents points de ce tableau, on présente tout d'abord les objectifs de chacun des modèles de cartes sémantiques, ainsi que leurs domaines d'application. Puis, on s'intéresse à la facilité de leurs constructions dans la section. Après cela, on présente comment les modèles s'équilibrent entre la visualisation de l'information et la signification de cette information. Enfin, on compare le choix de modélisation pour ces cartes.

		cartes mentales	cartes conceptuelles	graphes conceptuels	cartes cognitives
Rôle et objectifs	rôle du modèle	décrire une idée	répondre à une question	raisonner sur des connaissances	représenter une influence
	facilité de mémorisation	très fort	fort	moyen	moyen
	aide à l'apprentissage	fort	très fort	faible	moyen
	capacité de raisonnement	faible	faible	très fort	moyen
	aide à la décision de groupe	fort	moyen	moyen	très fort
Construction		très facile	facile	facile	facile
Visualisation et sémantique	importance de l'aspect visuel	très fort	fort	moyen	moyen
	importance de la sémantique	moyen	fort	très fort	fort
	précision de l'information	moyen	moyen	très fort	fort
Modélisation	type de graphe	arborescence	graphe biparti orienté étiqueté	multigraphe biparti non-orienté étiqueté	graphe orienté étiqueté
	étiquetage des sommets	libre	texte libre et concis	élément d'une ontologie	texte libre
	étiquetage des arêtes	libre	non	numérotation des liens	valeurs

FIGURE 1.16 – Comparaison des modèles de cartes sémantiques.

Objectifs

Chaque modèle a été défini pour des objectifs particuliers.

La mémorisation est un point fort des cartes mentales puisque c'est ce pour quoi elles ont été conçues. De par leur aspect relativement simple, les cartes conceptuelles sont facilement mémorisables. Les cartes cognitives et les graphes concep-

tuels permettent une facilitation de la lecture des connaissances, mais n'ont pas été conçus pour la mémorisation. Néanmoins, l'aspect graphique aide grandement à une telle tâche.

L'apprentissage est l'objectif principal des cartes conceptuelles. Elles sont donc particulièrement adaptées pour répondre à une question centrale. De part leur forte capacité de mémorisation, les cartes mentales sont également un bon outil pour l'aide à l'apprentissage. Les cartes cognitives peuvent être utiles pour représenter clairement et comprendre une influence. En revanche, les graphes conceptuels ne permettent pas un apprentissage aisé de l'information.

Le formalisme est un point fort des graphes conceptuels, qui ont été conçus pour raisonner sur des connaissances. Les cartes cognitives calculent des influences et permettent donc un raisonnement simple. Les cartes mentales et les cartes conceptuelles, quant à elles, ne permettent pas de raisonner sur des connaissances, mais il est toutefois possible d'en déduire indirectement de nouvelles.

L'aide à la prise de décision est une utilisation très courante des cartes cognitives : le calcul d'influence permet en effet d'avoir un aperçu des conséquences d'une telle décision. Il en va de même pour les cartes mentales qui se construisent souvent en groupe. Les cartes conceptuelles et les graphes conceptuels ne permettent pas vraiment une aide à la décision de groupe. Cependant, les raisonnements fournis par les graphes conceptuels peuvent être utiles pour l'aide à la décision, même si l'aspect collectif n'est pas pris en compte par le modèle.

Construction

Nous pouvons noter un point commun essentiel entre ces modèles de cartes : chacun d'eux permet une construction facile. Ces modèles ont été conçus pour représenter graphiquement des connaissances et pour faciliter leur modélisation. La carte mentale est la plus simple à construire car la moins contrainte et le graphe conceptuel le plus compliqué car étant le plus contraint. En effet, moins le modèle d'une carte est formel, plus elle est facile à construire, et plus le modèle est formel, plus il y a d'informations précises à fournir et par conséquent, plus sa construction est complexe.

Visualisation et sémantique

Ces modèles s'équilibrent entre deux aspects : la visualisation et la sémantique. En effet, une information précise et non ambiguë lui donne souvent un aspect graphique complexe. De même, une information représentée graphiquement de manière simple est parfois si concise qu'elle devient ambiguë. Typiquement, la visualisation des informations est très importante pour les cartes mentales aux dépens de leur sémantique, celles-ci étant conçues pour une mémorisation facile des informations, tandis qu'avec les graphes conceptuels, c'est l'inverse : la sémantique prend le pas sur leur visualisation, les graphes conceptuels étant en effet conçus pour supprimer toute ambiguïté, de manière à pouvoir raisonner. C'est aussi, du moins en partie, le cas avec les cartes cognitives, dont l'aspect visuel importe peu : comme elles calculent des influences, l'accent est donné sur les valeurs de ces influences et donc sur la sémantique. La visualisation n'est donc pas l'objectif premier

mais est tout de même essentielle pour que la carte garde une lisibilité correcte. Concernant les cartes conceptuelles, les deux aspects sont importants puisque la carte doit être à la fois facilement lisible et compréhensible pour l'apprentissage.

Modélisation

Chacune des cartes sémantiques présentée ici est définie sur un modèle de graphe particulier, mais elles ont globalement une structure similaire : des concepts représentés par des sommets liés entre eux. La différence entre ces modèles est essentiellement la représentation des liens. Pour les cartes mentales, les liens, lorsqu'ils sont étiquetés par un texte, sont représentés par des sommets et deviennent semblables aux concepts. Pour les cartes cognitives, un lien est un arc entre des sommets : il est étiqueté pour représenter la valeur de l'influence qu'il représente. Pour les cartes conceptuelles et les graphes conceptuels, un lien est aussi un sommet qui est étiqueté pour identifier la relation et ce sommet est relié à un ou plusieurs sommets concept par un arc ou une arête selon le modèle de graphe.

L'orientation des liens dans les cartes n'est pas toujours indiquée et a un sens différent selon les modèles. Celle-ci sert généralement à connaître le rôle de chacun des concepts impliqués dans une relation. Avec une carte mentale, il n'y a pas d'orientation : celle-ci ayant l'aspect d'une arborescence, on peut déduire une orientation implicite en partant du sommet racine vers les sommets feuille. Avec les graphes conceptuels, la numérotation des arêtes permet de spécifier le rôle de chacun des concepts impliqués dans une relation et par conséquent l'orientation des liens. Avec les cartes conceptuelles, l'orientation représente le sens de lecture de la carte tandis qu'avec les cartes cognitives, l'orientation permet d'identifier pour chaque influence quel est le concept-cause et quel est le concept-effet.

Pour être plus précis dans la description de la modélisation, on peut noter qu'il est possible d'étiqueter les sommets d'une carte mentale comme on le souhaite : texte, image, couleur, épaisseur de trait... Pour tous les autres modèles, les sommets doivent être étiquetés par un texte. Pour les cartes conceptuelles et les graphes conceptuels, il est également obligatoire d'étiqueter les relations. L'étiquetage dans les graphes conceptuels n'est pas libre. Concernant l'étiquetage des arêtes et des arcs, il est libre pour les cartes mentales. En ce qui concerne les cartes cognitives, les arcs ne peuvent être étiquetés que par des valeurs représentant les valeurs des influences entre deux concepts. Pour les cartes conceptuelles et les graphes conceptuels, les arêtes et arcs ne sont pas étiquetés : leur rôle est de lier des sommets concept à des sommets relation et qui eux sont étiquetés. Selon le type de connaissances à représenter, certains modèles sont donc plus adaptés que d'autres.

1.5.2 Influences

Comme expliqué précédemment, les informations représentées par une carte cognitive sont essentiellement des influences. Les autres modèles de cartes sémantiques présentés précédemment sont cependant beaucoup plus expressifs. Par conséquent, ils devraient être capables de modéliser une influence et donc d'exprimer les informations d'une carte cognitive. Nous montrons comment représenter les influences

d'une carte cognitive pour ces modèles, en utilisant le même exemple. Nous prenons un extrait de la carte cognitive de la figure 1.15 en nous limitant aux concepts *Filtrage d'Internet*, *Instruction*, *Contournement du filtrage* et *Liberté d'expression*.

Cartes mentales

Les cartes mentales étant centrées sur une idée principale, elles peuvent difficilement représenter une carte cognitive puisque celles-ci n'ont pas nécessairement de concept central. La notion d'arborescence limite de plus grandement les possibilités de liens entre des concepts. Néanmoins, on peut représenter une influence entre deux concepts grâce à un arc. La valeur de l'influence ainsi que son signe peuvent être représentés par exemple par l'épaisseur du trait et sa couleur, comme présenté sur la figure 1.17. L'influence de *Contournement du filtrage* vers *Filtrage d'Internet* n'a pas pu être représentée par un lien classique, en raison de la structure arborescente de la carte mentale. Elle a été substituée par un arc.

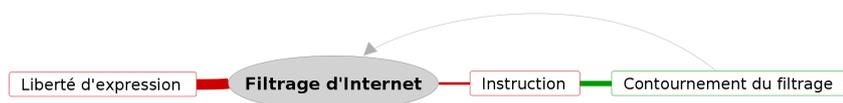


FIGURE 1.17 – Carte mentale représentant des relations d'influence.

Cartes conceptuelles

Les cartes conceptuelles n'ont pas de contraintes de structure. Les influences sont des relations ternaires qui associent un concept à un autre concept et à une valeur d'influence. On peut donc les représenter par trois concepts liés en série par des arcs, comme présenté sur la figure 1.18. La carte devient cependant illisible dès qu'un concept est influencé par valeurs différentes issues de deux concepts.

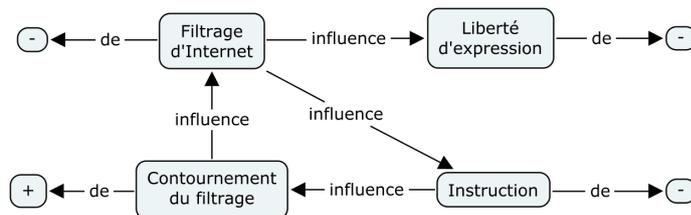


FIGURE 1.18 – Carte conceptuelle représentant des relations d'influence sous forme ternaire.

On peut également indiquer la valeur d'influence directement sur l'arc, comme présenté sur la figure 1.19. Cette représentation est plus juste et diminue les ambiguïtés mais les influences de la carte perdent en lisibilité.

Graphes conceptuels

Les graphes conceptuels peuvent quant à eux représenter de manière formelle les influences des cartes cognitives. Si l'ensemble de valeurs sur lequel la carte cognitive

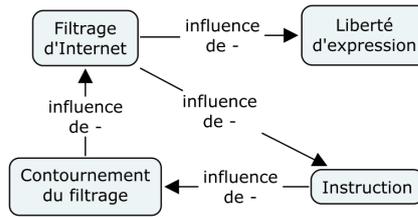


FIGURE 1.19 – Autre carte conceptuelle représentant des relations d’influence sous forme binaire.

est définie est un ensemble fini de valeurs tel que $\{+, -\}$ ou $\{nul < faible < moyen < fort\}$, les valeurs sont facilement représentables. En revanche, si c’est un ensemble infini tel que $[-1; 1]$, le modèle standard des graphes conceptuels est insuffisant puisque celui-ci ne gère pas les types de données. Il faut donc utiliser une extension du modèle pour qu’il puisse assurer cette gestion [Baget, 2007; Rudolph *et al.*, 2009].

Une influence peut être tout simplement représentée par une relation ternaire qui associe un concept-cause, un concept-effet et une valeur d’influence, comme présenté dans l’extrait de support de la figure 1.20. Le concept lié par l’arc étiqueté par 1 est le concept-cause, celui lié par l’arc 2 le concept-effet. L’arc 3 lie quant à lui la relation d’influence à une valeur. Représenter une influence revient donc simplement à représenter graphiquement une instance de cette relation, comme présenté dans le graphe conceptuel de la figure 1.21.

Type de relation (r)	$\sigma_1(r)$	$\sigma_2(r)$	$\sigma_3(r)$	Marqueur	Type
influence	Concept	Concept	Valeur	+	Valeur
				-	Valeur

FIGURE 1.20 – Extrait du support d’un graphe conceptuel figurant des relations d’influence ternaires.

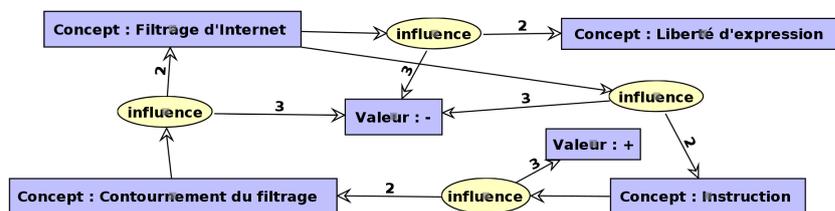


FIGURE 1.21 – Graphe conceptuel représentant des relations d’influence ternaires.

Cependant, comme pour les cartes conceptuelles, on peut également représenter une influence par une relation binaire entre les deux concepts liés. La valeur de l’influence fait partie du libellé de la relation. Néanmoins, l’ensemble de valeurs doit nécessairement être fini puisque l’intégralité des relations d’influence possibles doit être listée. L’extrait de support de la figure 1.22 décrit de telles relations d’influence. La figure 1.23 représente le même graphe conceptuel que précédemment, mais avec des relations d’influence binaires.

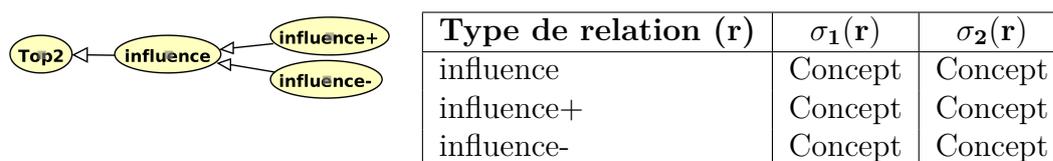


FIGURE 1.22 – Extrait du support d'un graphe conceptuel figurant des relations d'influence binaires.

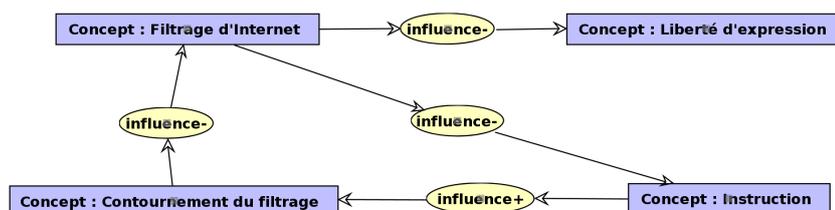


FIGURE 1.23 – Graphe conceptuel représentant des relations d'influence binaires.

Quelle que soit la modélisation d'influence choisie, aucun raisonnement concernant le calcul d'influence propagée n'est possible avec un graphe conceptuel. Pour cela, il faudrait étendre le modèle.

Conclusion

Au cours de ce chapitre, nous avons présenté différents modèles de cartes sémantiques et nous avons procédé à une étude comparative de ceux-ci.

Nous nous sommes intéressés principalement à la représentation, la visualisation et le raisonnement dans chaque modèle. Chaque type de carte a des objectifs différents. Pour certains modèles, la capacité de raisonnement est plus importante que pour d'autres. C'est pourquoi, pour ces modèles, la formalisation est essentielle. Leur principal point commun est qu'ils permettent tous une visualisation aisée de l'information : ils utilisent une représentation sous forme de graphe dont les sommets sont des concepts liés entre eux par des arcs représentant des relations entre ces concepts. Pour cette raison, la construction de ces cartes sémantiques est facile malgré l'aspect formel de certains modèles. Ces modèles permettent donc de représenter facilement des informations complexes. Il faut noter que ces aspects de visualisation sont liés à l'un des objectifs communs à ces modèles qui est de permettre une communication aisée au sein d'un groupe de personnes.

Les différences entre les modèles de cartes sémantiques s'expliquent par leurs objectifs distincts. Ces objectifs mêlent des aspects de sémantique et de visualisation de l'information. Il faut noter que ces aspects entrent en concurrence. En effet, du point de vue de la sémantique, une visualisation plus simple entraîne une sémantique moins précise de l'information alors qu'une sémantique précise nécessite une visualisation plus complexe. Les objectifs ont un impact sur la construction. Dans certains cas, la carte résulte d'un travail en commun entre plusieurs utilisateurs, dans d'autres cas, chaque utilisateur a la possibilité d'avoir un profil de la carte en fonction de ses intérêts. Enfin, la construction de la carte est d'autant plus aisée que la formalisation du modèle est faible alors qu'une formalisation plus

importante, plus contraignante pour la construction, assure une meilleure précision de la carte obtenue, voire une sémantique logique dans le cas des graphes conceptuels. Du point de vue de la visualisation, on peut voir que la richesse des attributs visuels proposés dans les cartes mentales pourrait être aisément appliquée aux autres modèles de cartes, par exemple dans le but de faciliter la mémorisation. D'un autre côté, nous avons noté que certains modèles n'ont pas été définis de manière précise. Dans un but de traitement automatique pour inférer des informations, il serait intéressant de réaliser une telle formalisation.

D'autres modèles représentent également sous forme graphique des informations et peuvent également être considérés comme des cartes sémantiques. Citons ici quelques modèles. Les cartes argumentatives [van Gelder, 2013] représentent sous la forme d'un graphe une série d'arguments et de contre-arguments qui se confortent ou se confrontent. Comme les cartes cognitives, elles sont attachées à un formalisme particulier et peuvent être liées à la théorie de l'argumentation [Dung, 1995]. Les cartes topiques [ISO/IEC JTC1/SC34, 2007] représentent formellement des informations à la manière d'un index. Leur formalisme est proche de celui de RDF [W3C, 2004a] et d'OWL [W3C, 2004b] dans le sens où elles représentent des connaissances ontologiques et peuvent être représentées sous la forme d'un document XML [W3C, 2008; ISO/IEC JTC1/SC34, 2008]. Elles ont cependant l'avantage d'être représentables facilement sous la forme d'un graphe.

Carte cognitive

Introduction

Nous avons présenté différents modèles de cartes sémantiques et nous avons procédé à une étude comparative de ceux-ci dans le chapitre 1. La plupart des modèles présentés étaient soit très formels mais difficiles à construire, tels les graphes conceptuels, soit peu formels mais faciles à construire, tels les cartes mentales. Les cartes cognitives combinent les avantages de ces deux aspects puisque c'est un modèle à la fois formel et simple à construire. Il constitue le modèle essentiel étudié dans notre thèse.

Ce chapitre a pour but de décrire formellement le modèle des cartes cognitives ainsi que les opérateurs associés de calcul de l'influence propagée d'un concept sur un autre. Nous définissons formellement le calcul de cette influence. Comme expliqué dans le chapitre précédent, une carte cognitive peut être définie sur des ensembles de valeurs très divers. On peut notamment utiliser les ensembles $\{+, -\}$, $\{nul < faible < moyen < fort\}$ ou encore $[-1; 1]$. Le calcul de l'influence propagée est effectué en agrégeant les valeurs portées par les influences, c'est-à-dire des valeurs appartenant à de tels ensembles.

Lionel Chauvin [Chauvin, 2009; Chauvin *et al.*, 2009] a étendu le modèle classique des cartes cognitives en lui associant une ontologie de concepts qui permet de regrouper plusieurs concepts de la carte cognitive au sein d'un seul concept plus grand. Grâce à cette ontologie, on est mesure de réaliser diverses opérations sur une carte, comme le calcul de l'influence ontologique ou la construction de vue. L'influence ontologique permet de calculer l'influence de n'importe quel concept de l'ontologie sur n'importe quel autre en effectuant un calcul sur les influences propagées entre les couples de concepts regroupés par chacun des concepts de l'ontologie. Nous réutilisons ces notions en remplaçant le terme d'« ontologie » par celui de « taxonomie » que nous pensons plus approprié pour la notion qu'il représente.

L. Chauvin a également introduit dans sa thèse la notion de vue de cartes

cognitives. Construire une vue d'une carte consiste à simplifier la dite carte en regroupant certains concepts au sein d'autres concepts grâce à l'ontologie, les influences étant calculés à partir de l'influence ontologique. Pour cela, on doit définir une échelle, c'est-à-dire les concepts qui apparaîtront dans la vue.

Enfin, comme nous l'avons vu, une carte cognitive peut être définie sur d'autres ensembles de valeurs. Dans la section 2.4, nous proposons d'autres opérateurs de calcul adaptés pour les ensembles valeurs $\{+, -\}$ et $\{nul < faible < moyen < fort\}$ pour l'influence propagée d'un concept sur un autre et pour l'influence taxonomique.

Pour présenter ce modèle, nous définissons tout d'abord comment calculer l'influence propagée d'un concept sur un autre pour l'ensemble de valeurs $[-1; 1]$ dans la section 2.1. Puis, nous présentons le modèle des cartes cognitives taxonomiques dans la section 2.2 qui nous sera utile pour les différentes notions définies dans ce mémoire de thèse. De ce modèle, nous présentons le mécanisme de vue dans la section 2.3 qui permet d'obtenir une représentation simplifiée d'une carte cognitive en adaptant nos définitions à la notion de taxonomie. Enfin, dans la section 2.4, nous montrons comment adapter les différents opérateurs définis dans ces différentes sections à d'autres ensembles de valeurs.

2.1 Définition du calcul de l'influence propagée

Nous avons déjà présenté formellement le modèle des cartes cognitives dans le chapitre 1 (définition 1.6) de manière générique pour n'importe quel ensemble de valeurs. Nous définissons à présent dans cette partie le calcul de l'influence propagée d'un concept sur un autre pour une carte cognitive dont les valeurs d'influence sont celles de l'intervalle de valeurs numériques $[-1; 1]$. Ces définitions sont basées sur celles décrites par L. Chauvin [Chauvin, 2009]. Avec cet ensemble de valeurs, une valeur d'influence $\alpha > 0$ d'un concept vers un autre signifie que ce concept influence positivement l'autre. Une valeur d'influence $\alpha < 0$ d'un concept sur un autre signifie cette fois que le concept influence négativement l'autre. Une valeur de -1 ou de 1 signifie que l'influence est totale. Une valeur de 0 signifie qu'il n'y a pas d'influence directe du concept sur l'autre. Une influence portant cette valeur n'est toutefois pas équivalente à une absence d'influence : sa valeur sera en effet prise en compte dans le calcul de l'influence propagée d'un concept sur un autre. En règle générale, cette valeur n'est donc jamais placée dans une carte en raison du comportement inattendu qu'elle peut provoquer. Nous la conservons néanmoins dans l'ensemble de valeurs car les définitions présentées ici restent valables pour cette valeur.

Dans cette partie, nous définissons le calcul de l'influence propagée de n'importe quel concept de la carte sur n'importe quel autre. L'idée est de considérer que chaque chemin permettant de lier ce concept à l'autre représente une manière possible pour le concept d'influencer l'autre. Chacun de ces chemins amène une valeur d'influence qu'il faut calculer. Enfin, on agrège les valeurs apportées par les différents chemins pour calculer l'influence propagée du concept sur l'autre. Ce calcul se fait donc en 3 étapes :

1. considérer l'ensemble des chemins liant ce concept à l'autre dans la carte (section 2.1.1) ;
2. calculer l'influence propagée sur chacun des chemins (section 2.1.2) ;
3. calculer enfin l'influence propagée d'un concept sur un autre (section 2.1.3).

Pour illustrer ces différents calculs, la carte cognitive sur laquelle nous appliquons les opérateurs définis dans cette partie est celle de l'exemple 2.1.

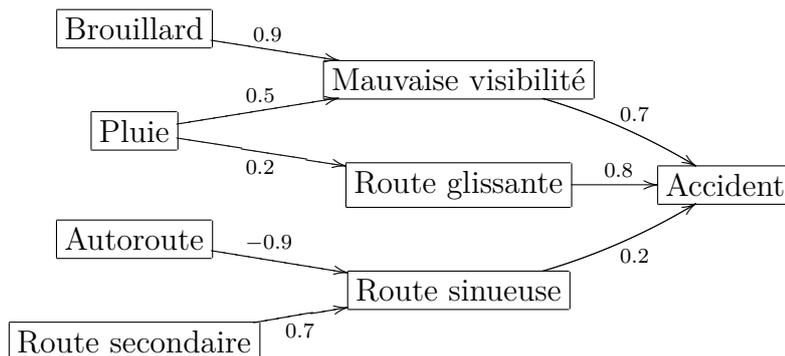


FIGURE 2.1 – La carte cognitive $CM1$ définie sur $[-1; 1]$.

Exemple 2.1. La carte $CM1$ (figure 2.1) représente différents facteurs pouvant influencer le risque qu'un accident de la route se produise. Cette carte cognitive est définie sur l'ensemble de valeurs $[-1; 1]$.

Si on considère l'influence de Pluie vers Mauvaise visibilité, si de la pluie tombe, elle influence positivement le risque que la visibilité soit mauvaise de 0.5. Si on considère l'influence de Autoroute vers Route sinueuse, le fait d'être sur une autoroute influence négativement le risque que la route soit sinueuse de 0.9. Si on considère les concepts Brouillard et Route glissante, il n'y a pas d'influence directe entre ces concepts.

2.1.1 Chemins minimaux

La première étape du calcul de l'influence propagée d'un concept sur un autre consiste à calculer l'intégralité des chemins qui permettent de lier ce concept à l'autre.

Avant toute chose, nous définissons ce qu'est un *chemin d'influence* d'un concept vers un autre. Un chemin d'influence est une séquence de couples de concepts basée sur les arcs d'une carte cognitive.

Définition 2.1 (Chemin d'influence d'un concept vers un autre)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts de CM .

On appelle un chemin d'influence P de c_1 vers c_2 une séquence de longueur $k \geq 1$ d'influences $(u_i, u_{i+1}) \in A$ avec $i \in [0; k - 1]$ telle que $u_0 = c_1$ et $u_k = c_2$.

On note un tel chemin d'influence $c_1 \rightarrow u_1 \rightarrow \dots \rightarrow u_{k-1} \rightarrow c_2$.

c_1 est la source du chemin d'influence P .

c_1 est la destination du chemin d'influence P .

Par souci de concision, nous parlerons souvent par la suite simplement de *chemin*.

On rappelle qu'une carte cognitive est un graphe. Si ce graphe autorise les circuits, il y a potentiellement une infinité de chemins permettant de lier un concept à un autre. Un ensemble fini de chemins est préférable pour calculer une influence. Pour cela, on définit la notion de chemin d'influence *minimal*. Cette définition permet d'assurer que l'ensemble des chemins minimaux soit fini.

Un *chemin minimal* est défini comme étant un chemin ne comportant aucun circuit, c'est-à-dire qu'aucun concept n'apparaît deux fois dans la séquence. Une telle définition autorise un circuit du concept-source au concept-destination mais pas au sein du chemin. D'autres définitions sont possibles et seront discutées dans la section 2.4.1.

Définition 2.2 (Chemin d'influence minimal)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

P est un chemin d'influence minimal ssi $\forall i, j \in [0; k-1], i \neq j \Rightarrow u_i \neq u_j \wedge u_{i+1} \neq u_{j+1}$.

La proposition 2.1 nous assure que, dans une carte cognitive sans circuit, n'importe quel chemin permettant de lier deux concepts est minimal.

Proposition 2.1. Soit $CM = (C, A, \text{label})$ une carte cognitive sans circuit. Soit P un chemin d'influence liant deux concepts de CM .

P est minimal.

Démonstration. Par définition, dans un graphe sans circuit, un chemin ne peut passer deux fois par le même sommet. P étant un chemin dans une carte cognitive sans circuit, il ne peut contenir deux fois le même sommet. On note k la longueur de P . On a donc $\forall i, j \in [0; k], i \neq j \Rightarrow u_i \neq u_j$. On en déduit trivialement que $\forall i, j \in [0; k-1], i \neq j \Rightarrow u_i \neq u_j \wedge u_{i+1} \neq u_{j+1}$.
Donc P est minimal. □

Maintenant que la notion de chemin minimal est définie, on peut définir l'ensemble des chemins minimaux d'un concept vers un autre. C'est sur cet ensemble de chemins que sera calculée l'influence propagée d'un concept sur un autre.

Définition 2.3 (Ensemble des chemins minimaux d'un concept vers un autre)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts de CM .

On note \mathcal{P}_{c_1, c_2} l'ensemble des chemins minimaux de c_1 vers c_2 .

Exemple 2.2. On considère les concepts *Pluie* et *Accident* dans la carte $CM1$ de l'exemple 2.1. Il y a deux chemins permettant de lier *Pluie* à *Accident* :

$$\begin{aligned} p_1 &= \{\text{Pluie} \rightarrow \text{Mauvaise visibilité} \rightarrow \text{Accident}\} \\ p_2 &= \{\text{Pluie} \rightarrow \text{Route glissante} \rightarrow \text{Accident}\} \end{aligned}$$

Ces deux chemins sont trivialement minimaux.

L'ensemble des chemins minimaux de Pluie vers Accident est donc $\mathcal{P}_{\text{Pluie, Accident}} = \{p_1, p_2\}$.

Au cours de ce mémoire, nous aurons besoin de la notion de parents d'un concept. L'ensemble des parents d'un concept est tout simplement l'ensemble des concepts parent de concept dans le graphe représentant la carte cognitive. La notion de *parents d'un concept* désigne donc l'ensemble des concepts qui influencent directement ce concept.

Définition 2.4 (Parents d'un concept)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $c \in C$ un concept.

Les parents de c sont :

$$\mathcal{C}(c) = \{c' \in C / (c', c) \in A\}$$

Exemple 2.3. Dans la carte cognitive CM1 (exemple 2.1), les parents du concept Route sinueuse sont Autoroute et Route. Le concept Pluie n'a pas de parent.

2.1.2 Influence propagée sur un chemin

Le calcul de l'*influence propagée sur un chemin d'influence* consiste à agréger les valeurs portées par les différentes influences qui composent ce chemin. Cette agrégation dépend bien évidemment de l'ensemble de valeurs sur lequel est définie la carte cognitive. Pour un ensemble de valeurs donné, il est également possible de définir cet opérateur de calcul de différentes manières.

Le calcul de l'influence propagée sur un chemin est défini ici, pour l'ensemble de valeurs $[-1; 1]$, comme étant un produit des valeurs d'influence composant ce chemin. Le produit permet de généraliser les opérateurs définis par Robert Axelrod [Axelrod, 1976] pour l'ensemble de valeurs $[-1; 1]$. En associant la valeur $+$ à 1 et la valeur $-$ à -1 , on retrouve en effet la même influence propagée sur un chemin.

Définition 2.5 (Influence propagée sur un chemin pour $[-1; 1]$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $I = [-1; 1]$. Soit P un chemin d'influence liant deux concepts de CM .

L'influence propagée sur P est :

$$\mathcal{IP}(P) = \prod_{i=0}^{k-1} \text{label}((u_i, u_{i+1}))$$

Exemple 2.4. On calcule les influences propagées sur les chemins p_1 et p_2 définis dans l'exemple 2.2 :

$$\begin{aligned} \mathcal{IP}(p_1) &= \times \begin{cases} \text{label}((\text{Pluie}, \text{Mauvaise visibilité})) \\ \text{label}((\text{Mauvaise visibilité}, \text{Accident})) \end{cases} = \times \begin{cases} 0.5 \\ 0.7 \end{cases} = 0.35 \\ \mathcal{IP}(p_2) &= \times \begin{cases} \text{label}((\text{Pluie}, \text{Route glissante})) \\ \text{label}((\text{Route glissante}, \text{Accident})) \end{cases} = \times \begin{cases} 0.2 \\ 0.8 \end{cases} = 0.16 \end{aligned}$$

2.1.3 Influence propagée d'un concept sur un autre

Le calcul de l'*influence propagée d'un concept sur un autre* consiste à agréger les influences propagées sur les différents chemins qui permettent de lier ce concept à l'autre. Cette agrégation dépend elle aussi de l'ensemble de valeurs sur lequel est définie la carte cognitive et il est aussi possible de définir cette agrégation de différentes manières pour un même ensemble de valeurs.

Pour l'ensemble de valeurs $[-1; 1]$, nous définissons ici l'influence propagée d'un concept sur un autre comme étant une moyenne des influences propagées sur les différents chemins qui permettent de lier ce concept à l'autre. Cette définition permet de quantifier la valeur ambiguë (?) qui apparaissait avec les opérateurs de R. Axelrod.

Définition 2.6 (Influence propagée d'un concept sur un autre sur $[-1; 1]$)
Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $I = [-1; 1]$. Soient $c_1, c_2 \in C$ deux concepts de CM .

L'influence propagée de c_1 sur c_2 est une fonction $\mathcal{I}: C \times C \rightarrow [-1; 1]$ telle que :

$$\mathcal{I}(c_1, c_2) = \begin{cases} 0 & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \frac{1}{|\mathcal{P}_{c_1, c_2}|} \times \sum_{P \in \mathcal{P}_{c_1, c_2}} \mathcal{IP}(P) & \text{sinon} \end{cases}$$

Il faut noter qu'ici, la sémantique de la valeur 0 est légèrement différente par rapport au cas où elle est portée par une influence. En effet, dans le cas d'une influence directe, elle signifie qu'il n'y a pas d'influence directe entre ces deux concepts. Ici, elle signifie que globalement, en considérant l'ensemble de la carte, il n'y a pas d'influence entre ces deux concepts.

Exemple 2.5. On calcule l'influence propagée de Pluie sur Accident dans $CM1$ grâce aux influences propagées sur les chemins calculées dans l'exemple 2.5.

$$\mathcal{I}(c_1, c_2) = \frac{1}{|\mathcal{P}_{\text{Pluie}, \text{Accident}}|} \times (\mathcal{IP}(p_1) + \mathcal{IP}(p_2)) = \frac{1}{2} \times (0.35 + 0.16) = 0.255$$

2.2 Carte cognitive taxonomique

Dans les différents chapitres qui composent cette thèse, nous aurons besoin d'un modèle de carte qui étend le modèle classique des cartes cognitives, le modèle des cartes cognitives taxonomiques. Ce modèle est celui des cartes cognitives ontologiques défini par L. Chauvin [Chauvin *et al.*, 2011] dans lequel le terme d'« ontologie » est remplacé par celui de « taxonomie ».

Dans un premier temps, nous définissons donc la notion de taxonomie dans la section 2.2.1, puis la notion de carte cognitive taxonomique en introduisant l'opérateur d'influence taxonomique associé dans la section 2.2.2.

2.2.1 Taxonomie

L'intérêt de la taxonomie est de regrouper plusieurs concepts d'une carte cognitive au sein d'un autre concept plus général. Un tel procédé permet ainsi par exemple

de simplifier une carte cognitive en réalisant une vue, comme expliqué dans la section 2.3. Dans un premier temps, nous allons définir cette taxonomie, puis différents opérateurs sur cette taxonomie qui nous seront utiles pour des définitions ultérieures.

Formellement, une *taxonomie* associe à un ensemble de concepts une relation d'ordre partiel de simple héritage. Une taxonomie peut donc être représentée par un ensemble d'arborescences.

Définition 2.7 (Taxonomie)

Soit C un ensemble de concepts.

Une taxonomie $T = (C, \leq)$ est un ensemble d'arborescences de concepts représentant une relation d'ordre partiel \leq .

La relation d'ordre d'une taxonomie étant une relation d'héritage, elle peut être interprétée comme étant une relation *est une sorte de*.

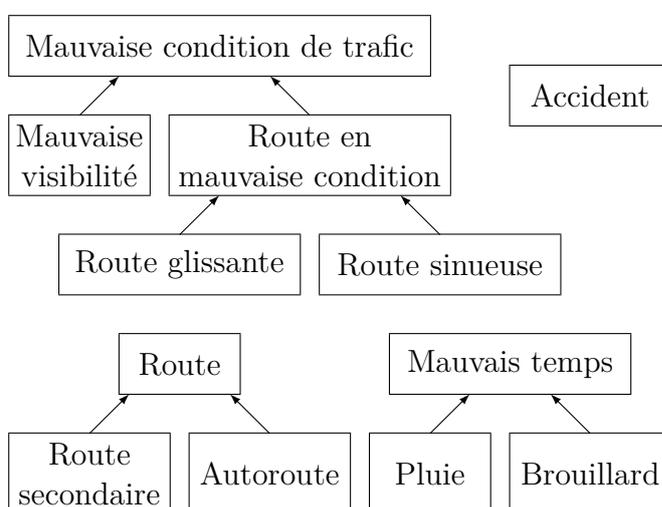


FIGURE 2.2 – La taxonomie de concepts $T1$.

Exemple 2.6. $T1$ (figure 2.2) est une taxonomie ordonnant des concepts. La relation d'ordre partiel \leq y est représentée par des liens fléchés. On peut par exemple y lire que la Pluie et le Brouillard sont des sortes de Mauvais temps.

Nous définissons à présent quelques notions liées à la taxonomie qui nous seront utiles dans les définitions à venir. Nous définissons ainsi la notion de comparabilité. Intuitivement, deux concepts sont *comparables* si l'un est plus petit que l'autre. Deux concepts sont *incomparables* s'ils ne sont pas comparables.

Définition 2.8 (Concepts comparables)

Soit $T = (C, \leq)$ une taxonomie. Soient $c_1, c_2 \in C$ deux concepts ordonnés par T .

c_1 et c_2 sont dits comparables, noté $c_1 \perp c_2$, ssi $(c_1 \leq c_2) \vee (c_2 \leq c_1)$.

c_1 et c_2 sont dits incomparables, noté $c_1 \parallel c_2$, ssi $\neg(c_1 \perp c_2)$.

Exemple 2.7. En reprenant la taxonomie $T1$ (exemple 2.6), Route glissante et Mauvaise condition de trafic sont comparables. En revanche, Mauvais temps et Mauvaise visibilité sont incomparables. On peut par ailleurs noter que Accident est incomparable avec tous les concepts ordonnés par $T1$, sauf avec lui-même.

Les concepts *minimaux* d'un ensemble de concepts par rapport à une taxonomie sont les concepts les plus petits par rapport à cette taxonomie. On définit de la même manière les concepts *maximaux* d'un ensemble de concepts comme étant les concepts les plus grands.

Définition 2.9 (Concepts minimaux & maximaux)

Soit $T = (C, \leq)$ une taxonomie de concepts C . Soit $C' \subseteq C$ un ensemble de concepts ordonnés par T .

Les concepts minimaux de C' par rapport à T sont :

$$\min_T(C') = \{c \in C' / \forall c' \in C', c' \leq c \Rightarrow c = c'\}$$

De la même manière, les concepts maximaux de C' par rapport à T sont :

$$\max_T(C') = \{c \in C' / \forall c' \in C', c \leq c' \Rightarrow c = c'\}$$

Exemple 2.8. On considère l'ensemble de concepts {Pluie, Brouillard, Mauvais temps, Mauvaise condition de trafic, Mauvaise visibilité, Autoroute, Accident}. Les concepts minimaux de cet ensemble par rapport à $T1$ (exemple 2.6) sont :

$$\{\text{Pluie, Brouillard, Mauvaise visibilité, Autoroute, Accident}\}$$

Les concepts maximaux de cet ensemble par rapport à $T1$ sont :

$$\{\text{Mauvais temps, Mauvaise condition de trafic, Autoroute, Accident}\}$$

Les concepts *élémentaires* d'une taxonomie sont les concepts les plus petits de cette taxonomie. Ce sont les concepts les plus spécialisés, c'est-à-dire ceux pour lesquels il n'y a pas de concept plus petit dans la taxonomie. Ce sont donc les feuilles des arbres représentant la relation d'héritage.

Définition 2.10 (Concepts élémentaires d'une taxonomie)

Soit $T = (C, \leq)$ une taxonomie de concepts.

Les concepts élémentaires de T sont $\text{elem}(T) = \min_T(C)$.

Les concepts élémentaires pour un concept selon une taxonomie sont les concepts élémentaires plus petits que ce concept dans cette taxonomie.

Définition 2.11 (Concepts élémentaires pour un concept)

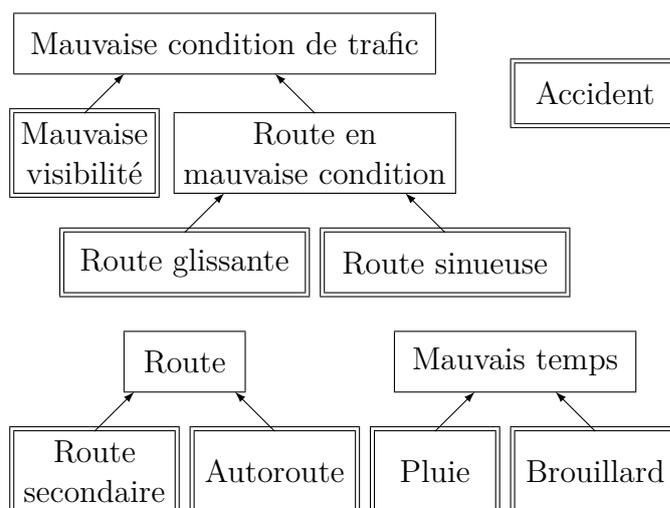
Soit $T = (C, \leq)$ une taxonomie de concepts et $c \in C$ un concept.

L'ensemble des concepts élémentaires pour c selon T est un sous-ensemble de $\text{elem}(T)$ tel que :

$$\text{elemPour}_T(c) = \{c' \in \text{elem}(T) / c' \leq c\}$$

Exemple 2.9. Les concepts élémentaires de $T1$ sont représentés doublement encadrés sur la figure 2.3.

Les concepts élémentaires pour Mauvaise condition de trafic sont {Mauvaise visibilité, Route sinueuse, Route glissante}.

FIGURE 2.3 – Concepts élémentaires de $T1$.

2.2.2 Influence taxonomique

L'influence taxonomique consiste à calculer l'influence d'un concept ordonné par la taxonomie sur un autre concept de la taxonomie par rapport à une carte cognitive. Pour calculer cette influence, il faut donc tout d'abord lier une taxonomie à une carte cognitive.

Une *carte cognitive taxonomique* est l'association d'une carte cognitive et d'une taxonomie. Les concepts élémentaires de la taxonomie sont les concepts de la carte cognitive.

Définition 2.12 (Carte cognitive taxonomique)

Une carte cognitive taxonomique TM définie sur un ensemble de valeurs I est l'association d'une taxonomie $T = (C, \leq)$ et d'une carte cognitive $(\text{elem}(T), A, \text{label})$ définie sur I .

Exemple 2.10. $TM1$ est une carte cognitive taxonomique associant la taxonomie $T1$ (exemple 2.6) et la carte cognitive $CM1$ (exemple 2.1).

L'*influence taxonomique* d'un concept d'une taxonomie sur un autre est une agrégation des influences propagées de tous les couples possibles constitués des concepts élémentaires pour le premier concept et ceux pour le second. Le calcul de l'influence taxonomique d'un concept sur un autre pour l'ensemble de valeurs $[-1; 1]$ est défini ici comme étant un intervalle entre la plus petite influence propagée des couples de concepts élémentaires et la plus grande.

Définition 2.13 (Influence taxonomique d'un concept sur un autre)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T définie sur $[-1; 1]$. Soient $c_1, c_2 \in C$ deux concepts ordonnés par T .

L'influence taxonomique de c_1 sur c_2 est une fonction $\mathcal{I}_T: C \times C \rightarrow \mathcal{P}([-1; 1])$

telle que :

$$\mathcal{I}_T(c_1, c_2) = \left[\min_{\substack{c'_1 \in \text{elemPour}_T(c_1) \\ c'_2 \in \text{elemPour}_T(c_2)}} \mathcal{I}(c'_1, c'_2) ; \max_{\substack{c'_1 \in \text{elemPour}_T(c_1) \\ c'_2 \in \text{elemPour}_T(c_2)}} \mathcal{I}(c'_1, c'_2) \right]$$

Exemple 2.11. *On considère les concepts Mauvais temps et Mauvaise condition de trafic dans la carte cognitive TM1 (exemple 2.10). Les concepts élémentaires pour Mauvais temps sont {Pluie, Brouillard}. Ceux pour Mauvaise condition de trafic ont déjà été énoncés dans l'exemple 2.9. On calcule les influences propagées de chacun des couples de concepts élémentaires de Mauvais temps sur Mauvaise condition de trafic.*

$$\begin{aligned} \mathcal{I}(\text{Brouillard}, \text{Mauvaise visibilité}) &= 0.9 \\ \mathcal{I}(\text{Brouillard}, \text{Route glissante}) &= 0 \\ \mathcal{I}(\text{Brouillard}, \text{Route sinueuse}) &= 0 \\ \mathcal{I}(\text{Pluie}, \text{Mauvaise visibilité}) &= 0.5 \\ \mathcal{I}(\text{Pluie}, \text{Route glissante}) &= 0.2 \\ \mathcal{I}(\text{Pluie}, \text{Route sinueuse}) &= 0 \end{aligned}$$

L'influence taxonomique de Mauvais temps sur Mauvaise condition de trafic est alors :

$$\mathcal{I}_T(\text{Mauvais temps}, \text{Mauvaise condition de trafic}) = [0; 0.9]$$

Utiliser un intervalle permet d'avoir une valeur synthétique représentant les valeurs possibles des influences propagées entre les concepts élémentaires. Un intervalle dont les bornes sont proches permet également de conclure que les influences propagées entre les concepts élémentaires sont homogènes, c'est-à-dire que les influences entre les concepts regroupés sont semblables. Au contraire, un intervalle dont les bornes sont éloignées signifie que les influences propagées sont hétérogènes.

2.3 Vue d'une carte cognitive taxonomique

Le mécanisme de vue a été formellement défini par L. Chauvin dans sa thèse [Chauvin, 2009]. Il permet de simplifier une carte cognitive en rassemblant plusieurs concepts d'une carte au sein d'un concept regroupant à l'aide d'une ontologie. Les influences entre les concepts de la carte originelle doivent être réaffectées aux concepts les regroupant et leurs valeurs calculées en fonction des influences qu'elles représentent. Pour sélectionner quels concepts de la carte cognitive seront présents dans la vue, l'utilisateur souhaitant réaliser cette vue doit choisir une échelle. Cette échelle représente le degré de détail qu'il souhaite obtenir dans la vue finale. Nous redéfinissons ici les notions d'échelle dans la section 2.3.1 et de vue dans la section 2.3.2 en utilisant la notion de taxonomie. Nous montrons enfin comment construire une vue adaptée à un usage particulier dans la section 2.3.3.

2.3.1 Échelle

Une *échelle* est l'ensemble des concepts qui seront présents dans la vue. Par conséquent, cette échelle doit respecter certaines contraintes. D'une part, les concepts

d'une échelle doivent être incomparables entre eux pour s'assurer qu'un même concept ne sera pas représenté deux fois dans la vue. D'autre part, une échelle doit être représentative des concepts de la carte cognitive. Un ensemble de concepts est *représentatif* d'un autre ensemble de concepts par rapport à une taxonomie les ordonnant si chaque concept du second ensemble a au moins un représentant dans le premier. Un concept *représente* un autre concept s'il lui est comparable.

De la notion de comparabilité (définition 2.8), nous définissons trivialement la notion d'*ensemble de concepts incomparables*.

Définition 2.14 (Ensemble de concepts incomparables)

Soit $T = (C, \leq)$ une taxonomie. Soit $C' \subseteq C$ un ensemble de concepts ordonnés par T .

C' est un ensemble de concepts incomparables ssi $\forall c_1, c_2 \in C', c_1 \neq c_2 \Rightarrow c_1 \parallel c_2$.

Exemple 2.12. On considère les trois ensembles de concepts suivants :

$$\begin{aligned} C_1 &= \{ \text{Mauvaise visibilité, Route, Mauvais temps, Accident} \} \\ C_2 &= \{ \text{Mauvaise condition de trafic, Route,} \\ &\quad \text{Autoroute, Mauvais temps, Accident} \} \\ C_3 &= \{ \text{Mauvaise visibilité, Route en mauvaise condition,} \\ &\quad \text{Route secondaire, Autoroute, Mauvais temps, Accident} \} \end{aligned}$$

C_1 et C_3 sont des ensembles de concepts incomparables selon T_1 (exemple 2.6). En revanche, C_2 n'est pas un ensemble de concepts incomparables puisque $\text{Autoroute} \leq \text{Route}$.

Pour vérifier simplement la représentativité, nous allons vérifier que l'ensemble des concepts élémentaires pour tous les concepts du premier ensemble inclut les concepts élémentaires du second.

Avant toute chose, il faut donc définir ce que sont les *concepts élémentaires pour un ensemble de concepts*.

Définition 2.15 (Concepts élémentaires pour un ensemble de concepts)

Soit $T = (C, \leq)$ une taxonomie. Soit $C' \subseteq C$ un ensemble de concepts ordonnés par T .

L'ensemble de concepts élémentaires pour C' est :

$$\text{elemPour}_T(C') = \bigcup_{c \in C'} \text{elemPour}_T(c)$$

Remarque. Pour une taxonomie $T = (C, \leq)$, $\text{elemPour}_T(C) = \min_T(C)$.

Exemple 2.13. On considère l'ensemble de concepts $\{\text{Mauvaise visibilité, Mauvaise route}\}$ ordonnés par la taxonomie T_1 (exemple 2.6). L'ensemble des concepts élémentaires pour cet ensemble est $\{\text{Mauvaise visibilité, Route glissante, Route sinueuse}\}$.

À présent, la notion d'ensemble représentatif est facile à définir.

Définition 2.16 (Ensemble représentatif d'un autre ensemble)

Soit $T = (C, \leq)$ une taxonomie. Soient $C_1, C_2 \subseteq C$ deux ensembles de concepts ordonnés par T .

C_1 est un ensemble représentatif de C_2 ssi $\text{elemPour}_T(C_2) \subseteq \text{elemPour}_T(C_1)$.

Exemple 2.14. Avec $T1$ (exemple 2.6), l'ensemble de concepts $\{\text{Mauvaise visibilité, Mauvaise route}\}$ est représentatif de $\{\text{Mauvaise condition de trafic}\}$. En effet, les concepts élémentaires pour $\{\text{Mauvaise condition de trafic}\}$ ($\{\text{Mauvaise visibilité, Route glissante, Route sinueuse}\}$) sont, étant égaux, inclus dans ceux de $\{\text{Mauvaise visibilité, Mauvaise route}\}$ qui ont déjà été calculés dans l'exemple 2.13.

On reprend les trois ensembles de concepts de l'exemple 2.12. De la même manière, on peut montrer que C_2 et C_3 sont représentatifs des concepts élémentaires de T . En revanche, C_1 n'est pas représentatif des concepts élémentaires de T puisque *Route glissante* et *Route sinueuse* ne trouvent pas de représentant dans C_1 .

Une *échelle* est un ensemble de concepts incomparables et représentatif des concepts d'une carte. Les concepts d'une carte taxonomique étant les concepts élémentaires de cette taxonomie, une échelle doit donc être représentative des concepts ordonnés par la taxonomie.

Définition 2.17 (Échelle)

Soit $T = (C, \leq)$ une taxonomie. Soit $C' \subseteq C$ un ensemble de concepts ordonnés par T .

C' est une échelle pour T ssi :

- C' est un ensemble de concepts incomparables ;
- C' est représentatif de C .

Exemple 2.15. On reprend à nouveau les ensembles de concepts de l'exemple 2.12. Comme déjà montré aux exemples 2.12 et 2.14, C_1 est un ensemble de concepts incomparables mais non représentatif de $T1$: C_1 n'est donc pas une échelle. C_2 est représentatif mais ses concepts ne sont pas incomparables : C_2 n'est donc pas une échelle non plus. En revanche, C_3 est un ensemble de concepts incomparables et représentatif des concepts de $T1$: C_3 est donc une échelle pour $T1$.

2.3.2 Vue pour une échelle

Maintenant que nous disposons des concepts de la vue sous la forme d'une échelle, on est en mesure de construire cette vue. Pour cela, il faut déterminer quelles seront les influences entre les concepts ainsi que les valeurs de ces influences.

Pour définir ces influences, on cherche à connaître les couples de concepts connectés dans une carte taxonomique. Deux concepts d'une taxonomie sont *connectés* dans une carte cognitive taxonomique s'il existe un concept élémentaire pour chacun d'entre eux liés par une influence dans la carte.

Définition 2.18 (Connexion entre deux concepts)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive

taxonomique associée à T . Soient $c_1, c_2 \in C$ deux concepts ordonnés par T .

c_1 et c_2 sont connectés dans TM ssi $\exists c'_1 \in \text{elemPour}_T(c_1), \exists c'_2 \in \text{elemPour}_T(c_2) / (c_1, c_2) \in A$.

Exemple 2.16. Dans $TM1$ (exemple 2.10), les concepts Mauvais temps et Mauvaise condition de trafic sont connectés selon $TM1$ car leurs concepts élémentaires respectifs Pluie et Route glissante sont directement liés par une influence dans la carte.

Dans la *vue*, les concepts sont ceux de l'échelle. On place une influence entre chaque couple de concepts connectés. La valeur de ces influences est l'influence taxonomique entre ces concepts. Cependant, si les deux concepts aux extrémités de cette influence sont des concepts élémentaires, alors cela signifie qu'il existe déjà une influence entre ces deux concepts dans la carte originale. Dans une telle situation, la valeur de cette influence originale est plus pertinente que celle de l'influence taxonomique et elle est donc conservée dans la vue.

Définition 2.19 (Vue d'une carte taxonomique)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T définie sur un ensemble de valeurs I . Soit C' une échelle pour T .

Une vue de TM pour C' est une carte cognitive (C', A', label') définie sur $I \cup \mathcal{P}(I)$ où :

- $A' = \{(c_1, c_2) / c_1, c_2 \in C' \wedge c_1 \text{ et } c_2 \text{ sont connectés dans } TM\}$;
- $\text{label}'((c_1, c_2)) = \begin{cases} \text{label}((c_1, c_2)) & \text{si } (c_1, c_2) \in A \\ \mathcal{I}_T(c_1, c_2) & \text{sinon} \end{cases}$

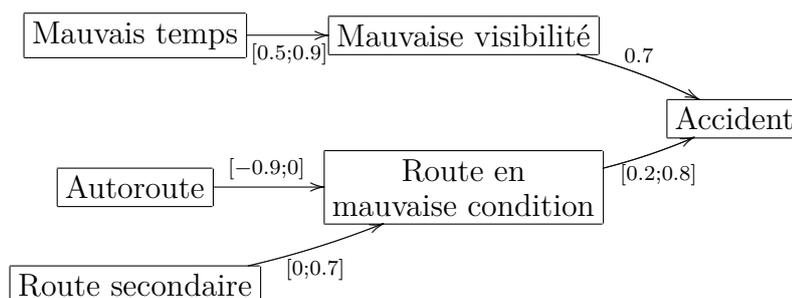


FIGURE 2.4 – Vue de $TM1$ pour l'échelle C_3 .

Exemple 2.17. La carte cognitive de la figure 2.4 est une vue de $TM1$ (exemple 2.10) pour l'échelle $C_3 = \{\text{Mauvaise visibilité, Route en mauvaise condition, Route secondaire, Autoroute, Mauvais temps, Accident}\}$.

2.3.3 Vues adaptées

Une vue d'une carte cognitive adaptée à un utilisateur est tout simplement une simplification de la carte originale détaillant en priorité les informations pertinentes. Ceci permet d'obtenir une vue dédiée à un usage particulier.

Pour adapter une vue à un usage particulier, on a besoin de définir une échelle qui correspond à un certain *profil d'utilisateur*. Un profil est tout simplement une échelle pour une taxonomie.

Définition 2.20 (Profil d'utilisateur)

Soit $T = (C, \leq)$ une taxonomie.

Un profil d'utilisateur P_u est une échelle pour T .

Exemple 2.18. On considère à nouveau la taxonomie $T1$ (exemple 2.6). Le profil $P_m = \{\text{Brouillard, Pluie, Route, Mauvaise condition de trafic}\}$ est un profil pour l'utilisateur « météorologiste ».

On définit trivialement la *vue pour un profil* comme étant la vue de l'échelle représentée par le profil.

Définition 2.21 (Vue pour un profil)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T définie sur un ensemble de valeurs I . Soit P_u un profil d'utilisateur.

Une vue de TM pour le profil P_u est la vue de TM pour P_u .

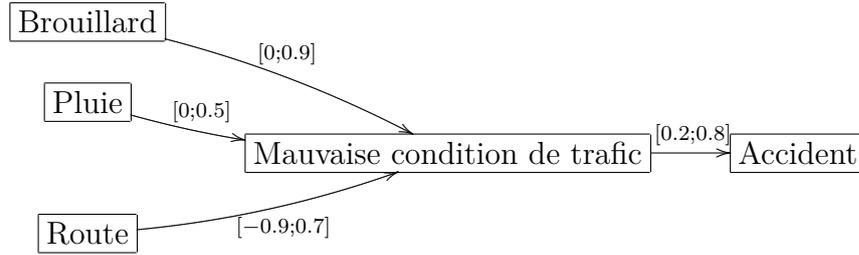


FIGURE 2.5 – Vue de $TM1$ pour le profil P_m .

Exemple 2.19. La carte cognitive de la figure 2.5 est une vue de $TM1$ (exemple 2.10) adaptée à l'usage de l'utilisateur « météorologiste » (exemple 2.18).

Lorsque plusieurs utilisateurs souhaitent partager une même carte pour travailler ensemble, une vue partagée doit être construite de telle manière qu'elle soit adaptée à tous les usages. Pour cela, on définit une nouvelle échelle basée sur les concepts partagés par les différents profils. Cet ensemble de concepts partagés est tout simplement l'union des ensembles de concepts représentés par les profils à laquelle on applique un min de sorte à fournir les concepts les plus spécialisés qui intéressent au moins un utilisateur. De plus, ceci nous permet de nous assurer que cet ensemble de concepts est bien une échelle.

Définition 2.22 (Concepts partagés)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T définie sur un ensemble de valeurs I . Soit $P = \{P_{u_1}, \dots, P_{u_n}\}$ un ensemble de profils d'utilisateurs.

L'ensemble des concepts partagés pour les profils P_{u_1}, \dots, P_{u_n} est :

$$\text{ConceptsPartagés}(P) = \min\left(\bigcup_{P_{u_i} \in P} P_{u_i}\right)$$

L'ensemble des concepts partagés d'un ensemble de profils étant une échelle, on peut aisément construire une vue basée sur cet ensemble.

Définition 2.23 (Vue partagée)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T définie sur un ensemble de valeurs I . Soit $P = \{P_{u_1}, \dots, P_{u_n}\}$ un ensemble de profils d'utilisateurs.

Une vue partagée de TM pour P est la vue de TM pour $\text{ConceptsPartagés}(P)$.

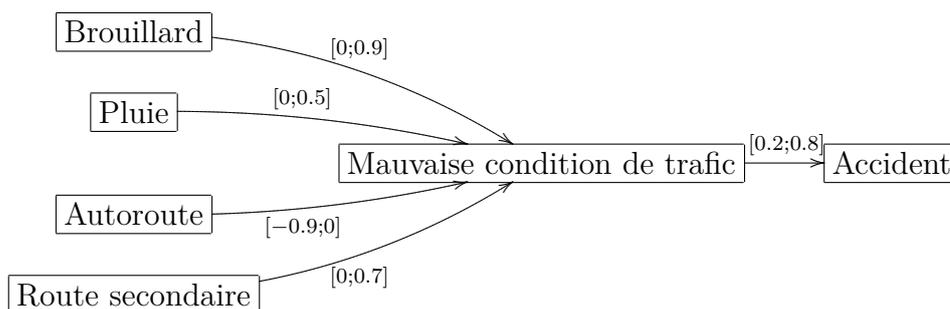


FIGURE 2.6 – Vue de $TM1$ pour les profils P_m et P_r .

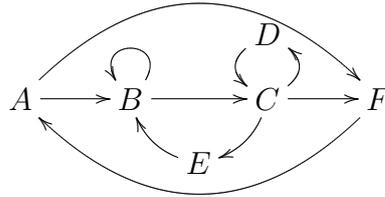
Exemple 2.20. On définit le profil « constructeur de route » $P_r = \{\text{Autoroute}, \text{Route secondaire}, \text{Mauvais temps}, \text{Mauvaise condition de trafic}, \text{Accident}\}$ pour $T1$ (exemple 2.6). La carte cognitive de la figure 2.6 représente une vue partagée de $TM1$ (exemple 2.10) de pour les profils « météorologiste » (exemple 2.18) et « constructeur de route ».

Dans sa thèse, L. Chauvin décrit différents mécanismes d'autres types de vues, comme les vues pour un ensemble de concepts sélectionnés ou les vues d'une influence. Nous ne détaillerons pas ici ces mécanismes, bien qu'ils puissent tout à fait s'appliquer à notre modèle.

2.4 Paramétrage

Dans cette section, nous discutons de la manière d'adapter nos définitions pour d'autres ensembles de valeurs que $[-1; 1]$. À noter qu'on peut définir de différentes manières les opérateurs de calcul d'influence propagée sur un chemin, d'influence propagée d'un concept sur un autre et d'influence taxonomique pour ce même ensemble $[-1; 1]$. Ce n'est pas l'objet de la discussion ici : le paramétrage concerne uniquement l'application du modèle à d'autres ensembles de valeurs. La discussion sur la sémantique des valeurs et l'adaptation des opérateurs en conséquence est davantage étayée dans le chapitre 5.

Nous discutons dans un premier temps des différents manières de définir la notion de chemin minimal dans la section 2.4.1. Puis nous montrons comment adapter les opérateurs d'influence pour l'ensemble de valeurs $\{+, -\}$ dans la section 2.4.2. Nous faisons de même pour l'ensemble de valeurs $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$ dans la section 2.4.3.

FIGURE 2.7 – Influences reliant un concept A à un concept F .

2.4.1 Chemins minimaux

Cette section discute de la possibilité d'adapter la définition de ce qu'est un chemin minimal en fonction de ce que l'utilisateur estime pertinent. On peut ainsi par exemple ignorer ou non les circuits ou encore limiter un chemin à un certain nombre d'influences.

Un chemin minimal peut être défini comme étant uniquement des influences directes. Une telle définition diminue l'intérêt du modèle des cartes cognitives puisque le calcul de l'influence propagée sur un chemin n'a plus d'intérêt, un chemin étant limité à une seule influence. Néanmoins, une telle définition permet d'indiquer qu'on ne considère que seules les informations fournies par le designer de la carte.

Définition 2.24 (Chemin minimal direct)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

P est minimal ssi $k = 1$.

On peut étendre cette définition en définissant un chemin minimal liant deux concepts comme étant un chemin quelconque liant ces deux concepts, mais limité à une certaine longueur. En effet, on peut considérer que la pertinence de l'influence sur un chemin diminue avec sa longueur. [Christiansen, 2011] limite ainsi les chemins à 3 influences.

Définition 2.25 (Chemin minimal limité)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k . Soit $K \in \mathbb{N}^*$ une limite de taille de chemin.

P est minimal ssi $k \leq K$.

Remarque. En posant $K = 1$, on retrouve la définition 2.24.

Exemple 2.21. La figure 2.7 représente une carte cognitive simplifiée dans laquelle de nombreuses influences permettent de lier le concept A au concept F .

Avec la définition classique de la notion de chemin minimal (définition 2.2), seuls 2 chemins sont minimaux : $A \rightarrow F$ et $A \rightarrow B \rightarrow C \rightarrow F$.

Avec la définition considérant uniquement les influences directes (définition 2.24), un seul chemin est minimal : $A \rightarrow F$.

Avec la définition limitant la taille des influences (définition 2.25), en posant $K = 3$, les chemins minimaux sont : $A \rightarrow F$, $A \rightarrow B \rightarrow C \rightarrow F$ et $A \rightarrow F \rightarrow A \rightarrow F$. En posant $K = 5$, il y a 6 chemins minimaux supplémentaires parmi lesquels $A \rightarrow B \rightarrow B \rightarrow C \rightarrow F$, $A \rightarrow B \rightarrow C \rightarrow D \rightarrow F$ et $A \rightarrow F \rightarrow A \rightarrow F \rightarrow A \rightarrow F$.

On peut définir un chemin minimal liant deux concepts comme étant un chemin quelconque liant ces deux concepts, mais dont chaque influence du chemin est unique. Une telle définition permet de s'assurer chaque influence d'un chemin n'est prise en compte qu'une seule fois lors du calcul de l'influence propagée sur ce chemin. Elle permet en outre à un chemin de passer plusieurs fois par le même concept à condition qu'il n'utilise pas la même influence.

Définition 2.26 (Chemin minimal avec influence unique)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

P est minimal ssi $\forall i, j \in [0; k - 1], i \neq j \Rightarrow u_i \neq u_j \vee u_{i+1} \neq u_{j+1}$.

Proposition 2.2. L'ensemble des chemins minimaux obtenu par la définition 2.2 est inclus dans l'ensemble des chemins minimaux obtenu par la définition 2.26.

Démonstration. Soit P un chemin minimal selon la définition 2.2. On a alors $\forall i, j \in [0; k - 1], i \neq j \Rightarrow u_i \neq u_j \wedge u_{i+1} \neq u_{j+1}$ et donc trivialement $\forall i, j \in [0; k - 1], i \neq j \Rightarrow u_i \neq u_j \vee u_{i+1} \neq u_{j+1}$.

P est donc minimal selon la définition 2.26. \square

Exemple 2.22. Nous considérons à nouveau la carte cognitive de la figure 2.7. Avec la définition basée sur l'influence unique (définition 2.26), nous ne listons pas l'intégralité des chemins minimaux car il y en a 13. On notera simplement que $A \rightarrow B \rightarrow B \rightarrow C \rightarrow D \rightarrow C \rightarrow F$ et $A \rightarrow B \rightarrow C \rightarrow F \rightarrow A \rightarrow F$ sont des chemins minimaux. $A \rightarrow B \rightarrow C \rightarrow E \rightarrow B \rightarrow C \rightarrow F$ n'en est pas un puisqu'il contient deux fois l'influence (B, C) (entre autres).

Une définition autorisant davantage de chemins sans pour autant amener à avoir un ensemble infini de chemins minimaux est celle qui interdit à un chemin minimal de passer deux fois par le même circuit. Une telle définition permet de s'assurer chaque circuit d'un chemin n'est pris en compte qu'une seule fois lors du calcul de l'influence propagée sur ce chemin. Un *circuit* est un chemin qui commence et termine au même concept.

Définition 2.27 (Circuit)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

P possède un circuit p ssi $\exists i, j \in [0; k], i \neq j \wedge u_i = u_j$.

On définit alors le circuit p comme étant un chemin de longueur $j - i$ d'influences (u'_l, u'_{l+1}) telles que $\forall l \in [0; j - i], u'_l = u_{i+l}$.

On note $i_p = i$ l'indice du circuit p dans le chemin P .

On définit alors un chemin minimal comme étant un chemin qui ne contient pas deux fois le même circuit.

Définition 2.28 (Chemin minimal avec circuit unique)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

P est minimal ssi $\forall p, p'$ circuits de $P, p = p' \Rightarrow i_p = i_{p'}$.

Proposition 2.3. *L'ensemble des chemins minimaux obtenu par la définition 2.26 est inclus dans l'ensemble des chemins minimaux obtenu par la définition 2.28. De plus, ceux obtenus par la définition 2.2 y sont également inclus.*

Démonstration. Soit P un chemin minimal selon la définition 2.26. Soient p, p' deux circuits de P . On considère (u_{i_p}, u_{i_p+1}) la première influence de p et $(u'_{i_{p'}}, u'_{i_{p'}+1})$ la première influence de p' . Si $p = p'$, alors nécessairement $u_{i_p} = u'_{i_{p'}}$ et $u_{i_p+1} = u'_{i_{p'}+1}$. Selon la définition 2.26, on en déduit $i_p = i_{p'}$.

Par conséquent, P est minimal selon la définition 2.28. \square

Concernant la seconde partie de la proposition, il suffit d'appliquer la proposition 2.2 qui spécifie que l'ensemble des chemins minimaux selon la définition 2.2 est inclus dans celui de la définition 2.26. On en déduit alors par la première partie de la proposition qu'ils sont également inclus dans l'ensemble des chemins minimaux de la définition 2.28. \square

Exemple 2.23. *Nous considérons à nouveau la carte cognitive de la figure 2.7. Avec la définition basée sur le circuit unique (définition 2.28), il y a 585 chemins minimaux. $A \rightarrow B \rightarrow C \rightarrow E \rightarrow B \rightarrow C \rightarrow F$ est désormais un chemin minimal. $A \rightarrow B \rightarrow C \rightarrow F \rightarrow A \rightarrow B \rightarrow C \rightarrow F$ est également minimal puisque $A \rightarrow B \rightarrow C \rightarrow F$, bien qu'il apparaisse deux fois, n'est pas un circuit. $A \rightarrow F \rightarrow A \rightarrow F$ est aussi un chemin minimal mais $A \rightarrow F \rightarrow A \rightarrow F \rightarrow A \rightarrow F$ ne l'est pas puisque le circuit $A \rightarrow F \rightarrow A$ apparaît deux fois.*

Pour le calcul de l'influence propagée, certains chemins s'avèrent redondants. En effet, deux chemins passant exactement par les mêmes influences le même nombre de fois mais pas dans le même ordre sont considérés comme différents et peuvent donc apparaître tout deux dans l'ensemble des chemins minimaux. De tels chemins sont donc identiques à l'ordre des influences près. Si cet ordre n'a pas d'importance dans le calcul de l'influence propagée sur un chemin, ils auront donc la même influence propagée. Afin d'obtenir une valeur d'influence propagée d'un concept sur un autre plus pertinente, il peut être judicieux de supprimer de tels doublons.

Cependant, cette suppression peut amener à une valeur d'influence propagée d'un concept sur un autre différente de celle calculée sans cette suppression. Par exemple, avec une moyenne (définition 2.6), on divise par le nombre de chemins. Or, si on supprime des chemins, ce nombre sera forcément différent. La sémantique de l'opérateur peut donc être modifiée.

Si deux chemins sont identiques à l'ordre des influences près, un de ces chemins doit être supprimé, au choix. Un ensemble de chemins minimaux ignorant l'ordre des influences n'est donc pas unique.

Définition 2.29 (Chemins minimaux ignorant l'ordre des influences)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit \mathcal{P}_{c_1, c_2} un ensemble de chemins minimaux. On note $\text{influences}(P)$ le multiensemble contenant les influences de P . Un ensemble de chemins \mathcal{P}'_{c_1, c_2} est un ensemble de chemins minimaux ignorant l'ordre des influences de \mathcal{P}_{c_1, c_2} si :

- $\forall P \in \mathcal{P}_{c_1, c_2}, \exists P' \in \mathcal{P}'_{c_1, c_2}, \text{influences}(P) = \text{influences}(P') ;$

- $\forall P', P'' \in \mathcal{P}'_{c_1, c_2}, P' \neq P'' \Rightarrow \text{influences}(P') \neq \text{influences}(P'')$.

Proposition 2.4. *Avec la définition classique des chemins minimaux (définition 2.2), il n'y a jamais de chemins identiques à l'ordre des influences près.*

Démonstration. Soit \mathcal{P}_{c_1, c_2} un ensemble de chemins minimaux entre deux concepts selon la définition 2.2.

Pour chaque $P \in \mathcal{P}_{c_1, c_2}$, il existe au moins un chemin P' tel que $\text{influences}(P) = \text{influences}(P')$, qui est P lui-même. La première condition est donc vérifiée.

Soient $P, P' \in \mathcal{P}_{c_1, c_2}$ tels que $\text{influences}(P) = \text{influences}(P')$. D'après la proposition 2.2, on sait qu'il n'y a pas deux fois la même influence dans P ou dans P' . Par conséquent, on en déduit que P et P' sont d'une même longueur $k = |\text{influences}(P)|$. On cherche à montrer par récurrence que pour tout concept u_i de P et pour tout concept u'_i de P' , on a $u_i = u'_i$ avec $i \in [0; k]$.

On considère u_0 et u'_0 . Puisque P et P' sont des chemins de c_1 vers c_2 , par définition, $u_0 = u'_0 = c_1$.

On suppose qu'on a $u_i = u'_i$ avec $i \in [0; k - 1]$. On cherche à montrer $u_i = u'_i \Rightarrow u_{i+1} = u'_{i+1}$. On considère les influences (u_i, u_{i+1}) et (u'_i, u'_{i+1}) . (u'_i, u'_{i+1}) est également une influence P puisque $\text{influences}(P) = \text{influences}(P')$. On ne connaît cependant pas son emplacement dans P . On la note (u_j, u_{j+1}) telle que $u_j = u'_i$ et $u_{j+1} = u'_{i+1}$. On a donc $u_i = u_j$. D'après la définition 2.2, $u_i = u_j \vee u_{i+1} = u_{j+1} \Rightarrow i = j$. On en déduit donc que $i = j$. Puisque $i = j$, $u_{i+1} = u_{j+1}$ et par conséquent, $u_{i+1} = u'_{i+1}$.

On a $u_0 = u'_0$ et $\forall i \in [0; k - 1], u_i = u'_i \Rightarrow u_{i+1} = u'_{i+1}$. Par récurrence, on a donc $\forall i \in [0; k], u_i = u'_i$. Les deux séquences sont donc équivalentes et par conséquent, $P = P'$. La seconde condition est donc vérifiée. \square

Exemple 2.24. *Nous considérons à nouveau la carte cognitive de la figure 2.7. En utilisant la définition 2.29 pour supprimer les chemins équivalents, le nombre de chemins minimaux avec un circuit unique tombe à 69. Ainsi, quand $A \rightarrow B \rightarrow B \rightarrow C \rightarrow E \rightarrow B \rightarrow C \rightarrow F$ et $A \rightarrow B \rightarrow C \rightarrow E \rightarrow B \rightarrow B \rightarrow C \rightarrow F$ étaient considérés comme deux chemins différents, maintenant, seul l'un des deux sera conservé.*

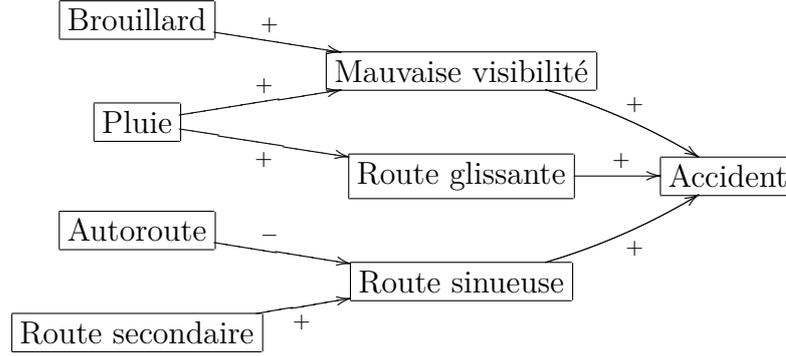
2.4.2 Définitions des opérateurs d'influence pour $\{+, -\}$

Pour appliquer nos opérateurs pour l'ensemble de valeurs $\{+, -\}$, nous utilisons les opérateurs définis par R. Axelrod [Axelrod, 1976]. Une influence d'un concept sur un autre portant la valeur $+$ signifie simplement que le premier concept influence positivement le second. Inversement, la valeur $-$ signifie qu'il l'influence négativement.

Exemple 2.25. *La carte cognitive CM2 de la figure 2.8 est identique à la carte cognitive CM1 (exemple 2.1) mais définie sur l'ensemble de valeurs $\{+, -\}$.*

Influence propagée sur un chemin

L'influence propagée sur un chemin est définie selon un opérateur spécial, \wedge . Cette définition est équivalente à l'application de la définition 2.5 pour une carte cognitive

FIGURE 2.8 – La carte cognitive $CM2$ définie sur $\{+, -\}$.

utilisant uniquement les valeurs -1 et 1 .

Définition 2.30 (Influence propagée sur un chemin pour $\{+, -\}$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $I = \{+, -\}$. Soit P un chemin d'influence liant deux concepts de CM .

L'influence propagée sur P est :

$$\mathcal{IP}(P) = \bigwedge_{i=0}^{k-1} \text{label}((u_i, u_{i+1}))$$

avec \wedge un opérateur commutatif et associatif défini sur $\{+, -\}$ tel que :

\wedge	+	-
+	+	-
-	-	+

Influence propagée d'un concept sur un autre

L'influence propagée d'un concept sur un autre introduit deux nouvelles valeurs. On rappelle la signification de ces valeurs, déjà expliquée dans la section 1.4.2 :

- la valeur nulle 0 indique qu'il n'y a aucun chemin d'influence entre les deux concepts ;
- la valeur ambiguë $?$ indique que l'influence entre les deux concepts est soit positive, soit négative.

Définition 2.31 (Influence propagée d'un concept sur un autre pour $\{+, -\}$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $I = \{+, -\}$. Soient $c_1, c_2 \in C$ deux concepts de CM .

L'influence propagée de c_1 sur c_2 est une fonction $\mathcal{I}: C \times C \rightarrow \{+, 0, -, ?\}$ telle que :

$$\mathcal{I}(c_1, c_2) = \begin{cases} 0 & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \bigvee_{P \in \mathcal{P}_{c_1, c_2}} \mathcal{IP}(P) & \text{sinon} \end{cases}$$

avec \vee un opérateur commutatif et associatif défini sur $\{+, 0, -, ?\}$ tel que :

\vee	+	-	?
+	+	?	?
-	?	-	?
?	?	?	?

La valeur 0 associée à $\{+, -\}$ n'a pas la même sémantique que celle de l'intervalle $[-1; 1]$ mais a celle décrite dans la section 2.1.3. Elle signifie qu'il n'y a pas d'influence globale entre deux concepts.

Exemple 2.26. *On reprend les concepts Pluie et Accident de l'exemple 2.2. L'influence propagée de Pluie sur Accident dans CM2 (exemple 2.25) est :*

$$\mathcal{I}(\text{Pluie}, \text{Accident}) = \mathcal{IP}(p_1) \vee \mathcal{IP}(p_2) = (+ \wedge +) \vee (+ \wedge +) = + \vee + = +$$

Influence taxonomique

L'influence taxonomique introduit elle aussi deux nouvelles valeurs d'influence entre deux concepts [Chauvin *et al.*, 2008] :

- la valeur \oplus indique que l'influence est positive ou nulle ;
- la valeur \ominus indique que l'influence est négative ou nulle.

Définition 2.32 (Influence taxonomique d'un concept sur un autre pour $\{+, -\}$)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T définie sur $\{+, -\}$. Soient $c_1, c_2 \in C$ deux concepts ordonnés par T .

L'influence taxonomique de c_1 sur c_2 est une fonction $\mathcal{I}_T: C \times C \rightarrow \{+, \oplus, 0, \ominus, -, ?\}$ telle que :

$$\mathcal{I}_T(c_1, c_2) = \bigodot_{\substack{c'_1 \in \text{elemPour}_T(c_1) \\ c'_2 \in \text{elemPour}_T(c_2)}} \mathcal{I}(c'_1, c'_2)$$

avec \odot un opérateur commutatif et associatif défini sur $\{+, \oplus, 0, \ominus, -, ?\}$ tel que :

\odot	+	\oplus	0	\ominus	-	?
+	+	\oplus	\oplus	?	?	?
\oplus	\oplus	\oplus	\oplus	?	?	?
0	\oplus	\oplus	0	\ominus	\ominus	?
\ominus	?	?	\ominus	\ominus	\ominus	?
-	?	?	\ominus	\ominus	-	?
?	?	?	?	?	?	?

Exemple 2.27. *On définit la carte taxonomique TM2 comme étant l'association de la carte CM2 (exemple 2.25) et la taxonomie T1 (exemple 2.6). On considère à nouveau les concepts Mauvais temps et Mauvaise condition de trafic, comme pour*

l'exemple 2.11. L'influence taxonomique de Mauvais temps sur Mauvaise condition de trafic est :

$$\mathcal{I}_T(\text{Mauvais temps}, \text{Mauvaise condition de trafic}) = \odot \left\{ \begin{array}{l} \mathcal{I}(\text{Brouillard}, \text{Mauvaise visibilité}) \\ \mathcal{I}(\text{Brouillard}, \text{Route glissante}) \\ \mathcal{I}(\text{Brouillard}, \text{Route sinueuse}) \\ \mathcal{I}(\text{Pluie}, \text{Mauvaise visibilité}) \\ \mathcal{I}(\text{Pluie}, \text{Route glissante}) \\ \mathcal{I}(\text{Pluie}, \text{Route sinueuse}) \end{array} \right\} = \odot \left\{ \begin{array}{l} + \\ 0 \\ 0 \\ + \\ + \\ 0 \end{array} \right\} = \oplus$$

2.4.3 Définitions des opérateurs d'influence pour $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$

L'ensemble de valeurs $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$ est un ensemble de valeurs totalement ordonné. La valeur *nul* est sémantiquement équivalente à la valeur 0 dans l'ensemble de valeurs $[-1; 1]$. Les autres valeurs représentent différents degrés de la force de l'influence. La valeur *fort* est la valeur maximale attribuable à une influence.

$\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$ étant un ensemble ordonné, il est adapté pour les cartes cognitives floues [Kosko, 1986]. Celles-ci seront présentées plus en détail dans le chapitre 5, notamment la section 5.1.2.

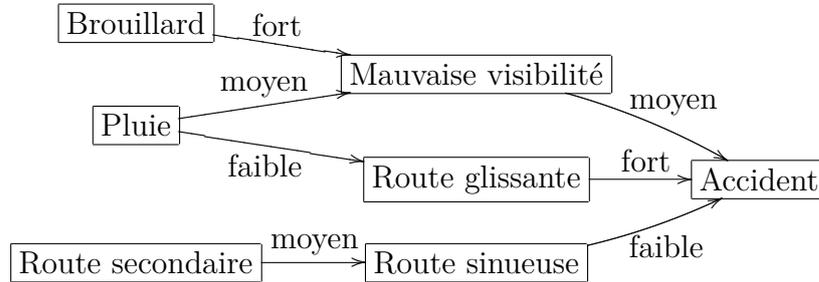


FIGURE 2.9 – La carte cognitive *CM3* définie sur $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$.

Exemple 2.28. La carte cognitive *CM3* de la figure 2.9 est identique à la carte cognitive *CM1* (exemple 2.1) mais définie sur l'ensemble de valeurs $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$. Cet ensemble ne contenant que des valeurs positives, le concept Autoroute a été retiré puisqu'il amenait une influence négative sur Route sinueuse.

Influence propagée sur un chemin

L'influence propagée sur un chemin est définie comme étant la valeur minimale portée par les influences de ce chemin.

Définition 2.33 (Influence propagée sur un chemin pour $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $I = \{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$. Soit P un chemin d'influence liant deux concepts

de C .

L'influence propagée sur P est :

$$\mathcal{IP}(P) = \min\left(\bigcup_{i=0}^{k-1} \{\text{label}((u_i, u_{i+1}))\}\right)$$

Influence propagée d'un concept sur un autre

L'influence propagée d'un concept sur un autre est définie comme étant la valeur maximale des influences propagées sur les différents chemins permettant de lier ce concept à l'autre.

Définition 2.34 (Influence propagée d'un concept sur un autre pour $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $I = \{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$. Soient $c_1, c_2 \in C$ deux concepts de CM .

L'influence propagée de c_1 sur c_2 est une fonction $\mathcal{I}: C \times C \rightarrow \{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$ telle que :

$$\mathcal{I}(c_1, c_2) = \begin{cases} \text{nul} & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \max\left(\bigcup_{P \in \mathcal{P}_{c_1, c_2}} \{\mathcal{IP}(P)\}\right) & \text{sinon} \end{cases}$$

Exemple 2.29. On reprend à nouveau les concepts *Pluie* et *Accident* de l'exemple 2.2. L'influence propagée de *Pluie* sur *Accident* dans $CM3$ (exemple 2.28) est :

$$\begin{aligned} \mathcal{I}(\text{Pluie}, \text{Accident}) &= \max(\{\mathcal{IP}(p_1), \mathcal{IP}(p_2)\}) \\ &= \max\left(\left\{\min_T(\{\text{moyen}, \text{moyen}\}), \min(\{\text{faible}, \text{fort}\})\right\}\right) \\ &= \max(\{\text{moyen}, \text{faible}\}) \\ &= \text{moyen} \end{aligned}$$

Influence taxonomique

Pour définir l'influence taxonomique pour $\{\text{nul} < \text{faible} < \text{moyen} < \text{fort}\}$, il n'y a pas réellement d'opérateur plus adapté qu'un autre. On peut donc utiliser un max, comme pour l'influence propagée d'un concept sur un autre (définition 2.34) ou un intervalle, comme pour l'influence taxonomique pour $[-1; 1]$ (définition 2.13) [Genest et Loiseau, 2007].

Conclusion

Les cartes cognitives sont un modèle efficace pour représenter visuellement des influences et pour raisonner. Le calcul de l'influence propagée permet de déduire des connaissances pertinentes pour un utilisateur.

Néanmoins, si obtenir une valeur unique d'influence est une information simple et concise, elle peut s'avérer insuffisante si on cherche à mettre en évidence les chemins d'influence portant une valeur différente de celles des autres. Des problèmes

de qualité de la carte peuvent aussi exister. Pour faire face à ce problème, nous proposons différents critères de validité d'une carte au chapitre 3.

Les vues partagées sont un système efficace pour manipuler une carte cognitive entre plusieurs utilisateurs. Cependant, rien n'est dit sur la construction de la carte elle-même. Cette construction peut en effet être le résultat d'un travail de groupe entre plusieurs designers de cartes. L'idée de synthèse qui est introduite au chapitre 4 propose des solutions pour construire une carte issue de différents designers en une carte unique. Elle utilise pour cela un ordre de préférence défini sur ces designers.

Enfin, les opérateurs de calcul de l'influence propagée d'un concept sur un autre ont en sens un soi. Néanmoins, la sémantique des ces valeurs n'est pas abordée. On ne peut donc savoir si les valeurs d'influence calculées ont effectivement un sens. Dans le chapitre 5, nous discutons de la sémantique qui peut être associée aux valeurs d'une carte cognitive et de la manière d'adapter les opérateurs de calcul d'influence propagée en conséquence.



3

Validation d'une carte cognitive

Introduction

Nous avons expliqué dans le chapitre précédent que l'influence propagée d'un concept sur un autre est calculée en agrégeant les valeurs des influences propagées sur les différents chemins permettant de lier ce concept à l'autre. Avec l'ensemble de valeurs $\{+, -\}$, obtenir une valeur d'ambiguïté par ce calcul est très fréquent : si l'intégralité des chemins liant ces concepts portent une valeur positive excepté un qui porte une valeur négative, cela suffit à inférer une valeur d'influence ambiguë (?) entre les deux concepts. Utiliser des valeurs numériques, telles que celles de l'intervalle $[-1; 1]$ avec des opérateurs adaptés, permet de faire disparaître cette notion d'ambiguïté en l'intégrant par exemple dans une moyenne. Cependant, l'information indiquant qu'un chemin apportait une valeur différente de celles des autres chemins est perdue. Il serait intéressant pour le designer de la carte cognitive de pouvoir détecter de telles ambiguïtés afin de les supprimer pour assurer la qualité de la carte ou de les mettre en avant pour provoquer une discussion entre plusieurs intervenants.

Ce chapitre propose une méthode pour *valider* une carte cognitive et ainsi assurer sa qualité. La validation est basée sur un *critère de qualité* qui est utilisé pour contrôler une base de connaissances. Ces critères vont nous permettre de relever les ambiguïtés discutées précédemment. La validation est souvent partagée en deux catégories : la vérification et le test [Ayel et Laurent, 1991]. La *vérification* est basée sur un critère de qualité qui ne nécessite pas d'information externe, autrement dit ce type de critère dépend uniquement de la cohérence interne de la base. Le *test* est basé sur un critère de qualité qui nécessite de l'information externe, souvent appelée *contrainte*, contrainte d'intégrité, jeu de test ou, plus généralement, *spécification*.

Pour vérifier une carte cognitive, nous introduisons la notion de carte non-ambiguë. Une carte cognitive est *non-ambiguë* si pour toute paire de concepts de

cette carte, l'influence propagée sur chacun des différents chemins liant ces deux concepts et l'influence propagée entre ces deux concepts sont égales.

Pour tester une carte cognitive, nous introduisons la notion de carte cognitive cohérente. Pour cela, une *spécification* est définie. Une spécification est un ensemble de contraintes. Une *contrainte* est un triplet composé d'un concept-source, d'un concept-destination ainsi que d'une valeur. Les contraintes permettent de spécifier une valeur d'influence attendue entre deux concepts. Cependant, définir des contraintes entre des concepts est coûteux et peut sembler redondant. C'est pourquoi les concepts de ces contraintes sont ordonnés par une taxonomie. Une contrainte est définie sur deux concepts de la taxonomie. Les feuilles de la taxonomie sont les concepts de la carte cognitive à valider. Grâce à cette spécification, nous définissons le critère de test de cohérence. Une carte cognitive est *cohérente* avec une contrainte si la valeur de l'influence taxonomique entre les deux concepts de la contrainte est égale à la valeur de cette contrainte. Par extrapolation, cette même carte est cohérente avec une spécification si elle est cohérente avec l'ensemble des contraintes de cette spécification.

Pour se représenter intuitivement ce que représentent les différents critères définis, nous nous restreignons dans un premier temps aux cartes cognitives définies sur l'ensemble de valeurs $\{+, -\}$. Nous montrons ensuite comment adapter les critères à d'autres ensembles de valeurs.

Nous proposons également une méthode pour aider le designer à corriger sa carte. On lui présente, pour chaque critère de qualité non validé, un ensemble d'influences qu'il devra modifier afin de valider le dit critère. Pour cela, nous nous basons sur les notions de conflit et de diagnostic [Reiter, 1987] reprises dans des travaux de validation de systèmes à base de règles [Bouali *et al.*, 1997]. Pour des raisons de simplicité, cette correction est limitée à l'ensemble de valeurs $\{+, -\}$.

Dans la section 3.1, nous définissons formellement deux critères de vérification, dont le critère de non-ambiguïté. Dans la section 3.2, nous introduisons le critère de test de cohérence ainsi que les notions de contrainte et de spécification. Dans la section 3.3, nous expliquons comment paramétrer les différents critères définis dans les sections précédentes pour d'autres ensembles de valeurs, notamment $[-1; 1]$. Enfin, dans la section 3.4, nous définissons des ensembles de conflits pour les critères de vérification.

3.1 Vérification

La vérification regroupe les critères de qualité qui valident un modèle sans apport d'information externe. Ils s'appliquent sans problème au modèle classique des cartes cognitives.

Nous définissons dans un premier temps le critère de propreté, qui peut être vu comme un sous-cas du critère de non-ambiguïté, dans la section 3.1.1. Nous définissons ensuite le critère de non-ambiguïté dans la section 3.1.2.

3.1.1 Propreté

Le premier critère de vérification que nous définissons est le critère de propreté. Avec ce critère, on s'intéresse uniquement aux concepts qui sont liés directement par une influence. Il permet de s'assurer que l'influence propagée d'un concept sur un autre est en accord avec l'influence directe de ce concept vers l'autre. L'idée derrière ce critère est de considérer qu'une influence directe représente l'influence globale et il est donc normal que l'influence propagée soit en accord avec l'influence directe.

Une carte cognitive est *propre* si pour chaque couple de concepts de la carte liés par une influence, l'influence propagée du premier concept vers le second est égale à la valeur de l'influence directe du premier vers le second.

Avant tout chose, nous définissons la propreté pour un couple de concepts.

Définition 3.1 (Couple de concepts propre)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur $\{+, -\}$. Soit $c_1, c_2 \in C$ deux concepts liés par une influence directe dans CM , i.e. $(c_1, c_2) \in A$.

Le couple (c_1, c_2) est propre ssi $\mathcal{I}(c_1, c_2) = \text{label}((c_1, c_2))$.

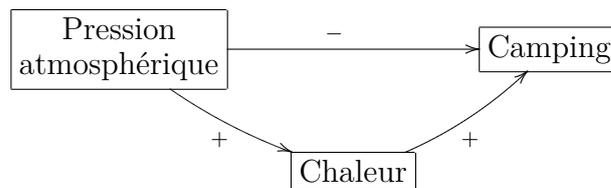


FIGURE 3.1 – Couple de concepts non-propre.

Exemple 3.1. Sur la figure 3.1, on peut lire qu'une pression atmosphérique élevée génère de la chaleur, ce qui attire les campeurs. Mais d'un autre côté, les campeurs n'aiment pas une pression atmosphérique trop lourde.

Le couple de concepts (Pression atmosphérique, Camping) n'est pas propre. En effet, on a :

$$\begin{aligned} \mathcal{I}(\text{Pression atmosphérique}, \text{Camping}) &= ? \\ \text{label}((\text{Pression atmosphérique}, \text{Camping})) &= - \end{aligned}$$

On a donc :

$$\mathcal{I}(\text{Pression atmosphérique}, \text{Camping}) \neq \text{label}((\text{Pression atmosphérique}, \text{Camping}))$$

Une carte cognitive *propre* est une carte cognitive dont tous les couples de concepts liés directement par une influence sont propres.

Définition 3.2 (Carte cognitive propre)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur $\{+, -\}$.

CM est propre ssi $\forall (c_1, c_2) \in A, (c_1, c_2)$ est propre.

Exemple 3.2. La carte cognitive $CM4$ (figure 3.2) n'est pas propre. En effet, comme déjà expliqué dans l'exemple 3.1, le couple de concepts (Pression atmosphérique, Camping) n'est pas propre.

En revanche, on peut noter que la paire de concepts (Pluie, Incendie) est propre.

On note par ailleurs que la notion de propreté ne s'applique pas au couple (Orage, Incendie) puisque ces deux concepts ne sont pas liés par une influence directe.

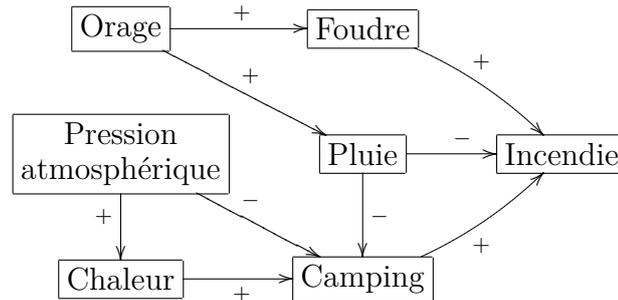


FIGURE 3.2 – La carte cognitive $CM4$, non-propre et ambiguë.

3.1.2 Non-ambiguïté

La propreté se limite aux concepts liés par une influence directe. La non-ambiguïté s'appuie sur cette notion en s'intéressant à tous les couples de concepts de la carte. En effet, on ne s'intéresse plus uniquement aux concepts liés directement par une influence mais à n'importe quel couple de concepts de la carte. Ce critère permet de s'assurer que l'influence propagée sur chacun des chemins liant le premier concept au second est en accord avec l'influence propagée du premier sur le second. L'idée derrière ce critère est de considérer que toutes les manières qu'a un concept d'en influencer un autre doivent être en accord avec l'influence globale de ce concept sur l'autre. Une carte cognitive est *non-ambiguë* si pour chaque couple de concepts de la carte, l'influence propagée sur chaque chemin permettant de lier le premier concept au second est égale à l'influence propagée du premier sur le second.

Avant tout chose, nous définissons la non-ambiguïté pour un couple de concepts.

Définition 3.3 (C couple de concepts non-ambigu)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur $\{+, -\}$. Soient $c_1, c_2 \in C$ deux concepts de CM .

Le couple (c_1, c_2) est non-ambigu ssi $\forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) = \mathcal{I}(c_1, c_2)$.

Exemple 3.3. Sur la figure 3.1, on peut lire d'un côté qu'un orage déclenche des incendies en provoquant des éclairs et de l'autre qu'un orage éteint les incendies en provoquant de la pluie.

Le couple de concepts (Orage, Incendie) est donc ambigu. En effet, on a :

$$\begin{aligned} \mathcal{IP}(\text{Orage} \rightarrow \text{Foudre} \rightarrow \text{Incendie}) &= + \\ \mathcal{IP}(\text{Orage} \rightarrow \text{Pluie} \rightarrow \text{Incendie}) &= - \\ \mathcal{I}(\text{Orage}, \text{Incendie}) &= ? \end{aligned}$$

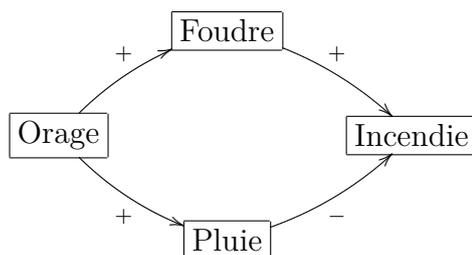


FIGURE 3.3 – Couple de concepts ambigu.

On a donc :

$$\begin{aligned} \mathcal{IP}(\text{Orage} \rightarrow \text{Foudre} \rightarrow \text{Incendie}) &\neq \mathcal{I}(\text{Orage}, \text{Incendie}) \\ \mathcal{IP}(\text{Orage} \rightarrow \text{Pluie} \rightarrow \text{Incendie}) &\neq \mathcal{I}(\text{Orage}, \text{Incendie}) \end{aligned}$$

Par ailleurs, comme expliqué dans l'exemple 3.2, on ne peut pas dire que ce couple de concept est propre puisque ceux-ci ne sont pas liés directement par une influence.

En revanche, le couple (Pression atmosphérique, Camping), en plus de ne pas être propre, est ambigu. En effet, on a :

$$\begin{aligned} \mathcal{IP}(\text{Pression atmosphérique} \rightarrow \text{Camping}) &= - \\ \mathcal{IP}(\text{Pression atmosphérique} \rightarrow \text{Chaleur} \rightarrow \text{Camping}) &= + \\ \mathcal{I}(\text{Pression atmosphérique}, \text{Camping}) &= ? \end{aligned}$$

Les influences propagées sur chacun des deux chemins sont donc différentes de l'influence propagée de Pression atmosphérique sur Camping.

La proposition 3.1 nous assure qu'un couple non-ambigu de concepts liés directement par une influence est également propre.

Proposition 3.1. Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $\{+, -\}$. Soit $(c_1, c_2) \in A$ un couple de concepts de CM liés par une influence.

$$(c_1, c_2) \text{ est non-ambigu} \implies (c_1, c_2) \text{ est propre}$$

Démonstration. Si (c_1, c_2) est non-ambigu, d'après la définition 3.3, on a $\forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) = \mathcal{I}(c_1, c_2)$. Puisque $(c_1, c_2) \in A$, on a $c_1 \rightarrow c_2 \in \mathcal{P}_{c_1, c_2}$. On a donc $\mathcal{IP}(c_1 \rightarrow c_2) = \mathcal{I}(c_1, c_2)$. Étant donné la définition de l'influence propagée sur un chemin (définition 2.30), on a $\mathcal{IP}(c_1 \rightarrow c_2) = \text{label}((c_1, c_2))$. On a donc $\text{label}((c_1, c_2)) = \mathcal{I}(c_1, c_2)$.

Donc, d'après la définition 3.1, (c_1, c_2) est propre. \square

Une carte cognitive *non-ambiguë* est une carte cognitive dont tous les couples de concepts sont non-ambigus.

Définition 3.4 (Carte cognitive non-ambiguë)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur $\{+, -\}$. CM est non-ambiguë ssi $\forall c_1, c_2 \in C, (c_1, c_2)$ est non-ambigu.

Exemple 3.4. *La carte cognitive CM4 (figure 3.2) est ambiguë. En effet, comme déjà expliqué dans l'exemple 3.3, le couple de concepts (Orage, Incendie) est ambigu.*

La proposition 3.2 nous assure qu'une carte cognitive non-ambiguë est également propre.

Proposition 3.2. *Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur l'ensemble de valeurs $\{+, -\}$.*

$$CM \text{ est non-ambiguë} \implies CM \text{ est propre}$$

Démonstration. Puisque CM est non-ambiguë, d'après la définition 3.4, $\forall c_1, c_2 \in C$, (c_1, c_2) est non-ambigu. Puisque $A \subseteq C \times C$, $(c_1, c_2) \in A \implies c_1, c_2 \in C$. On a donc $\forall (c_1, c_2) \in A$, (c_1, c_2) est non-ambigu. D'après la proposition 3.1, on a $\forall (c_1, c_2) \in A$, (c_1, c_2) est propre.

Donc, d'après la définition 3.3, (c_1, c_2) est non-ambigu. □

3.2 Test

Le test regroupe les critères de qualité qui valident un modèle en comparant les informations qu'il contient à de l'information externe. Nous définissons deux critères : le premier est le critère de cohérence qui va permettre de tester l'influence taxonomique entre deux concepts. Le second critère est le critère de compatibilité. Ce critère étend le critère de cohérence en lui fournissant davantage de souplesse.

Dans la section 3.2.1, nous définissons la notion de spécification qui contient un ensemble de contraintes qui seront comparées aux influences de la carte cognitive. Cette spécification est définie sur une taxonomie. Puisque cette raison, nous utilisons le modèle des cartes cognitives taxonomiques pour définir nos critères de test. Une fois cette spécification définie, nous définissons le critère de cohérence dans la section 3.2.2. Nous définissons ensuite le critère de compatibilité dans la section 3.2.3.

3.2.1 Spécification

Les contraintes spécifient des influences attendues entre deux concepts de la taxonomie. Elles sont fournies par le designer de la carte cognitive dans le but de valider celle-ci. Elles ne permettent pas d'inférer de nouvelles connaissances mais servent uniquement à tester la carte.

Une *contrainte* est un triplet fait d'un concept-source, d'un concept-destination et d'une valeur. Les concepts de la contrainte doivent être ordonnés par une taxonomie. Les contraintes s'appliquent donc sur une carte cognitive définie sur cette taxonomie et dont les concepts sont les feuilles de la taxonomie. Pour définir nos critères, nous utilisons donc le modèle des cartes cognitives taxonomiques défini au chapitre 2. L'utilité de la contrainte est de comparer l'influence taxonomique entre les concepts de cette contrainte à la valeur de cette même contrainte. La valeur de la contrainte doit donc prendre les mêmes valeurs que l'influence taxonomique.

Elle doit par conséquent appartenir au codomaine cette influence taxonomique¹. Dans le cas de l'ensemble $\{+, -\}$, ce codomaine est, d'après la définition 2.32, $\{+, \oplus, 0, \ominus, -, ?\}$.

Définition 3.5 (Contrainte)

Soit $T = (C, \leq)$ une taxonomie. Soient $c_1, c_2 \in C$ deux concepts ordonnés par T et $\alpha \in \{+, \oplus, 0, \ominus, -, ?\}$ une valeur.

$s = \langle c_1, c_2, \alpha \rangle$ est une contrainte définie sur T et $\{+, \oplus, 0, \ominus, -, ?\}$.

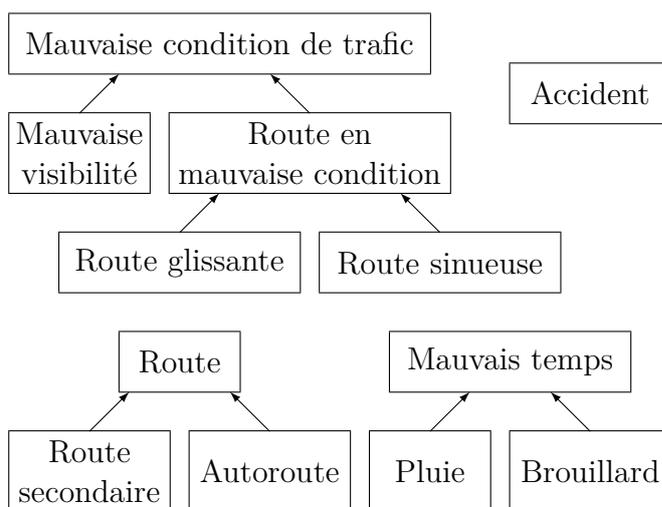


FIGURE 3.4 – Rappel de la taxonomie $T1$ (exemple 2.6).

Exemple 3.5. On reprend la taxonomie $T1$ (figure 3.4). Les 3 contraintes suivantes :

$$s_1 = \langle \text{Mauvaise condition de trafic}, \text{Accident}, + \rangle$$

$$s_2 = \langle \text{Mauvais temps}, \text{Mauvaise condition de trafic}, + \rangle$$

$$s_3 = \langle \text{Route}, \text{Route sinueuse}, - \rangle$$

sont définies sur $T1$ et l'ensemble de valeurs $\{+, \oplus, 0, \ominus, -, ?\}$.

Une *spécification* est un ensemble de contraintes. Une spécification est elle aussi définie sur une taxonomie et un ensemble de valeurs. Les contraintes d'une spécification doivent toutes être définies sur la même taxonomie et le même ensemble de valeurs que la spécification.

Définition 3.6 (Spécification)

Soit $T = (C, \leq)$ une taxonomie. Soit I un ensemble de valeurs. Soit S un ensemble de contraintes.

S est une spécification définie sur T et I ssi $\forall s \in S, s$ est définie sur T et I .

Exemple 3.6. On reprend les contraintes de l'exemple 3.5. Les spécifications $S_1 = \{s_1\}$, $S_2 = \{s_1, s_2\}$ et $S_3 = \{s_1, s_2, s_3\}$ sont des spécifications définies sur $T1$ et l'ensemble de valeurs $\{+, \oplus, 0, \ominus, -, ?\}$.

¹On rappelle que le codomaine d'une fonction, aussi appelé ensemble d'arrivée, est l'ensemble des valeurs que peut renvoyer cette fonction.

3.2.2 Cohérence

Maintenant que nous avons défini l'information externe au modèle, nous pouvons définir formellement nos critères de test qui comparent l'information contenue dans la carte à l'information externe. Nous définissons dans un premier temps formellement le critère de cohérence. Ce critère va simplement s'assurer que la valeur de l'influence taxonomique entre les concepts de la contrainte dans la carte est la même que celle spécifiée dans cette même contrainte.

Une carte cognitive taxonomique est *cohérente avec une contrainte* si l'influence taxonomique du premier concept de la contrainte sur le second dans cette carte est égale à la valeur de cette même contrainte.

Définition 3.7 (Cohérence avec une contrainte)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T et définie sur $\{+, -\}$. Soit $s = \langle c_1, c_2, \alpha \rangle$ une contrainte définie sur T et $\{+, \oplus, 0, \ominus, -, ?\}$.

TM est cohérente avec s ssi $\mathcal{I}_T(c_1, c_2) = \alpha$.

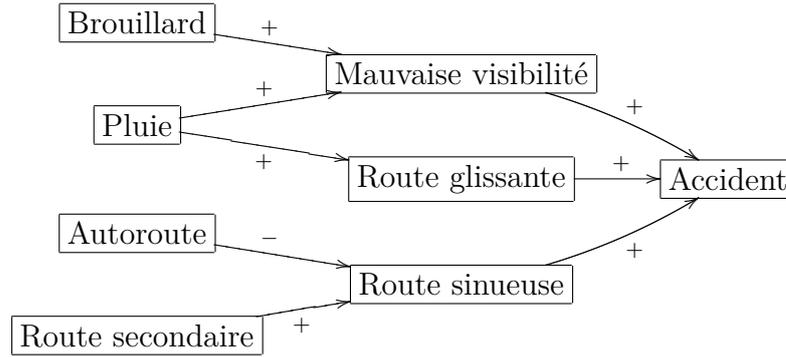


FIGURE 3.5 – Rappel de la carte cognitive CM2 (exemple 2.25).

Exemple 3.7. On reprend les contraintes de l'exemple 3.5. On étudie la carte cognitive taxonomique $TM2$ construite à partir de la taxonomie $T1$ et la carte cognitive $CM2$ (figure 3.5). $TM2$ est cohérente avec $s_1 = \langle \text{Mauvaise condition de trafic}, \text{Accident}, + \rangle$. En effet :

$$\mathcal{I}_T(\text{Mauvaise condition de trafic}, \text{Accident}) = +$$

+ est bien la valeur de la contrainte s_1 .

En revanche, $TM2$ n'est pas cohérente avec $s_2 = \langle \text{Mauvais temps}, \text{Mauvaise condition de trafic}, + \rangle$ puisque :

$$\mathcal{I}_T(\text{Mauvais temps}, \text{Mauvaise condition de trafic}) = \oplus \neq +$$

$TM2$ n'est pas non plus cohérente avec $s_3 = \langle \text{Route}, \text{Route sinueuse}, - \rangle$ puisque :

$$\mathcal{I}_T(\text{Route}, \text{Route sinueuse}) = ? \neq -$$

Une spécification est un ensemble de contraintes. Une carte cognitive taxonomique est *cohérente avec une spécification* si elle est cohérente avec l'ensemble des contraintes de cette spécification.

Définition 3.8 (Cohérence avec une spécification)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T et définie sur $\{+, -\}$. Soit S une spécification de contraintes définie sur T et $\{+, \oplus, 0, \ominus, -, ?\}$.

TM est cohérente avec S ssi $\forall s \in S, TM$ est cohérente avec s .

Exemple 3.8. On reprend les spécifications de l'exemple 3.6.

$TM2$ est cohérente avec $S_1 = \{s_1\}$ puisque $TM2$ est cohérente avec s_1 (exemple 3.7).

En revanche, $TM2$ n'est cohérente ni avec $S_2 = \{s_1, s_2\}$ ni avec $S_3 = \{s_1, s_2, s_3\}$ puisque $TM2$ n'est pas cohérente avec s_2 .

3.2.3 Compatibilité

Le critère de cohérence est basé sur une égalité entre une valeur calculée et une valeur attendue, spécifiée par une contrainte. Cette égalité peut parfois sembler trop stricte pour certaines valeurs, par exemple les valeurs $+$ et \oplus . Bien que ces deux valeurs soient sémantiquement proches, elles sont différentes. Par conséquent, si la valeur attendue est de $+$ et la valeur calculée \oplus , le critère de cohérence ne sera pas vérifié. Utiliser une autre relation pour souple que l'égalité permettrait d'ajouter de la souplesse au critère de cohérence en le validant même si deux valeurs ne sont pas parfaitement équivalentes.

Pour cela, nous définissons le critère de *compatibilité*. Ce critère étend le critère de cohérence en validant une contrainte même si l'influence taxonomique calculée n'est pas strictement égale à la valeur de la contrainte. Pour ce faire, on définit l'ensemble des *valeurs compatibles* avec une valeur donnée. L'idée de la compatibilité est de lister l'intégralité des valeurs de $\{+, \oplus, 0, \ominus, -, ?\}$ qui sont compatibles avec une valeur donnée. Chaque valeur doit être compatible avec elle-même.

Intuitivement, les valeurs positives ($+$, \oplus) ne sont pas compatibles avec les valeurs négatives ($-$, \ominus) et inversement : si le designer spécifie une contrainte de valeur $+$, il s'attend à avoir une influence positive. Une influence taxonomique de $-$ ou \ominus ne peuvent donc pas être compatibles avec cette valeur de contrainte. Une valeur \oplus est en revanche compatible avec $+$ puisqu'elle représente une influence positive ou nulle. Nous considérons en revanche que 0 n'est pas compatible avec $+$. En effet, la valeur $+$ sous-entend que l'influence doit être strictement positive. Or, la valeur 0 signifie qu'il n'y a pas d'influence entre deux concepts et n'est donc pas positive.

Spécifier une contrainte ayant pour valeur \oplus signifie formellement qu'on s'attend à une influence non-négative, c'est-à-dire positive ou nulle. Il semble donc tout à fait naturel que les valeurs $+$ et 0 soient compatibles avec \oplus .

Spécifier une contrainte ayant pour valeur 0 signifie qu'on s'attend à une influence nulle. On pourrait considérer que seules les influences taxonomiques de 0 sont compatibles avec cette contrainte. Cependant, ceci reviendrait à tester la

cohérence de la contrainte. Pour qu'une telle contrainte soit intéressante, nous considérons que le designer cherche à savoir si l'influence taxonomique est proche de 0. Pour cette raison, les valeurs \oplus et \ominus sont compatibles avec 0. Nous considérons également que ? est compatible avec 0. Cette valeur représente en effet une combinaison de valeurs positives et négatives et englobe par conséquent la valeur 0.

Spécifier une contrainte ayant pour valeur ? signifie qu'on s'attend à avoir une valeur ambiguë. Si on calcule une influence taxonomique de valeur autre que ?, il est évident qu'elle ne peut être compatible avec ?. Nous considérons donc que, puisque nous avons précisé plus haut que toute valeur devrait être compatible avec elle-même, seul ? est compatible avec ?. Ceci peut sembler étrange puisque cette valeur représente une valeur indéterminée. Néanmoins, ceci permet de s'assurer que les contraintes portant une valeur ? ont un intérêt. En effet, si aucune valeur n'était compatible avec la valeur ?, de telles contraintes ne seraient jamais validées et seraient donc inutiles.

Définition 3.9 (Valeurs compatibles)

Soient $\alpha_1, \alpha_2 \in \{+, \oplus, 0, \ominus, -, ?\}$ deux valeurs. α_1 représente la valeur de l'influence taxonomique. α_2 représente la valeur de la contrainte.

α_1 est compatible avec α_2 , noté $\alpha_1 \blacktriangleleft \alpha_2$ ssi :

		α_2					
		+	\oplus	0	\ominus	-	?
α_1	+	✓	✓	×	×	×	×
	\oplus	✓	✓	✓	×	×	×
	0	×	✓	✓	✓	×	×
	\ominus	×	×	✓	✓	✓	×
	-	×	×	×	✓	✓	×
	?	×	×	✓	×	×	✓

La relation \blacktriangleleft est réflexive.

On note bien qu'en considérant une valeur, davantage de valeurs sont compatibles qu'avec une égalité stricte, comme celle utilisée dans le critère de cohérence.

Nous définissons à présent le critère de compatibilité. Une carte cognitive est *compatible avec une contrainte* si l'influence taxonomique du premier concept de la contrainte sur le second dans cette carte est compatible avec la valeur de cette même contrainte. Pour définir simplement cette notion, nous nous inspirons la définition du critère de cohérence avec une contrainte (définition 3.7) dans laquelle nous remplaçons l'égalité entre l'influence taxonomique et la valeur de la contrainte par la notion de valeurs compatibles définie précédemment.

Définition 3.10 (Compatibilité avec une contrainte)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T et définie sur $\{+, \oplus, 0, \ominus, -, ?\}$. Soit $s = \langle c_1, c_2, \alpha \rangle$ une contrainte définie sur T et $\{+, \oplus, 0, \ominus, -, ?\}$.

TM est compatible avec s ssi $\mathcal{I}_T(c_1, c_2) \blacktriangleleft \alpha$.

Exemple 3.9. *On reprend les contraintes de l'exemple 3.7. Certaines des contraintes qui n'étaient pas cohérentes avec la carte cognitive sont à présent compatibles avec cette même carte.*

TM2 est cohérente et compatible avec la contrainte s_1 puisque :

$$\mathcal{I}_T(\text{Mauvaise condition de trafic}, \text{Accident}) = + \blacktriangleleft +$$

TM2 est maintenant compatible avec la contrainte s_2 puisque :

$$\mathcal{I}_T(\text{Mauvais temps}, \text{Mauvaise condition de trafic}) = \oplus \blacktriangleleft +$$

En revanche, TM2 est pas compatible avec la contrainte s_3 puisque :

$$\mathcal{I}_T(\text{Route}, \text{Route sinueuse}) = ? \blacktriangleleft -$$

Par souci de concision, nous ne donnons pas la définition de compatibilité avec une spécification. Cette définition est en effet triviale puisqu'elle est similaire à celle de la cohérence avec une spécification (définition 3.8).

Exemple 3.10. *Nous avons montré dans l'exemple 3.9 que TM2 est compatible avec les contraintes s_1 et s_2 mais pas avec la contrainte s_3 .*

Par conséquent, TM2 est compatible avec les spécifications $S_1 = \{s_1\}$ et $S_2 = \{s_1, s_2\}$.

Elle n'est en revanche toujours pas compatible avec $S_3 = \{s_1, s_2, s_3\}$.

La proposition 3.3 nous assure qu'une carte cohérente avec une spécification est également compatible avec cette même spécification.

Proposition 3.3. *Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T et définie sur $\{+, -\}$. Soit $s = \langle c_1, c_2, \alpha \rangle$ une contrainte définie sur T et $\{+, \oplus, 0, \ominus, -, ?\}$.*

$$TM \text{ est cohérente avec } s \implies TM \text{ est compatible avec } s$$

Démonstration. Si TM est cohérente avec s , alors $\mathcal{I}_T(c_1, c_2) = \alpha$ (définition 3.7). La notion de valeurs compatibles (définition 3.9) est réflexive. Donc $\mathcal{I}_T(c_1, c_2) \blacktriangleleft \alpha$ et donc TM est compatible avec s . \square

3.3 Paramétrage

Les critères de qualité que nous avons définis au cours de ce chapitre s'appliquent tous à une carte cognitive définie sur l'ensemble de valeurs $\{+, -\}$. Comme expliqué dans le chapitre 2, une carte cognitive peut être définie sur d'autres ensembles de valeurs, tels que l'intervalle de valeurs numériques $[-1; 1]$ ou un ensemble de valeurs symboliques totalement ordonné tel que $\{nul < faible < moyen < fort\}$. Excepté la compatibilité, les différents critères étant définis sur une égalité, on peut tout à fait les appliquer tels quels à ces deux ensembles de valeurs. Néanmoins, comme pour l'ensemble de valeurs $\{+, -\}$ et comme discuté dans la section 3.2.3, cette égalité est parfois trop stricte. La notion de valeurs compatibles a été définie

pour ajouter de la souplesse au critère de cohérence et ainsi définir le critère de compatibilité. Il faudrait disposer d'un mécanisme similaire pour $[-1; 1]$ et $\{nul < faible < moyen < fort\}$. Le designer pourrait ainsi être capable de valider un critère même si deux valeurs ne sont pas strictement égales, en se contentant du fait qu'elles soient compatibles. L'idée est donc de définir de nouveaux critères de qualité ajoutant de la souplesse pour ces ensembles des valeurs. Ces nouveaux critères seront aux critères précédemment définis ce que le critère de compatibilité est au critère de cohérence pour $\{+, -\}$. Nous devons pour cela définir la notion de valeurs compatibles pour ces deux ensembles de valeurs, puis expliquer comment adapter les critères de propreté, de non-ambiguïté et de compatibilité pour les rendre plus souples pour ces ensembles de valeurs.

Dans un premier temps, nous expliquons comment adapter nos critères de qualité aux ensembles de valeurs $[-1; 1]$ dans la section 3.3.1. Dans un second temps, nous expliquons comment les adapter pour l'ensemble de valeurs $\{nul < faible < moyen < fort\}$ dans la section 3.3.2.

3.3.1 Critères de qualité pour $[-1; 1]$

Si les critères de qualité définis précédemment peuvent s'appliquer sans problème sur les valeurs de l'ensemble $[-1; 1]$, le fait qu'ils soient tous basés sur une égalité les rend trop stricts. Pour leur ajouter de la souplesse, il faudrait pouvoir définir un intervalle de valeurs acceptables autour de chaque valeur considérée. L'idée est de définir un *seuil* qui indique la marge d'erreur qu'on s'autorise pour valider un critère. Cette valeur numérique nous sert à définir un intervalle autour d'une valeur donnée pour indiquer quelles autres valeurs de l'ensemble seront acceptées. Nous définissons alors un certain de seuil de tolérance : plus ce seuil est élevé, plus le nombre de valeurs acceptées est grand. Si au contraire ce seuil est nul, alors on retrouve une égalité stricte. Utiliser cette notion de seuil revient plus ou moins à définir la notion de valeurs compatibles (définition 3.9) pour l'ensemble $[-1; 1]$. On parle alors de *valeurs compatibles selon un seuil*.

Nous présentons tout d'abord comment adapter les critères de propreté et de non-ambiguïté en utilisant cette notion de valeurs compatibles selon un seuil. Nous définissons ensuite comment définir le critère de compatibilité pour $[-1; 1]$, qui représente un cas particulier.

Propreté et non-ambiguïté

Nous définissons la notion de valeurs compatibles selon un seuil. Nous devons donc avant toute chose définir ce qu'est un seuil.

Définition 3.11 (Seuil)

Un seuil $\delta \in \mathbb{R}^+$ est une valeur numérique positive représentant la différence maximale autorisée entre deux valeurs pour valider un critère de qualité.

Cette notion de seuil est utilisée pour définir la notion de *valeurs compatibles selon un seuil*. Deux valeurs sont compatibles entre elles selon un seuil si leur différence en valeur absolue est inférieure à ce seuil.

Définition 3.12 (Valeurs compatibles selon un seuil pour $[-1; 1]$)

Soient $\alpha_1, \alpha_2 \in [-1; 1]$ deux valeurs. Soit $\delta \in \mathbb{R}^+$ un seuil.

α_1 et α_2 sont compatibles selon δ ssi $|\alpha_1 - \alpha_2| \leq \delta$

Avec cette définition, on remarque qu'avec un seuil de 0, on retrouve une égalité stricte.

Exemple 3.11. On considère les valeurs numériques 0.5 et 0.6. Avec un seuil de 0.05, ces deux valeurs ne sont pas compatibles entre elles puisque $|0.5 - 0.6| = 0.1 > 0.05$. Seules les valeurs comprises entre $0.5 - 0.05 = 0.45$ et $0.5 + 0.05 = 0.55$ sont compatibles avec 0.5.

À partir de là, nous pouvons redéfinir les différents critères de qualité pour l'ensemble de valeurs $[-1; 1]$ en remplaçant les égalités strictes par la notion de valeurs compatibles selon un seuil. Toutefois, cette adaptation ne peut se faire que pour des critères manipulant des valeurs numériques, ce qui n'est pas le cas du critère de compatibilité.

Nous définissons les critères de propreté et de non-ambiguïté pour un concept de concepts selon un seuil pour l'ensemble de valeurs $[-1; 1]$ en adaptant très simplement les définitions 3.1 et 3.3.

Définition 3.13 (Couple de concepts propre selon un seuil pour $[-1; 1]$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur $[-1; 1]$. Soit $\delta \in \mathbb{R}^+$ un seuil. Soit $c_1, c_2 \in C$ deux concepts liés par une influence directe dans CM , i.e. $(c_1, c_2) \in A$.

(c_1, c_2) est propre selon δ ssi $|\mathcal{I}(c_1, c_2), \text{label}((c_1, c_2))| \leq \delta$.

Définition 3.14 (Couple de concepts non-ambigu selon un seuil pour $[-1; 1]$)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur $[-1; 1]$. Soit $\delta \in \mathbb{R}^+$ un seuil. Soient $c_1, c_2 \in C$ deux concepts de CM .

(c_1, c_2) est non-ambigu selon δ ssi $\forall P \in \mathcal{P}_{c_1, c_2}, |\mathcal{IP}(P), \mathcal{I}(c_1, c_2)| \leq \delta$.

Par souci de concision, nous ne donnons pas les définitions des notions de cartes cognitives propre et non-ambiguë selon un seuil car ce sont des simples adaptations des définitions 3.2 et 3.4. Par ailleurs, on peut facilement montrer que les propositions 3.1 et 3.2, qui nous assurent que la propreté est un cas particulier de la non-ambiguïté, sont toujours valables pour de telles définitions, à condition de considérer le même seuil pour les deux critères.

Exemple 3.12. On considère la carte cognitive $CM5$ (figure 3.6).

Avec un seuil de 0.3, $CM5$ n'est pas propre. En effet le couple (Pluie, Incendie) n'est pas propre selon 0.3 :

$$\begin{aligned} \mathcal{I}(\text{Pluie}, \text{Incendie}) &= -0.39 \\ \text{label}((\text{Pluie}, \text{Incendie})) &= -0.7 \end{aligned}$$

et $|-0.39 - (-0.7)| = 0.31 > 0.3$. Avec un seuil ≥ 0.31 , cette carte serait en revanche propre.

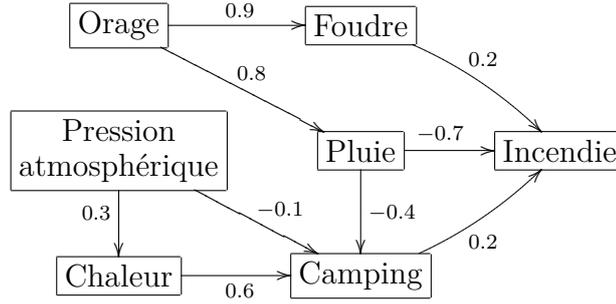


FIGURE 3.6 – La carte cognitive CM5.

Avec un seuil de 0.3, cette carte est aussi ambiguë. En effet, le couple (Pluie, Incendie) est également ambigu selon 0.3 :

$$\begin{aligned}\mathcal{IP}(Pluie \rightarrow Incendie) &= -0.7 \\ \mathcal{IP}(Pluie \rightarrow Camping \rightarrow Incendie) &= -0.08 \\ \mathcal{I}(Pluie, Incendie) &= -0.39\end{aligned}$$

et $|-0.7 - (-0.39)| = |-0.08 - (-0.39)| = 0.31 > 0.3$.

Avec un seuil de 0.4, ce couple n'est plus ambigu, mais cette carte l'est toujours par le couple (Orage, Incendie). En effet, on a :

$$\begin{aligned}\mathcal{IP}(Orage \rightarrow Pluie \rightarrow Incendie) &= -0.56 \\ \mathcal{I}(Orage, Incendie) &= -0.148\end{aligned}$$

et $|-0.56 - (-0.148)| = 0.412 > 0.4$. Avec un seuil ≥ 0.412 , cette carte est en revanche non-ambiguë.

Bien que les cartes cognitives soient définies sur des ensembles de valeurs différents, on remarque que, dans la carte cognitive de l'exemple 3.12 définie sur $[-1; 1]$, le couple (Pluie, Incendie) n'est pas propre selon un seuil de 0.3 alors qu'il l'était dans la carte cognitive de l'exemple 3.2 définie sur $\{+, -\}$. Au contraire, le couple (Pression atmosphérique, Camping) est propre selon le même seuil alors qu'il ne l'était pas dans l'autre exemple. Ceci est dû au fait que dans l'exemple 3.2, deux valeurs de signes opposés invalidaient nécessairement un critère. Avec la définition des critères selon un seuil, seul l'écart entre les valeurs compte.

On a :

$$\begin{aligned}\mathcal{I}(Pression\ atmosphérique, Camping) &= 0.04 \\ \text{label}((Pression\ atmosphérique, Camping)) &= -0.1\end{aligned}$$

et $|0.04 - (-0.1)| = 0.14$. Pour le couple de concepts (Pression atmosphérique, Camping), seul un seuil inférieur à 0.14 le rendrait donc non-propre. Ce couple est donc, selon notre définition, « plus propre » que (Pluie, Incendie) puisque que seul un seuil inférieur à 0.14 invaliderait le critère de propreté pour (Pression atmosphérique, Camping) alors qu'il est nécessaire d'avoir un seuil supérieur à 0.31 pour valider (Pluie, Incendie). Un moyen de résoudre ce problème serait de ne pas utiliser une simple différence entre deux valeurs mais une fonction sur ces valeurs définie de telle manière que l'écart entre ces deux valeurs est plus important si elles sont de signes opposés.

Compatibilité

Nous cherchons à présent à définir le critère de compatibilité pour l'ensemble de valeurs $[-1; 1]$, qui doit ajouter de la souplesse au critère de cohérence. Comme nous l'avons énoncé précédemment, la notion de valeurs compatibles selon un seuil n'est applicable qu'aux valeurs numériques. Or, le critère de cohérence ne manipule pas de telles valeurs. En effet, comme nous l'avons expliqué dans la section 3.2.1, la valeur de la contrainte doit appartenir au codomaine de l'influence taxonomique. Pour l'ensemble de valeurs $[-1; 1]$, d'après la définition de celle-ci (définition 2.13), ce codomaine est l'ensemble des intervalles sur $[-1; 1]$. La valeur de la contrainte doit donc être un intervalle de valeurs numériques. Une différence entre deux intervalles n'est pas une valeur numérique mais un autre intervalle. Or, un intervalle est incomparable à un seuil, qui est une valeur numérique. Nous ne pouvons donc simplement utiliser la notion de valeurs compatibles selon un seuil pour le critère de compatibilité.

Puisque ce critère manipule des intervalles de valeurs numériques, nous devons définir le fait que deux intervalles sont compatibles. On peut imaginer différentes manières de définir cette notion. On peut par exemple considérer que deux intervalles sont non-contradictoire s'ils ont des valeurs en commun, autrement dit s'ils se chevauchent. Cette définition semble pertinente : les intervalles représentent en effet une certaine incertitude dans la valeur de l'influence taxonomique. Si les deux intervalles ne se chevauchent pas, on sort du cadre de l'incertitude et on est donc sûr que les deux valeurs ne sont pas incompatibles. Cependant, un problème se pose lorsqu'une influence taxonomique est issue de valeurs hétérogènes et que l'intervalle qui la représente couvre quasiment tout l'ensemble $[-1; 1]$. En effet, une telle influence taxonomique validerait fréquemment une contrainte, même si la valeur spécifiée par la contrainte est un petit intervalle puisqu'il y aurait de grandes chances qu'elle le recouvre en partie. Pour cette raison, on définit le fait que deux intervalles sont compatibles en traitant différemment l'intervalle représentant l'influence taxonomique et celui représentant la valeur de la contrainte. Nous considérons qu'une contrainte n'est vérifiée que si l'intervalle représentant l'influence taxonomique entre les concepts de cette contrainte est inclus dans l'intervalle représentant la valeur de la contrainte. L'intervalle de la contrainte indique ainsi les différentes valeurs qu'on autorise pour l'influence taxonomique.

On considère donc que deux intervalles sont compatibles entre eux si le premier intervalle est inclus dans le second.

Définition 3.15 (Intervalles compatibles)

Soient $[\alpha_1, \beta_1], [\alpha_2, \beta_2] \subseteq [-1; 1]$ deux intervalles de valeurs numériques. $[\alpha_1, \beta_1]$ représente la valeur de l'influence taxonomique. $[\alpha_2, \beta_2]$ représente la valeur de la contrainte.

$[\alpha_1, \beta_1]$ et $[\alpha_2, \beta_2]$ sont compatibles ssi $[\alpha_1, \beta_1] \subseteq [\alpha_2, \beta_2]$.

La définition du critère de compatibilité avec une contrainte pour l'ensemble de valeurs $[-1; 1]$ est une simple adaptation de la définition du critère de compatibilité avec une contrainte (définition 3.10). On utilise cette fois la notion d'intervalles compatibles.

Définition 3.16 (Compatibilité avec une contrainte pour $[-1; 1]$)

Soit $T = (C, \leq)$ une taxonomie. Soit $TM = (\text{elem}(T), A, \text{label})$ une carte cognitive taxonomique associée à T et définie sur $[-1; 1]$. Soit I l'ensemble des intervalles possibles sur $[-1; 1]$. Soit $s = \langle c_1, c_2, \alpha \rangle$ une contrainte définie sur T et I .

TM est compatible avec s ssi $\mathcal{I}_T(c_1, c_2) \subseteq \alpha$.

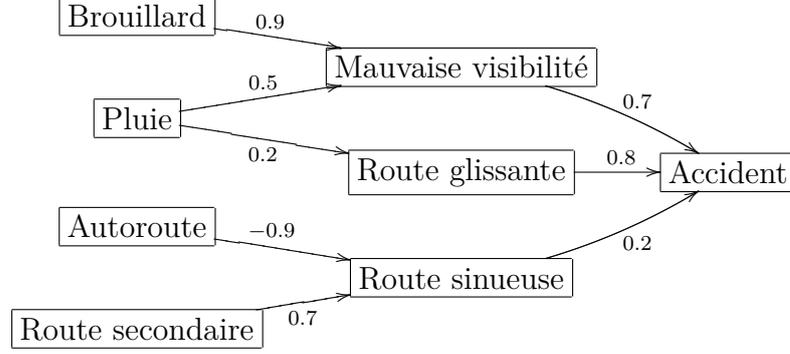


FIGURE 3.7 – Rappel de la carte cognitive $CM1$ (exemple 2.1).

Exemple 3.13. On définit les 3 contraintes suivantes, basées sur celles de l'exemple 3.5, définies sur la taxonomie $T1$ (exemple 2.6) et l'ensemble des intervalles de $[-1; 1]$:

$$s_1 = \langle \text{Mauvaise condition de trafic}, \text{Accident}, [0; 1] \rangle$$

$$s_2 = \langle \text{Mauvais temps}, \text{Mauvaise condition de trafic}, [0.5; 1] \rangle$$

$$s_3 = \langle \text{Route}, \text{Route sinueuse}, [-1; -0.5] \rangle$$

La contrainte s_1 signifie qu'on s'attend à une influence positive. La contrainte s_2 signifie qu'on s'attend à une influence positive et forte. La contrainte s_3 signifie qu'on s'attend à une influence négative et forte.

On considère la carte cognitive taxonomique $TM1$, construite à partir de la taxonomie $T1$ et la carte cognitive $CM1$ (figure 3.7).

$TM1$ est compatible avec la contrainte s_1 puisque :

$$\mathcal{I}_T(\text{Mauvaise condition de trafic}, \text{Accident}) = [0.2; 0.8] \subseteq [0; 1]$$

En revanche, $TM1$ n'est pas compatible avec la contrainte s_2 puisque :

$$\mathcal{I}_T(\text{Mauvais temps}, \text{Mauvaise condition de trafic}) = [0; 0.9] \not\subseteq [0.5; 1]$$

$TM1$ n'est pas non plus compatible avec la contrainte s_3 puisque :

$$\mathcal{I}_T(\text{Route}, \text{Route sinueuse}) = [-0.9; 0.7] \not\subseteq [-1; -0.5]$$

Par souci de concision, nous ne donnons pas la définition de la notion de carte cognitive compatible avec une spécification car cette définition est une simple adaptation de la définition 3.8.

Exemple 3.14. On considère à nouveau les contraintes de l'exemple 3.13. Nous avons montré dans l'exemple 3.13 que $TM1$ est compatible avec la contrainte s_1 mais pas avec les contraintes s_2 et s_3 .

Par conséquent, $TM1$ est compatible avec la spécification $S_1 = \{s_1\}$ mais pas avec les spécifications $S_2 = \{s_1, s_2\}$ et $S_3 = \{s_1, s_2, s_3\}$.

3.3.2 Critères de qualité pour $\{nul < faible < moyen < fort\}$

Les valeurs de l'ensemble $\{nul < faible < moyen < fort\}$ étant peu nombreuses, on peut considérer qu'une égalité stricte est suffisante. Cependant, de tels ensembles totalement ordonnés contiennent parfois davantage de valeurs. Dans une telle situation, on aimerait également de la souplesse, comme pour les ensembles de valeurs $\{+, -\}$ et $[-1; 1]$. Nous devons donc définir la notion de valeurs compatibles pour un ensemble fini totalement ordonné. On peut, comme pour l'ensemble de valeurs $\{+, \oplus, 0, \ominus, -, ?\}$, lister l'ensemble des valeurs compatibles avec une valeur donnée. Ceci peut toutefois s'avérer fastidieux si le nombre de valeurs est grand.

L'idée derrière la notion de valeurs compatibles pour un ensemble fini totalement ordonné est d'utiliser à nouveau un seuil. On considère que deux valeurs sont compatibles si le nombre de valeurs comprises entre ces deux valeurs dans l'ordre est inférieur à ce seuil. Pour cela, on donne à chaque valeur un indice qui représente sa place dans l'ordre. Si la différence entre les deux indices en valeur absolue est inférieure au seuil, alors les deux valeurs sont compatibles.

Définition 3.17 (Valeurs compatibles selon un seuil pour un ensemble fini totalement ordonné)

Soit I un ensemble de n éléments totalement ordonnés par une relation d'ordre $<$. Chaque $\alpha \in I$ peut donc être associé à un unique $i_\alpha \in [1; n]$ tel que $\forall \alpha_1, \alpha_2 \in I, \alpha_1 < \alpha_2 \Leftrightarrow i_{\alpha_1} < i_{\alpha_2}$. Soient $\alpha_1, \alpha_2 \in I$ deux valeurs.

α_1 et α_2 sont compatibles selon δ ssi $|i_{\alpha_1} - i_{\alpha_2}| \leq \delta$.

Exemple 3.15. Avec l'ensemble de valeurs $\{nul < faible < moyen < fort\}$, on a $i_{nul} = 1$, $i_{faible} = 2$, $i_{moyen} = 3$ et $i_{fort} = 4$. Avec un seuil de 1, la valeur moyen est donc uniquement compatible avec les valeurs faible et fort.

À partir de là, nous pouvons redéfinir les différents critères de propreté et de non-ambiguïté pour un ensemble fini de valeurs totalement ordonné de la même manière que pour l'ensemble $[-1; 1]$.

Définition 3.18 (Couple de concepts propre selon un seuil pour un ensemble fini totalement ordonné)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur un ensemble fini de valeurs totalement ordonné. Soit $\delta \in \mathbb{R}^+$ un seuil. Soit $c_1, c_2 \in C$ deux concepts liés par une influence directe dans CM , i.e. $(c_1, c_2) \in A$.

(c_1, c_2) est propre selon δ ssi $|i_{\mathcal{I}(c_1, c_2)}, i_{\text{label}((c_1, c_2))}| \leq \delta$.

Définition 3.19 (Couple de concepts non-ambigu selon un seuil pour un ensemble fini totalement ordonné)

Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur un ensemble fini de valeurs totalement ordonné. Soit $\delta \in \mathbb{R}^+$ un seuil. Soient $c_1, c_2 \in C$ deux concepts de CM . (c_1, c_2) est non-ambigu selon δ ssi $\forall P \in \mathcal{P}_{c_1, c_2}, |i_{\mathcal{IP}(P)}, i_{\mathcal{I}(c_1, c_2)}| \leq \delta$.

Nous ne donnons pas de définition pour le critère de compatibilité car nous n'avons donné de définition formelle pour l'influence taxonomique pour $\{nul < faible < moyen < fort\}$ dans le chapitre 2. Nous avons en effet considéré qu'on

pouvait utiliser un max ou intervalle de valeurs. Selon l'opérateur choisi, l'ensemble de valeurs de la contrainte n'est pas le même. Si c'est une valeur de $\{nul < faible < moyen < fort\}$, alors on peut redéfinir le critère de compatibilité en raisonnant de la même manière que pour les critères de propreté et de non-ambiguïté définis précédemment. Si c'est un intervalle de valeurs, alors on peut redéfinir le critère de compatibilité en raisonnant de la même manière que pour l'ensemble de valeurs $[-1; 1]$. Une simple adaptation des définitions déjà données permet donc de définir le critère de compatibilité pour $\{nul < faible < moyen < fort\}$.

3.4 Conflit

Nous avons présenté plusieurs critères de qualité pour valider une carte cognitive. Cependant, lorsqu'une carte cognitive ne valide pas un certain critère de qualité, nous n'avons rien proposé pour aider le designer à corriger sa carte de manière à ce qu'elle puisse valider ce critère. Pour cela, il faut chercher à comprendre pourquoi le critère de qualité n'a pas été validé. Pour aider le designer dans sa tâche de correction, deux approches s'offrent à nous.

D'une part, on peut lui présenter seulement une partie de la carte dont on sait qu'elle ne valide pas le critère. C'est ce qu'on appelle un *conflit*. Un conflit est un ensemble de composants d'un système qui suffit à invalider un critère de qualité donné [Reiter, 1987]. Corriger seulement un composant de ce conflit peut potentiellement corriger ce conflit. Pour chacun des critères de qualité que nous avons présentés, nous définissons donc les conflits associés à ce critère qui provoquent sa non-validation. Pour valider un critère de qualité, il faut choisir quels composants présentés dans chacun des conflits corriger.

D'autre part, on peut construire un ou plusieurs *diagnostics*, des ensembles de composants qui doivent être corrigés pour valider un critère de qualité [Reiter, 1987]. L'idée d'un diagnostic est de fournir une explication sur le comportement anormal d'un système. Ainsi, un diagnostic présente au designer un ensemble de composants à corriger pour assurer que le système valide le critère de qualité. Un diagnostic permet donc de corriger tous les conflits en proposant des composants à corriger. Intuitivement, un diagnostic est donc une intersection des composants de l'intégralité des conflits.

Nous nous limitons dans cette partie à l'ensemble de valeurs $\{+, -\}$. En effet, une carte cognitive définie sur d'autres ensembles de valeurs, notamment $[-1; 1]$, nécessite une étude beaucoup plus approfondie sur les raisons qui peuvent provoquer un conflit.

Nous définissons la notion de conflits de propreté dans la section 3.4.1. Nous définissons ensuite la notion de conflits d'ambiguïté dans la section 3.4.2. Par ailleurs, nous expliquons dans la section 3.4.3 pourquoi nous ne fournissons pas de définition pour la notion de conflits de cohérence. Enfin, nous expliquons comment calculer un diagnostic à partir d'un ensemble de conflits quelconque dans la section 3.4.4.

3.4.1 Conflit de propreté

Un *conflit de propreté* apparaît dès qu'un couple de concepts liés directement par une influence n'est pas propre. Le critère de propreté vérifie que l'influence propagée de ce couple de concepts est la même que la valeur de l'influence directe entre ces concepts. Si ce critère n'est pas validé, c'est donc que l'information utilisée par le calcul de l'influence propagée ou l'influence directe comporte une erreur. Pour calculer l'influence propagée, il faut considérer les influences de tous les chemins permettant de lier les deux concepts et réaliser un calcul sur les valeurs portées par cette influence. Par conséquent, les composants constituant un conflit de propreté sont des influences, associées à leurs valeurs.

On rappelle que le critère de propreté n'est pas validé si l'influence propagée du couple de concepts est différente de la valeur de l'influence directe entre ces concepts. En analysant la définition de l'influence propagée d'un concept sur un autre (définition 2.31), on se rend compte qu'on agrège la valeur de l'influence directe avec d'autres valeurs grâce à l'opérateur \vee . Pour que le résultat du calcul soit différent de l'influence directe, il faut nécessairement que l'influence directe ait été agrégée avec au moins une valeur différente. Cela signifie qu'il existe au moins un chemin entre les deux concepts tel que l'influence propagée sur ce chemin est différente de l'influence directe. En analysant la définition de l'influence propagée sur un chemin (définition 2.30), on se rend compte que si on supprime ou qu'on inverse la valeur de l'une des influences du chemin, celui-ci n'invalide plus le critère. C'est donc l'intégralité des influences de ce chemin associée à l'influence directe qui provoquent un conflit de propreté.

Puisqu'on considère les influences d'un chemin, on a besoin de l'ensemble des *influences qui composent ce chemin*.

Définition 3.20 (Influences d'un chemin)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

Les influences de P sont :

$$\text{influences}(P) = \bigcup_{i=0}^{k-1} \{(u_i, u_{i+1})\}$$

Pour savoir quels chemins provoquent un conflit, on considère uniquement les chemins dont l'influence propagée est différente de la valeur de l'influence directe. On considère deux concepts liés par une influence directe dans la carte cognitive. Si l'influence propagée sur un chemin entre ces deux concepts est différente de la valeur de l'influence directe, alors le conflit est défini comme étant l'union des influences composant ce chemin et de l'influence directe. Sinon, le conflit est défini comme un ensemble vide.

Définition 3.21 (Conflit de propreté)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $(c_1, c_2) \in A$ un couple de concepts de CM liés par une influence. Soit $P \in \mathcal{P}_{c_1, c_2}$ un chemin minimal de c_1 vers c_2 .

Le conflit de propreté de P est :

$$cc_P^{\text{propre}} = \begin{cases} \text{influences}(P) \cup \{(c_1, c_2)\} & \text{si } \mathcal{IP}(P) \neq \text{label}((c_1, c_2)) \\ \emptyset & \text{sinon} \end{cases}$$

Remarque. Puisqu'on a toujours $\mathcal{IP}(c_1 \rightarrow c_2) = \text{label}((c_1, c_2))$, on a l'assurance que l'influence directe ne sera jamais en conflit avec elle-même.

Exemple 3.16. Dans la carte cognitive CM4 (exemple 3.2), si on considère le couple de concepts (Pression atmosphérique, Camping), le chemin Pression atmosphérique \rightarrow Chaleur \rightarrow Camping provoque un conflit de propreté.

$$\left\{ \begin{array}{l} (\text{Pression atmosphérique, Chaleur}), (\text{Chaleur, Camping}), \\ (\text{Pression atmosphérique, Camping}) \end{array} \right\}$$

est par conséquent un conflit de propreté.

De cette définition, on définit l'ensemble des conflits de propreté pour un couple de concepts. Cet ensemble est l'ensemble des conflits de propreté pour chaque chemin entre les deux concepts. De cet ensemble, on retire toutefois les chemins dont l'ensemble des conflits de propreté est vide.

Définition 3.22 (Ensemble des conflits de propreté pour un couple de concepts)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $(c_1, c_2) \in A$ un couple de concepts de CM liés par une influence.

L'ensemble des conflits de propreté pour (c_1, c_2) est :

$$CC_{(c_1, c_2)}^{\text{propre}} = \bigcup_{\substack{P \in \mathcal{P}_{c_1, c_2} \\ CC_P^{\text{propre}} \neq \emptyset}} \{CC_P^{\text{propre}}\}$$

Exemple 3.17. Dans la carte cognitive CM4 (exemple 3.2), si on considère le couple de concepts (Pression atmosphérique, Camping), le chemin Pression atmosphérique \rightarrow Chaleur \rightarrow Camping est le seul chemin de Pression atmosphérique vers Camping provoquant un conflit de propreté, comme présenté dans l'exemple 3.16. L'ensemble des conflits de propreté pour (Pression atmosphérique, Camping) est donc :

$$\left\{ \left\{ \begin{array}{l} (\text{Pression atmosphérique, Chaleur}), (\text{Chaleur, Camping}), \\ (\text{Pression atmosphérique, Camping}) \end{array} \right\} \right\}$$

La proposition 3.4 nous assure que si un couple de concepts est propre, alors il n'y a aucun conflit de propreté et inversement.

Proposition 3.4. Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $(c_1, c_2) \in A$ un couple de concepts de CM liés par une influence.

$$(c_1, c_2) \text{ est propre} \iff CC_{(c_1, c_2)}^{\text{propre}} = \emptyset$$

Démonstration.

1. Si (c_1, c_2) est propre, alors $\mathcal{I}(c_1, c_2) = \text{label}((c_1, c_2))$. En analysant la définition de l'influence propagée d'un concept sur un autre (définition 2.31), on note que si $\exists P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) \neq \text{label}((c_1, c_2)) \Rightarrow \mathcal{I}(c_1, c_2) = ?$. Donc,

$\mathcal{I}(c_1, c_2) \neq ? \Rightarrow \forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) = \text{label}((c_1, c_2))$. Puisque $\mathcal{I}(c_1, c_2) = \text{label}((c_1, c_2))$, $\mathcal{I}(c_1, c_2)$ vaut donc soit $+$, soit $-$, donc a bien $\mathcal{I}(c_1, c_2) \neq ?$ et donc $\forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) = \text{label}((c_1, c_2))$. Par la définition 3.21, on a donc $\forall P \in \mathcal{P}_{c_1, c_2}, cc_P^{\text{propre}} = \emptyset$. Par la définition 3.22, on a donc finalement $CC_{(c_1, c_2)}^{\text{propre}} = \emptyset$.

Par conséquent, (c_1, c_2) est propre $\implies CC_{(c_1, c_2)}^{\text{propre}} = \emptyset$.

2. Si $CC_{(c_1, c_2)}^{\text{propre}} = \emptyset$, alors $\bigcup_{\substack{P \in \mathcal{P}_{c_1, c_2} \\ cc_P^{\text{propre}} \neq \emptyset}} \{cc_P^{\text{propre}}\} = \emptyset$. On a donc $\forall P \in \mathcal{P}_{c_1, c_2}, cc_P^{\text{propre}} =$

\emptyset et par conséquent $\forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) = \text{label}((c_1, c_2))$. Tous les chemins portent donc la même valeur que l'influence directe d'un concept sur un autre. Puisque $(c_1, c_2) \in A$, $c_1 \rightarrow c_2 \in \mathcal{P}_{c_1, c_2}$ et donc $\mathcal{P}_{c_1, c_2} \neq \emptyset$. Si $\mathcal{I}(c_1, c_2) = \text{label}((c_1, c_2)) = +$, par la définition de l'influence propagée d'un concept sur un autre, on en déduit donc que $\bigvee_{P \in \mathcal{P}_{c_1, c_2}} \mathcal{IP}(P) = +$. Si $\mathcal{I}(c_1, c_2) = \text{label}((c_1, c_2)) = -$, on déduit de la même manière que $\bigvee_{P \in \mathcal{P}_{c_1, c_2}} \mathcal{IP}(P) = -$.

Au final, $\mathcal{I}(c_1, c_2) = \text{label}((c_1, c_2))$ et donc (c_1, c_2) est propre.

Par conséquent, (c_1, c_2) est propre $\iff CC_{(c_1, c_2)}^{\text{propre}} = \emptyset$. \square

On déduit facilement de ces définitions l'ensemble des conflits de propreté d'une carte cognitive. Pour cela, on réalise l'union des conflits de propreté de chacun des couples de concepts liés par une influence directe. Pour obtenir ces couples de concepts, on se base tout simplement sur l'ensemble des influences de la carte.

Définition 3.23 (Ensemble des conflits de propreté d'une carte cognitive)

Soit $CM = (C, A, \text{label})$ une carte cognitive.

L'ensemble des conflits de propreté pour CM est :

$$CC^{\text{propre}} = \bigcup_{(c_1, c_2) \in A} CC_{(c_1, c_2)}^{\text{propre}}$$

Grâce à la proposition 3.4, on peut facilement montrer que si une carte cognitive est propre, alors il n'y a aucun conflit de propreté et inversement.

Exemple 3.18. Le seul conflit de propreté de $CM4$ (exemple 3.2) est celui présenté à l'exemple 3.17. L'ensemble des conflits de propreté de $CM4$ est donc :

$$\left\{ \left\{ \begin{array}{l} (\text{Pression atmosphérique}, \text{Chaleur}), (\text{Chaleur}, \text{Camping}), \\ (\text{Pression atmosphérique}, \text{Camping}) \end{array} \right\} \right\}$$

Modifier une seule de ces influences, c'est-à-dire la supprimer ou inverser sa valeur, suffit à permettre à la carte cognitive de valider le critère de propreté.

3.4.2 Conflit d'ambiguïté

L'ambiguïté a été démontrée comme étant une généralisation de la propreté (propositions 3.1 et 3.2). Par conséquent, pour chaque définition d'un ensemble de

conflits d'ambiguïté, nous montrons qu'elle inclut la définition de l'ensemble des conflits de propreté correspondante.

Tout comme un conflit de propreté, un *conflit d'ambiguïté* apparaît lui aussi dès qu'un couple de concepts est ambigu. Le critère de non-ambiguïté vérifie que, pour chaque chemin entre ces deux concepts, l'influence propagée sur ce chemin est égale à l'influence propagée du couple de concepts. En appliquant le même raisonnement que pour la propreté, on déduit que les conflits d'ambiguïté sont aussi basés sur la notion d'influence.

En analysant à nouveau la définition de l'influence propagée d'un concept sur un autre ainsi que celle de l'opérateur \vee , on se rend compte qu'une valeur agrégée par \vee est différente du résultat de l'agrégation si et seulement si elle a été agrégée avec une autre valeur qui lui est différente. Cela signifie qu'il existe au moins deux chemins entre les deux concepts tels que les influences propagées sur ces chemins sont différentes. En appliquant le même raisonnement que pour la propreté, on en déduit que c'est l'union des influences des deux chemins qui provoque un conflit.

On considère un couple de concepts de la carte cognitive. Un conflit d'ambiguïté entre deux chemins liant ces deux concepts apparaît donc dès que leurs influences propagées sont différentes. Il est défini comme étant l'union des influences qui composent ces deux chemins. Sinon, le conflit est défini comme un ensemble vide.

Définition 3.24 (Conflit d'ambiguïté)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $c_1, c_2 \in C$ deux concepts de CM . Soient $P_1, P_2 \in \mathcal{P}_{c_1, c_2}$ deux chemins minimaux de c_1 vers c_2 . Le conflit d'ambiguïté de P_1, P_2 est :

$$cc_{P_1, P_2}^{ambig} = cc_{P_2, P_1}^{ambig} = \begin{cases} \text{influences}(P_1) \cup \text{influences}(P_2) & \text{si } \mathcal{IP}(P_1) \neq \mathcal{IP}(P_2) \\ \emptyset & \text{sinon} \end{cases}$$

Exemple 3.19. Dans la carte cognitive $CM4$ (exemple 3.4), si on considère le couple de concepts (Orage, Incendie), les chemins Orage \rightarrow Foudre \rightarrow Incendie et Orage \rightarrow Pluie \rightarrow Incendie sont en conflit d'ambiguïté.

$$\left\{ \begin{array}{l} (\text{Orage}, \text{Foudre}), (\text{Foudre}, \text{Incendie}) \\ (\text{Orage}, \text{Pluie}), (\text{Pluie}, \text{Incendie}) \end{array} \right\}$$

est par conséquent un conflit d'ambiguïté.

La proposition 3.5 nous assure que, pour un couple de concepts liés directement par une influence, les conflits de propreté d'un chemin et les conflits d'ambiguïté de ce chemin et du chemin composé de l'influence directe sont égaux.

Proposition 3.5. Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $(c_1, c_2) \in A$ un couple de concepts de CM liés par une influence. Soit $P \in \mathcal{P}_{c_1, c_2}$ un chemin minimal de c_1 vers c_2 .

$$cc_P^{propre} = cc_{P, c_1 \rightarrow c_2}^{ambig}$$

Démonstration. Pour cette démonstration, on considère tout d'abord le cas où les deux conflits sont des ensembles vides, puis le cas où ils ne le sont pas.

- Si $cc_P^{\text{propre}} = \emptyset$, alors $\mathcal{IP}(P) = \text{label}((c_1, c_2))$ (définition 3.21). Puisque d'après la définition de l'influence propagée d'un concept sur un autre (définition 2.30), $\mathcal{IP}(c_1 \rightarrow c_2) = \text{label}((c_1, c_2))$, $\mathcal{IP}(P) = \mathcal{IP}(c_1 \rightarrow c_2)$ et donc $cc_{P, c_1 \rightarrow c_2}^{\text{ambig}} = \emptyset$ (définition 3.24).
D'où $cc_P^{\text{propre}} = \emptyset \Rightarrow cc_P^{\text{propre}} = cc_{P, c_1 \rightarrow c_2}^{\text{ambig}}$.
- Si $cc_P^{\text{propre}} \neq \emptyset$, alors $\mathcal{IP}(P) \neq \text{label}((c_1, c_2))$ et $cc_P^{\text{propre}} = \text{influences}(P) \cup \{(c_1, c_2)\}$. Puisque $\mathcal{IP}(c_1 \rightarrow c_2) = \text{label}((c_1, c_2))$, on a $\mathcal{IP}(P) \neq \mathcal{IP}(c_1 \rightarrow c_2)$. On a donc aussi $cc_{P, c_1 \rightarrow c_2}^{\text{ambig}} = \text{influences}(P) \cup \text{influences}(c_1 \rightarrow c_2) = \text{influences}(P) \cup \{(c_1, c_2)\}$.
D'où $cc_P^{\text{propre}} \neq \emptyset \Rightarrow cc_P^{\text{propre}} = cc_{P, c_1 \rightarrow c_2}^{\text{ambig}}$. \square

De cette définition, on définit l'ensemble des conflits d'ambiguïté pour un couple de concepts. Cet ensemble est l'ensemble des conflits d'ambiguïté pour chaque paire de chemins entre les deux concepts. De cet ensemble, on retire toutefois les paires de chemins dont l'ensemble des conflits d'ambiguïté est vide.

Définition 3.25 (Ensemble des conflits d'ambiguïté pour un couple de concepts)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts de CM .

L'ensemble des conflits d'ambiguïté pour (c_1, c_2) est :

$$CC_{(c_1, c_2)}^{\text{ambig}} = \bigcup_{P_1 \in \mathcal{P}_{c_1, c_2}} \bigcup_{\substack{P_2 \in \mathcal{P}_{c_1, c_2} \\ cc_{P_1, P_2}^{\text{ambig}} \neq \emptyset}} \{cc_{P_1, P_2}^{\text{ambig}}\}$$

Exemple 3.20. Dans la carte cognitive $CM4$ (exemple 3.4), si on considère le couple de concepts (Orage, Incendie), les chemins Orage \rightarrow Foudre \rightarrow Incendie et Orage \rightarrow Pluie \rightarrow Incendie sont en conflit d'ambiguïté pour le couple (Orage, Incendie), comme présenté dans l'exemple 3.19. Les chemins Orage \rightarrow Foudre \rightarrow Incendie et Orage \rightarrow Pluie \rightarrow Camping \rightarrow Incendie sont également en conflit d'ambiguïté. L'ensemble des conflits d'ambiguïté pour (Orage, Pluie) est donc :

$$\left\{ \left\{ \begin{array}{l} (\text{Orage, Foudre}), (\text{Foudre, Incendie}) \\ (\text{Orage, Pluie}), (\text{Pluie, Incendie}) \end{array} \right\}, \left\{ \begin{array}{l} (\text{Orage, Foudre}), (\text{Foudre, Incendie}) \\ (\text{Orage, Pluie}), (\text{Pluie, Camping}), (\text{Camping, Incendie}) \end{array} \right\} \right\}$$

La proposition 3.6 nous assure que si un couple de concepts est non-ambigu, alors il n'y a aucun conflit d'ambiguïté et inversement.

Proposition 3.6. Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts de CM .

$$(c_1, c_2) \text{ est non-ambigu} \iff CC_{(c_1, c_2)}^{\text{ambig}} = \emptyset$$

Démonstration.

1. Si (c_1, c_2) est non-ambigu, alors, d'après la définition 3.3, $\forall P \in \mathcal{P}_{c_1, c_2}$, $\mathcal{IP}(P) = \mathcal{I}(c_1, c_2)$. En analysant la définition de l'influence propagée d'un concept sur un autre (définition 2.31), on note que si $\exists P_1, P_2 \in \mathcal{P}_{c_1, c_2}$, $\mathcal{IP}(P_1) \neq \mathcal{IP}(P_2) \Rightarrow \mathcal{I}(c_1, c_2) = ?$. Donc, $\mathcal{I}(c_1, c_2) \neq ? \Rightarrow \forall P_1, P_2 \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P_1) = \mathcal{IP}(P_2)$. Puisque $\forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) = \mathcal{I}(c_1, c_2)$, $\mathcal{I}(c_1, c_2)$ vaut donc soit $+$, soit $-$, donc a bien $\mathcal{I}(c_1, c_2) \neq ?$ et donc $\forall P_1, P_2 \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P_1) = \mathcal{IP}(P_2)$. Par la définition 3.24, on a donc $\forall P_1, P_2 \in \mathcal{P}_{c_1, c_2}, cc_{P_1, P_2}^{\text{ambig}} = \emptyset$. Par la définition 3.25, on a donc finalement $CC_{(c_1, c_2)}^{\text{ambig}} = \emptyset$.
Par conséquent, (c_1, c_2) est non-ambigu $\implies CC_{(c_1, c_2)}^{\text{ambig}} = \emptyset$.

2. Si $CC_{(c_1, c_2)}^{\text{ambig}} = \emptyset$, alors $\bigcup_{P_1 \in \mathcal{P}_{c_1, c_2}} \bigcup_{\substack{P_2 \in \mathcal{P}_{c_1, c_2} \\ cc_{P_1, P_2}^{\text{ambig}} \neq \emptyset}} \{cc_{P_1, P_2}^{\text{ambig}}\} = \emptyset$. Si $\mathcal{P}_{c_1, c_2} = \emptyset$, alors

(c_1, c_2) est trivialement non-ambigu.

Si $\mathcal{P}_{c_1, c_2} \neq \emptyset$, on a $\forall P_1, P_2 \in \mathcal{P}_{c_1, c_2}, cc_{P_1, P_2}^{\text{ambig}} = \emptyset$ et, par conséquent, $\forall P_1, P_2 \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P_1) = \mathcal{IP}(P_2)$. Tous les chemins ont donc la même influence propagée. Si cette influence propagée est de $+$, d'après la définition de l'influence propagée d'un concept sur un autre, on en déduit que, puisque $\mathcal{P}_{c_1, c_2} \neq \emptyset$, $\mathcal{I}(c_1, c_2) = \bigvee_{P \in \mathcal{P}_{c_1, c_2}} \mathcal{IP}(P) = +$. Si elle est de $-$, on déduit de la même manière que $\mathcal{I}(c_1, c_2) = \bigvee_{P \in \mathcal{P}_{c_1, c_2}} \mathcal{IP}(P) = -$. Au final, $\forall P \in \mathcal{P}_{c_1, c_2}, \mathcal{IP}(P) =$

$\mathcal{I}(c_1, c_2)$ et donc (c_1, c_2) est non-ambigu.

Par conséquent, (c_1, c_2) est non-ambigu $\iff CC_{(c_1, c_2)}^{\text{ambig}} = \emptyset$. \square

La proposition 3.7 nous assure que les conflits de propriété d'un couple de concepts sont inclus dans les conflits d'ambiguïté de ce même couple de concepts.

Proposition 3.7. *Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $(c_1, c_2) \in A$ un couple de concepts de CM liés par une influence.*

$$CC_{(c_1, c_2)}^{\text{propre}} \subseteq CC_{(c_1, c_2)}^{\text{ambig}}$$

Démonstration. On a $CC_{(c_1, c_2)}^{\text{ambig}} = \bigcup_{P_1 \in \mathcal{P}_{c_1, c_2}} \bigcup_{\substack{P_2 \in \mathcal{P}_{c_1, c_2} \\ cc_{P_1, P_2}^{\text{ambig}} \neq \emptyset}} \{cc_{P_1, P_2}^{\text{ambig}}\}$ (définition 3.25).

Puisque $c_1 \rightarrow c_2 \in \mathcal{P}_{c_1, c_2}$, on a :

$$\begin{aligned} CC_{(c_1, c_2)}^{\text{ambig}} &\supseteq \bigcup_{P_1 \in \mathcal{P}_{c_1, c_2}} \bigcup_{\substack{P_2 \in \{c_1 \rightarrow c_2\} \\ cc_{P_1, P_2}^{\text{ambig}} \neq \emptyset}} \{cc_{P_1, P_2}^{\text{ambig}}\} \\ &\supseteq \bigcup_{\substack{P \in \mathcal{P}_{c_1, c_2} \\ cc_{P, c_1 \rightarrow c_2}^{\text{ambig}} \neq \emptyset}} \{cc_{P, c_1 \rightarrow c_2}^{\text{ambig}}\} \\ &\supseteq \bigcup_{\substack{P \in \mathcal{P}_{c_1, c_2} \\ cc_P^{\text{propre}} \neq \emptyset}} \{cc_P^{\text{propre}}\} && \text{(proposition 3.5)} \\ &\supseteq CC_{(c_1, c_2)}^{\text{propre}} && \text{(définition 3.22)} \quad \square \end{aligned}$$

On déduit facilement de ces définitions l'ensemble des conflits d'ambiguïté d'une carte cognitive. Pour cela, on réalise l'union des conflits d'ambiguïté de chacun des couples de concepts de la carte cognitive.

Définition 3.26 (Ensemble des conflits d'ambiguïté d'une carte cognitive)

Soit $CM = (C, A, \text{label})$ une carte cognitive.

L'ensemble des conflits d'ambiguïté pour CM est :

$$CC^{ambig} = \bigcup_{c_1 \in C} \bigcup_{c_2 \in C} CC_{(c_1, c_2)}^{ambig}$$

Grâce à la proposition 3.6, on peut facilement montrer que si une carte cognitive est non-ambiguë, alors il n'y a aucun conflit d'ambiguïté et inversement.

Exemple 3.21. Il y a 2 couples de concepts de CM4 (exemple 3.4) présentant un conflit d'ambiguïté en plus de celui de l'exemple 3.20 : (Pression atmosphérique, Camping) et (Pression atmosphérique, Incendie). L'ensemble des conflits d'ambiguïté de CM4 est alors :

$$\left\{ \begin{array}{l} \left\{ \begin{array}{l} (Orage, Foudre), (Foudre, Incendie) \\ (Orage, Pluie), (Pluie, Incendie) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Orage, Foudre), (Foudre, Incendie) \\ (Orage, Pluie), (Pluie, Camping), (Camping, Incendie) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Pression atmosphérique, Camping), \\ (Pression atmosphérique, Chaleur), (Chaleur, Camping) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Pression atmosphérique, Camping), (Camping, Incendie) \\ (Pression atmosphérique, Chaleur), \\ (Chaleur, Camping), (Camping, Incendie) \end{array} \right\} \end{array} \right\}$$

Chaque conflit représente ici un ensemble d'influences qui ne valident pas le critère d'ambiguïté. Cela signifie que, pour chaque conflit, au moins une influence est à corriger pour valider le critère. Un designer cherchant à corriger la carte doit donc corriger au moins une influence dans chacun des conflits présentés ici.

Grâce à la proposition 3.5, on peut également facilement montrer que l'ensemble des conflits de propreté d'une carte cognitive est inclus dans l'ensemble des conflits d'ambiguïté de cette même carte.

3.4.3 Conflits de cohérence et de compatibilité

Nous ne donnons pas de définition d'un conflit pour le critère de cohérence. En effet, un conflit de cohérence avec une contrainte apparaît dès que l'influence taxonomique du premier concept de la contrainte sur le second est différente de la valeur de cette contrainte. En appliquant le même raisonnement que pour la propreté et l'ambiguïté, on déduit à nouveau que les conflits de cohérence devraient être basés sur la notion d'influence. Les influences sont ici celles de tous les chemins qui lient entre eux les concepts élémentaires des concepts de la contrainte.

Pour savoir, parmi ces influences, lesquelles provoquent un conflit de cohérence, il faut chercher à comprendre pourquoi la valeur de l'influence taxonomique est

différente de la valeur de la contrainte. Les raisons provoquant cette différence dépendent de la valeur de la contrainte. Ainsi, une contrainte portant une valeur $+$ impose, d'après la définition de l'influence taxonomique, que l'influence propagée de chacun des couples de concepts élémentaires des concepts de la contrainte soit $+$. Il suffit donc qu'il existe au moins un couple de concepts élémentaires tel que leur influence propagée soit différente de $+$ pour le critère ne soit pas validé. Il est donc tout à fait possible que l'influence propagée entre les deux concepts soit de 0. Cela signifie qu'il n'y a aucun chemin permettant de lier le premier concept élémentaire à l'autre. Ce sont en principe les influences qui composent les chemins permettant de lier ces deux concepts élémentaires qui doivent définir le conflit de cohérence. C'est donc cette absence de chemin qui provoque un conflit. S'il n'y a aucun chemin permettant de lier les deux concepts, on ne peut définir aucune influence pour le conflit. Il existe donc certains cas où on ne peut fournir un conflit expliquant la non-validation du critère.

En raisonnant de la même manière pour les autres valeurs possibles de la contrainte, on se rend compte qu'il n'y a que pour le cas où la valeur de la contrainte est de 0 qu'il est possible de construire l'ensemble des conflits qui couvre toutes les raisons pour lesquelles le critère n'est pas validé. Pour cette raison, nous estimons qu'il n'est pas utile de fournir une définition complexe qui ne sera au mieux que partielle puisqu'elle ne permettra pas de construire des diagnostics permettant de corriger complètement la carte pour certaines valeurs de la contrainte.

Une analyse similaire sur le critère de compatibilité donne les mêmes résultats, même si cette fois-ci on peut également construire des ensembles de conflits convenables pour une valeur de contrainte de \oplus et \ominus .

3.4.4 Construction des diagnostics

Un diagnostic est un ensemble minimal de composants qui, s'ils sont supposés anormaux, expliquent le comportement anormal d'un système [Reiter, 1987]. Chaque conflit représente une preuve du fait que le critère n'a pas été validé. Tant que ces conflits ne seront pas corrigés, le critère ne sera pas validé. On comprend donc bien que, si un diagnostic présente plusieurs composants à corriger pour valider un critère, cela signifie qu'il doit contenir au moins un composant dans chacun des conflits.

Dans le cas de nos critères de qualité sur des cartes cognitives, nous avons expliqué que les composants à corriger étaient les influences entre les concepts, associées à leurs valeurs. Pour définir comment calculer ce diagnostic, nous introduisons différentes notions générales, applicables à n'importe quelle base de connaissances. Nous les appliquons ensuite aux cartes cognitives, notamment au critère de non-ambiguïté. Nous commençons par ce critère car, étant plus complexe que le critère de propreté, les différentes opérations successives à réaliser pour le calcul d'un diagnostic seront plus claires qu'avec le critère de propreté.

On considère qu'on dispose d'un ensemble de conflits. Dans le cadre du critère d'ambiguïté, c'est l'ensemble des conflits d'ambiguïté d'une carte cognitive (définition 3.26).

Avant de calculer un diagnostic, nous devons nous assurer que chaque conflit

de l'ensemble des conflits est minimal. Un conflit est *minimal* quand il ne peut pas être résolu en résolvant un autre conflit. Nous pouvons donc ignorer les conflits qui ne sont pas minimaux pour le calcul du diagnostic puisque résoudre les conflits minimaux permettront également de les résoudre.

L'ensemble des conflits minimaux d'un ensemble de conflits est donc l'ensemble de ces conflits qui sont minimaux.

Définition 3.27 (Ensemble des conflits minimaux)

Soit CC' un ensemble de conflits.

L'ensemble des conflits minimaux de CC' est un ensemble CC tel que :

$$CC = \{cc \in CC' / \forall cc' \in CC', cc' \not\subseteq cc\}$$

Exemple 3.22. On rappelle que l'ensemble des conflits d'ambiguïté calculé dans l'exemple 3.21 était :

$$\left\{ \begin{array}{l} \left\{ \begin{array}{l} (Orage, Foudre), (Foudre, Incendie) \\ (Orage, Pluie), (Pluie, Incendie) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Orage, Foudre), (Foudre, Incendie) \\ (Orage, Pluie), (Pluie, Camping), (Camping, Incendie) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Pression atmosphérique, Camping), \\ (Pression atmosphérique, Chaleur), (Chaleur, Camping) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Pression atmosphérique, Camping), (Camping, Incendie) \\ (Pression atmosphérique, Chaleur), \\ (Chaleur, Camping), (Camping, Incendie) \end{array} \right\} \end{array} \right\}$$

On remarque que l'avant-dernier conflit est inclus dans le dernier. En effet, on y retrouve les influences (Pression atmosphérique, Camping), (Pression atmosphérique, Chaleur) et (Chaleur, Camping). Le dernier conflit n'est donc pas minimal. Pour que cet ensemble des conflits soit minimal, il faut donc supprimer ce conflit.

Au final, l'ensemble des conflits minimaux d'ambiguïté de CM4 est :

$$\left\{ \begin{array}{l} \left\{ \begin{array}{l} (Orage, Foudre), (Foudre, Incendie) \\ (Orage, Pluie), (Pluie, Incendie) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Orage, Foudre), (Foudre, Incendie) \\ (Orage, Pluie), (Pluie, Camping), (Camping, Incendie) \end{array} \right\}, \\ \left\{ \begin{array}{l} (Pression atmosphérique, Camping), \\ (Pression atmosphérique, Chaleur), (Chaleur, Camping) \end{array} \right\} \end{array} \right\}$$

Un *diagnostic* fournit un ensemble de composants à corriger pour résoudre l'ensemble des conflits. Un diagnostic n'est pas forcément unique : certains diagnostics proposent de corriger un seul composant et d'autres d'en corriger plusieurs pour le même ensemble de conflits. Il n'est pas possible de savoir quel diagnostic est meilleur qu'un autre : seul le designer peut choisir parmi tous les diagnostics le meilleur à appliquer.

Avant de calculer un diagnostic, nous introduisons la notion d'ensemble explicatif. Un *ensemble explicatif* est un ensemble construit à partir de l'ensemble des conflits minimaux. La correction des éléments de l'ensemble explicatif permet de

résoudre l'intégralité des conflits. Tout comme la notion de diagnostic, il peut il y en avoir plusieurs pour le même ensemble de conflits. Un ensemble explicatif est défini à partir d'un ensemble de conflits minimaux tel qu'il a un élément en commun avec chaque conflit de l'ensemble des conflits.

Définition 3.28 (Ensemble explicatif)

Soit CC un ensemble de conflits minimaux. Soit $H \subseteq \bigcup_{cc \in CC} \bigcup_{o \in cc} \{o\}$ un ensemble de composants.

H est un ensemble explicatif pour CC ssi $\forall cc \in CC, H \cap cc \neq \emptyset$.

Exemple 3.23. Nous ne donnons pas l'intégralité des ensembles explicatifs possibles pour l'ensemble des conflits minimaux d'ambiguïté de l'exemple 3.22 car ceux-ci sont bien trop nombreux. Nous allons simplement en donner quelques uns :

$$\begin{aligned} & \{(Orage, Foudre), (Pression atmosphérique, Camping)\} \\ & \{(Orage, Foudre), (Foudre, Incendie), (Pression atmosphérique, Camping)\} \\ & \{(Pluie, Incendie), (Camping, Incendie), (Chaleur, Camping)\} \end{aligned}$$

Si on considère le premier ensemble explicatif, on voit que (Orage, Foudre) est présent dans les deux premiers conflits. Corriger cette influence permettrait donc de corriger ces deux conflits. En corrigeant également (Pression atmosphérique, Camping), on corrige également le dernier conflit. Au final, en appliquant cet ensemble explicatif en corrigeant les influences qui le composent, on corrige l'intégralité des conflits d'ambiguïté.

Un ensemble explicatif peut être recouvert par un autre ensemble explicatif : les éléments qu'il fournit sont alors tous inclus dans cet autre ensemble. Or, un ensemble explicatif est sensé fournir un ensemble de composants à corriger de manière à résoudre tous les conflits. Seule une partie des composants de l'autre ensemble explicatif suffit donc à résoudre tous les conflits. La notion de diagnostic applique le *principe de parcimonie*, c'est-à-dire que si la correction de certains composants suffit à corriger l'intégralité des conflits, il est inutile d'en considérer plus. Un diagnostic est donc un ensemble explicatif minimal. Un ensemble explicatif est *minimal* si aucun de ses sous-ensembles n'est un ensemble explicatif pour l'ensemble des conflits.

Définition 3.29 (Diagnostic)

Soit CC un ensemble de conflits. Soit H un ensemble explicatif pour CC .

H est un diagnostic pour CC ssi $\forall H' \subset H, H'$ n'est pas un ensemble explicatif pour CC .

Exemple 3.24. En analysant les ensembles explicatifs proposés dans l'exemple 3.23, on se rend compte que le deuxième ensemble explicatif n'est pas minimal. En effet, il inclut le premier ensemble explicatif. Ce ne peut donc être un diagnostic pour résoudre les conflits d'ambiguïté de la carte.

Nous n'allons pas lister l'intégralité des diagnostics car il y en a 15. Des 3 ensembles explicatifs de l'exemple 3.23, 2 sont minimaux et sont par conséquent des diagnostics :

$$\begin{aligned} & \{(Orage, Foudre), (Pression atmosphérique, Camping)\} \\ & \{(Pluie, Incendie), (Camping, Incendie), (Chaleur, Camping)\} \end{aligned}$$

L'utilisateur doit choisir le diagnostic qui lui semble le meilleur pour corriger la carte. Pour l'appliquer, pour chacune des influences du diagnostic, il doit soit la supprimer, soit inverser sa valeur.

La construction d'un diagnostic revient à calculer un *hitting set* sur l'ensemble des conflits [Reiter, 1987]. Le problème du *hitting set* a été démontré NP-complet [Karp, 1972]. Divers algorithmes permettent de calculer des diagnostics minimaux [de Kleer et Williams, 1987; Greiner *et al.*, 1989; Wotawa, 2001]. Nous pouvons sans difficulté appliquer un de ces algorithmes pour calculer les diagnostics minimaux pour corriger les conflits de propreté et d'ambiguïté.

Les définitions que nous avons données précédemment et qui permettent le calcul de diagnostics peuvent s'appliquer sans problème au critère de propreté en se basant sur l'ensemble des conflits de propreté d'une carte cognitive.

Exemple 3.25. *L'ensemble des conflits de propreté CM4 calculé dans l'exemple 3.18 était :*

$$\left\{ \left\{ \begin{array}{l} (\text{Pression atmosphérique, Chaleur}), (\text{Chaleur, Camping}), \\ (\text{Pression atmosphérique, Camping}) \end{array} \right\} \right\}$$

Tous les conflits de cet ensemble sont minimaux.

De cet ensemble, on calcule 3 diagnostics différents :

$$\begin{array}{c} \{(\text{Pression atmosphérique, Camping})\} \\ \{(\text{Pression atmosphérique, Chaleur})\} \\ \{(\text{Chaleur, Camping})\} \end{array}$$

Dans cet exemple, chaque diagnostic présente une influence unique. Supprimer l'une de ces influences ou inverser sa valeur permet de résoudre tous les conflits de propreté de la carte.

Conclusion

Nous avons présenté une formalisation de la validation des cartes cognitives pour laquelle nous avons introduit des critères de vérification tels que celui de propreté et de non-ambiguïté. Le modèle des cartes cognitives taxonomiques permet de tester la cohérence et la compatibilité d'une carte par rapport à une spécification de contraintes. Nous avons également fourni une définition d'un ensemble de conflits pour les critères de propreté et de non-ambiguïté. Ceci permet de corriger facilement une carte qui ne respecte pas le dit critère. De plus, nous avons proposé différentes solutions pour appliquer ces critères à des cartes cognitives définies sur d'autres ensembles de valeurs, notamment $[-1; 1]$.

Nous avons testé nos critères sur des cartes issues du monde de la pêche [Christiansen, 2011]. Les auteurs avaient défini des critères pour repérer des contradictions mais ceux-ci n'étaient pas spécifiquement dédiés au modèle des cartes cognitives. Nos critères ont permis de détecter entre autres les contradictions déjà repérées par les auteurs. D'autres contradictions non évoquées par l'auteur ont été

détectés par nos critères qui mériteraient elles aussi une discussion. De plus amples informations sont données en annexe A.

Notre approche permet d'indiquer si une carte ou parfois un couple de concepts respecte un critère de qualité ou non. Cette information peut avoir différentes applications. Elle permet d'une part de montrer à l'utilisateur quels points de la carte cognitive apportent le plus de controverse et donc sur quels points discuter lors d'une séance de brainstorming par exemple. Cette information permet d'autre part d'indiquer qu'une carte cognitive doit être corrigée afin justement de lever la controverse.



4

Synthèse de cartes cognitives

Introduction

Les cartes cognitives sont fréquemment utilisées lors de séances de brainstorming afin de prendre une décision de groupe. La construction elle-même des cartes cognitives est aussi souvent issue d'un travail de groupe. Cependant, il est souvent difficile pour un groupe de personnes de mettre en commun des connaissances et de concilier leurs différents points de vue. Il est plus facile pour chacun d'entre eux de construire sa propre carte cognitive représentant son point de vue sur un problème donné. Ces cartes sont ensuite mises en commun en vue de construire une carte unique représentant une synthèse des connaissances apportées par les différents intervenants. Cette synthèse se doit d'être concise de manière à être en accord avec les intérêts du modèle des cartes cognitives.

Ce chapitre propose une méthode automatique pour synthétiser une carte cognitive construite lors d'une séance de brainstorming entre différents intervenants. Chaque intervenant est le designer de sa propre carte. L'idée est d'utiliser une taxonomie pour lier entre eux les concepts utilisés par les différents designers. La taxonomie de concepts est fournie aux designers avant la construction de leur carte afin qu'elle puisse servir de vocabulaire, en listant les différents concepts qu'ils peuvent utiliser. On appelle une telle carte associée à une taxonomie une *carte attribuée*. Avant la construction des différentes cartes, il est également important d'établir des conventions entre les différents designers concernant la sémantique des valeurs d'influences et des concepts, afin d'obtenir un ensemble de cartes cohérent. Par exemple, il est judicieux d'utiliser le même ensemble de valeurs pour toutes les cartes. On appelle un tel ensemble de cartes respectant ces conditions un *ensemble univoque de cartes attribuées*, dans lequel chaque intervenant est le designer d'une et d'une seule carte.

Pour concilier les différents points de vue, des *préférences* [Kaci, 2009] doivent être établies sur les différents designers de l'ensemble univoque de cartes. Ces pré-

férences permettent de favoriser les connaissances apportées par certains designers par rapport à celles apportées par d'autres. Par exemple, lorsqu'une influence entre deux concepts porte une valeur différente dans deux cartes attribuées, la valeur que porte l'influence dans la carte attribuée dont le designer est préféré aux autres sera privilégiée. Grâce à ces informations, on est en mesure de construire étape par étape une carte cognitive unique synthétisant les connaissances apportées par les différents designers, la *carte synthétisée*.

La construction de la carte synthétisée s'effectue en trois étapes successives, chacune étant dédiée à un type d'information de la carte cognitive. La première est la sélection des concepts qui seront présents dans cette carte. Cet ensemble de concepts est appelé *ensemble des concepts synthétisés*. La seconde étape consiste à déterminer les influences entre les concepts, c'est-à-dire à construire l'ensemble des influences qui apparaîtront dans la carte synthétisée. On appelle cet ensemble d'influences *ensemble des influences synthétisées*. Ces deux étapes suffisent à obtenir une structure de carte qui peut servir de base pour un travail de groupe visant à donner une valeur à chacune des influences. Cette carte n'ayant aucune valeur, on l'appelle *carte synthétisée non-valorisée*. Il est aussi possible de donner une valeur à chacune des influences de manière automatique pour obtenir une *carte synthétisée valorisée*. Pour chaque influence synthétisée, on calcule ainsi sa *valeur synthétisée*, qui constitue la troisième étape de la construction d'une carte synthétisée.

À noter que Denis Poignonec [Poignonec, 2006] propose également une méthode pour la fusion de cartes cognitives. Cette thèse semi-formelle de sociologie utilise un dictionnaire hiérarchique des données pour composer les cartes à fusionner. Une carte unique extrêmement détaillée est construite sur les notions élémentaires de ce dictionnaire. Il est possible de sélectionner quels concepts seront présents dans la carte finale en sélectionnant un niveau dans le dictionnaire hiérarchique, à la manière d'une vue (section 2.3). Le travail de Gwen Christiansen [Christiansen, 2011] est basé sur la représentation sous forme de connaissances ontologiques des cartes cognitives. En mettant en avant les informations communes à plusieurs cartes, on est capable d'extraire un consensus parmi les différentes opinions représentées par les différentes cartes. Les mécanismes proposés sont cependant assez peu paramétrables et n'utilisent pas la notion de préférences.

La section 4.1 introduit les notions de carte attribuée, d'ensemble univoque et de préférences sur un ensemble de designers. La section 4.2 détaille la construction étape par étape de la carte cognitive synthétisée. Cette carte cognitive synthétisée étant une carte cognitive comme une autre, nous pouvons en construire une vue. Nous montrons comment adapter la taxonomie ayant servi à sa construction pour définir une échelle dans la section 4.3. Les définitions sont valables pour une carte cognitive définie sur l'ensemble de valeurs $[-1; 1]$. Nous discutons de la manière d'adapter cette démarche à des cartes cognitives définies sur d'autres ensembles de valeurs ainsi que d'autres opérateurs de synthèse, selon les besoins des utilisateurs dans la section 4.4.

4.1 Cartes attribuées et préférences

Dans un groupe de travail, chaque intervenant est le *designer* d'une carte, appelée *carte attribuée*. En considérant l'ensemble des cartes produites par les différents designers, on construit un *ensemble univoque de cartes attribuées*. Sur cet ensemble univoque, on définit des *préférences* sur ces designers.

Dans la section 4.1.1, nous définissons les notions de carte attribuée et d'ensemble univoque. Dans la section 4.1.2, nous définissons ce que sont les préférences et comment les utiliser.

4.1.1 Cartes attribuées

Dans nos définitions, un *designer* est un objet qui représente l'intervenant qui est à l'origine de la conception d'une carte. Nous le représentons simplement par un bref texte.

Définition 4.1 (Designer)

Un designer d est une étiquette linguistique identifiant le concepteur d'une carte.

Chaque designer produit une *carte attribuée*. Cette carte est l'association d'une carte cognitive classique et d'une taxonomie. Cependant, cette carte n'est pas une carte cognitive taxonomique comme définie au chapitre 2. En effet, une carte taxonomique est définie sur les concepts élémentaires d'une taxonomie. La taxonomie sert de vocabulaire et ce vocabulaire est partagé entre les différents designers. Certains de ces designers utilisent des concepts généraux tandis que d'autres utilisent des concepts spécialisés en tant qu'experts dans le domaine représenté par le concept. Les concepts utilisés dans les différentes cartes attribuées ne sont donc pas nécessairement élémentaires. Nous interdisons en revanche d'utiliser deux concepts comparables dans une même carte attribuée. En effet, nous considérons que représenter à la fois un concept et un concept plus général apporte une redondance d'information qui risque d'amener à une incohérence. Chaque carte attribuée est également associée à un designer.

Définition 4.2 (Carte attribuée)

Soit d un designer. Soit $T = (C, \leq)$ une taxonomie. Soit $CM = (C', A, \text{label})$ une carte cognitive.

Le triplet $CA = (d, T, CM)$ représente une carte attribuée construite par d si :

- $C' \subseteq C$;
- $\forall c_1, c_2 \in C', c_1 \parallel c_2$.

Exemple 4.1. *La carte attribuée de la figure 4.1 est une carte cognitive attribuée construite par le designer Préfet. Le vocabulaire de cette carte est celui de la taxonomie de la figure 4.2.*

Pour réaliser une synthèse de cartes cognitives, on doit disposer d'un ensemble de cartes attribuées. Pour que cet ensemble soit facilement manipulable, on doit

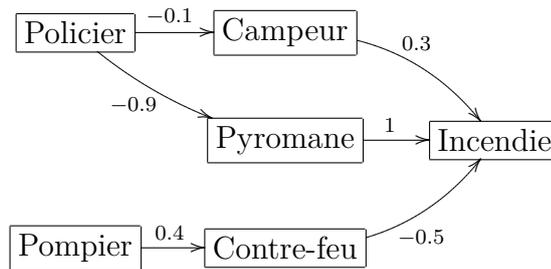


FIGURE 4.1 – La carte cognitive attribuée CA_P construite par le designer *Préfet* définie sur l'ensemble de valeurs $[-1; 1]$.

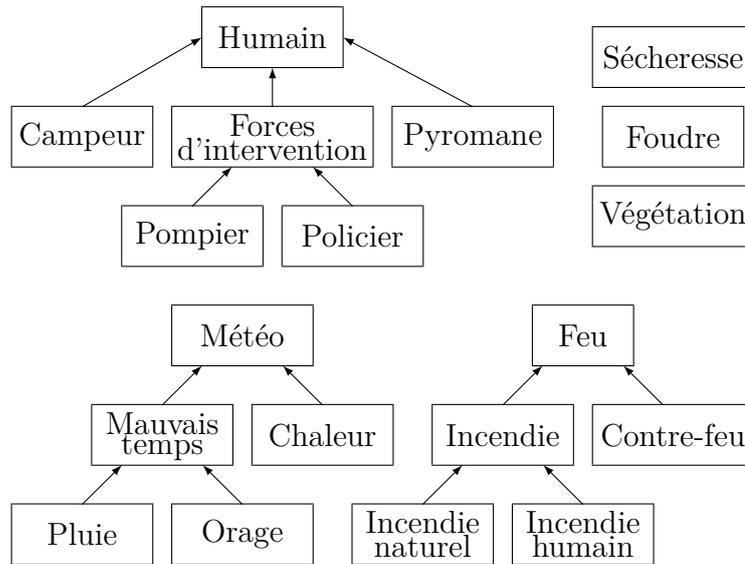


FIGURE 4.2 – La taxonomie T_2 .

s'assurer que l'ensemble est *univoque*, c'est-à-dire que toutes les cartes utilisent le même vocabulaire et les mêmes valeurs et que chaque designer a produit une seule carte.

Définition 4.3 (Ensemble univoque de cartes attribuées)

Soit U un ensemble de cartes attribuées. Soient T une taxonomie et I un ensemble de valeurs.

U est un ensemble univoque de cartes attribuées défini sur T et I ssi :

- $\forall (d, T', CM) \in U, T' = T$ et CM est définie sur I ;
- $\forall (d_1, T_1, CM_1), (d_2, T_2, CM_2) \in U, d_1 = d_2 \Rightarrow CM_1 = CM_2$.

Exemple 4.2. L'ensemble de cartes U (figure 4.3) est un ensemble univoque de cartes attribuées. Toutes ces cartes sont définies sur la taxonomie T_2 (figure 4.2) et le même ensemble de valeurs $[-1; 1]$. Ces cartes représentent les connaissances de différents experts sur les différents facteurs pouvant déclencher un incendie de forêt, selon leurs domaines d'expertise.

On peut voir que les différents designers utilisent différents niveaux de spécialisation pour les concepts. Ainsi, le *Préfet* qui est gère l'intervention de différents

décision. Il existe de nombreuses manières de représenter des préférences sur des objets. Nous nous limitons ici à une simple relation d'ordre sur ces objets.

Formellement, des préférences sont représentées par un pré-ordre sur un ensemble.

Définition 4.5 (Préférences)

Soit E un ensemble d'objets.

Des préférences sur E sont représentées par une relation de pré-ordre total \succeq sur les éléments de E .

Soient $e_1, e_2 \in E$ deux éléments de E .

On note \succ la relation de pré-ordre strict associée à \succeq , i.e. $(e_1 \succ e_2) \Leftrightarrow (e_1 \succeq e_2) \wedge \neg(e_2 \succeq e_1)$. On dit que e_1 est préféré à e_2 ssi $e_1 \succ e_2$.

Exemple 4.4. On considère le pré-ordre total \succeq suivant sur l'ensemble \mathcal{D}_U des designers de U (exemple 4.4) :

$$\begin{aligned} \text{Préfet} &\succeq \text{Garde-forestier} \\ \text{Préfet} &\succeq \text{Agent de tourisme} \\ \text{Préfet} &\succeq \text{Météorologue} \\ \text{Garde-forestier} &\succeq \text{Agent de tourisme} \\ \text{Garde-forestier} &\succeq \text{Météorologue} \\ \text{Agent de tourisme} &\succeq \text{Garde-forestier} \\ \text{Agent de tourisme} &\succeq \text{Météorologue} \end{aligned}$$

Garde-forestier est préféré à Météorologue et Préfet est préféré à tous les autres designers.

Deux objets sont *équitablement préférés* si aucun n'est préféré à l'autre.

Définition 4.6 (Objets équitablement préférés)

Soit E un ensemble d'objets. Soit \succeq des préférences sur E . Soient $e_1, e_2 \in E$ deux éléments de E .

e_1 est équitablement préféré à e_2 , noté $e_1 \approx e_2$ ssi $(e_1 \succeq e_2) \wedge (e_2 \succeq e_1)$.

Exemple 4.5. Avec les préférences \succeq (exemple 4.4), on a $\text{Garde-forestier} \approx \text{Agent de tourisme}$ puisque $\text{Garde-forestier} \succeq \text{Agent de tourisme}$ et $\text{Agent de tourisme} \succeq \text{Garde-forestier}$.

On peut représenter une relation de pré-ordre sous la forme d'une *stratification d'un ensemble*. Une stratification d'un ensemble selon une relation de pré-ordre total est une partition de cet ensemble. Elle est représentée par un tuple pour lequel chaque élément d'une composante du tuple est plus grand que chaque élément de la composante suivante selon la relation d'ordre. Cette stratification est unique et est donc équivalente à la relation de pré-ordre.

Définition 4.7 (Stratification d'un ensemble selon une relation de pré-ordre)

Soit E un ensemble d'objets. Soit \succeq une relation de pré-ordre total sur les éléments de E .

La stratification $\mathcal{S}_{E, \succeq}$ de E selon \succeq est un tuple $\langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$, tel que, $\forall i \in [1; m]$:

- $\mathcal{S}_i \neq \emptyset$;
- $\mathcal{S}_i = \{e \in (E - \bigcup_{k=1}^{i-1} \mathcal{S}_k) / \forall e' \in (E - \bigcup_{k=1}^{i-1} \mathcal{S}_k), e \succeq e'\}$.

La première strate de cette stratification est l'ensemble des objets qui sont tous préférés aux autres et donc les objets les plus grands selon la relation de pré-ordre.

Exemple 4.6. *La stratification de l'ensemble des designers de U (exemple 4.2) selon le pré-ordre \succeq (exemple 4.4) est un triplet $\langle \mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3 \rangle$ tel que :*

$$\begin{aligned}\mathcal{S}_1 &= \{\text{Préfet}\} \\ \mathcal{S}_2 &= \{\text{Garde-forestier, Agent de tourisme}\} \\ \mathcal{S}_3 &= \{\text{Météorologue}\}\end{aligned}$$

On peut donc écrire plus simplement le pré-ordre \succeq comment étant :

$$\text{Préfet} \succeq \text{Garde-forestier} \approx \text{Agent de tourisme} \succeq \text{Météorologue}$$

Une stratification étant unique, on peut attribuer à chaque élément de l'ensemble stratifié un *indice* représentant sa place dans la relation de pré-ordre par rapport aux autres éléments. On définit cet indice de telle manière qu'un indice élevé correspond aux objets les plus grands de l'ordre et un indice bas aux objets les plus petits.

Définition 4.8 (Indice d'un élément d'un ensemble selon une relation de pré-ordre)

Soit E un ensemble d'objets. Soit \succeq une relation de pré-ordre total sur les éléments de E . Soit $\mathcal{S}_{E,\succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ la stratification de E selon \succeq . Soit $e \in E$ un élément de E .

On sait que $\exists i \in [1; m] / e \in \mathcal{S}_m$. On définit l'indice de e dans E selon \succeq comme étant :

$$\mathcal{I}_{E,\succeq}(e) = m - i + 1$$

Exemple 4.7. *Avec l'ensemble des designers de U (exemple 4.2) selon le pré-ordre \succeq (exemple 4.4), les indices des designers sont :*

$$\begin{aligned}\mathcal{I}_{U,\succeq}(\text{Préfet}) &= 3 & \mathcal{I}_{U,\succeq}(\text{Garde-forestier}) &= 2 \\ \mathcal{I}_{U,\succeq}(\text{Agent de tourisme}) &= 2 & \mathcal{I}_{U,\succeq}(\text{Météorologue}) &= 1\end{aligned}$$

4.2 Opérateurs de synthèse

On dispose à présent d'un ensemble de cartes construites par des designers différents toutes définies sur la même taxonomie et le même ensemble de valeurs ainsi que de préférences sur les designers de ces cartes. Toutes les informations nécessaires sont disponibles pour réaliser une synthèse de cet ensemble de cartes. Cette synthèse s'effectue en 3 étapes :

1. la construction de l'*ensemble des concepts synthétisés* : les concepts qui apparaîtront dans la carte synthétisée;

2. la construction de l'ensemble des influences synthétisées : les influences qui apparaîtront dans la carte synthétisée, entre les concepts synthétisés ;
3. le calcul d'une valeur d'influence synthétisée pour chaque influence synthétisée : la valeur de cette influence dans la carte synthétisée.

À chacune de ces étapes correspond un opérateur de synthèse. L'étape 3 est optionnelle si on souhaite uniquement obtenir une structure de carte, amenant à une discussion sur les valeurs des influences de cette carte.

Dans la section 4.2.1, nous proposons une manière de synthétiser les concepts. Dans la section 4.2.2, nous proposons une manière de synthétiser les influences. Enfin, dans la section 4.2.3, nous proposons une manière de calculer les valeurs synthétisées.

4.2.1 Concepts synthétisés

Les *concepts synthétisés* sont les concepts qui apparaîtront dans la carte synthétisée. Comme expliqué précédemment, nous considérons que la présence dans une même carte de deux concepts comparables apporte une redondance d'information. Par conséquent, on ne peut définir l'ensemble des concepts synthétisés comme étant simplement l'union des concepts utilisés par les différents designers, ceux-ci risquant fort de ne pas être incomparables. De plus, la carte synthétisée est sensée représenter une synthèse de l'information apportée par les différents designers. Cette information doit donc être concise, sinon le modèle des cartes cognitives perdrait un de ses principaux intérêts.

Nous considérons donc qu'un concept fait partie de l'ensemble des concepts synthétisés s'il est présent dans la carte d'un designer et qu'il n'existe pas de concept qui lui comparable présent dans la carte d'un autre designer qui lui est préféré. En considérant les préférences sur les designers, la construction de l'ensemble de concepts synthétisés se fait strate par strate.

Lorsqu'un concept est présent dans la carte d'un designer, on dit que ce designer *utilise* ce concept. Pour un ensemble univoque de cartes attribuées, les *concepts utilisés par un ensemble de designers* sont les concepts utilisés par au moins un des designers de cet ensemble dans sa carte.

Définition 4.9 (Concepts utilisés par un ensemble de designers)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soit $D \subseteq \mathcal{D}_U$ un ensemble de designers de cartes de U .

L'ensemble des concepts utilisés par D est :

$$\text{conceptsUtilisésPar}_U(D) = \bigcup_{d \in D} C' \text{ avec } (d, T, (C', A, \text{label})) \in U$$

Exemple 4.8. En reprenant l'ensemble univoque de cartes attribuées U (exemple 4.2), les concepts utilisés par l'ensemble de designers {Garde-forestier, Agent de tourisme} sont :

$$\left\{ \begin{array}{l} \text{Campeur, Pompier, Policier, Foudre, Sécheresse, Végétation,} \\ \text{Mauvais temps, Chaleur, Incendie, Incendie naturel, Incendie humain} \end{array} \right\}$$

La synthèse des concepts s'effectue en raisonnant strate par strate en suivant l'ordre des préférences. On construit ainsi, pour chaque strate de l'ensemble des designers basée sur les préférences, un ensemble de concepts correspondant aux concepts utilisés par les designers de cette strate. L'idée est de réaliser l'union de ces ensembles. Cependant, nous devons nous assurer que cette union est incomparable. Nous écartons les concepts comparables avec ceux utilisés par un designer préféré. Ceci nous assure d'avoir un ensemble de concepts incomparables et que les informations apportées par le designer préféré sont privilégiées. Les concepts utilisés par les designers de cette strate doivent donc être incomparables avec ceux de tous les ensembles correspondant à des strates préférées. Avant d'effectuer l'union, nous devons donc nous assurer, pour chaque strate, qu'aucun concept utilisé par un designer de cette strate n'est comparable avec les concepts des strates préférées.

Néanmoins, lors de la construction de l'ensemble des concepts utilisés par les designers d'une strate, il se peut que deux concepts soient comparables entre eux, chacun d'entre eux ayant été choisi par deux designers appartenant à cette strate. Si le cas se présente, il ne faut choisir qu'un seul de ces deux concepts car les concepts de ces ensembles doivent être incomparables. Les préférences ne peuvent permettre de trancher ici, puisque les deux designers appartenant à la même strate, ils sont équitablement préférés. Entre ces deux concepts, choisir le plus petit permet de s'assurer que les influences auxquelles est lié le concept le plus grand sont toujours valables si on les applique au plus petit. En effet, selon le principe de substitution de Liskov [Liskov et Wing, 1999], si une propriété est vraie pour tout objet d'un type, alors elle doit aussi être vraie pour tout objet de n'importe quel sous-type de ce type. La relation d'ordonnement de la taxonomie étant une relation *est une sorte de*, elle entre dans le cadre de ce principe. C'est pourquoi nous ne conservons que les concepts minimaux de chacun des ensembles de concepts par strate.

Définition 4.10 (Concepts synthétisés d'une strate)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit la stratification $\mathcal{S}_{\mathcal{D}_U, \succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ de \mathcal{D}_U selon \succeq .

$\forall i \in [1; m]$, l'ensemble des concepts synthétisés de la strate \mathcal{S}_i est :

$$\mathcal{CS}_{U, \succeq}(\mathcal{S}_i) = \min \left(\left\{ c \in \text{conceptsUtilisésPar}_U(\mathcal{S}_i) / \forall c' \in \bigcup_{k=1}^{i-1} \mathcal{CS}_{U, \succeq}(\mathcal{S}_k), c \parallel c' \right\} \right)$$

Exemple 4.9. On reprend l'ensemble univoque de cartes attribuées U (exemple 4.2) et les préférences \succeq (exemple 4.4). Les concepts synthétisés de la strate $\mathcal{S}_1 = \{\text{Préfet}\}$ (exemple 4.6) sont :

$$\mathcal{CS}_{U, \succeq}(\mathcal{S}_1) = \{\text{Campeur, Pompier, Policier, Pyromane, Incendie, Contre-feu}\}$$

Les concepts synthétisés de la strate $\mathcal{S}_2 = \{\text{Garde-forestier, Agent de tourisme}\}$ sont :

$$\mathcal{CS}_{U, \succeq}(\mathcal{S}_2) = \{\text{Foudre, Sécheresse, Végétation, Mauvais temps, Chaleur}\}$$

On note que par rapport à l'exemple 4.8, les concepts *Campeur, Pompier, Policier* et *Incendie* ne sont pas présents puisqu'ils le sont déjà dans $\mathcal{CS}_{U, \succeq}(\mathcal{S}_1)$. En ce qui

concerne les concepts *Incendie naturel* et *Incendie humain*, ils ne sont pas présents puisqu'ils sont comparables avec *Incendie*, qui lui est présent dans $\mathcal{CS}_{U,\succeq}(\mathcal{S}_1)$. Les concepts synthétisés de la strate $\mathcal{S}_3 = \{\text{Météorologue}\}$ sont :

$$\mathcal{CS}_{U,\succeq}(\mathcal{S}_3) = \{\}$$

Cet ensemble est vide car chacun des concepts utilisés par le Météorologue est soit déjà présent dans les concepts synthétisés d'autres strates, soit comparable avec certains de ces concepts.

De là, on définit l'ensemble des concepts synthétisés d'un ensemble univoque de cartes attribuées selon des préférences sur les designers de cet ensemble comme étant l'union des concepts synthétisés des strates de la stratification de l'ensemble des designers selon ces préférences.

Définition 4.11 (Concepts synthétisés)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit la stratification $\mathcal{S}_{\mathcal{D}_U,\succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ de \mathcal{D}_U selon \succeq .

L'ensemble des concepts synthétisés de U selon \succeq est :

$$\mathcal{CS}_{U,\succeq} = \bigcup_{i=1}^m \mathcal{CS}_{U,\succeq}(\mathcal{S}_i)$$

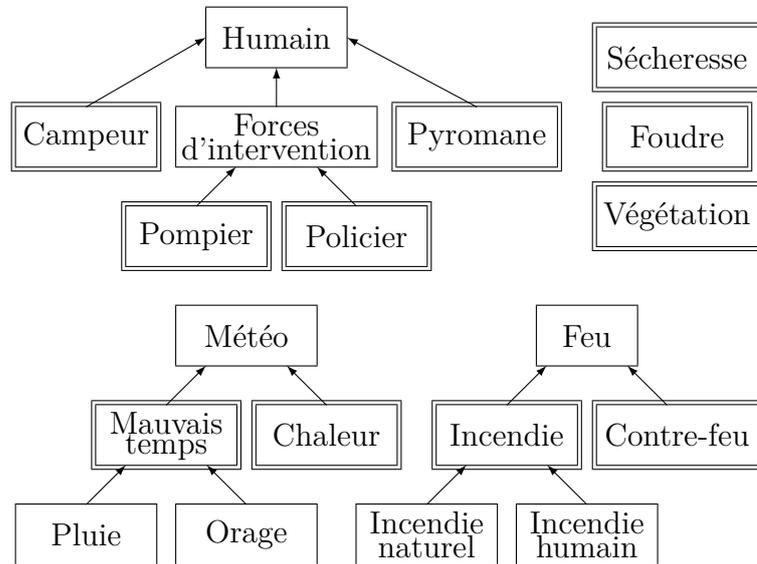


FIGURE 4.4 – L'ensemble des concepts synthétisés de U selon \succeq .

Exemple 4.10. L'ensemble des concepts synthétisés de U (exemple 4.2) selon les préférences \succeq (exemple 4.4) est :

$$\left\{ \begin{array}{l} \text{Campieur, Pompier, Policier, Pyromane, Sécheresse, Foudre,} \\ \text{Végétation, Mauvais temps, Chaleur, Incendie, Contre-feu} \end{array} \right\}$$

Ils sont représentés par un double encadrement dans la figure 4.4. On note qu'ils sont bien incomparables.

Dans cet ensemble de concepts, on voit que les concepts utilisés par le Préfet, le designer préféré, sont tous présents. En revanche, les concepts du Météorologue, le designer le moins préféré, sont souvent représentés par des concepts plus généraux.

Cette définition de l'ensemble de concepts synthétisés permet de nous assurer que chaque concept utilisé par un designer a au moins un représentant dans cet ensemble. On rappelle qu'un concept *représente* un autre concept s'il lui est comparable, comme expliqué dans la section 2.3.1.

4.2.2 Influences synthétisées

Les *influences synthétisées* sont les influences qui apparaîtront dans la carte synthétisée entre les concepts synthétisés. Synthétiser les influences signifie donc choisir les arcs qui seront présents dans la carte finale. Bien évidemment, ces influences ne peuvent exister qu'entre les concepts précédemment sélectionnés pour être dans cette carte.

Pour construire cet ensemble, on cherche à construire des influences qui *représentent* les influences des cartes originales dans la carte synthétisée. Construire l'ensemble des influences synthétisées revient ensuite simplement à réaliser l'union des influences représentantes.

Pour définir cette représentation, on utilise la notion de représentation pour les concepts. Nous considérons que toutes les influences placées par un designer dans sa carte doivent chacune être représentée par une influence la représentant dans la carte synthétisée. Un tel procédé permet de s'assurer qu'aucune information n'est perdue. On rappelle qu'un objectif de la carte synthétisée est la concision de l'information. Le fait de garder toutes les influences semble en désaccord avec cet objectif; cependant, nous estimons que le travail de concision a déjà été effectué lors du choix des concepts qui seront présents dans la carte synthétisée.

On dit qu'une influence synthétisée *représente* une influence originale si ses extrémités sont comparables avec celles de l'influence originale. On identifie ici une influence synthétisée par un couple de concepts.

Définition 4.12 (Couple de concepts représentant une influence)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soit $(d, T, (C', A, \text{label})) \in U$ une carte attribuée. Soient $c_1, c_2 \in C$ deux concepts. Soit $(c'_1, c'_2) \in A$ une influence.

On dit que (c_1, c_2) représente (c'_1, c'_2) ssi $c_1 \perp c'_1 \wedge c_2 \perp c'_2$.

Exemple 4.11. On considère à nouveau l'ensemble univoque de cartes attribuées U (exemple 4.2) et les préférences \succeq (exemple 4.4) et plus particulièrement la carte du Météorologue CA_M . Le couple de concepts (Mauvais temps, Foudre) représente l'influence (Orage, Foudre) dans cette carte.

L'ensemble des influences synthétisées est simplement l'union des influences représentant des influences originales.

Définition 4.13 (Influences synthétisées)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U .

Les influences synthétisées de U selon \succeq sont les couples de concepts de l'ensemble :

$$\mathcal{IS}_{U, \succeq} = \{(c_1, c_2) \mid c_1, c_2 \in \mathcal{CS}_{U, \succeq} \\ \wedge \exists (d, T, (C', A, \text{label})) \in U, \exists (c'_1, c'_2) \in A, \\ (c_1, c_2) \text{ représente } (c'_1, c'_2)\}$$

Exemple 4.12. On considère à nouveau l'ensemble univoque de cartes attribuées U (exemple 4.2) et les préférences \succeq (exemple 4.4). L'ensemble des influences synthétisées issues du Préfet sont :

$$\left\{ \begin{array}{l} (\text{Policier, Campeur}), (\text{Campeur, Incendie}), (\text{Policier, Pyromane}), \\ (\text{Pyromane, Incendie}), (\text{Pompier, Contre-feu}) \end{array} \right\}$$

Étant donné que le Préfet est le designer préféré selon l'ordre, l'ensemble des concepts qu'il a utilisés sont présents dans l'ensemble de concepts synthétisés (exemple 4.10). Par conséquent, toutes les influences qu'il a placées sont présentes dans l'ensemble d'influences synthétisées avec leurs extrémités originales.

L'ensemble des influences synthétisées issues du Garde-forestier sont :

$$\left\{ \begin{array}{l} (\text{Foudre, Incendie}), (\text{Végétation, Incendie}), (\text{Pompier, Incendie}), \\ (\text{Sécheresse, Incendie}), (\text{Campeur, Incendie}) \end{array} \right\}$$

Ici, toute influence dont le concept-effet était Incendie naturel ou Incendie humain a été réaffectée au concept Incendie.

L'ensemble des influences synthétisées issues de l'Agent de tourisme sont :

$$\left\{ \begin{array}{l} (\text{Végétation, Campeur}), (\text{Policier, Campeur}), (\text{Incendie, Campeur}), \\ (\text{Mauvais temps, Campeur}), (\text{Chaleur, Campeur}) \end{array} \right\}$$

Ici, toutes les influences sont représentées par elles-mêmes car, bien que l'Agent de tourisme ne soit pas un des designers préférés, les concepts qu'il utilise font tous partie de l'ensemble des concepts synthétisés.

Enfin, l'ensemble des influences synthétisées issues du Météorologue sont :

$$\left\{ \begin{array}{l} (\text{Mauvais temps, Foudre}), (\text{Mauvais temps, Mauvais temps}), \\ (\text{Mauvais temps, Pompier}), (\text{Mauvais temps, Policier}), \\ (\text{Mauvais temps, Sécheresse}), (\text{Chaleur, Sécheresse}) \end{array} \right\}$$

Ici, les influences liées à Orage ou Pluie ont été réaffectées à Mauvais temps. Les influences étant liées à Forces d'intervention ont été réaffectées à Policier et à Pompier. Par conséquent, une même influence du Météorologue est représentée par deux influences différentes dans l'ensemble des influences synthétisées.

Sur l'ensemble des influences synthétisées, on note que l'influence (Campeur, Incendie) représente une influence de la carte du Préfet et une influence de la carte du Garde-forestier. De la même manière l'influence (Policier, Campeur) représente une influence de la carte du Préfet et une influence de la carte de l'Agent de tourisme.

On dispose à présent des concepts et des influences de la carte synthétisée. En soi, il y a suffisamment d'information pour construire une ébauche de carte synthétisée. Cette carte est composée uniquement de concepts et d'influences non valuées. Une telle carte a pour but d'amener à une discussion entre les différents intervenants sur la valeur des influences.

Une *carte synthétisée* est une carte cognitive construite à partir d'un ensemble univoque de cartes attribuées et de préférences sur les designers de ces cartes. Pour l'instant, nous définissons une carte dont aucune influence n'a de valeur. Or, selon notre définition formelle du modèle des cartes cognitives (définition 1.6), chaque influence d'une carte doit être associée à une valeur. C'est pourquoi nous introduisons une nouvelle valeur qui représente une *valeur non-renseignée*, notée \emptyset .

Définition 4.14 (Carte synthétisée non valuée)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit $I = \{\emptyset\}$ un ensemble de valeurs contenant uniquement la valeur non-renseignée.

Une carte synthétisée non-valuée est une carte cognitive $CS_{U, \succeq} = (CS_{U, \succeq}, \mathcal{IS}_{U, \succeq}, \text{label})$ définie sur I et construite à partir de U selon \succeq avec :

$$\begin{aligned} \text{label: } \mathcal{IS}_{U, \succeq} &\mapsto I \\ \text{label}((c_1, c_2)) &= \emptyset \end{aligned}$$

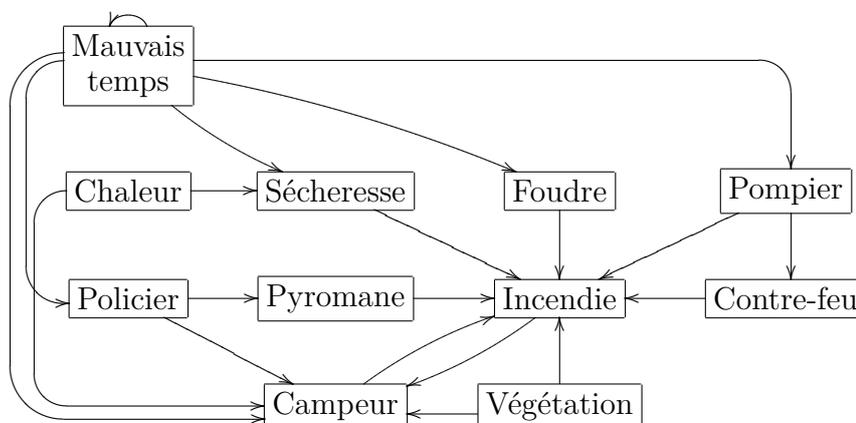


FIGURE 4.5 – Carte synthétisée non-valuée construite à partir de U selon \succeq .

Exemple 4.13. La carte cognitive de la figure 4.5 est une carte synthétisée non-valuée construite à partir de l'ensemble univoque de cartes attribuées U (exemple 4.2) et des préférences \succeq (exemple 4.4). La valeur \emptyset n'est pas représentée sur la figure par souci de concision.

Une fois ce graphe obtenu, on a un premier résultat qui met en lumière les influences existant entre les différents concepts (sans en évaluer la force). Les cartes cognitives aidant la prise de décision collective, on peut s'arrêter ici et proposer cette carte à tous les designers soit comme base de discussion collective, soit afin de leur permettre de proposer de nouvelles valeurs évaluant les influences. Néanmoins, on peut proposer une solution pour obtenir une carte synthétisée contenant les valeurs pour les influences. C'est ce qui est fait dans la section suivante.

4.2.3 Valeurs synthétisées

Les *valeurs synthétisées* sont les valeurs étiquetant les influences de la carte synthétisée. La synthèse des valeurs est dépendante de l'ensemble de valeurs utilisé dans les cartes. En effet, elle nécessite de manipuler ces valeurs et de les combiner entre elles. Ainsi, on ne peut pas proposer un opérateur d'agrégation des valeurs irréfutable, valable pour n'importe quel ensemble de valeurs. Néanmoins, pour pouvoir raisonner, on est obligé de combiner les différentes valeurs. Pour l'heure, nous nous intéressons uniquement aux ensembles univoques de cartes attribuées définis sur l'ensemble de valeurs $[-1; 1]$. La latitude inhérente aux solutions d'agrégation des valeurs se retrouve dans les possibilités de paramétrage expliquées en section 4.4.

On doit calculer une valeur synthétisée pour chaque influence synthétisée. Pour cela, on a besoin de diverses informations. Une influence synthétisée lie deux concepts, l'un étant le concept-cause et l'autre le concept-effet. Chaque influence synthétisée représente une ou plusieurs influences dans les cartes attribuées originales. Chacune de ces influences originales porte une valeur et a été construite par un designer. La synthèse des valeurs doit donc utiliser toutes ces informations, tout en utilisant les préférences sur les designers, pour calculer une valeur unique.

Avant toute chose, il faut donc construire, pour chaque influence synthétisée, l'ensemble de ces informations. Pour chaque influence originale représentée par l'influence synthétisée considérée, on retient son designer, son concept-cause, son concept-effet ainsi que sa valeur. On appelle un tel regroupement d'information une *valeur d'influence extraite*. L'opérateur de synthèse des valeurs d'influence utilise alors les informations de cet ensemble pour calculer une valeur.

Définition 4.15 (Valeurs d'influences extraites)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit $(c_1, c_2) \in \mathcal{IS}_{U, \succeq}$ une influence synthétisée.

Les valeurs d'influences extraites de (c_1, c_2) sont un ensemble de quadruplets tels que :

$$\text{extract}((c_1, c_2)) = \left\{ \left(d, c'_1, c'_2, \text{label}((c'_1, c'_2)) \right) / \left(d, T, (C, A, \text{label}) \right) \in U \right. \\ \left. \wedge (c_1, c_2) \in A \text{ représente } (c'_1, c'_2) \right\}$$

Exemple 4.14. On considère les influences synthétisées de l'exemple 4.12 et plus particulièrement l'influence (Campeur, Incendie). Elle représente l'influence (Campeur, Incendie) du Préfet ayant une valeur de 0.3 et l'influence (Campeur, Incendie humain) du Garde-forestier ayant une valeur de 0.2. Les valeurs d'influences extraites de (Campeur, Incendie) sont donc :

$$\left\{ \begin{array}{l} (\text{Préfet}, \text{Campeur}, \text{Incendie}, 0.3), \\ (\text{Garde-forestier}, \text{Campeur}, \text{Incendie humain}, 0.2) \end{array} \right\}$$

Nous avons ici choisi de définir la valeur synthétisée d'une influence comme étant la moyenne pondérée des valeurs de cet ensemble. La pondération d'une valeur est basée sur l'indice du designer dans les préférences. C'est une manière

simple de prendre en compte les points de vue de tous les concepteurs tout en privilégiant ceux des designers préférés. De plus, cette manière de procéder est en accord avec la logique des pénalités [Dupin de Saint-Cyr, 1996] dans laquelle le choix de l'information à considérer dépend de son importance mais aussi du nombre de fois où elle apparaît. Ainsi, une valeur d'influence placée par un designer préféré peut être supplantée par les valeurs d'influences placées par les autres designers si celles-ci sont assez nombreuses.

Définition 4.16 (Valeur synthétisée d'une influence)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T et sur l'ensemble de valeurs $I = [-1; 1]$. Soient \succeq des préférences sur \mathcal{D}_U . Soit $(c_1, c_2) \in \mathcal{IS}_{U, \succeq}$ une influence synthétisée.

La valeur synthétisée de (c_1, c_2) dans U selon \succeq est :

$$\mathcal{VS}_{U, \succeq}((c_1, c_2)) = \frac{\sum_{(d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))} (\mathcal{I}_{U, \succeq}(d) \times \alpha)}{\sum_{(d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))} \mathcal{I}_{U, \succeq}(d)}$$

Exemple 4.15. On reconsidère les valeurs d'influences extraites de l'exemple 4.14. La valeur synthétisée de l'influence (Campeur, Incendie) est :

$$\frac{0.3 \times \mathcal{I}_{U, \succeq}(\text{Préfet}) + 0.2 \times \mathcal{I}_{U, \succeq}(\text{Garde-forestier})}{3 + 2} = \frac{0.3 \times 3 + 0.2 \times 2}{3 + 2} = 0.26$$

On dispose à présent de toutes les informations nécessaires pour construire une carte synthétisée dont les influences sont valuées par des valeurs synthétisées. On appelle une telle carte une *carte synthétisée valuée*.

Définition 4.17 (Carte synthétisée valuée)

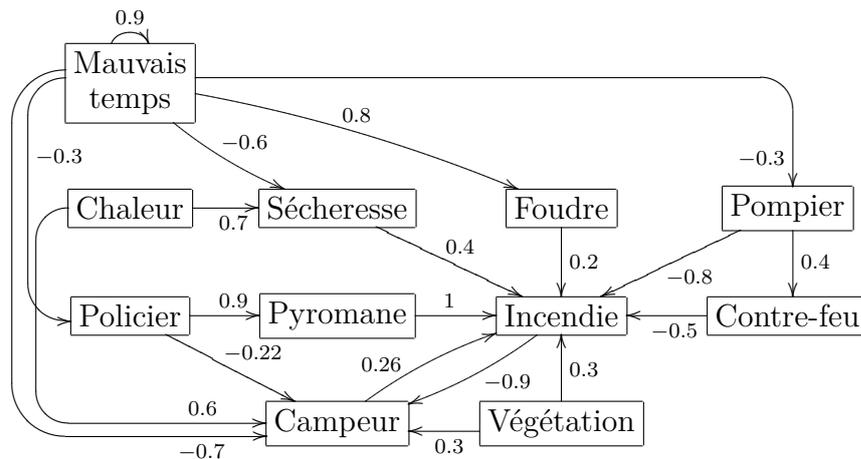
Soient T une taxonomie et I un ensemble de valeurs. Soit U un ensemble univoque de cartes attribuées défini sur T et I . Soient \succeq des préférences sur \mathcal{D}_U . Une carte synthétisée valuée est une carte cognitive $CS_{U, \succeq} = (\mathcal{CS}_{U, \succeq}, \mathcal{IS}_{U, \succeq}, \text{label})$ définie sur I et construite à partir de U selon \succeq avec :

$$\begin{aligned} \text{label: } \mathcal{IS}_{U, \succeq} &\longmapsto I \\ \text{label}((c_1, c_2)) &= \mathcal{VS}_{U, \succeq}((c_1, c_2)) \end{aligned}$$

Exemple 4.16. La carte cognitive de la figure 4.6 est une carte synthétisée valuée construite à partir de l'ensemble univoque de cartes attribuées U (exemple 4.2) et des préférences \succeq (exemple 4.4). On note que l'influence (Campeur, Incendie) porte bien la valeur 0.26, comme calculée à l'exemple 4.15.

4.3 Vue d'une carte synthétisée

Dans le chapitre 2, nous avons présenté dans la section 2.3 un mécanisme permettant de construire une vue d'une carte cognitive. Cette vue simplifie une carte en

FIGURE 4.6 – Carte synthétisée valuée construite à partir de U selon \succeq .

présentant à un utilisateur une version simplifiée de cette carte. La carte synthétisée étant une carte cognitive comme une autre, il est tout à fait possible d'en construire une vue. Nous pouvons réutiliser la taxonomie qui a servi à sa construction pour définir une échelle. Elle doit cependant être adaptée. Nous présentons comment construire une vue d'une carte cognitive synthétisée dans la section 4.3.1.

L'opérateur de synthèse des concepts permet de sélectionner les concepts d'une taxonomie grâce à des préférences exprimées sur des designers. Cela ressemble beaucoup à la construction d'une vue adaptée à différents profils (section 2.3.3 et plus particulièrement la définition 2.23). En effet, ces deux définitions permettent de construire un ensemble de concepts d'une taxonomie à partir de plusieurs ensembles de concepts représentant une sélection des concepts que souhaitent voir différentes personnes. Dans la section 4.3.2, nous allons donc montrer comment nous inspirer de la définition des concepts synthétisés pour définir autrement la notion de concepts partagés. Ceci permet d'introduire la notion de préférences lors de la construction d'une vue partagée.

4.3.1 Vue d'une carte synthétisée

On rappelle que pour construire une vue, on a besoin d'une taxonomie de concepts et d'une échelle construite sur cette taxonomie. Pour construire la carte synthétisée, on a utilisé une taxonomie. Cependant, on ne peut l'utiliser directement. En effet, pour construire une vue d'une carte cognitive, la dite carte doit être associée à une taxonomie dont les concepts élémentaires sont les concepts de la carte. Or, ce n'est pas nécessairement le cas avec la carte synthétisée et sa taxonomie.

Avant de définir une échelle, on doit donc adapter la taxonomie pour la construction d'une vue. Pour cela, on retire de cette taxonomie les concepts plus petits que les concepts utilisés dans la carte synthétisée de manière à ce que ces concepts soient les concepts élémentaires de la taxonomie. La vue consistant uniquement à simplifier une carte et non à la détailler, on n'a pas besoin de tels concepts pour la construire. On appelle une telle taxonomie une *taxonomie synthétisée*.

Définition 4.18 (Taxonomie synthétisée)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit $\mathcal{CS}_{U, \succeq}$ l'ensemble des concepts synthétisés de U selon \succeq . La taxonomie synthétisée $T_{\mathcal{CS}_{U, \succeq}}$ est définie comme étant :

$$(\{c \in C / \exists c' \in \mathcal{CS}_{U, \succeq}, c' \leq c\}, \leq)$$

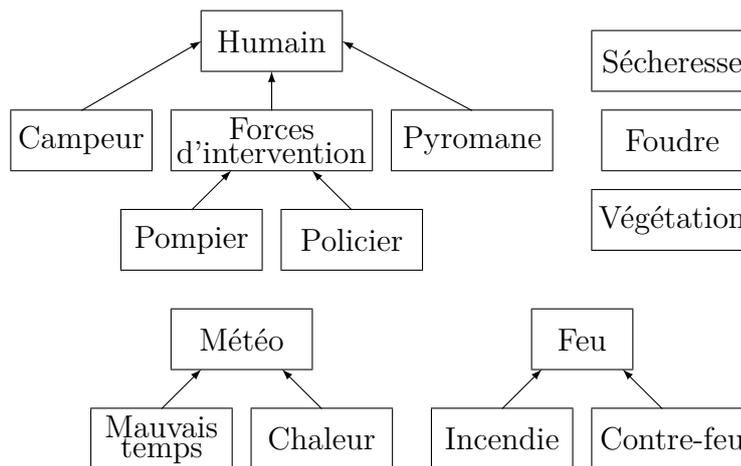


FIGURE 4.7 – La taxonomie synthétisée T2b.

Exemple 4.17. La taxonomie T2b (figure 4.7) est la taxonomie T2 (exemple 4.2) synthétisée selon les concepts synthétisés définis à l'exemple 4.10. Les concepts Orage et Pluie ont été supprimés de la taxonomie car ils sont plus petits que Mauvais temps, qui est présent dans la carte synthétisée.

Grâce à cette taxonomie, on peut aisément construire une carte cognitive taxonomique associant la carte synthétisée à la dite taxonomie. À partir de là, on peut facilement définir une échelle sur la taxonomie pour construire une vue de la carte.

Exemple 4.18. La carte cognitive de la figure 4.8 est une vue de la carte cognitive taxonomique construite à partir de la carte synthétisée de l'exemple 4.16 et de la taxonomie T2b de l'exemple 4.17 pour l'échelle $\{\text{Campeur, Forces d'intervention, Pyromane, Sécheresse, Foudre, Végétation, Mauvais temps, Chaleur, Feu}\}$.

4.3.2 Vue adaptée à plusieurs profils et préférences

Dans cette section, nous allons introduire la notion de préférences pour construire une vue d'une carte cognitive partagée entre plusieurs utilisateurs. Pour cela, nous allons définir une *échelle partagée* entre plusieurs utilisateurs selon des préférences sur ces utilisateurs. Cette définition peut être vue comme une amélioration de celle de concepts partagés (définition 2.22). Pour cela, nous allons nous inspirer de la définition des concepts synthétisés (définitions 4.10 et 4.11).

Avant même de définir comment construire cette échelle partagée, il faut savoir qu'on ne peut pas utiliser la notion de profil pour représenter les concepts choisis

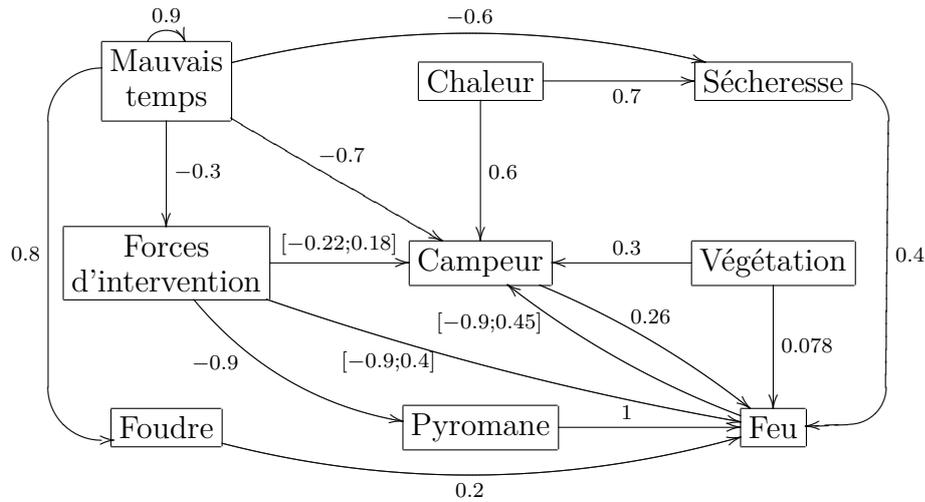


FIGURE 4.8 – Vue de la carte synthétisée.

par un utilisateur. En effet, un profil est par définition une échelle (définition 2.20). Une échelle étant représentative de la taxonomie, chaque concept de la taxonomie est comparable avec au moins un concept du profil. Si on appliquait telle quelle la définition 4.11, seul les concepts du profil préféré seraient présents dans l'échelle partagée. On peut montrer facilement qu'elle serait équivalente à construire une vue partagée entre les profils de la première strate des préférences. Nous définissons donc la notion d'*ensemble de concepts utilisateur* qui représente l'ensemble des concepts qu'un utilisateur souhaite voir dans la vue qui sera construite. Les concepts de cet ensemble doivent être incomparables et pas forcément représentatifs de la taxonomie.

Définition 4.19 (Ensemble de concepts utilisateur)

Soit $T = (C, \leq)$ une taxonomie. Soit $C' \subseteq C$ un ensemble de concepts ordonnés par la taxonomie.

C' est un ensemble de concepts utilisateur pour un utilisateur u défini sur T si C' est un ensemble de concepts incomparables.

Exemple 4.19. Sur la taxonomie $T2b$ (exemple 4.17), on définit pour l'utilisateur *Campeur* l'ensemble de concepts $C_{\text{Campeur}} = \{\text{Campeur}, \text{Mauvais temps}, \text{Chaleur}, \text{Forces d'intervention}, \text{Feu}\}$. Le *Campeur* souhaite en effet obtenir des informations sur lui, sur la météo ainsi que sur divers facteurs qui pourraient troubler ses vacances.

On définit pour l'utilisateur *Policier* l'ensemble de concepts $C_{\text{Policier}} = \{\text{Policier}, \text{Pyromane}\}$. Le *Policier* ne s'intéresse qu'aux interactions entre lui et un pyromane potentiel.

Enfin, on définit pour l'utilisateur *Écologiste* l'ensemble de concepts $C_{\text{Écologiste}} = \{\text{Végétation}, \text{Humain}\}$. L'*Écologiste* s'intéresse uniquement à l'impact de l'être humain sur la végétation.

La notion d'échelle partagée selon des préférences est donc basée sur la notion d'ensemble de concepts utilisateur plutôt que sur celle de profil. Néanmoins,

une simple adaptation de la définition 4.11 ne suffit pas pour définir une échelle partagée selon des préférences. Une échelle doit en effet être représentative de la taxonomie, chaque concept de la taxonomie étant comparable avec au moins un concept du profil.

Nous construisons donc l'échelle partagée en deux temps. On adapte dans un premier temps la définition 4.11 pour un ensemble de concepts utilisateur, afin de construire un ensemble de concepts représentant les attentes d'un ensemble d'utilisateurs. On appelle un tel ensemble de concepts une *échelle partagée partielle*. L'échelle n'est que partielle car elle n'est pas encore représentative de la taxonomie. Dans un second temps, on ajoute des concepts à cette échelle partielle de manière à ce qu'elle soit représentative de la taxonomie.

Définition 4.20 (Échelle partagée partielle selon des préférences)

Soit $T = (C, \leq)$ une taxonomie. Soient C_{u_1}, \dots, C_{u_n} des ensembles de concepts utilisateur définis sur T . Soient \succeq des préférences sur $\{C_{u_1}, \dots, C_{u_n}\}$. Soit la stratification $\mathcal{S}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ de $\{C_{u_1}, \dots, C_{u_n}\}$ selon \succeq .

$\forall i \in [1; m]$, on définit les concepts de l'échelle partagée partielle de la strate $n^o i$ comme étant :

$$\text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}(\mathcal{S}_i) = \min\left(\left\{c \in \bigcup_{C_{u_j} \in \mathcal{S}_i} C_{u_j} / \forall c' \in \bigcup_{k=1}^{i-1} \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}(\mathcal{S}_k), c \parallel c'\right\}\right)$$

L'échelle partagée partielle de $\{C_{u_1}, \dots, C_{u_n}\}$ selon \succeq est :

$$\text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq} = \bigcup_{i=1}^m \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}(\mathcal{S}_i)$$

Exemple 4.20. En reprenant les utilisateurs de l'exemple 4.19, on définit les préférences suivantes :

$$\text{Policier} \succeq \text{Écologiste} \succeq \text{Campeur}$$

La stratification de cet ordre est triviale.

Pour la strate $\{\text{Policier}\}$, l'échelle partagée partielle est :

$$\{\text{Policier}, \text{Pyromane}\}$$

Pour la strate $\{\text{Écologiste}\}$, l'échelle partagée partielle est :

$$\{\text{Végétation}\}$$

Humain est comparable avec Policier et Pyromane et n'est donc pas conservé.

Pour la strate $\{\text{Campeur}\}$, l'échelle partagée partielle est :

$$\{\text{Campeur}, \text{Mauvais temps}, \text{Chaleur}, \text{Feu}\}$$

Forces d'intervention étant comparable avec Policier, il n'est pas conservé.

L'échelle partagée partielle entre le Campeur, le Policier et l'Écologiste selon les préférences définies précédemment est donc :

$$\{\text{Policier}, \text{Pyromane}, \text{Végétation}, \text{Campeur}, \text{Mauvais temps}, \text{Chaleur}, \text{Feu}\}$$

En analysant l'échelle partagée partielle de l'exemple 4.20, on se rend bien compte qu'elle n'est pas représentative de la taxonomie. En effet, les concepts *Pompier*, *Sécheresse* et *Foudre* de la taxonomie ne sont comparables avec aucun concept de l'échelle partielle. Il faut donc ajouter des concepts à l'échelle partielle de manière à ce qu'elle soit représentative.

Une solution simple serait d'ajouter les concepts élémentaires de la taxonomie qui ne sont comparables avec aucun concept de l'échelle partielle. Cependant, afficher des concepts élémentaires dans la vue revient à afficher un niveau de détail maximal pour une partie de la carte qui n'intéresse aucun utilisateur, ce serait complètement contre-intuitif. Il est donc plus judicieux de choisir au contraire les concepts les plus grands de la taxonomie qui ne soient comparables avec aucun concept de l'échelle partagée partielle.

Définition 4.21 (Échelle partagée selon des préférences)

Soit $T = (C, \leq)$ une taxonomie. Soient C_{u_1}, \dots, C_{u_n} des ensembles de concepts utilisateur définis sur T . Soient \succeq des préférences sur $\{C_{u_1}, \dots, C_{u_n}\}$.

L'échelle partagée de $\{C_{u_1}, \dots, C_{u_n}\}$ selon \succeq est :

$$\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq} = \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq} \cup \max\left(\{c \in C \mid \forall c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}, c \parallel c'\}\right)$$

La proposition 4.1 nous assure que l'échelle partagée selon des préférences est bien une échelle.

Proposition 4.1. Soit $T = (C, \leq)$ une taxonomie. Soient C_{u_1}, \dots, C_{u_n} des ensembles de concepts utilisateur définis sur T . Soient \succeq des préférences sur $\{C_{u_1}, \dots, C_{u_n}\}$.

$\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ est une échelle pour T .

Démonstration. On montre tout d'abord que les concepts de $\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ sont incomparables, puis qu'ils sont représentatifs des concepts de T .

1. En analysant la définition de l'échelle partagée partielle (définition 4.20), on note que chaque strate est un ensemble de concepts minimaux. Par conséquent, les concepts de chaque échelle partagée partielle d'une strate donnée sont incomparables. Si l'échelle partagée partielle est définie comme l'union de ces échelles partagées partielles par strate, on a également l'assurance qu'elle est composée de concepts incomparables. En effet, chaque strate est définie comme étant un ensemble de concepts incomparables avec les strates inférieures. Ainsi, $\text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ est un ensemble de concepts incomparables. Par la définition de l'échelle partagée (définition 4.21), on note que les concepts qu'on ajoute à $\text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ sont incomparables avec ses concepts. De plus, ces concepts sont incomparables entre eux puisqu'on leur applique un max. Par conséquent, $\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ est un ensemble de concepts incomparables.
2. On cherche à montrer que $\text{elem}(T) \subseteq \text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$. Pour cela, on va considérer $c \in \text{elem}(T)$ un concept élémentaire de T .

On cherche à montrer que ce concept appartient également à $\text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$.

On suppose dans un premier temps que c est comparable avec un concept c' (éventuellement lui-même) d'un ensemble de concepts utilisateur u_j . On a donc $c \leq c'$ et $c' \in C_{u_j}$. Si $c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$, alors il est évident que $c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq} \cup \max(\{c \in C / \forall c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}, c \parallel c'\})$. De plus, puisque $c \in \text{elem}(T)$, il n'existe aucun concept plus petit que lui. Donc, puisque $c \leq c'$, $c \in \text{elemPour}_T(\text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq} \cup \max(\{c \in C / \forall c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}, c \parallel c'\}))$. D'où $c \in \text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$.

Si maintenant $c' \notin \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$, puisque $c' \in C_{u_j}$, d'après la définition de l'échelle partagée partielle, cela signifie qu'il existe un autre concept $c'' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ tel que $c' \perp c''$ qui appartient à l'ensemble de concepts d'un autre utilisateur préféré à u_j . On a donc $c \perp c''$ et donc $c \in \text{elemPour}_T(\text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$ car \perp est transitif. Puisque c est élémentaire, on a nécessairement $c \leq c''$ et donc, de la même manière, on a $c \in \text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$.

On suppose dans un second temps que c n'est comparable avec aucun concept d'un quelconque ensemble de concepts utilisateur. On a donc $\forall c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}, c \parallel c'$. Il existe donc un concept c' tel que $c \leq c'$ et $c' \in \max(\{c \in C / \forall c' \in \text{CEP}'_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}, c \parallel c'\})$. Puisque c est élémentaire, en raisonnant à nouveau de la même manière, on en déduit que $c \in \text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$.

On a donc $c \in \text{elem}(T) \Rightarrow c \in \text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$. D'où $\text{elem}(T) \subseteq \text{elemPour}_T(\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq})$ et donc $\text{CEP}_{\{C_{u_1}, \dots, C_{u_n}\}, \succeq}$ est représentatif de T . \square

Exemple 4.21. *En reprenant les ensembles de concepts utilisateur et les préférences de l'exemple 4.20, on ajoute à l'échelle partagée partielle les concepts Pompier, Sécheresse et Foudre. L'échelle partagée entre le Campeur, le Policier et l'Écologiste selon les préférences précédemment définies est donc :*

$$\left\{ \begin{array}{l} \text{Policier, Pyromane, Végétation, Campeur, Mauvais temps,} \\ \text{Chaleur, Feu, Pompier, Sécheresse, Foudre} \end{array} \right\}$$

On peut à présent sans problème utiliser cette échelle pour construire une vue de la carte cognitive synthétisée de l'exemple 4.16. Nous ne figurons pas une telle vue puisque le principe de la vue d'une carte cognitive a déjà été présenté dans l'exemple 4.18.

On peut envisager des préférences plus complexes et mêler plusieurs relations d'ordre. Par exemple, on peut demander à chaque utilisateur d'ordonner les concepts comparables. Ceci lui permet d'indiquer qu'il préfère voir un concept plutôt qu'autre dans une carte mais si les besoins d'un autre utilisateur empêchent de sélectionner son premier choix, alors on se rabat sur le deuxième. L'idée est de parvenir à une solution optimale entre tous les utilisateurs. Il est également possible de conserver des préférences sur les utilisateurs de manière à trancher certains conflits ou bien à atteindre un juste milieu entre cet ordre de préférences et les besoins de chaque utilisateur.

4.4 Personnalisation

Comme expliqué au cours de la section 4.2, la synthèse d'un ensemble de cartes attribuées s'effectue en 3 étapes :

1. la synthèse des concepts ;
2. la synthèse des influences ;
3. la synthèse des valeurs pour chaque influence.

Pour chacune de ces étapes, on définit un opérateur chargé de réaliser une partie de la synthèse. Il est tout à fait possible de définir d'autres opérateurs prenant en compte ou non les préférences exprimées sur les designers.

Dans la section 4.4.1, nous définissons d'autres opérateurs de synthèse des concepts. Dans la section 4.4.2, nous définissons d'autres opérateurs de synthèse des influences. Enfin, dans la section 4.4.3, nous définissons d'autres opérateurs de synthèse des valeurs et notamment pour d'autres ensembles de valeurs que $[-1; 1]$.

4.4.1 Synthèse des concepts

Nous proposons ici d'autres manières de définir la synthèse des concepts. Ces définitions se substituent à la définition des concepts synthétisés (définition 4.11).

Il est tout à fait possible de définir un opérateur de synthèse qui ne tient pas compte des préférences. Par exemple, si on souhaite obtenir une carte avec le maximum de détail possible, on peut choisir de synthétiser l'ensemble des concepts élémentaires de la taxonomie, en se limitant néanmoins à ceux représentant au moins un concept utilisé par un designer.

Définition 4.22 (Synthèse des concepts élémentaires)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U .

L'ensemble des concepts synthétisés élémentaires de U selon \succeq est :

$$\left\{ c \in \text{elem}(T) \mid \exists (d, T, (C', A, \text{label})) \in U, \exists c' \in C', c \perp c' \right\}$$

On remarque que les préférences ne sont pas utilisées dans cette définition.

Exemple 4.22. L'ensemble des concepts synthétisés élémentaires de l'ensemble univoque de cartes attribuées U (exemple 4.2) selon les préférences \succeq (exemple 4.4) est :

$$\left\{ \begin{array}{l} \text{Campeur, Pompier, Policier, Pyromane, Sécheresse, Foudre, Végétation,} \\ \text{Pluie, Orage, Chaleur, Incendie naturel, Incendie humain, Contre-feu} \end{array} \right\}$$

De la même manière, si on souhaite avoir un niveau de détail minimal, on peut choisir de ne synthétiser que les concepts les plus grands de la taxonomie.

Définition 4.23 (Synthèse des concepts racine)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U .

L'ensemble des concepts synthétisés racine de U selon \succeq est :

$$\{c \in \max_T(C) / \exists (d, T, (C', A, \text{label})) \in U, \exists c' \in C', c \perp c'\}$$

On remarque ici aussi que les préférences ne sont pas utilisées dans cette définition.

Exemple 4.23. L'ensemble des concepts synthétisés racine de l'ensemble univoque de cartes attribuées U (exemple 4.2) selon les préférences \succeq (exemple 4.4) est :

$$\{\text{Humain, Sécheresse, Foudre, Végétation, Météo, Feu}\}$$

On peut considérer les préférences de manière stricte et imposer que seuls les concepts du designer préféré soient conservés.

Définition 4.24 (Synthèse des concepts stricte)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit la stratification $\mathcal{S}_{\mathcal{D}_U, \succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ de \mathcal{D}_U selon \succeq .

L'ensemble des concepts synthétisés stricts de U selon \succeq est :

$$\{c / d \in \mathcal{S}_1, (d, T, (C', A, \text{label})) \in U, c \in C'\}$$

Exemple 4.24. L'ensemble des concepts synthétisés stricts de l'ensemble univoque de cartes attribuées U (exemple 4.2) selon les préférences \succeq (exemple 4.4) est :

$$\{\text{Campeur, Pompier, Policier, Pyromane, Incendie, Contre-feu}\}$$

Ce sont donc uniquement les concepts utilisés par le Préfet.

On peut améliorer la définition classique de la synthèse des concepts. En effet, il est possible que dans certains cas, des concepts de la taxonomie ne sont pas représentés alors qu'ils sont indirectement utilisés par un designer. Supposons qu'on ajoute un concept *Milice*, plus petit que *Forces d'intervention* dans la taxonomie T_2 (figure 4.2). Selon la définition classique des concepts synthétisés, ce concept ne peut appartenir à l'ensemble des concepts synthétisés puisqu'il faudrait que ce concept soit utilisé par au moins un designer. Or, le *Météorologue* a exprimé des influences sur *Forces d'intervention*. Celles-ci ont logiquement été réaffectées aux concepts qui lui étaient comparables dans la taxonomie, ici *Policier* et *Pompier*. Mais selon le principe de Liskov [Liskov et Wing, 1999], ces influences devraient aussi être réaffectées à *Milice*. Si on souhaite avoir la plus grande quantité possible d'informations représentées par les différents designers, il faut ajouter de tels concepts à l'ensemble de concepts synthétisés.

C'est le problème que nous avons entre l'échelle partagée partielle (définition 4.20) et l'échelle partagée (définition 4.21). En effet, nous devons ajouter à l'échelle partagée partielle des concepts de la taxonomie suffisamment grands pour

qu'ils soient les moins spécialisés possibles mais aussi suffisamment petits pour qu'ils soient incomparables avec les concepts déjà présents. C'est ce principe que nous appliquons à la définition classique de la synthèse des concepts en vue de l'améliorer.

Définition 4.25 (Synthèse des concepts étendue)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit $\mathcal{CS}_{U, \succeq}$ l'ensemble des concepts synthétisés de U selon \succeq d'après la définition 4.11. L'ensemble des concepts synthétisés étendus de U selon \succeq est :

$$\mathcal{CS}_{U, \succeq} \cup \max\left(\{c \in C / \forall c' \in \mathcal{CS}_{U, \succeq}, c \parallel c' \wedge \exists c'' \in \text{conceptsUtilisésPar}_U(\mathcal{D}_U), c \leq c''\}\right)$$

4.4.2 Synthèse des influences

Nous proposons ici d'autres manières de définir la synthèse des influences. Les influences synthétisées sont définies comme étant l'ensemble des couples de concepts représentant une ou plusieurs influences. Une manière simple de redéfinir l'opérateur de synthèse d'influence est donc tout simplement de redéfinir la notion de représentation. Les définitions proposées ici se substituent donc à la définition de couple de concepts représentant une influence (définition 4.12).

Par exemple, on peut imposer que les influences représentées soient uniquement celles dont les extrémités originales sont dans l'ensemble de concepts synthétisés. Un tel opérateur permet de s'assurer que les influences représentées n'ont pas été modifiées.

Définition 4.26 (Couple de concepts représentant une influence originale)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit $(d, T, (C', A, \text{label})) \in U$ une carte attribuée. Soit $(c_1, c_2) \in A$ une influence.

On dit que (c_1, c_2) se représente lui-même ssi $c_1, c_2 \in \mathcal{CS}_{U, \succeq}$.

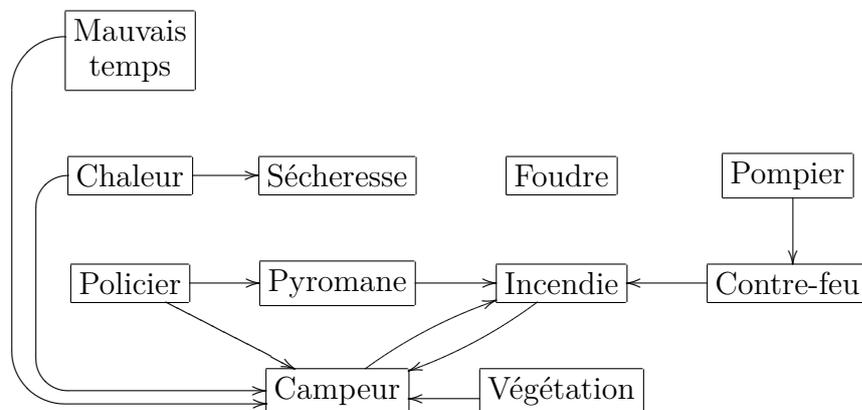


FIGURE 4.9 – Influences présentes dans au moins une des cartes attribuées.

Exemple 4.25. La carte synthétisée non-valorisée de la figure 4.9 de l'ensemble univoque de cartes attribuées U (exemple 4.2) selon les préférences \succeq (exemple 4.4) ne représente que les influences présentes dans au moins une carte de U avec les mêmes extrémités.

La définition classique de synthèse des influences (définition 4.13) n'utilise pas les préférences sur les designers. En effet, les préférences ont peu d'intérêt pour la synthèse des influences. Elles sont pratiques pour pondérer une information entre plusieurs designers. Or, une telle chose est impossible avec des influences : elles sont soit présentes, soit absentes. On pourrait cependant utiliser les préférences pour indiquer qu'on garde les influences définies par des designers suffisamment hauts dans l'ordre, au-delà d'un certain seuil.

Définition 4.27 (Représentation d'une influence selon les préférences)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soient \succeq des préférences sur \mathcal{D}_U . Soit la stratification $\mathcal{S}_{\mathcal{D}_U, \succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ de \mathcal{D}_U selon \succeq . Soit $k \in [1; m]$ un seuil de préférence. Soit $d \in \bigcup_{i=1}^k \mathcal{S}_i$ un designer en dessous du seuil k . Soit $(d, T, (C', A, \text{label})) \in U$ la carte attribuée de d . Soient $c_1, c_2 \in C$ deux concepts. Soit $(c'_1, c'_2) \in A$ une influence. On dit que (c_1, c_2) représente (c'_1, c'_2) ssi $c_1 \perp c'_1 \wedge c_2 \perp c'_2$.

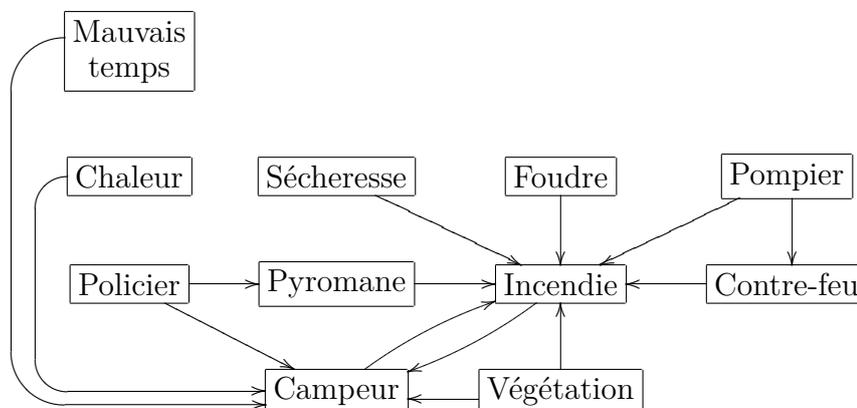


FIGURE 4.10 – Influences présentes dans les cartes attribuées des designers dont l'indice dans l'ordre de préférence est plus grand que ou égal à 2.

Exemple 4.26. La carte synthétisée non-valorisée de la figure 4.10 de l'ensemble univoque de cartes attribuées U (exemple 4.2) selon les préférences \succeq (exemple 4.4) avec un seuil de préférence 2 ne représente que les influences dont l'indice du designer est plus grand que ou égal à 2. Cette carte ne fait donc apparaître que les influences du Préfet, du Garde-forestier et de l'Agent de tourisme.

On peut aussi définir l'ensemble des influences synthétisées comme étant les influences définies par un nombre de designers différents suffisamment élevé, au-delà d'un certain seuil.

Définition 4.28 (Influences synthétisées selon l'utilisation)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T . Soit $k \in \mathbb{N}$ un seuil d'utilisation. Soient $c_1, c_2 \in C$ deux concepts. On note $A_{c_1, c_2} = \{(c'_1, c'_2) / (d, T, (C', A, \text{label})) \in U, (c'_1, c'_2) \in A, c_1 \perp c'_1 \wedge c_2 \perp c'_2\}$ l'ensemble des influences dont les extrémités sont comparables avec (c_1, c_2) . On dit que (c_1, c_2) représente toutes les influences de A_{c_1, c_2} si $|A_{c_1, c_2}| \geq k$.

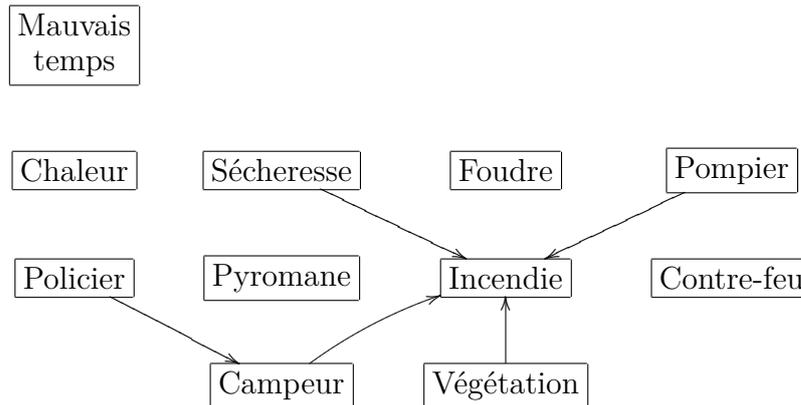


FIGURE 4.11 – Influences présentes dans au moins 2 cartes attribuées différentes.

Exemple 4.27. La carte synthétisée non-valorisée de la figure 4.10 de l'ensemble univoque de cartes attribuées U (exemple 4.2) selon les préférences \succeq (exemple 4.4) ne représente que les influences présentes dans au moins 2 cartes attribuées de U dont les extrémités sont comparables aux concepts de la carte. Le seuil d'utilisation défini est donc de 2.

On peut imaginer combiner les définitions 4.27 et 4.28 pour définir l'ensemble des influences synthétisées comme étant les influences définies par un nombre de designers différents suffisamment grand ou bien celles dont les designers sont suffisamment hauts dans l'ordre en définissant un double seuil.

4.4.3 Synthèse des valeurs

Dans cette partie, nous discutons tout d'abord d'autres opérateurs possibles pour la synthèse d'influence, puis nous montrons comment réaliser une synthèse de cartes cognitives définies sur d'autres ensembles de valeurs que $[-1; 1]$.

Autres opérateurs de synthèse de valeurs pour $[-1; 1]$

Pour la synthèse des valeurs, les préférences sont réellement importantes puisque c'est l'étape de construction de la synthèse qui comporte le plus d'informations potentiellement contradictoires. Nous l'avons définie comme étant une moyenne pondérée, la pondération d'une valeur étant basée sur la place de son designer dans l'ordre de préférence. On peut tout à fait définir n'importe quelle autre fonction de pondération, par exemple en utilisant une exponentielle de manière à privilégier encore davantage les informations fournies par le designer préféré. Cette fonction

de pondération pourrait également utiliser toutes les informations fournies par les valeurs d'influences extraites. Ainsi, dans l'exemple 4.15, si on considère l'influence de *Campeur* sur *Incendie*, la valeur apportée par le *Garde-forestier* se rapporte elle au concept *Incendie humain*, qui est un sous-concept de *Incendie*. Les influences sur *Incendie humain* ne représentent en réalité qu'une partie des influences sur *Incendie*. Il serait donc judicieux de diminuer le poids de cette valeur afin de mieux correspondre aux informations apportées par le designer. On peut par exemple diviser le poids normal de la valeur par le nombre d'enfants de *Incendie*, ce qui représente d'une manière naïve la part du concept *Incendie humain* dans le concept *Incendie*.

La moyenne pondérée telle qu'elle est définie a un défaut : si une influence synthétisée représente deux influences d'un même designer, celui-ci fournit deux valeurs lors du calcul. La pondération totale de ce designer dans ce calcul est alors le double du poids qui devrait normalement lui être attribué. Pour corriger ce problème, il faudrait réaliser une moyenne non pondérée des valeurs d'influences extraites fournies par un même designer avant le calcul de la moyenne pondérée.

Enfin, on peut tout simplement utiliser un autre opérateur qu'une moyenne pondérée. Par exemple, l'idée de la synthèse des concepts est de privilégier en priorité les concepts du designer préféré. Si celui-ci ne fournit pas d'information, alors on considère les concepts des designers suivant dans l'ordre. On peut appliquer le même principe pour les valeurs en ne gardant que la valeur du designer le plus haut dans l'ordre. Un problème se pose là aussi lorsque deux valeurs sont fournies par des designers de la même strate. Pour palier ce problème, on peut réaliser une moyenne non pondérée de ces valeurs.

Définition 4.29 (Valeur synthétisée selon les préférences)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T et sur l'ensemble de valeurs $I = [-1; 1]$. Soient \succeq des préférences sur \mathcal{D}_U . Soit la stratification $\mathcal{S}_{\mathcal{D}_U, \succeq} = \langle \mathcal{S}_1, \dots, \mathcal{S}_m \rangle$ de \mathcal{D}_U selon \succeq . Soit $(c_1, c_2) \in \mathcal{IS}_{U, \succeq}$ une influence synthétisée.

Soit \mathcal{S}_i la première strate telle qu'au moins un de ses designers a fourni une valeur, c'est-à-dire que :

- $\exists d \in \mathcal{S}_i, \exists c'_1, c'_2 \in C, \exists \alpha \in I, (d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))$;
- $\forall d \in \mathcal{S}_i, \forall d' \in \mathcal{D}_U, \forall c'_1, c'_2 \in C, \forall \alpha \in I, (d', c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2)) \Rightarrow d \succeq d'$.

La valeur synthétisée de (c_1, c_2) dans U selon \succeq est :

$$\mathcal{VS}_{U, \succeq}((c_1, c_2)) = \frac{\sum_{\substack{d \in \mathcal{S}_i \\ (d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))}} \alpha}{\left| \bigcup_{d \in \mathcal{S}_i} \{(d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))\} \right|}$$

Exemple 4.28. On reconsidère les valeurs d'influences extraites de l'exemple 4.14. La valeur synthétisée de l'influence (*Campeur*, *Incendie*) est 0.3 puisqu'on ne conserve que la valeur fournie par le *Préfet*.

Synthèse pour d'autres ensembles de valeurs

Au cours de la synthèse, les valeurs d'une carte cognitive ne sont manipulées que lors de la synthèse des valeurs. Il suffit donc de redéfinir l'opérateur de synthèse des valeurs pour un ensemble de valeurs donné pour que le processus entier de synthèse soit compatible avec cet ensemble de valeurs. L'opérateur classique de la synthèse de valeurs (définition 4.16) est spécifique aux valeurs numériques puisqu'une moyenne pondérée est effectuée.

Pour l'ensemble de valeurs $\{+, -\}$, puisque les valeurs d'influence sont soit $+$, soit $-$, l'opérateur \vee (définition 2.31) peut synthétiser des valeurs. Néanmoins, il n'est pas possible avec \vee d'attribuer un poids à chaque valeur.

Définition 4.30 (Valeur synthétisée d'une influence pour $\{+, -\}$)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T et sur l'ensemble de valeurs $\{+, -\}$. Soient \succeq des préférences sur \mathcal{D}_U . Soit $(c_1, c_2) \in \mathcal{IS}_{U, \succeq}$ une influence synthétisée.

La valeur synthétisée de (c_1, c_2) dans U selon \succeq est :

$$\mathcal{VS}_{U, \succeq}((c_1, c_2)) = \bigvee_{(d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))} \alpha$$

Il est à noter qu'une carte synthétisée construite en utilisant cette définition n'est pas définie sur l'ensemble de valeurs $\{+, -\}$ mais sur $\{+, -, ?\}$. On remarque que les préférences sont inutiles dans cette définition puisqu'il n'y a pas de pondération. Pour les prendre en considération, on pourrait s'inspirer de la définition 4.29. Il faut cependant considérer le cas où deux designers de la même strate utilisent des valeurs opposées.

Exemple 4.29. L'ensemble univoque de cartes attribuées U' (figure 4.12) est défini sur la taxonomie T2 (figure 4.2) et sur l'ensemble de valeurs $\{+, -\}$. La carte cognitive de la figure 4.13 est une carte synthétisée évaluée construite à partir de l'ensemble univoque de cartes attribuées U' et des préférences \succeq (exemple 4.4).

L'ensemble $\{+, -\}$ ne possédant que deux valeurs, on peut aussi se baser sur le nombre d'utilisations de chacun des valeurs en choisissant la plus utilisée parmi les deux.

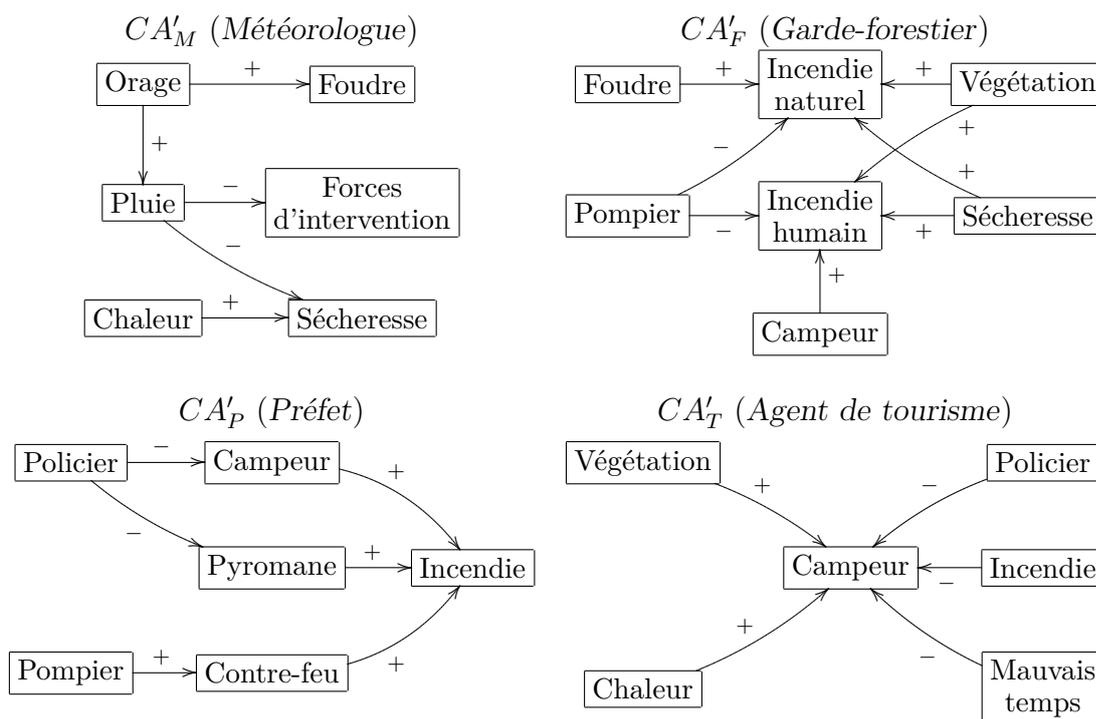
Définition 4.31 (Valeur synthétisée basée sur le nombre d'utilisations)

Soit $T = (C, \leq)$ une taxonomie. Soit I un ensemble de valeurs fini. Soit U un ensemble univoque de cartes attribuées défini sur T et sur I . Soient \succeq des préférences sur \mathcal{D}_U . Soit $(c_1, c_2) \in \mathcal{IS}_{U, \succeq}$ une influence synthétisée. Soit $\alpha \in I$ la valeur la plus utilisée telle que :

$$\forall \alpha' \in I, \left| \left\{ (d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2)) \right\} \right| \geq \left| \left\{ (d', c''_1, c''_2, \alpha') \in \text{extract}((c_1, c_2)) \right\} \right|$$

La valeur synthétisée de (c_1, c_2) dans U selon \succeq est alors :

$$\mathcal{VS}_{U, \succeq}((c_1, c_2)) = \alpha$$

FIGURE 4.12 – L'ensemble univoque de cartes attribuées U' défini sur $\{+, -\}$.

Cette définition n'est pas parfaite puisque si deux valeurs sont utilisées le même nombre de fois, rien ne permet de choisir l'une plutôt que l'autre. Les préférences peuvent permettre de trancher entre deux valeurs. On peut également s'en servir pour que l'utilisation d'une valeur par un designer ait plus de poids que celle d'un autre si le designer lui est préféré.

Pour l'ensemble de valeurs $\{nul < faible < moyen < fort\}$, il n'est pas non plus possible d'attribuer un poids à chaque valeur. On peut en revanche construire un intervalle entre la valeur la plus petite fournie par un designer et la plus grande.

Définition 4.32 (Valeur synthétisée d'une influence pour $\{nul < faible < moyen < fort\}$)

Soit $T = (C, \leq)$ une taxonomie. Soit U un ensemble univoque de cartes attribuées défini sur T et sur l'ensemble de valeurs $\{nul < faible < moyen < fort\}$. Soient \succeq des préférences sur \mathcal{D}_U . Soit $(c_1, c_2) \in \mathcal{IS}_{U, \succeq}$ une influence synthétisée.

La valeur synthétisée de (c_1, c_2) dans U selon \succeq est :

$$\mathcal{VS}_{U, \succeq}((c_1, c_2)) = \left[\min_{(d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))} \alpha ; \max_{(d, c'_1, c'_2, \alpha) \in \text{extract}((c_1, c_2))} \alpha \right]$$

Les préférences sont aussi inutiles dans cette définition. Pour les prendre en considération, on peut également s'inspirer de la définition 4.29, comme pour l'ensemble de valeurs $\{+, -\}$. Il faut cependant considérer là encore le cas où deux designers de la même strate utilisent des valeurs opposées. On remarque par ailleurs que cette définition peut s'appliquer sans problème à l'ensemble de valeurs $[-1; 1]$, comme présenté sur la figure 4.14. Par ailleurs, la définition 4.31 peut également s'appliquer sans problème à $\{nul < faible < moyen < fort\}$.

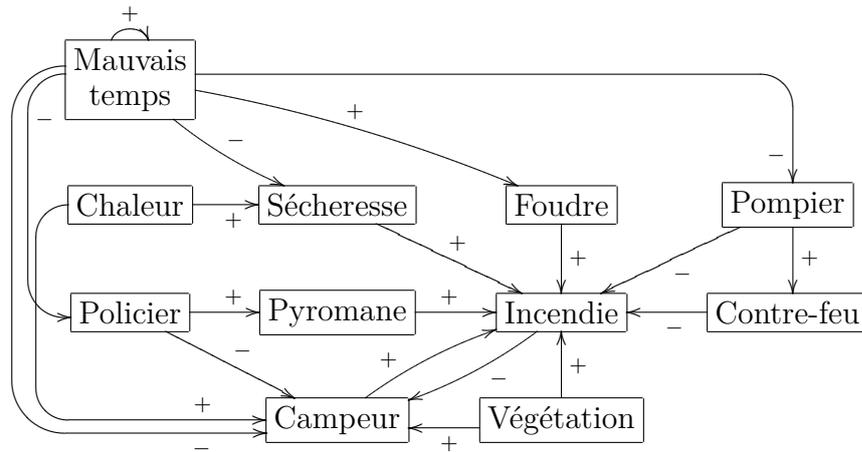


FIGURE 4.13 – Carte synthétisée évaluée définie sur $\{+, -, ?\}$ construite à partir de U' selon \succeq .

Remarque. Si la première strate d'un ordre de préférence n'a qu'un seul designer, en utilisant la synthèse de concepts de la définition stricte (définition 4.24), la représentation des influences selon les préférences avec un seuil de 1 (définition 4.27) et la synthèse de valeurs selon les préférences (définition 4.29), on retrouve exactement la carte de ce designer.

Conclusion

Dans ce chapitre, nous avons présenté une méthode pour synthétiser un ensemble de cartes cognitives construites par différents designers en une carte synthétisée unique au moyen de préférences sur ces designers. La synthèse de cartes s'effectue en trois étapes successives : la synthèse des concepts, la synthèse des influences et enfin la synthèse des valeurs pour chaque influence, cette dernière étant facultative. Nous avons également proposé différentes manières d'utiliser les notions introduites par la synthèse de cartes pour la construction d'échelles et de vues de cartes cognitives. Enfin, nous avons discuté de la pertinence des opérateurs de synthèse pour chacune des trois étapes. Selon les attentes des designers qui souhaitent synthétiser un ensemble de cartes, certains opérateurs seront plus adaptés que d'autres. Nous avons également montré comment adapter nos définitions pour d'autres ensembles de valeurs que $[-1; 1]$.

Les préférences sur les designers que nous avons utilisées sont définies comme étant une simple relation de pré-ordre sur ces designers. On pourrait tout à fait envisager d'utiliser des préférences plus complexes, comme par exemple une relation de pré-ordre définie sur les parties de l'ensemble des designers. Ceci nous permettrait d'indiquer qu'une valeur fournie à la fois par deux designers a plus de poids que si on considère indépendamment ces deux designers. On peut ainsi spécifier que deux designers ont plus de poids qu'un designer seul. Il existe déjà des méthodes pour calculer des moyennes qui prennent en compte des multi-pondérations, comme l'intégrale de Choquet [Grabisch, 1995; Grabisch, 1996]. Il reste néanmoins à voir comment adapter les autres opérateurs de synthèse à de telles préférences.

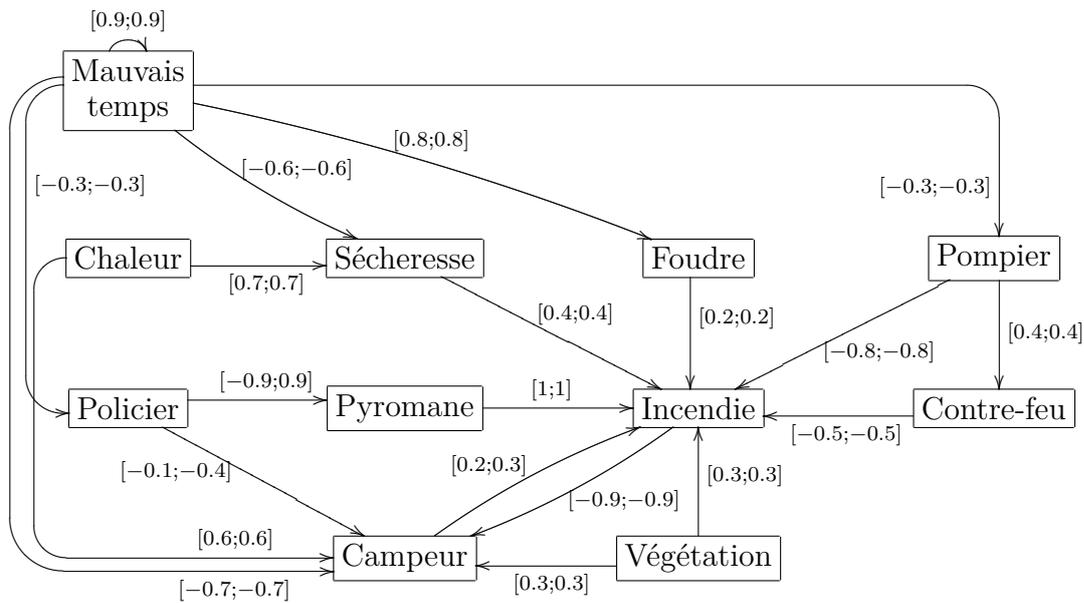


FIGURE 4.14 – Carte synthétisée valuée figurant les intervalles des valeurs des différents designers construite à partir de U selon \succeq .

Pour déduire un ordre de préférence sur les designers, on peut utiliser des formalismes issus la logique des préférences [Benferhat *et al.*, 2001; Benferhat *et al.*, 2002; Coste-Marquis *et al.*, 2004; Domshlak *et al.*, 2003; Gérard *et al.*, 2007; Wilson, 2006]. Les CP-nets [Boutilier *et al.*, 1999; Boutilier *et al.*, 2004] ont l'avantage d'avoir une représentation graphique facilement utilisable par un non-spécialiste et d'être basés sur un principe intuitif qu'est le *ceteris paribus*. Ce principe consiste, lorsque plusieurs choix définis par différents critères s'offrent une personne, à comparer ces différents choix selon un critère en considérant que tous les autres critères n'ont pas changé. Sven Ove Hansson [Hansson, 1996] donne un exemple de ce principe en considérant un serveur d'un restaurant qui propose deux types de table différents à ses clients : ronde ou carrée. Dire qu'on préfère la table carrée à la table ronde ne signifie pas qu'on préférera toutes les tables carrées à toutes les tables rondes mais simplement que si deux tables sont quasi-identiques mais que si l'une est ronde et l'autre carrée, alors on préférera la carrée. Les CP-nets ont cependant l'inconvénient de produire des ordres partiels plutôt que des ordres totaux. Par conséquent, ils ne peuvent être utilisés la plupart du temps avec les opérateurs définis ici.

Excepté l'opérateur de synthèse des valeurs, la définition de la synthèse reste suffisamment générique pour pouvoir être appliquée facilement à d'autres types de modèles graphiques. En prenant en compte les spécificités des modèles, la synthèse devrait donc être applicable aux modèles de cartes sémantiques présentés dans le chapitre 1.



5

Discussion sur les valeurs

Introduction

Dans le chapitre 2 dédié à la définition du modèle des cartes cognitives, nous avons défini entre autres le calcul de l'influence propagée d'un concept sur un autre dans une carte. Nous avons présenté différentes manières d'effectuer ce calcul en fonction des valeurs manipulées mais tout en gardant la même idée générale. Si ces opérateurs ont intuitivement un sens, nous n'avons pas défini de sémantique formelle pour la carte justifiant leur choix.

L'objectif de ce chapitre est d'étudier la sémantique pour les cartes cognitives de manière à savoir ce que représentent formellement les valeurs de ces cartes. La sémantique des valeurs est fortement liée à celles des concepts et des influences. De plus, le calcul de l'influence propagée d'un concept sur un autre est également lié à cette sémantique. Avoir une sémantique claire nous assure donc d'effectuer un calcul pertinent.

Pour définir une telle sémantique, nous étudions d'autres modèles disposant d'une sémantique forte. Dans le modèle des cartes cognitives floues [Kosko, 1986], les concepts sont des ensembles flous. Dans une telle carte, les valeurs sont nécessairement positives. Pour représenter des valeurs négatives, Bart Kosko propose de représenter sur une même carte cognitive un concept et son contraire. Il présente également une manière de calculer l'influence propagée d'un concept ou son contraire sur un autre d'une manière proche de celle déjà présentée. Les cartes cognitives floues dynamiques [Liu et Zhang, 2003] fonctionnent de manière itérative et permettent d'étudier l'évolution du système représentée par une carte cognitive en fonction du temps. D'autres modèles que les cartes cognitives représentent sous forme graphique des liens entre des concepts du monde réel. Des valeurs liées aux arcs sont également présentes pour quantifier la relation entre deux concepts. Ainsi, les réseaux bayésiens [Pearl, 1988; Pearl, 2009] figurent des liens de dépendance probabiliste entre des variables aléatoires. Les valeurs sont portées par chacun des

sommets via une table de probabilités qui indique la probabilité de chacune des valeurs possibles de la variable sachant l'état de ses parents dans le graphe. Les réseaux bayésiens causaux étendent le modèle classique des réseaux bayésiens en donnant une sémantique causale aux liens entre les variables. Les QPN [Wellman, 1990] représentent aussi des probabilités entre des variables aléatoires mais les valeurs permettent ici de spécifier des contraintes.

Dans ce chapitre, nous définissons également un nouveau modèle de carte cognitive, le modèle des *cartes cognitives probabilistes*. L'idée de ce modèle est de considérer que les valeurs portées par les influences sont apparentées à des probabilités. Nous étudions en conséquence la sémantique qu'on peut donner aux concepts et aux influences ainsi que la manière de calculer l'influence propagée d'un concept sur un autre. Pour prouver la pertinence de notre modèle, nous proposons une traduction permettant de représenter une telle carte cognitive en un réseau bayésien et nous montrons qu'il y a correspondance entre ces deux modèles.

Nous présentons différents modèles ainsi que leurs liens avec le modèle des cartes cognitives dans la section 5.1. Puis, nous présentons le modèle des cartes cognitives probabilistes dans la section 5.2. Enfin, nous résumons les notions introduites par les différents modèles dans la section 5.3 où nous discutons de la sémantique des concepts et des influences.

5.1 Interprétation des valeurs

Dans cette section, nous nous interrogeons sur la manière d'interpréter les valeurs d'une carte cognitive. Nous analysons ici deux manières d'interpréter ces valeurs, soit comme des probabilités, soit comme des valeurs floues. En ce qui concerne les probabilités, nous présentons deux modèles figurant, comme les cartes cognitives, un graphe et des valeurs dans la section 5.1.1. Il s'agit des réseaux bayésiens et des QPN. En ce qui concerne les valeurs floues, certains modèles de cartes cognitives ont déjà été développés pour interpréter de la sorte leurs valeurs. Nous présentons ces modèles dans la section 5.1.2.

5.1.1 Réseaux probabilistes

La structure graphique d'une carte cognitive et les valeurs associées à chaque lien entre deux sommets est similaire à celle des réseaux bayésiens. Les QPN sont, selon Michael Wellman leur inventeur, une généralisation des cartes cognitives. Nous estimons donc qu'il est naturel de présenter ces modèles et d'étudier leurs points de similitudes et de divergence avec les cartes cognitives.

Réseaux bayésiens

Les réseaux bayésiens [Pearl, 1988; Pearl, 2009] sont des modèles graphiques représentant des relations de dépendance probabiliste entre des variables discrètes évaluées par des probabilités conditionnelles. Chaque variable prend ses valeurs dans un ensemble prédéfini d'états. Dans un tel graphe, une variable est représentée par un sommet et un arc représente une relation de dépendance probabiliste

entre deux variables. Chacun des sommets est associé à une table de probabilités conditionnelles dans laquelle sont renseignées les probabilités de chacun des états de la variable sachant les différents états possibles de ses parents. L'un des intérêts des réseaux bayésiens est de calculer les probabilités des états des variables selon l'observation d'une ou plusieurs autres variables dans le réseau. La structure graphique du réseau est une donnée utilisée pour simplifier ces calculs en utilisant les relations d'indépendance entre les variables, notamment grâce à la notion de *d-séparation* [Verma et Pearl, 1988]. Cependant, dans le cas général, le calcul de l'inférence bayésienne est un problème NP-complet [Chickering, 1996].

Définition 5.1 (Réseau bayésien)

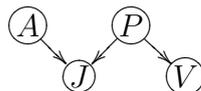
Un réseau bayésien est un graphe orienté sans circuit (DAG – directed acyclic graph) $G = (V, E)$ où :

- V est l'ensemble des sommets du graphe ;
- E est l'ensemble des arcs du graphe ;
- chaque sommet V_i représente une variable aléatoire associée à une table de probabilités conditionnelle $\mathbb{P}(V_i \mid \mathcal{C}(V_i))$ où $\mathcal{C}(V_i)$ est l'ensemble des parents de V_i dans le graphe.

Grâce à un réseau bayésien, on retrouve la probabilité jointe par :

$$\mathbb{P}(V_1, \dots, V_n) = \prod_{i=1}^n \mathbb{P}(V_i \mid \mathcal{C}(V_i))$$

Exemple 5.1. Cet exemple est inspiré de [Naïm et al., 2004]. Soit le réseau bayésien suivant :



où chaque variable de ce réseau est décrite dans le tableau suivant :

A	<i>J'ai laissé mon arroseur en marche cette nuit.</i>
P	<i>Il a plu cette nuit.</i>
J	<i>La pelouse de mon jardin est humide.</i>
V	<i>La pelouse du jardin de mon voisin est humide.</i>

Dans cet exemple, les variables représentent des événements. Leur domaine, c'est-à-dire l'ensemble des valeurs qu'elles peuvent prendre est donc $\{\mathbf{vrai}, \mathbf{faux}\}$. Lorsque la variable A est dans l'état $A = \mathbf{vrai}$, cela correspond donc à l'événement « J'ai laissé mon arroseur en marche cette nuit. ». Au contraire, l'état $A = \mathbf{faux}$ correspond donc à l'événement « Mon arroseur était éteint cette nuit. ». Pour simplifier l'écriture, l'état $A = \mathbf{vrai}$ est noté par A et $A = \mathbf{faux}$ par \bar{A} .

On associe à chaque sommet une table de probabilités. On commence par les sommets sans parent A et P . Ces variables n'ont pas de parent et ne dépendent donc d'aucun événement. Les probabilités à fournir dans ces deux tables sont donc des probabilités a priori. On estime qu'en l'absence de toute autre information, la

probabilité que j'ai laissé mon arroseur en marche et la probabilité qu'il a plu sont toutes deux à 0.4.

\mathbb{P}	
A	0.4
\bar{A}	0.6

\mathbb{P}	
P	0.4
\bar{P}	0.6

Les variables J et V ont un ou plusieurs parents. Les tables de ces variables représentent les probabilités de chacun de leurs états sachant les états de leurs parents respectifs. Ainsi, si on considère la table du sommet J , à partir du moment où mon arroseur est resté en marche ou bien s'il a plu, je suis sûr que mon jardin est humide. Au contraire, si mon arroseur n'est pas resté en marche ou s'il n'a pas plu, je suis sûr que mon jardin n'est pas humide. Il est aussi décrit dans la table du sommet V que le jardin de mon voisin ne peut être humide qui s'il a plu.

\mathbb{P}	AP	$A\bar{P}$	$\bar{A}P$	$\bar{A}\bar{P}$	
J	1	1	1	0	\mathbb{P}
\bar{J}	0	0	0	1	P
					\bar{P}
					V
					\bar{V}
					1
					0
					1

À partir de ce réseau, on peut déduire quelques informations, comme par exemple savoir si deux variables sont indépendantes ou non ou bien la probabilité d'un état d'une variable après observation de l'état d'autres variables du réseau.

Par exemple, en sortant de ma maison, je me rends compte que l'herbe de mon jardin est humide. L'herbe ne peut être mouillée que par la pluie ou par mon arroseur automatique. Je me demande alors si j'ai pu laisser allumé mon arroseur automatique. Grâce à ce réseau, on est en mesure de calculer $\mathbb{P}(A|J)$, qui nous donne une valeur de 0.625. Cette valeur est plus grande que $\mathbb{P}(A)$. Par conséquent, savoir que mon jardin est humide augmente la probabilité que mon arroseur soit resté allumé cette nuit. Cependant, on trouve également une valeur de 0.625 pour $\mathbb{P}(P|J)$. On ne peut donc pas trancher sur l'événement qui a provoqué l'humidité de mon jardin. À présent, je me rends compte que le jardin de mon voisin n'est pas humide. S'il a plu cette nuit, alors nos deux jardins devraient être humides. On cherche donc à calculer la probabilité que mon arroseur soit allumé sachant que mon jardin est humide, sans que celui de mon voisin ne le soit. On calcule donc $\mathbb{P}(A|J\bar{V})$, qui nous donne une valeur de 1. Je suis donc à présent sûr que mon arroseur automatique est resté allumé cette nuit.

Les cartes cognitives et les réseaux bayésiens ont donc pour points communs leur structure graphique et le fait que chacun des sommets est associé à un concept du monde réel, nécessairement une variable aléatoire dans le cas des réseaux bayésiens. Quelques différences subsistent :

- les cartes cognitives sont des graphes orientés autorisant les circuits tandis que les réseaux bayésiens sont des graphes orientés sans circuit ;
- les arcs d'une carte cognitive expriment des relations d'influence, c'est-à-dire de causalité entre des concepts tandis que ceux des réseaux bayésiens expriment des relations de dépendance probabiliste (éventuellement nulle) ;
- les cartes cognitives sont évaluées au niveau des arcs tandis que les réseaux bayésiens sont évalués au niveau des sommets ;

- les cartes cognitives permettent de calculer l'influence d'un concept sur un autre tandis que les réseaux bayésiens permettent de calculer la valeur de probabilité de l'état d'un concept sachant l'état d'un autre concept ;
- un des aspects essentiels des réseaux bayésiens est qu'une information peut « remonter » un lien car connaître la probabilité d'un enfant modifie la probabilité qu'on attribue au parent, ce qui est impossible avec une carte cognitive.

Pour expliquer ce dernier point, si j'observe qu'il a plu, je peux en déduire une évolution de l'état de l'humidité de mon jardin (raisonnement déductif [Charniak et McDermott, 1985]). Au contraire, si j'observe que mon jardin est humide, je peux quand même en déduire une évolution de l'état de la pluie (raisonnement abductif [Charniak et McDermott, 1985]). Ces deux types de raisonnement sont possibles à partir du même lien dans un réseau bayésien. En revanche, avec les cartes cognitives, si je peux bien décrire une influence de la pluie sur l'humidité de mon jardin, je ne peux rien en déduire de l'influence de l'humidité de mon jardin sur la pluie. En effet, une influence se limite à un raisonnement déductif. C'est la différence entre une relation causale et une relation de dépendance probabiliste.

Il faut noter que le fait que, dans un réseau bayésien, l'information peut être transmise d'un sommet à l'autre sans suivre le sens des arcs ne peut en aucun cas être équivalent à un circuit dans une carte cognitive. En effet, une carte cognitive possédant un circuit décrit une relation causale d'un concept sur lui-même : cela reste un raisonnement déductif. Dans un réseau bayésien, le raisonnement peut à la fois être déductif et abductif. Ce sont donc sémantiquement deux relations différentes.

Certaines extensions du modèle des réseaux bayésiens existent pour prendre en compte cette dimension causale de la notion d'influence. Ainsi, le modèle des réseaux bayésiens causaux [Pearl, 2000; Pearl, 2009] est similaire à celui des réseaux bayésiens classiques. La différence principale est que, alors que les arcs d'un réseau classique peuvent représenter n'importe quelle sorte de dépendance probabiliste, il s'agit nécessairement de dépendances causales dans un réseau bayésien causal. De plus, les réseaux bayésiens causaux distinguent les notions d'observation et d'intervention.

L'*observation* correspond à la méthode d'inférence des réseaux bayésiens classiques : observer l'état d'une ou plusieurs variables me permet de déduire de nouvelles probabilités sur les états des autres variables du réseau. Cette propagation d'information s'effectue à travers les arcs, en respectant leur sens ou non. L'observation permet donc un raisonnement à la fois abductif et déductif.

L'*intervention* est propre aux réseaux bayésiens causaux et se base sur la sémantique causale des liens entre les variables du réseau. Elle consiste à intervenir sur la valeur d'une variable du réseau, c'est-à-dire provoquer un état particulier de la variable. Les parents de cette variable ne sont pas la cause de son changement d'état. Par conséquent, l'information circule uniquement aux enfants de la variable, c'est-à-dire dans le sens des arcs. Lors d'une intervention sur une variable, seuls les descendants de cette variable sont donc influencés par ce changement. L'intervention permet donc un raisonnement strictement déductif.

Pour comprendre cette notion d'intervention, on considère les concepts J et P de l'exemple 5.1. On cherche à calculer la probabilité qu'il ait plu sachant l'état de l'humidité de mon jardin. Si on observe que notre jardin est humide, on calcule $\mathbb{P}(P|J)$, comme présenté précédemment. On se demande maintenant quelle est la probabilité qu'il ait plu sachant qu'on a rendu la pelouse humide. Pour représenter cette connaissance, on ajoute un nouvel opérateur, $\text{do}(\cdot)$ [Goldszmidt et Pearl, 1992] qui signifie qu'on provoque l'état d'une variable. On représente alors la probabilité qu'il ait plu sachant qu'on a provoqué l'humidité de ma pelouse par $\mathbb{P}(P|\text{do}(J))$. On dit qu'on est *intervenu* sur la variable J .

D'un point de vue algébrique, l'application de l'opérateur $\text{do}(\cdot)$ consiste à ne pas transmettre l'information concernant le changement de l'état de la variable aux parents. Ainsi, en reprenant notre exemple, rendre humide notre jardin n'a aucun effet sur le fait qu'il pleuve ou non. Par conséquent, $\mathbb{P}(P|\text{do}(J)) = \mathbb{P}(P)$. En revanche, le fait de provoquer la pluie par quelque manière que ce soit a bien une conséquence sur le fait que le jardin soit humide, identique à celle d'une pluie naturelle. On a donc $\mathbb{P}(J|\text{do}(P)) = \mathbb{P}(J|P)$. Structurellement, l'intervention sur l'état d'une variable revient à supprimer les arcs arrivant sur le sommet associé à cette variable dans le réseau bayésien et donc à le séparer de ses parents [Spirtes *et al.*, 2001]. Une autre modélisation possible est d'ajouter une variable au réseau représentant la « force » responsable de l'intervention [Pearl, 1993], moyennant une modification des tables de probabilités.

Dans un cas plus général, on peut s'intéresser à la probabilité que le jardin de mon voisin soit humide sachant que j'ai provoqué l'humidité de mon jardin. On cherche donc à calculer $\mathbb{P}(V|\text{do}(J))$. Comme expliqué précédemment, on a $\mathbb{P}(P|\text{do}(J)) = \mathbb{P}(P)$. Puisque V ne dépend que de P , on peut facilement montrer que $\mathbb{P}(V|\text{do}(J)) = \mathbb{P}(V)$.

Pour toutes ces raisons, le modèle des réseaux bayésiens causaux est plus adapté pour représenter les informations d'une carte cognitive que celui des réseaux bayésiens classiques. En effet, la notion d'influence étant une relation causale, étudier son mécanisme est assimilable à intervenir sur la valeur d'une variable.

Un autre argument nous pousse à considérer ce modèle : il existe deux types de graphes représentant les dépendances entre des concepts : les graphes de dépendances (D-map) et les graphes d'indépendance (I-map) [Pearl, 1988]. Les D-map garantissent que deux concepts connectés sont dépendants ; en revanche, deux concepts non connectés ne sont pas nécessairement indépendants. Les I-map garantissent que deux concepts séparés sachant d'autres variables sont conditionnellement indépendants ; en revanche, deux concepts connectés ne sont pas nécessairement dépendants. Une carte cognitive est donc une D-map tandis qu'un réseau bayésien est une I-map. Les réseaux bayésiens causaux, eux, sont à la fois une I-map et une D-map [Nadkarni et Shenoy, 2004]. Ils sont donc parfaits pour représenter un lien entre les deux modèles.

Qualitative Probabilistic Networks (QPN)

Les réseaux probabilistes qualitatifs (QPN) [Wellman, 1990] sont similaires aux réseaux bayésiens et sont par conséquent eux aussi des graphes orientés sans circuit.

Ils diffèrent cependant de ces derniers par l'expression de contraintes, appelées *relations qualitatives* sur la distribution de probabilité jointe. Dans le modèle de base des QPN, ces contraintes sont uniquement qualitatives et ne représentent donc pas des valeurs. Par conséquent, ce modèle ne permet pas de calculer des probabilités conditionnelles comme les réseaux bayésiens mais permet en revanche de déduire des relations entre les différentes variables. Ceci en fait un modèle adapté pour la prise de décision, entre autres. Deux types de relations qualitatives peuvent être définies : des *influences qualitatives* entre les variables et des *synergies qualitatives* entre ces influences. Comme pour les réseaux bayésiens, les algorithmes utilisent la structure du graphe pour calculer de manière efficace les relations entre les variables.

Les sommets des QPN sont des variables aléatoires associées à une loi de probabilité dont le domaine n'est pas nécessairement formellement défini. Cependant, ce domaine est supposé totalement ordonné. Il peut être discret ou continu, fini ou infini.

Les arcs d'un QPN représentent une dépendance probabiliste entre deux variables et correspondent aux influences qualitatives. Un arc peut donc être vu comme une influence associée à un signe parmi $+$, $-$, 0 ou $?$. Considérons deux variables d'un QPN A et B liées par un arc de A vers B . Chacune de ces variables peut prendre différentes valeurs parmi un ensemble de valeurs ordonné. Une valeur de $+$ de A sur B signifie que, plus A est grand, plus la probabilité que B atteigne une certaine valeur est importante *ceteris paribus*¹. Ainsi, pour toutes les valeurs a_i, a_j de A telles que $a_i > a_j$, b de B , et pour tous les assignements X des autres parents de B dans le QPN :

- si l'arc porte le signe $+$:

$$\mathbb{P}(B \geq b \mid a_i X) \geq \mathbb{P}(B \geq b \mid a_j X)$$

- si l'arc porte le signe $-$:

$$\mathbb{P}(B \geq b \mid a_i X) \leq \mathbb{P}(B \geq b \mid a_j X)$$

- si l'arc porte le signe 0 :

$$\mathbb{P}(B \geq b \mid a_i X) = \mathbb{P}(B \geq b \mid a_j X)$$

À noter que les arcs portant le signe 0 sont implicites dans un QPN par convention. Ils peuvent en effet être déduits en assumant l'hypothèse du monde clos. Dans une telle situation, les variables sont conditionnellement indépendantes sachant tous les parents de B .

- si l'arc porte le signe $?$, l'influence de A sur B est ambiguë. Par conséquent, aucune relation entre les probabilités conditionnelles des variables ne peut être exprimée. De tels arcs ne sont généralement pas représentés dans un QPN puisqu'ils représentent justement une absence de contrainte.

¹Pour une définition de la notion de *ceteris paribus*, se référer à la conclusion du chapitre 4.

Les propriétés d'indépendances des variables dans un QPN sont les mêmes que dans un réseau bayésien. Pour les mêmes raisons, les relations de dépendances entre les variables ne suivent pas la direction indiquée par un arc : comme pour les raisons bayésiennes, l'information peut « remonter » les arcs (raisonnement abductif). Si un arc existe d'une variable A vers une variable B portant un certain signe, la contrainte exprimée par cet arc est également valable de B vers A . Néanmoins, la *force* de cette influence n'est pas nécessairement identique. Cependant, comme les QPN sont des graphes sans circuit, on ne construit jamais un tel arc. En revanche, lors du calcul du signe de l'influence d'une variable sur une autre, on ne suit pas nécessairement la direction indiquée par les arcs.

Un QPN permet également de définir des synergies qualitatives qui correspondent aux hyper-arcs. Les synergies qualitatives permettent par exemple d'exprimer le fait qu'une probabilité sachant la valeur des deux parents est supérieure à la somme des probabilités sachant la valeur de chacun des parents. Nous n'allons pas développer ici cette notion, puisqu'il existe différentes sortes de synergies et que ces relations ne sont généralement pas prises en compte dans le modèle des cartes cognitives.

Formellement, un QPN est un graphe orienté sans circuit exprimant des influences entre des variables dont les arcs sont des influences qualitatives et les hyper-arcs des synergies qualitatives.

Définition 5.2 (QPN)

Un QPN est tuple $Q = (G, \Delta)$ où :

- $G = (V, A)$ est un graphe orienté sans circuit où :
 - V est l'ensemble des sommets du graphe, chacun d'entre eux représentant une variable ;
 - A est l'ensemble des arcs du graphe ;
- $\Delta = \mathcal{S} \cup \mathcal{X}$ est l'ensemble des relations qualitatives tel que :
 - \mathcal{S} est l'ensemble des influences qualitatives, chacune étant associée à un arc de G ;
 - \mathcal{X} est l'ensemble des synergies qualitatives, chacune étant associée à un hyper-arc de G .

Une influence qualitative entre deux variables établit une relation d'influence entre ces variables. C'est une relation symétrique et transitive.

Définition 5.3 (Influence qualitative)

Soit $Q = ((V, A), \mathcal{S} \cup \mathcal{X})$ un QPN. Soient $B, C \in V$ deux variables. Soit $\delta \in \{+, 0, -, ?\}$ un signe.

$S_G^\delta(B, C) \in \mathcal{S}$ est une influence qualitative entre B et C de signe δ .

Pour le calcul du signe de l'influence qualitative entre deux variables quelconques, on a besoin de deux opérateurs :

- \otimes combine les signes sur un chemin ;

- \oplus combine les signes entre différents chemins.

Comme pour les cartes cognitives, l'idée est de lister les différents chemins permettant de lier les deux variables, d'utiliser l'opérateur \otimes sur les signes des arcs de ce chemin, puis d'utiliser l'opérateur \oplus sur les résultats obtenus pour obtenir la valeur de l'influence qualitative. Ce principe est basé sur celui de l'inférence dans les diagrammes d'influence [Shachter, 1986]. Différents algorithmes permettent de calculer cette influence, dont certains sont polynomiaux [Druzdzel et Henrion, 1993].

Définition 5.4 (Calcul de l'influence qualitative)

Soit $Q = ((V, A), \mathcal{S} \cup \mathcal{X})$ un QPN. Soient $B, C \in V$ deux variables.

Le signe de l'influence qualitative entre B et C est :

$$\bigoplus_{P \in \mathcal{P}_{B,C}} \bigotimes_{\substack{(u,v) \in P \\ \mathcal{S}_G^\delta(u,v) \in \mathcal{S}}} \delta$$

où \otimes et \oplus sont définis comme étant :

\otimes	+	0	-	?		\oplus	+	0	-	?
+	+	0	-	?		+	+	+	?	?
0	0	0	0	0		0	+	0	-	?
-	-	0	+	?		-	?	-	-	?
?	?	0	?	?		?	?	?	?	?

et où $P_{B,C}$ désigne l'ensemble des chemins entre B et C ².

On note que les QPN ont une structure et un mécanisme d'inférence semblables à ceux des cartes cognitives, notamment celles définies sur l'ensemble de valeurs $\{+, -\}$. M. Wellman a comparé les deux modèles [Wellman, 1994] et liste quelques différences.

Les opérateurs \otimes et \oplus (définition 5.4) des QPN sont quasiment identiques respectivement aux opérateurs \wedge (définition 2.30) et \vee (définition 2.31) des cartes cognitives, du moins si on en restreint le domaine de définition. Les cartes cognitives n'utilisent en effet que les signes $+$ et $-$. Par conséquent, les signes 0 et $?$ ne font pas partie du domaine de définition de l'opérateur \wedge . Cependant, puisque le signe 0 est équivalent à une absence d'arc dans un QPN, ce signe est implicite dans une carte cognitive. On remarque par ailleurs que ce signe est absorbant avec l'opérateur \otimes , ce qui signifie finalement qu'un chemin est coupé si un arc n'est pas présent dans le graphe selon ce chemin. Le signe $?$ pose quant à lui quelques problèmes. Si un concepteur le plaçait dans une carte cognitive, cela signifierait qu'il pense qu'il y a une influence entre deux concepts mais qu'il est incapable de la quantifier, que ce soit par ignorance ou bien parce que la valeur de cette influence dépend de trop nombreux paramètres. Si c'est la première raison, on peut penser que le concepteur préférera ne pas mettre d'arc. Si c'est la seconde raison, construire différemment la carte cognitive peut permettre d'exprimer l'information

²Ici, la définition d'un chemin est différente de celle de la définition 2.1 puisqu'on ne suit pas nécessairement le sens des arcs. De plus, il faut prendre en compte les notions de d-séparation [Wellman, 1990] ainsi que le fait que la valeur d'une variable ait été observée ou non.

de manière non-ambiguë. Il est aussi toujours possible de créer un modèle de carte cognitive prenant en compte ce signe. Avec l'opérateur \vee , la valeur 0 est muette, ce qui signifie qu'une absence de chemin est ignorée, hypothèse effectuée lors de la construction de l'ensemble des chemins minimaux.

L'autre différence majeure entre les deux modèles est la définition de l'ensemble des chemins entre deux variables. Puisque les QPN sont des graphes sans circuit, tous les chemins entre deux sommets d'une carte cognitive ayant la même structure sont obligatoirement minimaux. Or, comme expliqué précédemment, un chemin d'un QPN n'est pas forcément un chemin d'une carte cognitive puisque les chemins d'un QPN ne suivent pas forcément les directions des arcs. Néanmoins, le signe de l'influence entre deux variables A et B d'un QPN est le même que celui entre deux concepts A et B d'une carte cognitive ayant la même structure si A et B n'ont pas d'ancêtre commun dans le graphe.

Pour Wellman, les cartes cognitives définies sur $\{+, -\}$ représentent des cas particuliers des QPN dont les opérateurs doivent être légèrement adaptés pour représenter formellement les informations.

5.1.2 Cartes cognitives floues

Nous présentons à présent des modèles de cartes cognitives. Ceux-ci disposent d'une sémantique bien définie basée sur la logique floue. Le modèle des cartes cognitives floues a été défini par Bart Kosko en 1986 [Kosko, 1986]. Ces cartes cognitives représentent en réalité des relations causales entre des concepts. Par conséquent, les valeurs portées par ces cartes ne sont pas réellement des valeurs floues. En revanche les concepts sont des ensembles flous au sens de Lotfi Zadeh [Zadeh, 1965].

Sémantique d'une influence

Une influence causale est sémantiquement différente d'une implication logique. Par exemple, considérer dans le cadre d'une étude sur les accidents de la route que « la vitesse cause des accidents » ne permet pas de déduire que « ne pas avoir d'accident cause un ralentissement ». Une influence n'est donc pas assimilable à une implication logique. En revanche, cela permet bien de déduire que « ralentir cause des non-accidents », autrement dit, rouler lentement diminue le risque d'accident. En conséquence, une influence entre deux concepts est également valable pour des concepts qui représenteraient les notions opposées à ces concepts. Cependant, on ne peut rien déduire sur la relation entre le fait de ralentir et le fait d'avoir un accident, ni sur la relation entre le fait d'accélérer et le fait de ne pas avoir d'accident. En effet, si on assimile les concepts à des ensembles flous, ce n'est pas parce qu'une influence augmente un concept qu'elle diminue le concept opposé.

Chaque concept d'une carte cognitive floue représente une échelle de valeurs qui, selon si on se place haut ou bas dans cette échelle, influencera positivement ou négativement un autre concept. Si un concept représente un événement ou une action, il ne représente pas une échelle mais simplement une variable binaire : un événement a lieu ou n'a pas lieu. Par conséquent, si toutes les influences partant de ce concept peuvent facilement trouver un sens, il est plus difficile d'interpréter

les influences arrivant sur un tel concept. On mesure dans ce cas les chances ou les risques que cet événement se déclenche, comme le risque d'avoir un cancer du poumon. Cependant, si on considère que « fumer cause le cancer du poumon », déduire que « ne pas fumer cause le fait de ne pas avoir de cancer du poumon » semble discutable. Si ne pas fumer diminue bien les risques qu'un cancer se déclare, il semble étrange qu'une inaction cause quoi que ce soit. En réalité, ce qu'il faut comprendre est que le fait de ne pas fumer par rapport au fait de fumer diminue les risques de cancer. Une influence positive d'un concept *Vitesse* vers un concept *Accident* représente donc le fait que « augmenter sa vitesse augmente le risque d'accident » et que « diminuer sa vitesse augmente le risque de non-accident », c'est-à-dire que « diminuer sa vitesse diminue le risque d'accident ».

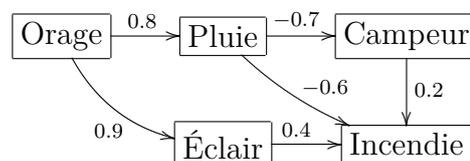


FIGURE 5.1 – Une carte cognitive floue.

Exemple 5.2. La carte cognitive de la figure 5.1 représente l'influence de différents concepts sur le risque qu'un incendie se déclare. On peut y lire qu'un temps orageux augmente le risque de la présence d'éclair de 0.9. On peut également y lire que la pluie diminue le risque qu'un incendie se déclare de 0.6. La valeur de 0.4 entre Éclair et Incendie, en comparaison avec la valeur de 0.8 entre Orage et Pluie ne signifie pas que les orages provoquent deux fois plus d'éclairs que les éclairs ne déclarent d'incendie. Les valeurs représentent une échelle. Cela signifie qu'on peut uniquement dire que 0.8 est plus grand que 0.4 et donc simplement que les orages provoquent plus d'éclairs que les éclairs ne déclarent d'incendie. De la même manière, on ne peut pas dire qu'une valeur est l'opposée d'une autre.

Modifications de la carte cognitive

Pour calculer les influences entre les concepts d'une carte cognitive floue, il est plus facile de disposer d'une carte dont toutes les influences sont positives. Grâce à la relation établie entre un concept et son opposé, B. Kosko propose de construire une nouvelle carte cognitive à partir de la carte originale contenant à la fois les concepts originaux et les opposés de ces concepts dont les influences sont positives. Cette carte intermédiaire sert en réalité d'intermédiaire pour simplifier les définitions des opérateurs d'influence entre concepts. L'opposé d'un concept n'est pas forcément son complément mais plutôt son complément local. Par exemple, le contraire de « cancer du poumon » n'est pas « cancer du non-poumon » mais plutôt « non-cancer du poumon », autrement dit le fait de ne pas avoir de cancer. En effet, un concept est généralement constitué d'une *quantité causale*, ici le cancer, et d'un *modificateur*, ici le poumon. Ce qui nous intéresse est la négation de la quantité causale ; en revanche, le modificateur reste identique de manière à ce que l'opposé d'un concept reste local. Pour chaque quantité causale q , on note son opposé $\sim q$.

On définit alors un *concept causal* comme étant l'union de deux quantités causales opposées, associée à un modificateur.

Définition 5.5 (Concept causal)

Soit q une quantité causale et $\sim q$ son opposé. Soit m un modificateur.

$c = (q \cup \sim q) \cap m$ est un concept causal. On note $c' = q \cap m$ et $\sim c' = \sim q \cap m$.

Exemple 5.3. On considère le concept *Pluie* de l'exemple 5.2. Ce concept peut être vu comme la quantité de précipitations. Les quantités causales associées à ce concept sont donc *Beaucoup de* et son opposé *Peu de*. Le modificateur est *pluie*. Le concept causal est donc représenté par les deux concepts *Peu de pluie* et *Beaucoup de pluie*.

\sim est un opérateur de négation et représente donc une partition d'un ensemble. Par conséquent, l'opposé de l'opposé d'une quantité causale est la quantité causale elle-même, autrement dit $\sim\sim q = q$. La confusion est souvent faite entre un concept causal et la quantité qu'il représente. Ainsi, lorsqu'on parle du concept causal c , on fait souvent référence à c' , notamment lors du calcul des influences.

Notons que B. Kosko définit une carte cognitive floue comme étant un graphe causal flou sans circuit. La définition que nous donnons ici d'une carte cognitive floue n'est pas exactement celle donnée par B. Kosko mais est basée sur notre définition formelle d'une carte cognitive que nous avons donnée auparavant (définition 1.6).

Définition 5.6 (Carte cognitive floue)

Soit I un ensemble de valeurs ordonné. Soit C un ensemble de concepts causaux.

$CM = (C, A, \text{label})$ est une carte cognitive floue si CM n'a pas de circuit.

Une *influence causale* établit un lien d'inclusion (*subsethood*) entre deux concepts causaux.

Définition 5.7 (Influence causale)

Soit $CM = (C, A, \text{label})$ est une carte cognitive floue. Soient $c_1 = (q_1 \cup \sim q_1) \cap m_1$ et $c_2 = (q_2 \cup \sim q_2) \cap m_2$ deux concepts liés par une influence, i.e. $(c_1, c_2) \in A$.

c_1 influence positivement de manière causale c_2 , noté $c_1 \subset c_2$ ssi $(q_1 \cap m_1) \subset (q_2 \cap m_2)$ et $(\sim q_1 \cap m_1) \subset (\sim q_2 \cap m_2)$.

c_1 influence négativement de manière causale c_2 , noté $c_1 \subset \sim c_2$ ssi $(q_1 \cap m_1) \subset (\sim q_2 \cap m_2)$ et $(\sim q_1 \cap m_1) \subset (q_2 \cap m_2)$.

La valeur de l'influence définit le *degré d'inclusion* [Bandler et Kohout, 1980] du concept-cause dans le concept-effet.

Définition 5.8 (Valeur d'influence causale)

Soit I un ensemble de valeurs ordonné. Soit $CM = (C, A, \text{label})$ est une carte cognitive floue définie sur I . Soient c_1 et c_2 deux concepts liés par une influence, i.e. $(c_1, c_2) \in A$.

Si $\text{label}((c_1, c_2))$ est positif, alors $\pi(c_1 \subset c_2) = \text{label}((c_1, c_2))$.

Si $\text{label}((c_1, c_2))$ est négatif, alors $\pi(c_1 \subset \sim c_2) = |\text{label}((c_1, c_2))|$.

Comme une influence négative sur un concept équivaut à une influence positive sur le concept opposé, on peut remplacer toutes les influences négatives en supprimant chacune d'entre elles et en la remplaçant par une influence positive de même valeur en valeur absolue vers le concept opposé. L'ensemble de valeurs d'une telle carte cognitive est donc un ensemble ordonné de valeurs positives. Pour simplifier la définition des calculs et pour mieux figurer les informations qu'elle représente, on construit donc une carte cognitive figurant chaque concept et son opposé. Cette carte cognitive est définie sur un ensemble de valeurs positives.

Définition 5.9 (Modification d'une carte cognitive floue)

Soit I un ensemble de valeurs ordonné. Soit $CM = (C, A, \text{label})$ une carte cognitive floue définie sur I . On modifie CM en une carte cognitive $CM' = (C', A', \text{label}')$ définie sur un ensemble de valeurs I' telle que :

- $C' = \bigcup_{c=(q \cup \sim q) \cap m \in C} \{q \cap m, \sim q \cap m\}$;
- $\forall (c_1, c_2) \in A$, avec $c_1 = (q_1 \cup \sim q_1) \cap m_1$, $c_2 = (q_2 \cup \sim q_2) \cap m_2$ et $\text{label}((c_1, c_2)) = \alpha$:
 - si α est positif :
 - * $(q_1 \cap m_1, q_2 \cap m_2), (\sim q_1 \cap m_1, \sim q_2 \cap m_2) \in A'$;
 - * $\text{label}'((q_1 \cap m_1, q_2 \cap m_2)) = \text{label}'((\sim q_1 \cap m_1, \sim q_2 \cap m_2)) = \alpha$;
 - si α est négatif :
 - * $(q_1 \cap m_1, \sim q_2 \cap m_2), (\sim q_1 \cap m_1, q_2 \cap m_2) \in A'$;
 - * $\text{label}'((q_1 \cap m_1, \sim q_2 \cap m_2)) = \text{label}'((\sim q_1 \cap m_1, q_2 \cap m_2)) = |\alpha|$;
- $I' = \{|\alpha| / \alpha \in I\}$.

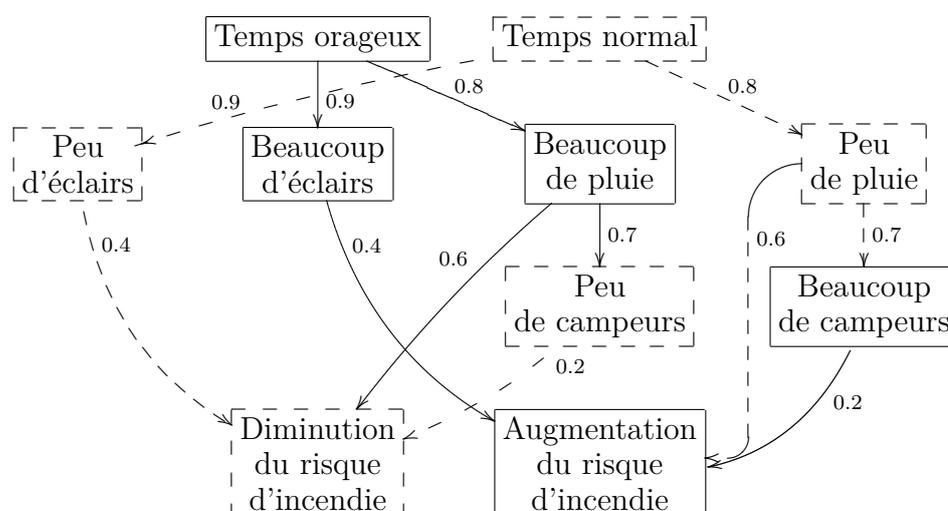


FIGURE 5.2 – Modification de la carte cognitive floue de la figure 5.1 pour le calcul des influences.

Exemple 5.4. *En reprenant la carte de l'exemple 5.2, on obtient la carte modifiée de la figure 5.2. Les dénominations des concepts ont été modifiées de manière à mieux cerner ce qu'ils représentent. Les concepts opposés ainsi que les influences qui partent de ces concepts sont représentés par des traits non continus.*

Calcul de l'influence entre deux concepts

À partir de cette nouvelle carte, on peut calculer l'influence d'un concept sur n'importe quel autre, que ce concept soit un concept opposé ou non. L'influence propagée sur un chemin est définie comme étant la plus petite valeur portée par une des influences de ce chemin. Cette définition est une généralisation de la définition 2.33.

Définition 5.10 (Influence propagée sur un chemin dans une carte floue)

Soit I un ensemble de valeurs ordonné. Soit $CM = (C, A, \text{label})$ une carte cognitive floue définie sur I . Soit $CM' = (C', A', \text{label}')$ définie sur I' la modification de CM . Soit P un chemin d'influence liant deux concepts de CM' de longueur k .

L'influence propagée sur le chemin d'influence P est :

$$\mathcal{IP}'(P) = \min \left(\bigcup_{i=0}^{k-1} \{ \text{label}((u_i, u_{i+1})) \} \right)$$

On rappelle que les cartes cognitives de B. Kosko n'autorisent pas les circuits. Par conséquent, le nombre de chemins d'influence d'un concept vers un autre est limité et tous les chemins sont minimaux. L'influence propagée entre deux concepts est définie comme étant la plus grande valeur portée par un des chemins d'un concept vers l'autre si ceux-ci sont liés par au moins un chemin. Cette définition est une généralisation de la définition 2.34.

Définition 5.11 (Influence propagée d'un concept sur un autre dans une carte floue)

Soit I un ensemble de valeurs ordonné. Soit $CM = (C, A, \text{label})$ une carte cognitive floue définie sur I . Soit $CM' = (C', A', \text{label}')$ définie sur I' la modification de CM .

L'influence propagée entre c_1 et c_2 est une fonction $\mathcal{I}' : C \times C \rightarrow I'$ telle que :

$$\mathcal{I}'(c_1, c_2) = \max_{P \in \mathcal{P}_{c_1, c_2}'} \mathcal{IP}'(P)$$

N'importe quels opérateurs de t-norme et t-conorme [Klement *et al.*, 2000] peuvent être utilisés pour calculer \mathcal{IP}' et \mathcal{I}' respectivement. B. Kosko recommande cependant l'utilisation de min et max quand rien ne permet de décider de la manière dont ces opérateurs doivent se comporter.

Exemple 5.5. *On cherche à calculer l'influence de l'orage sur le risque d'incendie.*

On doit donc calculer l'influence propagée du concept causal Orage sur le concept causal Incendie. Les concepts liés à ces concepts causaux dans la carte modifiée sont respectivement Temps orageux et Augmentation du risque d'incendie. En reprenant la carte modifiée de l'exemple 5.4, on note qu'il n'y a qu'un chemin entre le concept Temps orageux et Augmentation du risque d'incendie :

$p = \text{Temps orageux} \rightarrow \text{Beaucoup d'éclairs} \rightarrow \text{Augmentation du risque d'incendie}$.
On a donc :

$$\mathcal{IP}'(p) = \min(\{0.9, 0.4\}) = 0.4$$

On a donc trivialement $\mathcal{I}(\text{Orage}, \text{Incendie}) = 0.4$. On note qu'en raisonnant sur les concepts Temps normal et Diminution du risque d'incendie, on obtient exactement la même valeur, conformément à la sémantique causale d'une influence.

Une analyse complète du problème nécessite aussi d'étudier l'influence propagée du concept causal Orage sur le concept causal $\sim\text{Incendie}$. Les concepts liés à ces concepts causaux dans la carte modifiée sont respectivement Temps orageux et Diminution du risque d'incendie. On note qu'il y a deux chemins entre Temps orageux et Diminution du risque d'incendie : $p_1 = \text{Temps orageux} \rightarrow \text{Beaucoup de pluie} \rightarrow \text{Diminution du risque d'incendie}$ et $p_2 = \text{Temps orageux} \rightarrow \text{Beaucoup de pluie} \rightarrow \text{Peu de campeurs} \rightarrow \text{Diminution du risque d'incendie}$. On a donc :

$$\begin{aligned}\mathcal{IP}'(p_1) &= \min(\{0.8, 0.6\}) = 0.6 \\ \mathcal{IP}'(p_2) &= \min(\{0.8, 0.7, 0.2\}) = 0.2\end{aligned}$$

D'où :

$$\mathcal{I}'(\text{Orage}, \sim\text{Incendie}) = \max(\{\mathcal{IP}'(p_1), \mathcal{IP}'(p_2)\}) = \max(\{0.6, 0.2\}) = 0.6$$

On obtient également la même valeur de 0.6 pour $\mathcal{I}'(\sim\text{Orage}, \text{Incendie})$. L'analyse de l'influence d'un orage sur le risque d'incendie est donc une combinaison de deux valeurs qui quantifient deux concepts opposés. D'un côté, on peut dire qu'un orage déclenche un incendie avec une tendance de 0.4. De l'autre côté, on peut dire qu'un orage éteint un incendie avec une tendance de 0.6. De ces deux informations, on peut déduire qu'un orage a plus tendance à éteindre un incendie qu'à l'allumer sans pouvoir quantifier exactement cette tendance.

A présent, il est plus simple d'expliquer pourquoi les cartes cognitives floues n'autorisent pas de circuit. Puisque les influences décrivent des relations d'inclusion d'ensembles flous, un circuit d'influences signifierait donc qu'un concept est inclus dans lui-même. Quelque soit ce concept, son degré d'inclusion dans lui-même est nécessairement de 1. Par conséquent, le calcul de l'influence de ce concept sur lui-même devrait donner une valeur de 1. Or, d'après les définitions données précédemment, il faudrait que, sur au moins un chemin permettant de lier ce concept à lui-même, toutes les influences de ce chemin portent la valeur 1. Une telle contrainte étant difficile à respecter, il est préférable d'interdire les circuits.

La notion de chemin minimal que nous avons définie au chapitre 2 permet de calculer l'influence propagée d'un concept sur un autre dans les cartes cognitives possédant un circuit. Avec les cartes cognitives floues, si le calcul de cette influence dans une carte ayant un circuit est mathématiquement possible en utilisant la notion de chemin minimal, elle est sémantiquement fautive.

Cartes cognitives floues dynamiques

D'autres algèbres existent pour manipuler des cartes cognitives floues. L'algèbre définie par B. Kosko ne peut s'appliquer aux cartes cognitives contenant un circuit.

Certaines opérations sont construites de manière itérative et ne donnent la valeur de l'influence d'un concept sur un autre qu'au bout d'un certain temps t . On peut ainsi également étudier vers quelles valeurs vont tendre les influences lorsque le nombre d'itérations tend vers l'infini. De telles algèbres autorisent des circuits dans les cartes cognitives.

Dans certains modèles, les concepts peuvent être valués et par conséquent ont une valeur de départ au temps $t = 0$ [Dickerson et Bart, 1994]. Cette valeur permet de quantifier encore davantage les influences partant du concept ou de leur donner un état. Cela permet d'étudier l'évolution d'un système au cours du temps. Un tel mécanisme d'inférence dans une carte cognitive peut être vu comme une suite. Comme toute suite, le système peut alors converger vers un point fixe ou diverger. On peut donc étudier les conséquences des différentes entrées sur l'évolution du système. Nous n'allons cependant pas aborder ici ces méthodes de calcul, puisque nous nous consacrons à l'étude des valeurs des influences et plus généralement à l'influence d'un concept sur un autre. Certains modèles permettent néanmoins d'utiliser ces données pour calculer cette influence, comme celui présenté par Zhi-Qiang Liu et Jian Zhang [Liu et Zhang, 2003].

Avant toute chose, une fonction de seuil doit être définie. Son rôle est d'indiquer l'état d'un concept. Deux états différents sont possibles pour un concept. Un concept est soit en état d'influencer d'autres concepts, soit il ne peut en influencer aucun. On représente l'état où le concept peut en influencer d'autres par la valeur 1 et l'autre état par la valeur 0.

La fonction de seuil prend en paramètre une valeur floue portée par le concept. L'état du concept est alors déduit de cette valeur. Si la valeur floue est suffisamment élevée, alors on considère que le concept peut en influencer d'autres. Chaque concept peut être défini sur sa propre fonction de seuil. Nous avons légèrement modifié les définitions de manière équivalente pour que celles-ci soient plus facilement compatibles avec les notions que nous avons déjà définies.

Définition 5.12 (Seuil)

Soit $C = \{c_1, \dots, c_n\}$ un ensemble de concepts. Soient α une valeur floue et $c_i \in C$ un concept.

La fonction de seuil \mathcal{S}_i associée à c_i est définie sur un seuil T_i telle que :

$$\mathcal{S}_i(\alpha) = \begin{cases} 1 & \text{si } v \geq T_i \\ 0 & \text{sinon} \end{cases}$$

Les fonctions de seuil peuvent être définies différemment d'un concept à l'autre, sur des seuils différents.

On attribue ensuite à chaque concept une valeur représentant son état en fonction du temps. Pour cela, on calcule les valeurs d'influences qu'amènent les parents de ce concept dans la carte cognitive s'ils étaient dans un état influençant au temps précédent. On applique ensuite la fonction de seuil pour savoir dans quel état se trouve le concept.

Définition 5.13 (État d'un concept)

Soit $C = \{c_1, \dots, c_n\}$ un ensemble de concepts. Soit I un ensemble de valeurs numériques. Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur I . Soient

$c_i \in C$ un concept et $t \in \mathbb{N}$ un temps.

L'état du concept c_i au temps $t + 1$ est une fonction $\Delta: C \times \mathbb{N}$ telle que :

$$\Delta(c_i, t + 1) = \mathcal{S}_i \left(\sum_{c \in \mathcal{C}(c_i)} \text{label}((c, c_i)) \times \Delta(c, t) \right)$$

$\Delta(c_i, 0)$ est l'état initial du concept c_i .

L'influence propagée sur un chemin est elle aussi définie en fonction d'un temps t puisqu'elle se base sur l'état des concepts. L'idée du calcul est similaire à celle de la définition 2.5 : on multiplie les valeurs portées par les influences composant le chemin uniquement si le concept-cause de cette influence est dans un état influençant. Pour cela, on multiplie la valeur de l'influence par l'état du concept.

Définition 5.14 (Influence propagée sur un chemin selon un temps)

Soit I un ensemble de valeurs numériques. Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur I . Soit P un chemin d'influence liant deux concepts de CM de longueur k . Soit $t \in \mathbb{N}$ un temps.

L'influence propagée sur un P au temps t est :

$$\mathcal{IP}(P, t) = \prod_{i=0}^{k-1} \Delta(u_i, t) \times \text{label}((u_i, u_{i+1}))$$

L'influence propagée d'un concept sur un autre est également définie en fonction d'un temps t puisqu'elle agrège les valeurs de l'influence propagée sur différents chemins. L'idée de ce calcul est de regrouper les chemins d'influence en fonction du dernier concept de ces chemins influençant directement le concept-destination. Sur chacun de ces groupes, on considère uniquement l'influence propagée maximale sur un des ces chemins. On réalise ensuite la somme de ces valeurs.

Avant toute chose, nous avons besoin de définir, à partir d'un ensemble de chemins minimaux, l'ensemble de ses chemins passant par un concept donné, parent du dernier concept du chemin. Ceci nous permet de diviser l'ensemble des chemins minimaux d'un concept vers un autre en sous-ensembles selon chacun des parents de cet autre concept.

Définition 5.15 (Ensemble des chemins minimaux passant par un parent)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts. Soit $c'_2 \in \mathcal{C}(c_2)$ un concept parent de c_2 .

L'ensemble des chemins minimaux entre c_1 et c_2 passant par c'_2 est :

$$\mathcal{P}_{c_1, c'_2, c_2} = \{P \in \mathcal{P}_{c_1, c_2} / (c'_2, c_2) \text{ est la dernière influence de } P\}$$

La définition de l'influence propagée d'un concept sur un autre est alors simple.

Définition 5.16 (Influence propagée d'un concept sur un autre selon un temps)

Soit I un ensemble de valeurs numériques. Soit $CM = (C, A, \text{label})$ une carte cognitive définie sur I . Soient $c_1, c_2 \in C$ deux concepts. Soit $t \in \mathbb{N}$ un temps.

L'influence propagée entre c_1 et c_2 au temps t est :

$$\mathcal{I}(c_1, c_2, t) = \sum_{c'_2 \in \mathcal{C}(c_2)} \max_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \mathcal{IP}(P, t)$$

Le choix des opérateurs dépend essentiellement de ce que représentent les cartes. Prendre en compte le facteur temps dans une carte cognitive sans circuit n'a que peu d'intérêt.

Par ailleurs, avec une telle algèbre, une carte cognitive peut être vue comme une suite mathématique. Ainsi, on peut étudier son état lorsque le nombre d'itérations tend vers l'infini. On peut également étudier à partir de quelle itération la carte cognitive atteint un certain état.

Notons enfin que diverses extensions existent pour les cartes cognitives floues [Satur et Liu, 1999; Hagiwara, 1992]. Ainsi, [Song *et al.*, 2006] utilise des probabilités de manière à prendre en compte la part d'aléatoire inhérente à certains événements. Cependant, la notion de probabilité n'est qu'une composante d'un concept et participe uniquement en partie aux calculs d'influences.

5.2 Carte cognitive probabiliste

Nous avons présenté différents modèles de cartes cognitives ou autres modèles de représentation graphique de connaissances figurant des liens valués entre des concepts. Parmi ces modèles, on a vu que la sémantique attachée au modèle a des implications sur celles des concepts et des valeurs mais aussi sur la manière d'inférer des connaissances via le calcul de l'influence propagée. Si les QPN (section 5.1.1) sont bien attachés à une sémantique probabiliste, les valeurs portées par les arcs ne représentent pas directement des valeurs de probabilité. Elles représentent simplement un signe permettant de déduire des contraintes sur les probabilités conditionnelles attachées à chacun des concepts.

Dans cette section, nous définissons un modèle de carte cognitive dans lequel les valeurs portées par les influences sont apparentées à des probabilités. Ce choix de sémantique pour les valeurs a des conséquences sur la sémantique donnée aux concepts et influences. De plus, le calcul de l'influence propagée d'un concept sur un autre doit également prendre en compte cette sémantique. On appelle un tel modèle de carte basée sur des probabilités le modèle des cartes cognitives probabilistes.

Pour cela, nous attribuons tout d'abord une sémantique aux concepts et aux influences qui est compatible avec la notion de probabilité dans la section 5.2.1. Puis, nous proposons un opérateur de calcul de l'influence propagée d'un concept sur un autre adapté à cette sémantique dans la section 5.2.2. Enfin, pour prouver la pertinence de cet opérateur, nous détaillons un processus pour représenter une carte cognitive par un réseau bayésien, en respectant bien évidemment la sémantique associée à ce modèle. Nous montrons alors que notre définition de l'influence propagée d'un concept sur un autre correspond à une certaine probabilité conditionnelle dans le réseau bayésien dans la section 5.2.3.

5.2.1 Sémantique des concepts et des influences

Dans un premier temps, nous étudions quelle sémantique peut être donnée aux concepts et aux influences d'une carte cognitive dont les valeurs sont apparentées à des probabilités. Nous nous focalisons dans un premier temps sur la sémantique des concepts, puis sur la sémantique des influences.

Sémantique des concepts

Considérons une carte cognitive très simple définie sur un ensemble de valeurs I quelconque dans laquelle il y a une unique influence étiquetée $\alpha \in I$ entre un concept A et un concept B . Une telle carte est représentée sur la figure 5.3. Cette carte nous aidera à mieux expliquer l'idée derrière notre approche : les liens entre influence, valeur et probabilité sont en réalité plus complexes.

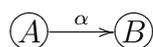


FIGURE 5.3 – Une carte cognitive simple.

Cette carte signifie que A influence B au niveau α . Comme α est apparentée à une probabilité, les concepts A et B doivent nécessairement être associés à des variables aléatoires.

Toute variable aléatoire est associée à plusieurs valeurs disjointes décrivant la totalité des états possibles de la variable. Nous aimerions que cet ensemble soit le plus simple possible et soit le même pour toutes les variables associées aux concepts. Ces valeurs doivent représenter une information du monde réel.

Dans une carte cognitive, un concept est souvent associé à une information du monde réel qui est quantifiable. Pour illustrer cette idée, on considère un concept A qui représente un arroseur automatique. Ce concept est à distinguer de la variable aléatoire A de l'exemple 5.1 qui représente le fait qu'un arroseur soit allumé ou non. Comme avec les concepts flous de l'exemple 5.4, la quantité associée au concept A pourrait être vue comme étant la force de l'arroseur, c'est-à-dire la quantité d'eau que l'arroseur fournit. On définit les valeurs possibles de la variable aléatoire associée à un concept grâce à cette quantité. Cependant, nous ne pouvons utiliser directement ses valeurs pour définir celles de la variable aléatoire puisque de telles quantités, même bornées, peuvent être continues. Or, le nombre d'états possibles pour une variable aléatoire doit nécessairement être fini.

Pour avoir les mêmes valeurs pour toutes les variables aléatoires associées aux concepts, nous définissons deux valeurs, en nous inspirant de [Cheah *et al.*, 2007]. L'une spécifie que la quantité associée au concept est en train d'augmenter ; l'autre qu'elle est en train de diminuer. On pourrait tout à fait ajouter d'autres valeurs pour la variable représentant par exemple une forte augmentation. Cependant, un tel choix complexifierait le modèle alors que nous cherchons à conserver sa simplicité.

Définition 5.17 (Variable aléatoire associée à un concept)

Soit A un concept d'une carte cognitive probabiliste.

Le concept A est associé à une information quantifiable. La variable aléatoire X_A associée au concept A d'une carte cognitive peut prendre deux valeurs :

- *+ signifie que la quantité associée à A est en état d'augmentation ;*
- *- signifie que la quantité associée à A est en état de diminution.*

$\mathbb{P}(X_A = +)$ (resp. $\mathbb{P}(X_A = -)$) dénote la probabilité que la quantité associée à A est en train d'augmenter (resp. de diminuer).

Exemple 5.6. Soit un concept A représentant un arroseur automatique. La quantité associée à A est la quantité d'eau que cet arroseur déverse. On associe au concept A la variable aléatoire X_A .

L'état $X_A = +$ signifie que l'arroseur déverse de plus en plus d'eau.

L'état $X_A = -$ signifie qu'il en déverse de moins en moins.

On note qu'il n'y pas d'état $A=0$ qui représenterait le fait que la quantité associée à A stagne. Un tel état signifierait ainsi que la quantité associée à un concept se retrouve dans un état figé. Puisque cet état est absent, cette modélisation implique que la quantité associée au concept ne peut pas stagner et doit donc nécessairement augmenter ou diminuer. Néanmoins, les cartes cognitives ont pour but d'étudier les influences entre des concepts, pas les états des concepts eux-mêmes. Par conséquent, l'absence d'un tel état n'a pas de conséquence sur le calcul de l'influence.

Notons également que [Cheah *et al.*, 2007] utilise également cette représentation binaire des concepts. La valeur $+$ signifie que l'*effet causal* du concept est positif tandis que $-$ signifie que l'effet est négatif. Nous associons ici une sémantique plus forte à ses valeurs et nous utilisons cette sémantique pour définir la notion d'influence.

On note enfin que dans les exemples de carte cognitive du chapitre 2, on a un concept *Autoroute*. Or, dire qu'une autoroute est en état d'augmentation ou de diminution n'a pas de sens : une route est ou n'est pas une autoroute. Toutefois, on peut voir ce concept comme étant les attributs qu'on donne habituellement aux autoroutes (voies multiples, longues lignes droites...). On peut alors considérer que plus ces attributs sont vérifiés, plus leur influence est importante. On note également que, vu la représentation des concepts considérée, il n'est pas possible d'avoir un concept *Sexe* avec deux valeurs *Homme* ou *Femme*. Or, le sexe d'une personne peut avoir certaines influences, comme par exemple la taille ou le poids. Cependant, le lien de causalité entre ces concepts n'est pas évident : s'il y a bien corrélation statistique, il n'est pas sûr que ce soit un lien de causalité, ce que sont sensées être les influences. Pour tous les exemples de concepts discrets dont les valeurs seraient autres que $+$ et $-$ que nous avons considérés, nous n'avons pas pu les placer dans un état influençant.

L'idée derrière la notion d'influence dans une carte cognitive probabiliste est d'étudier la modification qu'induit cette influence sur la probabilité qu'on accorde à un concept. Il faut donc exprimer comment la loi de probabilité s'applique sur les variables aléatoires associées aux concepts et notamment définir la probabilité a priori. Nous n'avons aucune information concernant la valeur a priori à donner à chacun des états possibles d'une variable. Nous considérons donc que chacun des états est équiprobable.

Définition 5.18 (Probabilité a priori d'une variable aléatoire associée à un concept)

Soit A un concept d'une carte cognitive. Soit X_A la variable aléatoire associée à A .

On a $\mathbb{P}(X_A = +) = \mathbb{P}(X_A = -) = 0.5$.

Sémantique des influences

Pour évaluer l'influence d'un concept sur un autre, l'idée est d'étudier comment le concept influencé réagit selon les différents états du concept influençant. Nous étudions donc la probabilité des états du concept influencé sachant les différents états du concept influençant. La notion d'influence est donc liée à la notion de probabilité conditionnelle.

Si on reprend la carte cognitive simple de la figure 5.3, A a deux manières d'influencer B : quand A est en train d'augmenter ou quand A est en train de diminuer. L'influence devrait donc avoir deux valeurs : une quand $X_A = +$ et une autre quand $X_A = -$. [Sedki et Bonneau de Beaufort, 2012a] donne ainsi à chaque influence deux valeurs. Cependant, nous aimerions conserver la simplicité du modèle des cartes cognitives et par conséquent nous limiter à une valeur par influence. Conformément à [Kosko, 1986], on considère que, dans la mesure où la relation d'influence est causale, l'effet de l'augmentation de A sur l'augmentation de B est égal à l'effet de la diminution de A sur la diminution de B . Ainsi, la probabilité de $X_B = +$ quand $X_A = +$ est égale à la probabilité de $X_B = -$ quand $X_A = -$. Par conséquent, la probabilité de X_B quand $X_A = +$ est le complément de la probabilité de X_B quand $X_A = -$.

Définition 5.19 (Relation causale entre un concept-cause et un concept-effet)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste. Soit $B \in C$ et X_B la variable aléatoire qui lui est associée. Soient $A \in \mathcal{C}(B)$ un parent de B , associé à la variable aléatoire X_A .

On a :

$$\begin{aligned}\mathbb{P}(X_B = + | X_A = +) &= \mathbb{P}(X_B = - | X_A = -) \\ \mathbb{P}(X_B = + | X_A = -) &= \mathbb{P}(X_B = - | X_A = +)\end{aligned}$$

Plus simplement, si on note a une valeur de X_A (i.e. $a = +$ ou $a = -$) et \bar{a} la valeur opposée (i.e. $a = -$ ou $a = +$) :

$$\mathbb{P}(X_B = + | X_A = a) = \mathbb{P}(X_B = - | X_A = \bar{a})$$

Nous considérons que la valeur d'influence α représente la modification de la probabilité accordée à B quand à la fois A et B sont en train d'augmenter. Donner une valeur α à l'influence directe entre A et B amènerait donc à se poser des questions telles que : « Sachant que A augmente, comment la probabilité que B augmente est-elle modifiée ? ». La valeur α de l'influence quantifie cette modification de la probabilité a priori de B causée par A . α permet donc d'exprimer la différence entre la probabilité de B sachant A et la probabilité a priori de B . Nous considérons donc que α est liée à la différence entre la probabilité de B sachant que A augmente et la probabilité a priori de B , c'est-à-dire $\mathbb{P}(X_B = + | X_A = +) - \mathbb{P}(X_B = +)$.

Pour montrer la pertinence de cette relation, supposons une valeur d'influence nulle $\alpha = 0$, qui signifie que A n'a aucune influence directe sur B . La probabilité conditionnelle de $X_B = +$ sachant $X_A = +$ ne devrait donc pas être modifiée par rapport à la probabilité a priori de $X_B = +$. Par conséquent, on aurait dans une telle situation $\mathbb{P}(X_B = + | X_A = +) = \mathbb{P}(X_B = +)$. En réécrivant cette expression,

on note que $\mathbb{P}(X_B = + | X_A = +) - \mathbb{P}(X_B = +) = 0$, ce qui nous permet bien de retrouver la valeur de α .

Cette relation entre les notions d'influence et de probabilité conditionnelle a des conséquences sur la structure de la carte cognitive. En effet, pour calculer l'influence propagée d'un concept sur un autre, on agrège des influences et donc des probabilités conditionnelles. Par conséquent, l'influence propagée d'un concept sur lui-même est liée à la probabilité conditionnelle de la variable associée à ce concept sur elle-même. Dans un tel cas, la probabilité conditionnelle doit respecter certaines propriétés : elle doit être égale à 0 ou 1 selon les valeurs de la variable spécifiées. Ainsi, si un chemin d'influence lie un concept à lui-même, les valeurs portées par les influences de ce chemin doivent être en accord avec cette propriété. Cette contrainte est difficile à respecter, ce qui entrave la simplicité du modèle des cartes cognitives. Nous interdisons donc les circuits dans une carte cognitive probabiliste pour qu'elles restent simples à construire.

Nous exprimons à présent formellement le lien entre α et $\mathbb{P}(X_B = + | X_A = +) - \mathbb{P}(X_B = +)$. Puisque $\mathbb{P}(X_B = +) = 0.5$ et $\mathbb{P}(X_B = + | X_A = +)$ est une valeur de probabilité et doit donc appartenir à $[0; 1]$, α doit appartenir à $[-0.5; 0.5]$. Cependant, cet ensemble de valeurs peut être difficile à utiliser par rapport à un ensemble de valeurs plus classique. L'idée est de quantifier les influences avec des valeurs d'un ensemble plus courant, tel que $[-1; 1]$ et d'utiliser une fonction de conversion qui convertira les valeurs de la carte cognitive en valeurs de $[-0.5; 0.5]$.

La *fonction de conversion* est définie telle que, quel que soit l'ensemble de valeurs I sur lequel est définie la carte cognitive, chaque valeur de I est associée à une valeur de $[-0.5, 0.5]$. Cette fonction n'a pas besoin de respecter d'autres contraintes que celles déjà présentées pour être utilisée lors de la traduction. Pour respecter la sémantique donnée aux valeurs d'influences, cette fonction doit cependant être monotone et croissante. De même, si une valeur de I correspond à une valeur nulle, son image doit être 0 par la fonction de conversion. De la même manière, la valeur maximale (resp. minimale) de I doit avoir pour image 0.5 (resp. -0.5) par la fonction de conversion.

De plus, on définit une *fonction de conversion inverse* qui doit réaliser l'opération inverse de la fonction de conversion, c'est-à-dire attribuer une valeur de I à chaque valeur de $[-0.5; 0.5]$. On peut ainsi retrouver la valeur initiale une fois qu'elle a été convertie. Cette fonction nous sera utile pour déterminer la valeur de l'influence définie sur I pour le calcul de l'influence propagée d'un concept sur un autre. Si la fonction de conversion est bijective, la fonction de conversion inverse est tout simplement la fonction réciproque.

Définition 5.20 (Fonction de conversion)

Soit I un ensemble de valeurs.

$\mathcal{F}: I \rightarrow [-0.5; 0.5]$ est une fonction de conversion qui associe une valeur de $[-0.5; 0.5]$ à chaque valeur de I .

$\mathcal{F}^{-1}: [-0.5; 0.5] \rightarrow I$ est une fonction de conversion inverse de \mathcal{F} ssi $\forall \alpha \in I$, $\mathcal{F}^{-1} \circ \mathcal{F}(\alpha) = \alpha$.³

³ $\mathcal{F} \circ \mathcal{F}^{-1}(\alpha) = \alpha$ n'est pas nécessairement vrai.

Ainsi, dans notre carte cognitive simple (figure 5.3), l'influence α de A sur B signifie que $\mathcal{F}(\alpha) = \mathbb{P}(X_B = + | X_A = +) - \mathbb{P}(X_B = +) = \mathbb{P}(X_B = + | X_A = +) - 0.5$.

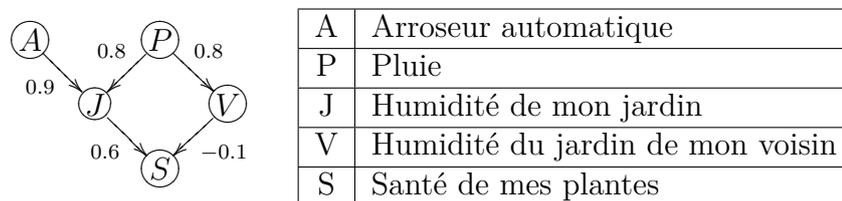


FIGURE 5.4 – Une carte cognitive définie sur l'ensemble de valeurs $[-1; 1]$.

Exemple 5.7. La carte cognitive CM6 (figure 5.4) représente les influences de concepts à propos des conséquences de l'humidité sur la santé des plantes de mon jardin. Celle-ci est inspirée du réseau bayésien de l'exemple 5.1 pour mieux comprendre les différences entre les deux modèles. L'influence négative entre V et S , signifie que l'humidité du jardin de mon voisin influence la santé de mes plantes car ses arbres poussent et font de l'ombre à mon jardin.

Dans la mesure où CM6 est définie sur $I = [-1; 1]$, on définit la fonction de conversion telle que $\forall \alpha \in I$, $\mathcal{F}(\alpha) = \frac{\alpha}{2}$. On a bien $\mathcal{F}(1) = 0.5$, $\mathcal{F}(0) = 0$ et $\mathcal{F}(-1) = -0.5$.

5.2.2 Influence propagée probabiliste

Nous définissons à présent l'opération d'influence propagée dans une carte cognitive probabiliste, que nous appelons l'*influence propagée probabiliste*. Nous considérons qu'une telle influence doit prendre ses valeurs dans le même ensemble de valeurs que celui sur lequel la carte cognitive est définie. Cependant, nous avons spécifié que la valeur d'une influence directe est liée à la différence entre une probabilité conditionnelle et la probabilité a priori et que cette différence prend ses valeurs dans $[-0.5; 0.5]$. L'influence propagée étant la combinaison de plusieurs influences, sa valeur devrait donc aussi prendre ses valeurs dans $[-0.5; 0.5]$. Or, cet ensemble de valeurs n'est pas celui sur lequel la carte cognitive est définie. Nous définissons donc une valeur intermédiaire, que nous appelons l'*influence propagée probabiliste partielle*, qui représente cette différence. Nous manipulons ensuite cette influence propagée probabiliste partielle pour calculer l'influence propagée probabiliste en la reconvertissant en une valeur de l'ensemble de valeurs original. Pour cela, puisque l'influence propagée probabiliste partielle prend ses valeurs dans $[-0.5; 0.5]$, on utilise la fonction de conversion inverse.

Pour calculer l'influence propagée probabiliste partielle, nous suivons la procédure classique du calcul de l'influence propagée, présentée dans le chapitre 2. Nous listons tout d'abord les chemins permettant de lier les deux concepts. Puis, nous calculons la valeur d'influence de chaque chemin, que nous appelons *influence propagée probabiliste sur un chemin*. Enfin, nous agrégeons ces valeurs pour calculer l'influence propagée probabiliste partielle.

Une carte cognitive probabiliste étant sans circuit, l'ensemble des chemins liant les deux concepts est nécessairement fini. Par la propriété 2.1, nous pouvons donc utiliser la définition classique d'un chemin minimal (définition 2.2).

Puisque l'influence propagée probabiliste partielle d'un concept sur un autre prend ses valeurs dans l'intervalle $[-0.5; 0.5]$, l'agrégation des valeurs composant un chemin devrait également appartenir à l'intervalle $[-0.5; 0.5]$. Si on considère un chemin de n influences, on ne peut définir l'agrégation comme un simple produit des valeurs des influences converties car nous obtiendrions une valeur appartenant à $[-0.5^n; 0.5^n]$. Pour éviter cela, nous multiplions donc chaque valeur convertie par 2 de manière à avoir une valeur appartenant à $[-1; 1]$. On réalise ensuite le produit de ces valeurs pour obtenir une influence appartenant à $[-1^n; 1^n] = [-1; 1]$. Nous divisons enfin ce produit par 2, de manière à retrouver une valeur appartenant à $[-0.5; 0.5]$. Ainsi, un chemin dont les influences ne portent que des valeurs maximales a bien une influence propagée probabiliste de 0.5.

Définition 5.21 (Influence propagée probabiliste sur un chemin)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste définie sur un ensemble de valeurs I . Soit P un chemin d'influence liant deux concepts de CM de longueur k . Soit \mathcal{F} une fonction de conversion pour I .

L'influence propagée probabiliste sur P est :

$$\mathcal{IP}_{\mathbb{P}}(P) = \frac{1}{2} \times \prod_{i=0}^{k-1} 2 \times \mathcal{F}(\text{label}((u_i, u_{i+1})))$$

Exemple 5.8. On considère le chemin $p_1 = P \rightarrow J \rightarrow S$ dans la carte cognitive $CM6$ (exemple 5.7). On utilise la fonction de conversion \mathcal{F} définie dans cet exemple. L'influence propagée probabiliste sur p_1 est :

$$\begin{aligned} \mathcal{IP}_{\mathbb{P}}(p_1) &= \frac{1}{2} \times \left(2 \times \mathcal{F}(\text{label}((P, J))) \right) \times \left(2 \times \mathcal{F}(\text{label}((J, S))) \right) \\ &= \frac{1}{2} \times \left(2 \times \mathcal{F}(0.6) \right) \times \left(2 \times \mathcal{F}(0.8) \right) = 0.24 \end{aligned}$$

Pour calculer l'influence propagée probabiliste partielle, nous devons agréger les influences propagées probabilistes sur les différents chemins. Pour cela, nous définissons un coefficient pour chaque chemin que nous appelons la *part* de ce chemin. Dans la définition classique de l'influence propagée (définition 2.6), chaque chemin avait la même part. Ici, la part d'un chemin représente sa part dans les causes de B .

Supposons que nous cherchons à calculer l'influence propagée probabiliste d'un concept A sur un concept B . Pour calculer les parts des chemins entre A et B , nous les parcourons en sens inverse, en partant de B . Nous considérons que chaque parent de B reçoit la même part. Puis, pour chacun de ces parents de B , nous partageons équitablement la part reçue précédemment aux parents des parents et ainsi de suite... Cette récursion se termine soit sur A , soit sur un concept sans parent, la carte étant sans circuit. Si on termine sur A , alors la part courante est la part du chemin que nous avons suivi, en partant de B . Si on termine sur un concept sans parent autre que A , alors nous n'avons pas suivi un chemin entre A et B .

Une représentation graphique de ce calcul est donnée sur la figure 5.5. Nous considérons par exemple le chemin de gauche. En partant de B , nous donnons à chaque parent de B la même part. Puisque B a 2 parents, nous donnons donc à chacun d'entre eux une part de $\frac{1}{2}$. Puis, nous réalisons la même opération pour le parent de B sur le chemin considéré. Puisque celui-ci a lui-même 3 parents, le chemin a à présent une part de $\frac{1}{2} \times \frac{1}{3} = \frac{1}{6}$. Nous répétons à nouveau l'opération pour le dernier arc, pour terminer sur A , et ainsi obtenir une part de $\frac{1}{12}$. Puisque A est le concept source de l'influence, la part de ce chemin est $\frac{1}{12}$.

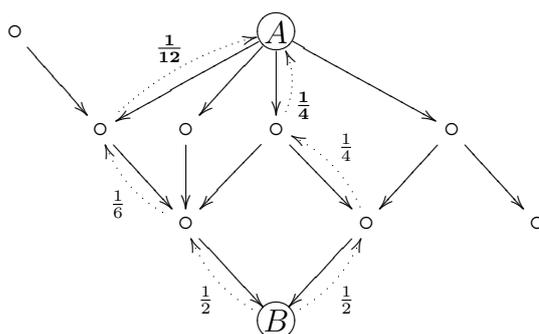


FIGURE 5.5 – Illustration du calcul des parts de deux différents chemins de A vers B

Pour résumer, la part d'un chemin est définie comme étant 1 divisé par le nombre de parents de chacun des concepts traversés par ce chemin, excepté le tout premier concept de ce chemin.

Définition 5.22 (Part d'un chemin)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste. Soit P un chemin d'influence liant deux concepts de CM de longueur k .

La part de P est :

$$\text{part}(P) = \prod_{i=1}^k \frac{1}{|\mathcal{C}(u_i)|}$$

Exemple 5.9. On considère à nouveau le chemin $p_1 = P \rightarrow J \rightarrow S$ de l'exemple 5.8. La part de p_1 est :

$$\text{part}(p_1) = \frac{1}{|\mathcal{C}(J)|} \times \frac{1}{|\mathcal{C}(S)|} = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$$

En utilisant cette part et l'influence propagée probabiliste sur un chemin, nous pouvons calculer l'influence propagée probabiliste partielle. L'influence propagée probabiliste partielle est définie comme étant la somme des produits entre la part et la valeur de chaque chemin de l'ensemble des chemins minimaux entre les deux concepts. Avec une telle définition, lorsqu'il n'y a aucun chemin d'un concept vers un autre, l'influence propagée probabiliste partielle est 0, ce qui est la valeur attendue puisqu'il n'y a aucun moyen pour le premier concept d'influencer le second.

Toutefois, nous ajoutons une exception à cette définition pour le cas du calcul de l'influence propagée probabiliste partielle d'un concept sur lui-même. Nous avons en effet considéré que l'influence propagée probabiliste partielle est liée à la

différence entre une probabilité conditionnelle et une probabilité a priori. Or, nous devrions avoir, pour chaque concept A , $\mathbb{P}(X_A = + | X_A = +) = 1$.⁴ Nous définissons donc l'influence propagée probabiliste partielle d'un concept sur lui-même comme étant 0.5.

Définition 5.23 (Influence propagée probabiliste partielle d'un concept sur un autre)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste définie sur un ensemble de valeurs I . Soient $c_1, c_2 \in C$ deux concepts.

L'influence propagée probabiliste partielle de c_1 sur c_2 est :

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \begin{cases} 0.5 & \text{si } c_1 = c_2 \\ \sum_{P \in \mathcal{P}_{c_1, c_2}} \text{part}(P) \times \mathcal{I}_{\mathbb{P}}(P) & \text{sinon} \end{cases}$$

Exemple 5.10. On reprend la carte CM6 de l'exemple 5.7. On cherche à calculer l'influence propagée probabiliste partielle de P sur S . Il y a deux chemins permettant de lier P à S dans CM6 : $p_1 = P \rightarrow J \rightarrow S$ et $p_2 = P \rightarrow V \rightarrow S$.

D'après les exemples 5.8 et 5.9, on a $\mathcal{I}_{\mathbb{P}}(p_1) = 0.24$ et $\text{part}(p_1) = \frac{1}{4}$. De la même manière, on calcule $\mathcal{I}_{\mathbb{P}}(p_2) = -0.04$ et $\text{part}(p_2) = \frac{1}{2}$.

L'influence propagée probabiliste partielle de P sur S est donc :

$$\begin{aligned} \mathcal{I}_{\mathbb{P}}'(P, S) &= \text{part}(p_1) \times \mathcal{I}_{\mathbb{P}}(p_1) + \text{part}(p_2) \times \mathcal{I}_{\mathbb{P}}(p_2) \\ &= \frac{1}{4} \times 0.24 + \frac{1}{2} \times -0.04 = 0.04 \end{aligned}$$

L'influence propagée probabiliste partielle de V sur A , puisqu'il n'y a aucun chemin permettant de lier ces deux concepts est $\mathcal{I}_{\mathbb{P}}'(V, A) = 0$.

L'influence propagée probabiliste partielle de A sur A est $\mathcal{I}_{\mathbb{P}}'(A, A) = 0.5$.

Pour le calcul de l'influence propagée probabiliste d'un concept sur un autre, nous convertissons l'influence propagée probabiliste partielle en une valeur appartenant à l'ensemble de valeurs sur lequel est définie la carte cognitive. Nous utilisons pour cela la fonction de conversion inverse. De plus, le calcul de l'influence propagée probabiliste partielle peut être vu comme une moyenne pondérée par les parts des chemins entre deux concepts. Pour que la somme des poids (donc des parts) des chemins soit égale à 1, on normalise en divisant chaque part de chemin par la somme des parts de tous les chemins. On applique enfin la fonction de conversion inverse pour obtenir une valeur d'influence appartenant à l'ensemble initial des valeurs d'influence.

Une telle opération permet d'augmenter la valeur de l'influence propagée qui est souvent proche de 0.5, tout en restant dans l'intervalle $[0; 1]$ avant reconversion. De plus, cela permet de s'assurer que le calcul de l'influence propagée probabiliste correspond à la notion d'influence. En effet, si on considère les concepts A et J de l'exemple 5.7, puisque le seul chemin entre ces deux concepts est l'influence directe, on a l'assurance que $\mathcal{I}_{\mathbb{P}}(A, J) = \text{label}((A, J))$.

Cependant, il faut s'assurer que la somme des parts soit différente de 0 pour s'assurer que la division soit toujours possible. Cela correspond au cas où il n'y a

⁴Pour n'importe quelle variable aléatoire X et pour n'importe quelle valeur x de X , si $\mathbb{P}(X = x) \neq 0$, on a $\mathbb{P}(X = x | X = x) = 1$.

pas de chemin entre deux concepts. Dans une telle situation, on reconvertit directement l'influence propagée probabiliste partielle, sans effectuer de normalisation.

Définition 5.24 (Influence propagée probabiliste d'un concept sur un autre)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste définie sur un ensemble de valeurs I . Soient $c_1, c_2 \in C$ deux concepts.

L'influence propagée probabiliste de c_1 sur c_2 est :

$$\mathcal{I}_{\mathbb{P}}(c_1, c_2) = \begin{cases} \mathcal{F}^{-1} \left(\mathcal{I}_{\mathbb{P}'}(c_1, c_2) \right) & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \mathcal{F}^{-1} \left(\frac{\mathcal{I}_{\mathbb{P}'}(c_1, c_2)}{\sum_{P \in \mathcal{P}_{c_1, c_2}} \text{part}(P)} \right) & \text{sinon} \end{cases}$$

Exemple 5.11. On reprend toujours la carte $CM6$ de l'exemple 5.7. On cherche cette fois à calculer l'influence propagée probabiliste de P sur S . Pour calculer cette influence, on a besoin d'une fonction de conversion inverse. Ici, comme $\mathcal{F}(\alpha) = \frac{\alpha}{2}$ est bijective, on a $\mathcal{F}^{-1}(\alpha) = \alpha \times 2$.

L'influence propagée probabiliste de P sur S est, d'après l'influence propagée probabiliste partielle calculée dans l'exemple 5.10 :

$$\mathcal{I}_{\mathbb{P}}(P, S) = \mathcal{F}^{-1} \left(\frac{\mathcal{I}_{\mathbb{P}'}(P, S)}{\text{part}(p_1) + \text{part}(p_2)} \right) = \left(\frac{0.04}{\frac{1}{4} + \frac{1}{2}} \right) \times 2 = 0.1067$$

Puisqu'il n'y a aucun chemin entre V et A , l'influence propagée probabiliste est :

$$\mathcal{I}_{\mathbb{P}}(V, A) = \mathcal{F}^{-1}(\mathcal{I}_{\mathbb{P}'}(V, A)) = 0 \times 2 = 0$$

Pour la même raison, l'influence propagée probabiliste de A sur A est :

$$\mathcal{I}_{\mathbb{P}}(A, A) = \mathcal{F}^{-1}(\mathcal{I}_{\mathbb{P}'}(A, A)) = 0.5 \times 2 = 1$$

5.2.3 Codage d'une carte cognitive probabiliste en un réseau bayésien

Pour prouver que notre opérateur d'influence propagée probabiliste défini pour une carte cognitive probabiliste est pertinent, nous proposons une méthode de traduction pour représenter une carte cognitive probabiliste en un réseau bayésien. On montre ensuite que le calcul de l'influence propagée probabiliste d'un concept sur un autre dans la carte cognitive mène au même résultat que le calcul d'une probabilité conditionnelle correspondante dans le réseau bayésien.

Nous détaillons ce processus en plusieurs temps. Le réseau bayésien est construit de telle manière que chaque sommet dans la carte cognitive (concept) est codé comme un sommet dans le réseau bayésien. Chaque influence entre deux concepts de la carte est représentée par un arc entre les deux sommets dans le réseau bayésien qui représentent ces concepts. Au final, le réseau bayésien a donc la même structure graphique que la carte cognitive. Nous donnons donc le même nom aux nœuds de la carte cognitive et du réseau bayésien. Chaque nœud du

réseau bayésien est associé à une variable aléatoire qui correspond à la variable aléatoire associée au concept de la carte cognitive correspondant à ce nœud. La table de probabilités associée à une variable est calculée à partir des valeurs des influences attachées aux arcs qui arrivent sur le concept correspondant.

Nous expliquons tout d'abord l'intuition puis comment représenter un concept d'une carte cognitive ayant au plus un parent par un sommet dans un réseau bayésien. Puis, nous examinons le cas général avec un concept ayant un nombre quelconque de parents. Ceci nous permet alors de définir une méthode de traduction complète, que nous appliquons sur la carte cognitive de l'exemple 5.7. Enfin, nous énonçons un théorème qui illustre la correspondance entre l'influence propagée d'un concept sur un autre et une probabilité conditionnelle.

Cas d'un concept ayant au plus un parent

La définition 5.18 nous donne la probabilité a priori de l'ensemble des sommets d'un réseau bayésien représentant une carte cognitive. On en déduit trivialement la table de probabilités des sommets sans parent.

Définition 5.25 (Table de probabilités des sommets sans parent)

Soit $CM = (C, A, \text{label})$ une carte cognitive. Soit $B \in C$ un concept sans parent, i.e. $\mathcal{C}(B) = \emptyset$. Soit X_B la variable aléatoire associée à B .

La table de probabilités de X_B est :

\mathbb{P}	
$X_B = +$	0.5
$X_B = -$	0.5

Pour les concepts ayant exactement un parent, on reprend notre exemple de carte cognitive simple de la figure 5.3. Le sommet B a un seul parent A . On doit calculer quatre valeurs de probabilités, chacune correspondant aux différentes configurations possibles des états des variables aléatoires associées aux concepts ($X_B = +$ et $X_A = +$; $X_B = +$ et $X_A = -$; $X_B = -$ et $X_A = +$; $X_B = -$ et $X_A = -$). On a déjà la valeur de $\mathbb{P}(X_B = + | A = +)$, donnée à la fin de la section 5.2.1. Nous avons en effet $\mathcal{F}(\alpha) = \mathbb{P}(X_B = + | X_A = +) - 0.5$, d'où $\mathbb{P}(X_B = + | X_A = +) = 0.5 + \mathcal{F}(\alpha)$. La valeur de $\mathbb{P}(X_B = - | X_A = +)$ peut être déduite par la relation $\mathbb{P}(B = - | A = +) = 1 - \mathbb{P}(B = + | A = +)$. Nous avons également considéré que $\mathbb{P}(X_B = + | X_A = -)$ est le complément de $\mathbb{P}(X_B = + | X_A = +)$ (définition 5.19). Nous pouvons donc renseigner les quatre valeurs de probabilité et construire la table de probabilités d'une variable aléatoire associée à un concept n'ayant qu'un seul parent.

Définition 5.26 (Table de probabilités d'un concept ayant un seul parent)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste définie sur un ensemble de valeurs I . Soit \mathcal{F} une fonction de conversion pour I . Soit $B \in C$ un concept tel que B a un seul parent A amenant une valeur d'influence α (i.e. $\mathcal{C}(B) = \{A\}$ et $\text{label}((A, B)) = \alpha$). Soient X_B et X_A les variables aléatoires associées respectivement à B et A .

La table de probabilités de X_B est :

\mathbb{P}	$X_A = +$	$X_A = -$
$X_B = +$	$0.5 + \mathcal{F}(\alpha)$	$0.5 - \mathcal{F}(\alpha)$
$X_B = -$	$0.5 - \mathcal{F}(\alpha)$	$0.5 + \mathcal{F}(\alpha)$

Exemple 5.12. Dans cet exemple, nous prenons un extrait de CM6 (exemple 5.7) pour obtenir une nouvelle carte, définie sur $[-1; 1]$:



On cherche à représenter les tables de probabilités des variables aléatoires associées à chacun de ces concepts. On reprend la fonction de conversion $\mathcal{F}(\alpha) = \frac{\alpha}{2}$. Les tables de probabilités des variables aléatoires associées aux concepts A , J et S sont :

\mathbb{P}		\mathbb{P}	$X_A = +$	$X_A = -$
$X_A = +$	0.5	$X_J = +$	$0.5 + \mathcal{F}(0.9) = 0.95$	$\mathbb{P}(X_J = - X_A = +) = 0.05$
$X_A = -$	0.5	$X_J = -$	$0.5 - \mathcal{F}(0.9) = 0.05$	$\mathbb{P}(X_J = + X_A = +) = 0.95$

\mathbb{P}	$X_J = +$	$X_J = -$
$X_S = +$	$0.5 + \mathcal{F}(0.8) = 0.9$	$\mathbb{P}(X_S = - X_J = +) = 0.1$
$X_S = -$	$0.5 - \mathcal{F}(0.8) = 0.1$	$\mathbb{P}(X_S = + X_J = +) = 0.9$

Cas d'un concept ayant un nombre quelconque de parents

On considère la carte cognitive simple de la figure 5.6 qui représente deux concepts A et B influençant un concept C avec des valeurs d'influence respectivement de α et β . Supposons que α soit la valeur maximale de l'ensemble de valeurs sur lequel est définie la carte cognitive et β la valeur minimale. Avec une fonction de conversion convenablement définie, on a $\mathcal{F}(\alpha) = 0.5$ et $\mathcal{F}(\beta) = -0.5$. Si on applique naïvement la définition 5.26, on calcule $\mathbb{P}(X_C = + | X_A = +) = 0.5 + \mathcal{F}(\alpha) = 1$. De la même manière, on a $\mathbb{P}(X_C = + | X_B = +) = 0.5 + \mathcal{F}(\beta) = 0$. Or, avec de telles valeurs pour ces probabilités conditionnelles, il est impossible de trouver une valeur pour $\mathbb{P}(X_C = + | X_A = +, X_B = +)$.

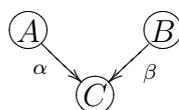


FIGURE 5.6 – Une carte cognitive figurant un concept avec deux parents.

Les définitions 5.25 et 5.26 sont en réalité uniquement des cas particuliers d'une définition plus générale. Les tables de probabilités doivent être calculées différemment dans ce cas général. Il faut noter qu'il y a moins de valeurs disponibles dans la carte cognitive que de valeurs à fournir pour les tables de probabilités. Ainsi, pour un concept ayant n parents, on a n valeurs, chacune correspondant à la valeur d'influence amenée par un parent. Pour une variable aléatoire associée à un tel concept, on a besoin de 2^n valeurs pour renseigner en entier sa table de probabilités. Néanmoins, il existe plusieurs méthodes pour calculer toutes les valeurs de la table de probabilités d'un réseau bayésien à partir de quelques valeurs données par les experts.

Certaines de ces méthodes sont dédiées à la représentation d'une carte cognitive en un réseau bayésien. [Cheah *et al.*, 2007] propose ainsi une méthode pour représenter par un réseau bayésien des cartes cognitives définies sur l'intervalle $[-1; 1]$. Cependant, cette méthode amène à obtenir une probabilité de 1 dans chaque table de probabilités. L'influence combinée de plusieurs parents peut ainsi être totale même si la valeur de chaque influence est basse. Ce problème s'illustre de manière triviale dans le cas d'un seul parent dans la mesure où une influence de 0.1 et une influence de 0.9 sont représentées avec la même valeur de 1 dans la table. La valeur initiale de l'influence est ainsi perdue. La même critique peut être adressée à [Sedki et Bonneau de Beaufort, 2012a]. Toutefois, [Sedki et Bonneau de Beaufort, 2012b] étend cette méthode en traitant spécifiquement ce cas particulier en créant une exception pour les sommets ayant un seul parent. Cependant, le problème reste toujours présent pour les autres sommets.

Dans le cadre des réseaux bayésiens, le modèle du OU-bruité [Lemmer et Gossink, 2004] amène à calculer la table à partir de probabilités conditionnelles individuelles. Dans ce modèle, les variables aléatoires doivent être binaires et l'influence combinée de plusieurs parents n'importe pas. Autrement dit, l'influence combinée de deux concepts sur le même concept n'est pas plus importante que si on considère indépendamment l'influence de chaque concept. Ceci correspond au modèle des cartes cognitives dans le sens où la valeur de l'influence d'un concept est donnée en ignorant celles des autres influences ayant pour effet le même concept. Cependant, avec le modèle du OU-bruité, il est nécessaire de supposer que les probabilités données correspondent au cas où on fixe un parent à une certaine valeur et tous les autres parents à la valeur opposée. Ceci signifie qu'il faut donner des valeurs de la forme $\mathbb{P}(B = + | A_1 = -, \dots, A_{i-1} = -, A_i = +, A_{i+1} = -, \dots, A_n = -)$. Or, ceci n'est pas cohérent avec le fait que la notion d'influence des cartes cognitives est indépendante des autres parents.

[Das, 2004] utilise une moyenne pondérée de plusieurs valeurs. Ces valeurs et poids sont donnés par un expert. Chaque valeur d'expert représente la probabilité d'un fils sachant un seul de ses parents. Chaque poids représente la force relative de l'influence d'un parent. Cette méthode est adéquate pour les cartes cognitives. La question posée à l'expert est en effet : « Sachant que la valeur du parent Y est y , compatible avec les valeurs des autres parents, quelle serait la répartition des probabilités des valeurs du fils X ? ». La notion de *compatible* signifie par exemple qu'un parent Y_i de valeur y_i est compatible avec un autre parent Y_j de valeur y_j si, dans l'esprit de l'expert, la valeur $Y_i = y_i$ peut cohabiter avec la valeur $Y_j = y_j$ [Das, 2004]. Ce fait permet à l'expert de donner une valeur sans prendre en compte les autres informations.

Pour calculer la moyenne pondérée définie dans [Das, 2004], pour un sommet ayant n parents, on a besoin :

- d'un ensemble de poids w_i pour chaque parent tel que $\sum_{i=1}^n w_i = 1$: ce poids permet de quantifier l'importance relative de chaque parent et donc de chaque valeur lors du calcul de la probabilité conditionnelle ;
- d'une distribution de probabilités $\{\dots, \mathbb{P}(x^j | \{\text{Comp}(Y_i = y_i)\}), \dots\}$ où x^j est la valeur de la variable X : la notation $\text{Comp}(Y_i = y_i)$ signifie que la valeur

de la variable Y_i est y_i et que les valeurs des autres parents sont compatibles avec cette valeur.

Les probabilités conditionnelles sont alors données par :

$$\mathbb{P}(x^j | y_1, \dots, y_n) = \sum_{i=1}^n w_i \times \mathbb{P}(x^j | \{\text{Comp}(Y_i = y_i)\}).$$

Nous appliquons ce système pour les concepts d'une carte cognitive probabiliste ayant un nombre quelconque de parents. Pour cela, on considère un concept B ayant n parents A_i , chacun d'entre eux amenant respectivement une valeur d'influence de α_i (figure 5.7). Comme présenté précédemment, on a besoin de diverses

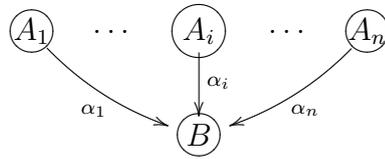


FIGURE 5.7 – Une carte cognitive figurant un concept avec plusieurs parents.

informations pour calculer la moyenne pondérée.

- Dans une carte cognitive, il n'est pas possible d'indiquer que l'influence d'un concept est plus importante que celle d'un autre. On considère donc que les valeurs des influences sont de même importance et ont donc le même poids $w_i = \frac{1}{n}$.
- On considère que chaque valeur d'expert est spécifiée par une valeur d'influence. Nous avons établi un lien entre la valeur d'influence α et une probabilité conditionnelle telle que $\mathcal{F}(\alpha) = \mathbb{P}(X_B = + | X_A = +) - 0.5$. Les valeurs d'expert étant des probabilités conditionnelles, nous les définissons telles que $\mathbb{P}(X_B = + | \{\text{Comp}(X_{A_i} = +)\}) = 0.5 + \mathcal{F}(\alpha_i)$. Une telle définition généralise donc les définitions 5.25 et 5.26. Puisque nous avons spécifié que les probabilités conditionnelles étaient complémentaires (définition 5.19), nous définissons également $\mathbb{P}(X_B = + | \{\text{Comp}(X_{A_i} = -)\}) = 0.5 - \mathcal{F}(\alpha_i)$.

La question à se poser lors de la construction d'une carte cognitive probabiliste pour renseigner une valeur d'influence serait alors : « Sachant que A augmente, cette augmentation étant compatible avec les états des autres parents de B , comment la probabilité que B augmente doit-elle augmenter ? ».

On peut maintenant calculer la table de probabilités pour un concept ayant un nombre quelconque de parents.

Définition 5.27 (Table de probabilités pour un concept ayant un nombre quelconque de parents)

Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste définie sur un ensemble de valeurs I . Soit \mathcal{F} une fonction de conversion pour I . Soit $B \in C$ un concept et X_B la variable aléatoire qui lui est associée. Soient $A_i \in \mathcal{C}(B)$ les parents de B , chacun d'eux étant associé à une variable aléatoire X_{A_i} . On note, pour chaque A_i :

- $\alpha_i = \text{label}((A_i, B))$;
- a_i la valeur de la variable aléatoire associée à A_i .

La table de probabilités de X_B est :

\mathbb{P}	...	$X_{A_1} = a_1, \dots, X_{A_n} = a_n$...
$X_B = +$		$0.5 + \frac{1}{n} \sum_{i=1}^n c(a_i)$	
$X_B = -$		$0.5 - \frac{1}{n} \sum_{i=1}^n c(a_i)$	

$$\text{où } c(a_i) = \begin{cases} \mathcal{F}(\alpha_i) & \text{si } a_i = + \\ -\mathcal{F}(\alpha_i) & \text{si } a_i = - \end{cases}$$

Cette définition généralise trivialement les définitions 5.25 et 5.26. De plus, elle vérifie la proposition 5.1 qui étend la définition 5.19 pour un nombre quelconque de parents.

Proposition 5.1. *Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste. Soit $B \in C$ un concept et X_B la variable aléatoire qui lui est associée. Soient $A_i \in \mathcal{C}(B)$ les parents de B , chacun d'eux étant associé à une variable aléatoire X_{A_i} . Soit a_i une valeur de la variable X_{A_i} et \bar{a}_i la valeur opposée.*

On a :

$$\mathbb{P}(X_B = + | X_{A_1} = a_1, \dots, X_{A_n} = a_n) = \mathbb{P}(X_B = - | X_{A_1} = \bar{a}_1, \dots, X_{A_n} = \bar{a}_n)$$

Démonstration. D'après la définition 5.27, on a :

$$\begin{aligned} \mathbb{P}(X_B = + | X_{A_1} = a_1, \dots, X_{A_n} = a_n) &= 0.5 + \frac{1}{n} \sum_{i=1}^n c(a_i) \\ &= 0.5 + \frac{1}{n} \sum_{\substack{i=1 \\ a_i=+}}^n c(a_i) + \frac{1}{n} \sum_{\substack{i=1 \\ a_i=-}}^n c(a_i) \\ &= 0.5 + \frac{1}{n} \sum_{\substack{i=1 \\ a_i=+}}^n \mathcal{F}(\alpha_i) + \frac{1}{n} \sum_{\substack{i=1 \\ a_i=-}}^n -\mathcal{F}(\alpha_i) \\ &= 0.5 + \frac{1}{n} \sum_{\substack{i=1 \\ a_i=+}}^n \mathcal{F}(\alpha_i) - \frac{1}{n} \sum_{\substack{i=1 \\ a_i=-}}^n \mathcal{F}(\alpha_i) \end{aligned}$$

On a également :

$$\begin{aligned} \mathbb{P}(X_B = - | X_{A_1} = \bar{a}_1, \dots, X_{A_n} = \bar{a}_n) &= 0.5 - \frac{1}{n} \sum_{i=1}^n c(\bar{a}_i) \\ &= 0.5 - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=+}}^n c(\bar{a}_i) - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=-}}^n c(\bar{a}_i) \\ &= 0.5 - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=-}}^n c(\bar{a}_i) - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=+}}^n c(\bar{a}_i) \\ &= 0.5 - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=-}}^n \mathcal{F}(\alpha_i) - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=+}}^n -\mathcal{F}(\alpha_i) \\ &= 0.5 - \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=-}}^n \mathcal{F}(\alpha_i) + \frac{1}{n} \sum_{\substack{i=1 \\ \bar{a}_i=+}}^n \mathcal{F}(\alpha_i) \end{aligned}$$

L'égalité devient alors triviale. □

Exemple 5.13. Nous considérons le concept S de CM6 (exemple 5.7). On reprend à nouveau la fonction de conversion $\mathcal{F}(\alpha) = \frac{\alpha}{2}$. On calcule, pour chaque combinaison possible des variables parentes de S une probabilité conditionnelle. Ainsi, on calcule par exemple :

$$\mathbb{P}(X_S = + | X_J = -, X_V = +) = 0.5 + \frac{1}{2}(-\mathcal{F}(0.6) + \mathcal{F}(-0.1)) = 0.325$$

Ainsi, la table de probabilités de la variable aléatoire associée au concept S est :

\mathbb{P}	$X_J = +, X_V = +$	$X_J = +, X_V = -$	$X_J = -, X_V = +$	$X_J = -, X_V = -$
$X_S = +$	0.625	0.675	0.325	0.375
$X_S = -$	0.375	0.325	0.675	0.625

Traduction complète

Nous disposons à présent de toutes les informations nécessaires pour représenter une carte cognitive en un réseau bayésien. Avant toute chose, nous devons nous assurer que tous les concepts peuvent être associés à une information quantifiable. Il faut également s'assurer que la carte cognitive ne contient aucun circuit. Si c'est malgré tout le cas, il est possible de construire différemment une carte cognitive de manière à en supprimer les circuits. [Nadkarni et Shenoy, 2001; Nadkarni et Shenoy, 2004] décrivent ainsi comment obtenir une structure de carte adéquate pour un réseau bayésien. Le meilleur moyen d'y parvenir est de discuter avec le designer de la carte pour qu'il explique quelle est la signification des liens afin de supprimer les redondances ou les inconsistances. Par exemple, il est relativement courant pour un designer de placer une influence directe d'un concept vers un autre alors qu'un troisième concept est intermédiaire à l'action de l'un sur l'autre. Il arrive également parfois qu'il y ait confusion entre le raisonnement déductif et le raisonnement abductif. Si, après ces modifications, la carte cognitive contient toujours un circuit, il est possible de créer un nouveau concept qui représente un concept de ce circuit au temps suivant. On représente ainsi un concept au temps t et le même concept au temps $t + 1$. Ceci permet d'évaluer l'évolution d'un système au cours du temps, à la manière des cartes cognitives floues dynamiques (voir section 5.1.2).

Définition 5.28 (Traduction d'une carte cognitive en un réseau bayésien)

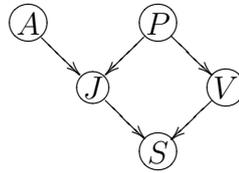
Soient $C = \{C_1, \dots, C_m\}$ un ensemble de concepts. Chacun de ces concepts est associé à une variable aléatoire X_{C_i} à deux valeurs, $+$ et $-$. Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste définie sur un ensemble de valeurs I .

Un réseau bayésien représentant CM est un réseau bayésien $G = (V, E)$ tel que :

- chaque $V_i \in V$ est un sommet associé la variable aléatoire X_{C_i} ;
- $E = \{(V_i, V_j) / (C_i, C_j) \in A\}$;
- $\mathbb{P}(X_{C_i} = + | X_{C'_1} = c'_1, \dots, X_{C'_n} = c'_n) = 0.5 + \frac{1}{n} \sum_{j=1}^n c(c'_j)$,

$$\text{avec } C'_j \in \mathcal{C}(C_i) \text{ et } c(c'_j) = \begin{cases} \mathcal{F}(\text{label}((C'_j, C_i))) & \text{si } c'_j = + \\ -\mathcal{F}(\text{label}((C'_j, C_i))) & \text{si } c'_j = - \end{cases}$$

Exemple 5.14. On effectue la traduction complète de CM6 (exemple 5.7) en un réseau bayésien. CM6 n'ayant pas de circuit, il n'est pas nécessaire de l'adapter pour la traduction. La structure graphique du réseau bayésien est équivalente à celle de la carte cognitive.



On associe une variable aléatoire à chaque concept de CM6. Il faut renseigner les tables de probabilités de chacune des variables. On définit pour cela une fonction de conversion, ici à nouveau $\mathcal{F}(\alpha) = \frac{\alpha}{2}$.

\mathbb{P}	
$X_A = +$	0.5
$X_A = -$	0.5

\mathbb{P}	
$X_P = +$	0.5
$X_P = -$	0.5

\mathbb{P}	$X_P = +$	$X_P = -$
$X_V = +$	0.9	0.1
$X_V = -$	0.1	0.9

\mathbb{P}	$X_A = +, X_P = +$	$X_A = +, X_P = -$	$X_A = -, X_P = +$	$X_A = -, X_P = -$
$X_J = +$	0.925	0.525	0.475	0.075
$X_J = -$	0.075	0.475	0.525	0.925

\mathbb{P}	$X_J = +, X_V = +$	$X_J = +, X_V = -$	$X_J = -, X_V = +$	$X_J = -, X_V = -$
$X_S = +$	0.625	0.675	0.325	0.375
$X_S = -$	0.375	0.325	0.675	0.625

On remarque quelques différences entre les nombres obtenus dans l'exemple 5.14, représentant une carte cognitive probabiliste, et ceux de l'exemple 5.1, représentant un réseau bayésien classique. La première différence évidente est le fait que les sommets sans parent n'ont pas les mêmes probabilités a priori que dans l'exemple 5.14,. Les cartes cognitives ne disposent en effet d'aucune information permettant d'obtenir ce renseignement. Néanmoins, cette information pourrait éventuellement être donnée si la carte cognitive disposait d'un « super-parent » qui serait le parent de tous les concepts sans parent. Ce concept distribuerait les probabilités a priori via les valeurs d'influences entre lui et les autres concepts.

On remarque également que dans la table de probabilités de la variable aléatoire associée à V , on n'a plus de valeurs totales comme 0 et 1, mais des valeurs plus modérées. Ceci vient du fait que la valeur de l'influence entre P et V n'est ni 1 ni -1 . En effet, la sémantique des concepts n'est pas la même dans les deux exemples. Le domaine de définition des variables de l'exemple 5.1 était $\{\mathbf{vrai}, \mathbf{faux}\}$, tandis que celui des variables de l'exemple 5.14 est $\{+, -\}$. Les valeurs n'ont donc par conséquent pas la même signification. Quand on disait auparavant que « s'il pleut, alors je suis sûr que l'herbe de mon jardin est humide » on dit ici « s'il pleut de plus en plus, alors il est fort probable que mon jardin soit de plus en plus humide ». C'est une des raisons pour lesquelles la table de probabilités de la variable aléatoire associée à J est aussi différente.

Cependant, on note également que les valeurs de $\mathbb{P}(X_J = + | X_A = +, X_P = -)$ et $\mathbb{P}(X_J = + | X_A = -, X_P = +)$ sont deux fois plus petites que $\mathbb{P}(J|A, \overline{P})$ et $\mathbb{P}(J|\overline{A}, P)$ respectivement. Le concept P représente l'augmentation ou la diminution de la pluie, pas la présence ou l'absence de pluie. Le fait qu'il pleuve de moins en moins provoque donc une diminution de l'humidité du jardin. Ceci peut sembler contre-intuitif puisque même s'il pleut peu, l'humidité devrait continuer d'augmenter. En réalité, la sémantique associée au modèle suppose qu'en l'absence d'influence de tout autre concept, chaque concept tend à atteindre un état d'équilibre, c'est-à-dire qu'il est équiprobable que la pluie augmente ou qu'elle diminue. Ainsi, d'autres causes de la diminution de l'humidité ne sont pas représentées, comme par exemple le degré d'ensoleillement. Pour mieux prendre en compte ce facteur, on pourrait l'ajouter à la carte cognitive.

Comparaison entre l'influence propagée probabiliste et la probabilité conditionnelle

Après avoir codé une carte cognitive en un réseau bayésien, nous devons vérifier que l'influence propagée probabiliste définie dans la section 5.2.2 correspond bien à une méthode d'inférence dans le réseau bayésien correspondant. Comme nous l'avons vu, une influence directe d'un concept A vers un concept B donne une information sur $\mathbb{P}(X_B = + | X_A = +)$. De la même manière, l'influence propagée de A sur B doit être liée à $\mathbb{P}(X_B = + | X_A = +)$.

Nous avons déjà montré dans la section 5.1.1 que le modèle des réseaux bayésiens causaux est plus proche des cartes cognitives que le modèle des réseaux bayésiens classiques. En effet, étudier l'influence d'un concept dans une carte est similaire à intervenir sur la valeur d'une variable dans un réseau. Dans le modèle des réseaux bayésiens causaux, quand on étudie les conséquences d'une augmentation de A , on s'intéresse alors à la valeur de $\mathbb{P}(X_B = + | \text{do}(X_A = +))$. L'influence propagée de A sur B est donc liée à $\mathbb{P}(X_B = + | \text{do}(X_A = +))$. Il y a en effet une relation directe entre la définition de l'influence propagée probabiliste partielle dans une carte cognitive et la probabilité conditionnelle dans le réseau bayésien correspondant. On en déduit une relation entre l'influence propagée probabiliste et une probabilité conditionnelle. Cette relation est donnée par le théorème 1.

Théorème 1. *Soit $CM = (C, A, \text{label})$ une carte cognitive probabiliste. Soit un réseau bayésien causal traduisant cette carte cognitive avec une loi de probabilité \mathbb{P} . On note les variables aléatoires du réseau bayésien de la même manière que les concepts auxquels ils sont associés. Soient $c_1, c_2 \in C$ deux concepts. Soient X_{c_1} et X_{c_2} les variables aléatoires associées respectivement aux concepts c_1 et c_2 .*

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \mathbb{P}(X_{c_2} = + | \text{do}(X_{c_1} = +)) - 0.5$$

On en déduit que :

$$\mathcal{I}_{\mathbb{P}}(c_1, c_2) = \begin{cases} \mathcal{F}^{-1} \left(\mathbb{P}(X_{c_2} = + | \text{do}(X_{c_1} = +)) - 0.5 \right) & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \mathcal{F}^{-1} \left(\frac{\mathbb{P}(X_{c_2} = + | \text{do}(X_{c_1} = +)) - 0.5}{\sum_{P \in \mathcal{P}_{c_1, c_2}} \text{part}(P)} \right) & \text{sinon} \end{cases}$$

La preuve de ce théorème est donnée en annexe B.

Influence taxonomique probabiliste

Nous n'avons pas proposé d'opérateur pour une éventuelle influence taxonomique probabiliste. Cette influence, tout comme l'influence propagée probabiliste d'un concept sur un autre doit être liée à une méthode d'inférence dans le réseau bayésien correspondant. Une influence taxonomique s'exprime entre des concepts d'une taxonomie, c'est-à-dire entre des concepts qui en regroupent plusieurs autres. L'inférence correspondante dans le réseau bayésien devrait donc être une probabilité conditionnelle à plusieurs variables.

Reste à savoir si ces variables sont liées par une conjonction ou une disjonction. Si on considère que la taxonomie est exhaustive, un concept parent est défini comme étant l'union de ses enfants. Il semblerait donc logique de placer une disjonction entre les variables de la probabilité conditionnelle.

Nous n'avons trouvé aucun travail décrivant l'intervention sur plusieurs variables à la fois dans un même réseau bayésien causal. Or, on a nécessairement besoin d'évaluer cette intervention multiple puisqu'une disjonction nécessite d'étudier l'action conjointe des variables. En effet, on doit prendre en compte toutes les actions conjointes des différentes variables de la condition : seules, deux à deux, trois à trois... Pour la même raison, si de telles probabilités se manipulent comme les probabilités conditionnelles classiques, alors la complexité du calcul de l'influence taxonomique devrait être exponentielle.

Il serait aussi intéressant d'étudier les réseaux bayésiens hiérarchiques [Ben Messaoud *et al.*, 2009; Ben Messaoud, 2012]. Dans ce modèle, une hiérarchie permet de regrouper plusieurs variables aléatoires d'un réseau bayésien au sein d'autres variables aléatoires, à la manière de la taxonomie d'une carte cognitive. Ce modèle a été utilisé essentiellement pour l'apprentissage de réseaux bayésiens ainsi que pour l'aide à la visualisation. Le calcul de probabilités des variables de la hiérarchie par rapport aux réseaux bayésiens n'a pas été étudié mais le modèle peut servir de base pour notre influence taxonomique.

5.3 Sémantique des concepts et des influences

Dans cette section, nous discutons du meilleur choix à adopter concernant le modèle de carte cognitive. Ce choix se fait en fonction du problème considéré ainsi que de divers autres facteurs.

Le choix du modèle de carte cognitive se résume souvent à la sémantique d'une influence d'un concept sur un autre et de sa valeur. Nous considérons donc une carte élémentaire figurant une simple influence d'un concept A sur un concept B ayant pour valeur α .

$$\textcircled{A} \xrightarrow{\alpha} \textcircled{B}$$

La manière la plus simple de lire cette carte est de dire que « A influence B de α ». Cependant, cette information ne nous permet pas de déduire grand chose. Tout ce qu'on peut dire c'est que A modifie ou apporte quelque chose à B et que cette modification ou apport est quantifié par α . Nous devons donc donner une interprétation formelle à cette notion d'influence. La sémantique associée à

ce concept ainsi que la définition de l'algèbre manipulant les valeurs d'influence en découlent naturellement. De cette sémantique peuvent également être déduites d'autres relations entre A et B .

Nous allons donc répondre aux questions suivantes :

- Que signifie « A influence B de α » ?
- Que représentent les concepts A et B ?
- Comment combiner les valeurs d'influence ?
- Quelles autres relations entre A et B peut-on déduire ?

Chacun des modèles que nous avons présentés apporte une réponse différente à chacune de ces questions.

5.3.1 Cartes cognitives floues

Avec le modèle des cartes cognitives floues de B. Kosko, « A influence B de α » signifie formellement que « le degré d'inclusion de B dans A est de α ». α étant un degré d'inclusion, il doit être strictement positif. Si ce n'est pas le cas, la structure de la carte doit être modifiée pour le calcul des influences. De telles cartes sont nécessairement sans circuit.

Les concepts A et B sont des ensembles flous. Chacun de ces concepts est défini comme étant l'union d'une certaine quantité, de la quantité inverse et d'un modificateur. Un concept opposé peut être défini pour chacun des concepts. Ce concept est pris en considération pour représenter les influences dont la valeur est négative.

Pour combiner les valeurs d'influence et calculer l'influence propagée de A sur B , on considère tout d'abord tous les chemins permettant de lier les deux concepts. Cette opération est simple puisqu'une carte floue ne contient pas de circuit. Pour chacun des chemins, on combine les valeurs des influences de ce chemin grâce à une t -norme. Enfin, on combine les valeurs de chacun des chemins grâce à une t -conorme.

Comme expliqué précédemment, les concepts opposés sont pris en considération pour les influences dont la valeur est négative. Si on note, pour un concept A son concept opposé $\sim A$, alors une influence de A sur B de α signifie qu'on a également une influence de $\sim A$ sur $\sim B$ de α . On ne peut en revanche rien déduire de l'influence de A sur $\sim B$, ni de $\sim A$ sur B .

5.3.2 Cartes cognitives floues dynamiques

Avec le modèle des cartes cognitives floues dynamiques, « A influence B de α » signifie formellement que « A augmente B de α ». Contrairement aux cartes cognitives floues classiques, α peut être négatif sans engendrer de modification de la structure de la carte pour le calcul des influences. De telles cartes peuvent contenir un circuit.

Les concepts A et B sont donc associés à une certaine quantité qui augmente ou diminue en fonction de diverses influences. Si la quantité à laquelle ils sont associés est suffisamment élevée, ils peuvent influencer d'autres concepts. Chaque concept peut donc être dans deux états : l'un influençant et l'autre non.

Pour combiner les valeurs d'influence et calculer l'influence propagée de A sur B , une algèbre plus complexe que celle du modèle de Kosko est mise en place prenant en compte les influences entre les concepts, les valeurs de ces influences et l'état de chacun des concepts. Cette algèbre fonctionne étape par étape en fonction du temps. À chaque temps, un concept n'influence donc que les concepts qu'il influence directement. Le temps n'est donc pas le même que le temps physique mais représente simplement une itération supplémentaire lors du calcul des influences. On peut également étudier l'évolution du système représenté par la carte cognitive au bout d'un temps infini, à la manière d'une suite.

Avec un tel modèle, il n'y a pas vraiment d'autres informations qui peuvent être déduites entre deux concepts à partir d'une simple influence. On peut toutefois, en analysant la structure de la carte, déduire qu'il existe des influences indirectes d'un concept sur un autre, comme pour n'importe quel modèle de carte cognitive dont les influences peuvent être combinées sur un même chemin.

5.3.3 Qualitative Probabilistic Networks (QPN)

Avec le modèle des QPN, la signification formelle de « A influence B de α » n'est pas triviale. En effet, dans le modèle de base, α n'est pas une valeur mais un signe. Une influence n'est qu'une contrainte dont l'expression formelle dépend de ce signe. Ainsi, si $\alpha = +$, « A influence B de α » signifie formellement que « en supposant les autres parents influençant B dans le même état, plus A est grand, plus la probabilité que B soit grand est importante ». Dans un QPN, il n'y a nécessairement aucun circuit.

Les concepts A et B sont associés à des variables dont le domaine de définition est totalement ordonné. Les valeurs de ces variables n'ont pas besoin d'être fournies : l'ordre seul suffit à exprimer la relation d'influence entre A et B .

Pour combiner les valeurs d'influence et calculer l'influence propagée de A sur B , il est nécessaire de considérer les chemins permettant de lier les deux concepts. Ces chemins ne suivent cependant pas nécessairement l'ordre des arcs. Les valeurs sont combinées sur un même chemin puis sur l'ensemble des chemins grâce à deux opérateurs définis par M. Wellman. Ce modèle a l'avantage de donner un formalisme fort au modèle original de R. Axelrod.

Diverses informations non représentées explicitement dans un QPN peuvent être déduites du formalisme associé à ce modèle. La plupart sont simplement des informations implicites. Ainsi, si on effectue l'hypothèse du monde clos, une absence d'arc entre deux concepts est équivalente à la présence d'un arc portant le signe 0. En règle générale, en l'absence de toute information contraire, il y a toujours une relation ayant pour signe ? entre deux concepts. De plus, un arc entre deux concepts portant un certain signe signifie qu'il existe également un arc implicite dans l'autre sens ayant le même signe.

5.3.4 Cartes cognitives probabilistes

Avec le modèle des cartes cognitives probabilistes défini dans la section 5.2, « A influence B de α » signifie formellement que si A augmente, alors la probabilité que B augmente est fonction de α , en supposant les autres parents de B dans un état compatible avec l'augmentation de A ». Cette « fonction de α » est définie par la fonction de conversion. De telles cartes sont nécessairement sans circuit.

Les concepts A et B sont associés à des variables aléatoires binaires. Chaque concept est associé à une échelle de valeurs représentant une certaine quantité du monde réel. Les deux valeurs de chaque variable représentent un état d'augmentation ou un état de diminution de la valeur de cette échelle. Ce modèle considère que, à la manière des cartes cognitives floues, la probabilité que le concept B augmente sachant que le concept A augmente est égale à celle que B diminue sachant que A diminue.

Pour combiner les valeurs d'influence et calculer l'influence propagée de A sur B , on considère tout d'abord tous les chemins permettant de lier les deux concepts. Pour chacun des chemins, on utilise un opérateur pour calculer sa valeur grâce aux valeurs des influences de ce chemin converties via la fonction de conversion et sa part. Le calcul de l'influence propagée consiste alors à réaliser une moyenne pondérée des valeurs de chaque chemin par leurs parts respectives. Enfin, on convertit cette valeur grâce à la fonction de conversion inverse de manière à retrouver une valeur appartenant à l'ensemble de valeurs sur lequel est définie la carte cognitive. Ce calcul correspond à la loi de probabilité associée à un réseau bayésien causal représentant une telle carte cognitive.

Les informations d'une telle carte étant représentables par un réseau bayésien, les mécanismes d'inférence associés à ce modèle peuvent être appliqués à la carte pour déduire de nouvelles informations.

Conclusion

Nous avons présenté différents modèles de cartes cognitives ou proches des cartes cognitives et étudié leurs différences sémantiques et structurelles. Nous avons également proposé un nouveau modèle de carte cognitive dans lequel les valeurs des influences sont apparentées à des probabilités. Nous avons montré la pertinence de ce modèle en montrant qu'il était équivalent à une représentation de la carte en un réseau bayésien. De plus, même si ce n'était pas le but initial, nous pouvons aussi voir le travail présenté dans cet article comme une première étape pour l'apprentissage de réseaux bayésiens lorsque l'information est exprimée par un utilisateur grâce à une carte cognitive, une carte cognitive étant un modèle facile pour décrire de la connaissance informelle.

Nous n'avons pas étudié le mécanisme inverse, à savoir représenter un réseau bayésien par une carte cognitive. Les réseaux bayésiens contiennent beaucoup plus d'informations que les cartes cognitives et, par conséquent, exprimer un réseau bayésien par une carte cognitive entraîne forcément une perte d'information ou bien nécessite de construire une carte cognitive si complexe qu'elle en perdrait sa concision, un des principaux intérêts du modèle. Cependant, une carte concise

serait intéressante puisqu'elle permettrait de présenter à un utilisateur un aperçu d'un réseau bayésien d'une manière moins formelle. Les cartes cognitives deviendraient alors un moyen efficace de visualiser les données d'un réseau bayésien. La principale difficulté est de savoir quelles informations supprimer.

Lors de la construction d'une carte cognitive en un réseau bayésien, pour le cas d'un sommet ayant plusieurs parents, nous avons utilisé une moyenne. Nous pourrions également étudier ce que donneraient les modèles log-linéaires [Christensen, 1997]. Ces modèles permettent de décomposer une loi de probabilité d'un sommet et de ses parents en décrivant les interactions entre les sommets. Si ces modèles sont applicables pour aider à la représentation d'une carte cognitive, le modèle de carte qui en découlerait serait probablement différent de celui des cartes cognitives probabilistes. En étudiant les différences serait intéressant.

Les cartes cognitives sont un cas particulier des QPN selon l'inventeur de ce modèle. Différentes extensions du modèle des QPN ont été proposées pour que ceux-ci manipulent d'autres signes que $+$, 0 , $-$ et $?$, notamment pour qualifier la force de l'influence. [Renooij et van der Gaag, 1999] propose ainsi d'ajouter des valeurs supplémentaires telles que $++$ ou $--$ qui permettent de représenter différents niveaux de force d'influence. [Renooij et van der Gaag, 2002] utilise quant à lui des intervalles de valeurs numériques et [Renooij *et al.*, 2003] utilise des valeurs Kappa [Spohn, 1988]. Il serait intéressant de définir des cartes cognitives qui utiliseraient de telles valeurs.

Enfin, on peut également étudier d'autres algèbres de valeurs qui pourraient être appliquées aux cartes cognitives. Les possibilités [Dubois et Prade, 1988] pourraient faire office de valeur pour les influences des cartes cognitives. En effet, elles sont issues de la logique floue et elles évaluent les incertitudes, tout comme les probabilités, en donnant une valeur de possibilité d'un événement ou de l'état d'une variable. Les possibilités sont une autre manière que la probabilité d'attribuer une valeur d'incertitude à un événement. Elles utilisent pour cela une représentation ordinaire de l'incertitude. Elles peuvent d'ailleurs être utilisées pour définir des réseaux, à la manière des réseaux bayésiens [Benferhat *et al.*, 1999]. Le formalisme d'une carte cognitive pourrait également être basé sur les fonctions de croyance [Shafer, 1976]. Les fonctions de croyance généralisent les probabilités et les possibilités et peuvent être utilisées elles aussi pour définir des réseaux [Ben Yaghlane *et al.*, 2003]. Certains de ces réseaux peuvent même prendre en compte la notion de causalité et d'intervention [Boukhris *et al.*, 2011]. On peut enfin étudier ce que donneraient d'autres logiques multi-valuées [Dubois et Prade, 2001].



6

Systeme

Introduction

Dans ce chapitre, nous présentons une implémentation des différents mécanismes introduits au cours de notre thèse. Un logiciel a en effet été développé parallèlement au travail de recherche de manière à tester les diverses notions introduites. Il se nomme *VSPCC* est disponible à l'adresse <http://forge.info.univ-angers.fr/~ledorze/vspcc>. Ce logiciel nous a également permis de réaliser des tests sur des cartes issues du monde réel telles que des cartes issues du monde de la pêche [Christiansen, 2011] ainsi que de nous assurer que les mécanismes que nous avons introduits s'appliquent sur des cartes complexes et de grandes tailles.

Le prototype que nous avons conçu permet de construire un ensemble de cartes cognitives, chacune d'entre elles ayant été réalisée par un designer différent. Ces cartes sont définies sur une même taxonomie, qui peut elle aussi être construite via le logiciel. Sur chacune de ces cartes, nous pouvons calculer l'influence propagée d'un concept sur un autre ou l'influence taxonomique sur des concepts de la taxonomie. On peut également construire une vue de la carte cognitive en utilisant la taxonomie ou bien valider cette carte selon différents critères de qualité. Enfin, on peut effectuer une synthèse d'une partie ou de l'intégralité des cartes en fournissant un ordre de préférence sur les designers de ces cartes.

Dans la section 6.1, nous présentons comment nous avons implémenté le modèle des cartes cognitives (section 1.4 et chapitre 2). Nous présenterons également l'implémentation des notions de taxonomie, de carte taxonomique, de vue et d'échelle. Nous présentons et expliquons notre choix d'implémentation du mécanisme de calcul des influences (chapitre 2) dans la section 6.2. Dans la section 6.3, nous présentons la modélisation des notions relatives à la validation de cartes cognitives (chapitre 3). Nous présentons l'implémentation et la modélisation des notions relatives à la synthèse de cartes cognitives, telles que les préférences (chapitre 4) dans la section 6.4. La section 6.5 est la dernière section décrivant nos choix d'im-

plémentation et est consacrée à l'interface graphique que nous avons développée pour manipuler aisément les différentes notions présentées précédemment. Enfin, dans la section 6.6, nous présentons quelques cas d'utilisation pour expliquer comment se servir du logiciel.

6.1 Cartes cognitives

Le logiciel a été programmé en Java ¹. Pour une meilleure organisation, le code du logiciel est découpé essentiellement en trois packages distincts :

model contient les classes métiers, c'est-à-dire les classes décrivant le modèle de base (carte cognitive, taxonomie...) ainsi que les différents mécanismes introduits dans cette thèse (validation, synthèse) ;

io contient le code permettant de lire ou stocker des cartes cognitives depuis ou dans un fichier ; contient également les descriptions des différents algorithmes ;

ui contient le code gérant l'interface graphique et l'interaction avec l'utilisateur.

Les classes de ces différents packages se font appel entre elles d'un package à l'autre, comme présenté à la figure 6.1.

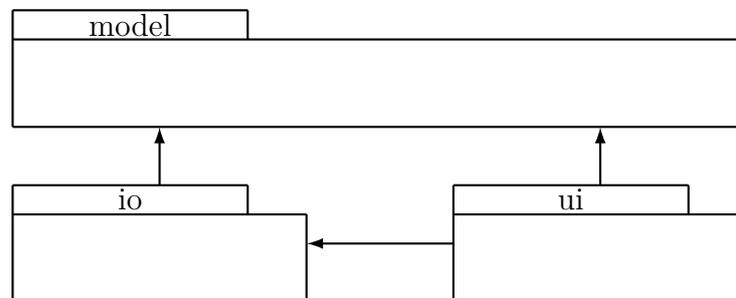


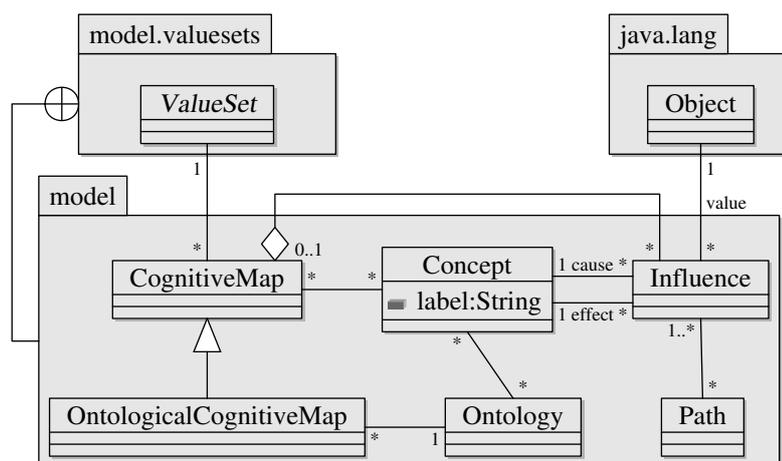
FIGURE 6.1 – Organisation générale des packages.

Le package **model** en lui-même est découpé en plusieurs parties. Directement au sein du package **model** se trouvent les classes principales, essentielles pour toutes les autres classes. Il est présenté dans la section 6.1.1. La définition des ensembles de valeurs des cartes cognitives est assez complexe et nécessite de nombreuses classes. Celles-ci sont donc réunies dans un sous-package à part, **model.valuesets**. Il est présenté dans la section 6.1.2.

6.1.1 Cartes cognitives et taxonomie

Le package **model** comporte les classes de base pour manipuler des cartes cognitives. Ces classes ainsi que leurs liens sont représentés sur le diagramme UML de la figure 6.2.

¹<http://www.java.net/>

FIGURE 6.2 – Diagramme de classes du package `model`.

Chaque concept (voir section 1.4.2) est représenté par une instance de la classe `Concept`. Cette classe ne renseigne qu'un texte bref pour décrire le concept. Néanmoins, ce texte n'identifie pas le concept : plusieurs concepts peuvent avoir le même texte dans une même carte.

La classe `CognitiveMap` représente une carte cognitive (définition 1.6). Elle est définie sur un ensemble de valeurs, représenté par une instance de l'interface `ValueSet` (voir section 6.1.2). Une carte cognitive n'est pas définie sur un ensemble de concepts figé. Elle contient en réalité plusieurs influences qu'on ajoute au fur et à mesure de la construction de la carte. L'ensemble des concepts sur lequel elle est définie est calculé à partir des extrémités de ces influences.

Chaque influence est représentée par une instance de la classe `Influence`. Chaque instance est liée à deux concepts, l'un représentant sa cause et l'autre son effet. Elle est également liée à une valeur, représentée par une instance de la classe `Object`. Ce choix d'implémentation est plus amplement discuté dans la section 6.1.2.

Chaque chemin d'influence est représenté par une instance de la classe `Path` (définition 2.1). Celle-ci définit une liste d'influences telle que, pour chaque influence de cette liste, son concept-cause est le concept-effet de l'influence précédente et son concept-effet est le concept-cause de l'influence suivante. Deux méthodes existent pour accéder facilement au concept-source et au concept-destination du chemin, correspondant respectivement au concept concept-cause de la première influence et au concept-effet de la dernière. Cette classe est beaucoup utilisée par d'autres packages, notamment lors du calcul des influences.

La classe `Ontology` représente une taxonomie de concepts (définition 2.7). Le choix du changement de terme d'« ontologie » pour celui de « taxonomie » s'est fait assez tard. Par conséquent, l'ancienne dénomination a été conservée. Dans nos textes explicatifs, nous parlerons cependant toujours de « taxonomie ». Cette classe associe à chaque concept qu'elle ordonne un autre concept qui représente son parent dans la taxonomie et un ensemble d'autres concepts qui représentent ses enfants. À chaque ajout et suppression, on s'assure que la taxonomie reste consistante : il

est ainsi impossible de spécifier qu'un concept est son propre parent.

Enfin, la classe `OntologicalCognitiveMap` représente une carte cognitive taxonomique (définition 2.12). Nous avons ici aussi conservé l'ancienne dénomination. Cette classe hérite de `CognitiveMap`, ce qui permet de retrouver toutes ses méthodes. Elle est également définie sur une taxonomie de concepts. Au moment de l'instanciation de la classe, deux options sont possibles. On peut ainsi autoriser les concepts d'une carte taxonomique à ne pas être nécessairement minimaux. On peut également autoriser deux concepts comparables à être présents dans la carte. Dans la pratique, on ne choisit que la première option. Ceci semble est en désaccord avec la définition du modèle. En réalité, cette classe n'est pas réellement une carte cognitive taxonomique représente plutôt l'association d'une carte cognitive et d'une taxonomie, à la manière d'une carte attribuée (définition 4.2). Imposer que les concepts soient minimaux permet cependant de définir une carte cognitive taxonomique.

6.1.2 Ensembles de valeurs

Le package `model.valuesets` contient les différentes classes et interfaces représentant des ensembles de valeurs. Ces ensembles ne contiennent pas de valeurs à proprement parler mais indiquent simplement si une valeur est contenue dans cet ensemble ou non.

Organisation générale

Les interfaces et classes de base de ce package et leurs liens sont représentés dans le diagramme UML de la figure 6.3.

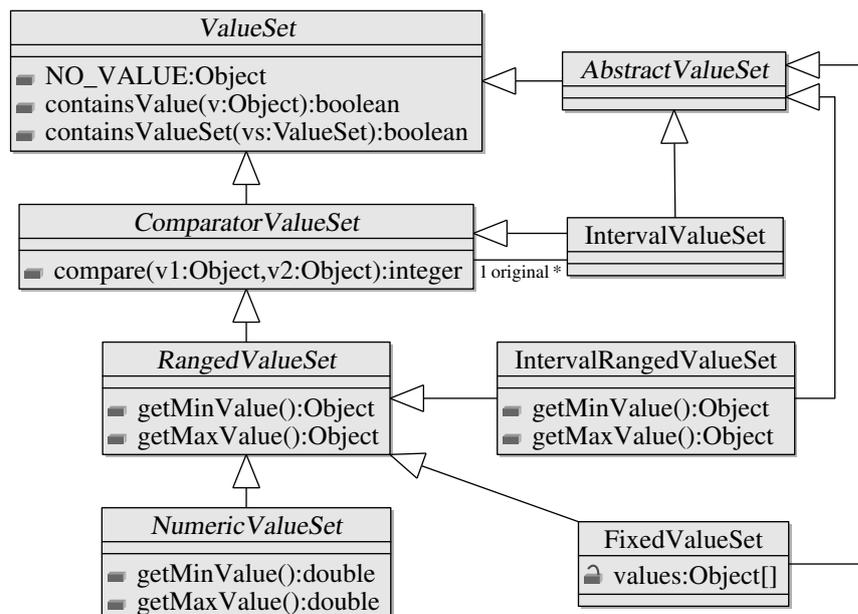


FIGURE 6.3 – Diagramme des principales classes du package `model.valuesets`.

Une valeur est une instance de la classe `Java Object`. Le choix de cette classe nous assure la plus grande généricité possible pour les valeurs d'une carte. Dans les faits, on utilise le plus souvent des instances de la classe `String` ou `Double`. Les classes représentant des ensembles de valeurs doivent donc disposer de différentes méthodes pour savoir si des valeurs appartiennent ou non à cet ensemble.

L'interface `ValueSet` est l'interface initiale, mère de toutes classes représentant des ensembles de valeurs. Un ensemble de valeurs est simplement défini par deux méthodes : l'une permet d'indiquer si une valeur appartient ou non à l'ensemble considéré. L'autre permet d'indiquer si un ensemble de valeurs en contient un autre. Cette méthode est particulièrement utile pour s'assurer que différents algorithmes sont compatibles entre eux. Cette interface fournit en outre une valeur spéciale, `NO_VALUE`. Cette valeur représente la valeur non-renseignée \emptyset (définition 4.14). Tout ensemble de valeur doit contenir cette valeur. Ce choix n'est pas en accord avec le modèle. Il était néanmoins essentiel pour permettre à l'utilisateur du logiciel en train de construire une carte de laisser des influences sans valeur pour ne les fournir qu'ensuite. Elle représente finalement une valeur par défaut.

L'interface `ComparatorValueSet` est un ensemble de valeurs dont les valeurs sont comparables. Il permet donc de définir une relation d'ordre sur les éléments de l'ensemble. Certains algorithmes tels que `min` ou `max` nécessitent en effet une relation d'ordre sur les valeurs d'un ensemble. La relation d'ordre n'est pas nécessairement totale : certaines valeurs peuvent être incomparables à toutes les autres, typiquement `NO_VALUE`.

L'interface `RangedValueSet` est un ensemble de valeurs borné, c'est-à-dire un ensemble de valeurs qui possède une valeur minimale et une valeur maximale. Certains algorithmes nécessitent également une telle contrainte. De plus, connaître les valeurs minimale et maximale d'un ensemble permet d'optimiser certains algorithmes.

Enfin, l'interface `NumericValueSet` représente un ensemble de valeurs numériques. Ces valeurs sont représentées par le type `Java double`. On redéfinit ainsi les méthodes de l'interface `RangedValueSet` telles que les valeurs minimale et maximale sont des `double` en utilisant la covariance du type de retour. Les valeurs minimale et maximale peuvent éventuellement être infinies.

Différentes classes implémentent ces interfaces. Ainsi, la classe `AbstractValueSet` implémente naïvement l'interface `ValueSet`. La seule valeur reconnue par cet ensemble est `NO_VALUE`. Pour savoir si un autre ensemble de valeurs est contenu dans cet ensemble, cette classe tente d'itérer sur les valeurs de l'autre ensemble et vérifie si chaque valeur lui appartient. Cette implémentation sert de base à tous les autres ensembles de valeurs.

La classe `FixedValueSet` représente un ensemble fini de valeurs. Ces valeurs sont listées au moment de la création de l'ensemble. L'ordre de ces valeurs est important puisqu'il définit la relation d'ordre sur ces valeurs. Les valeurs minimale et maximale sont respectivement la dernière et la première valeur.

La classe `IntervalRangedValueSet` représente un ensemble dont les valeurs appartiennent à un intervalle. La notion d'intervalle est implémentée par une classe qui n'est pas représentée ici. Elle est définie sur une relation d'ordre et sur deux valeurs représentant les bornes de cet intervalle.

Enfin, la classe `IntervalValueSet` est un ensemble de valeurs défini sur un autre ensemble, appelé ensemble original. Il accepte à la fois les valeurs de cet ensemble et n'importe quel intervalle défini sur ces valeurs. L'ensemble original doit donc nécessairement posséder une relation d'ordre de manière à pouvoir y définir des intervalles. Cet ensemble de valeurs permet ainsi de représenter l'ensemble de sortie de l'influence taxonomique de la définition 2.13.

Ensembles de valeurs courants

Les ensembles de valeurs présentés précédemment ne correspondent à aucun ensemble de valeurs présenté dans cette thèse. Ce ne sont en effet pour l'instant que des bases génériques pour définir des ensembles de valeurs utilisables. Les classes représentant les ensembles de valeurs liés à $\{+, -\}$ sont présentées dans le diagramme UML de la figure 6.4.

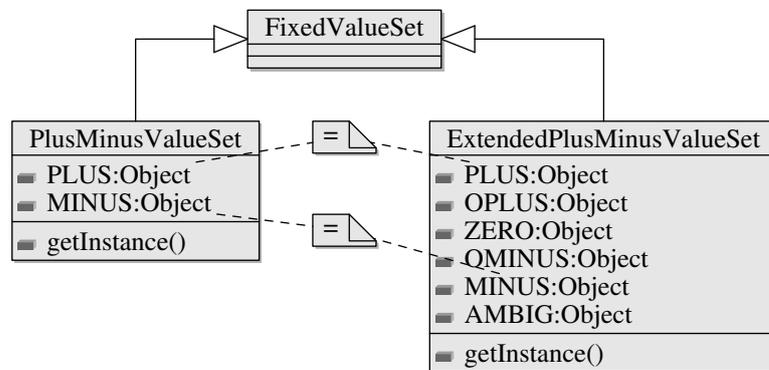


FIGURE 6.4 – Diagramme de classes des ensembles de valeurs liés à $\{+, -\}$.

La classe `PlusMinusValueSet` fait bien évidemment référence à l'ensemble de valeurs $\{+, -\}$. Cette classe-singleton est un ensemble fini de valeurs. Les attributs `PLUS` et `MINUS` sont des constantes de classe accessibles publiquement de manière à faciliter l'écriture du code et à comparer directement les valeurs.

La classe `ExtendedPlusMinusValueSet` représente quant à elle l'ensemble de valeurs $\{+, \oplus, 0, \ominus, -, ?\}$. Cette classe-singleton est elle aussi un ensemble fini de valeurs. Les attributs `PLUS`, `OPLUS`, `ZERO`, `OMINUS`, `MINUS` et `AMBIG` sont aussi accessibles publiquement. Les attributs `PLUS` et `MINUS` représentent les mêmes objets que ceux de la classe `PlusMinusValueSet`. Nous n'avons pas défini de classe exclusive pour l'ensemble $\{+, 0, -, ?\}$, pourtant cité quelquefois. Nous considérons en effet que déclarer une classe juste pour cet ensemble risquait de complexifier le code. La rareté de son utilisation pose finalement peu de problèmes.

Les classes représentant les ensembles numériques sont présentés dans le diagramme UML de la figure 6.5.

La classe `NumericIntervalValueSet` représente un intervalle de valeurs numériques, typiquement l'ensemble de valeurs $[-1; 1]$.

La classe `NumericFixedValueSet` représente un ensemble fini de valeurs numériques. Celles-ci sont automatiquement ordonnées selon l'ordre naturel. De tels

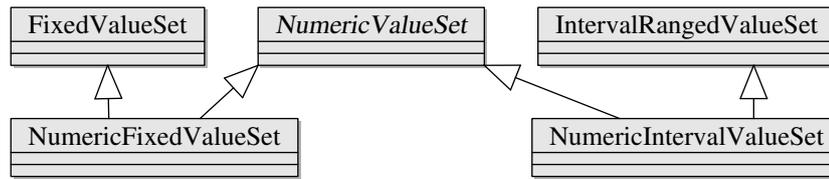


FIGURE 6.5 – Diagramme de classes des ensembles de valeurs numériques.

ensembles n'ont pas été présentés dans cette thèse mais il arrive parfois que des ensembles tels que $\{-3, -2, -1, 0, 1, 2, 3\}$ soient utilisés [Christiansen, 2011].

L'ensemble de valeurs $\{nul < faible < moyen < fort\}$ n'a pas de classe qui lui est dédié. Il peut en revanche être facilement représenté par une instance de la classe `FixedValueSet`.

6.2 Calcul d'influences

Le package `model.influences` est dédié à la définition des opérateurs de calcul d'influence. Pour présenter ces différentes définitions, nous introduisons d'abord la manière dont nous avons implémenté la notion d'algorithme dans la section 6.2.1. Nous présentons comment nous avons implémenté la notion de chemin minimal ainsi que les différentes manières de la définir dans la section 6.2.2. Nous présentons ensuite comment nous avons implémenté les différents opérateurs de calcul d'influence dans la section 6.2.3. Enfin, nous présentons le cas particulier de l'influence propagée probabiliste dans la section 6.2.4.

6.2.1 Algorithmes

Le calcul d'influences entre concepts peut être défini de différentes manières. L'implémentation doit donc être suffisamment générique pour s'adapter aux différentes définitions possibles. Cette généricité se retrouve dans de nombreux autres mécanismes introduits dans cette thèse. Pour cette raison, chacun de ces mécanismes doit se définir au sein de la hiérarchie d'interfaces représentée dans le diagramme UML de la figure 6.6.

L'interface `Algorithm` est une interface générique qui est l'interface-mère de tous les algorithmes. Cette interface est purement sémantique puisqu'elle ne nécessite de définir aucune méthode et ne contient aucun attribut.

L'interface `ValueSetAlgorithm` représente n'importe quel algorithme défini sur un ou plusieurs ensembles de valeurs. Typiquement, une classe implémentant cette interface définit une méthode dont les paramètres sont une ou plusieurs valeurs appartenant à un ou plusieurs ensembles de valeurs. Par exemple, l'opérateur \wedge (définition 2.30) définit une méthode qui manipule différentes valeurs $+$ et $-$ et renvoie un résultat. Ces valeurs doivent toutes appartenir à un seul ensemble de valeurs, ici l'instance de `PlusMinusValueSet`. La méthode `isCompatibleWith(vss:ValueSet[]):boolean` permet de s'assurer que les valeurs passées en paramètres de la méthode sont bien applicables à cet algorithme.

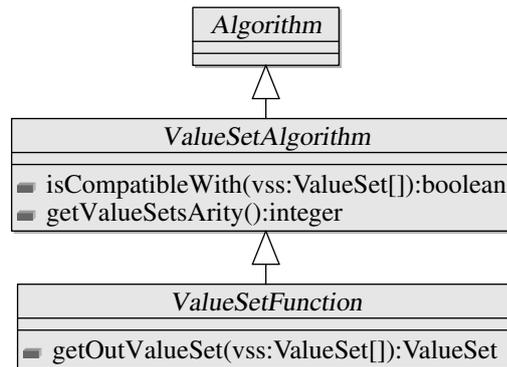


FIGURE 6.6 – Diagramme de classes des interfaces d’algorithmes.

Pour cela, on vérifie que les ensembles de valeurs auxquels elles appartiennent sont compatibles avec l’algorithme. Pour l’opérateur \wedge , on vérifie ainsi que l’ensemble de valeurs est bien un `PlusMinusValueSet`. On note qu’on utilise un tableau d’ensemble de valeurs comme paramètre. En effet, les valeurs manipulées par un algorithme n’appartiennent pas toutes nécessairement au même ensemble de valeurs. Dans une telle situation, il faut passer tous les différents ensembles de valeurs en paramètres afin de s’assurer qu’ils soient tous compatibles avec l’algorithme. Dans le cas de l’opérateur \wedge , bien que celui-ci manipule plusieurs valeurs, elles doivent tout appartenir au même ensemble, typiquement celui de la carte cognitive. Le tableau ne doit donc contenir qu’un seul ensemble de valeurs. Pour savoir combien d’ensembles de valeurs doivent être passés en paramètres et donc ainsi la taille du tableau, on utilise la méthode `getValueSetsArity():integer`. Celle-ci renvoie l’*arité* de l’algorithme, c’est-à-dire le nombre d’ensemble de valeurs sur lequel il est défini. Ces méthodes permettent donc de spécifier une partie du domaine de l’algorithme. Dans le cas de l’opérateur \wedge , cette méthode renvoie donc 1.

L’interface `ValueSetFunction` représente une fonction de valeurs. C’est un algorithme sur des ensembles de valeurs qui renvoie une valeur. Cette valeur appartient elle aussi à un ensemble de valeurs. La méthode `getOutValueSet(vss:ValueSet[]):ValueSet` permet de savoir quel est cet ensemble de valeurs. On appelle un tel ensemble l’ensemble de sortie ou codomaine de la fonction. Le tableau `vss` passé en paramètre de la méthode est le même que celui de la méthode `isCompatibleWith(vss:ValueSet[]):boolean`.

6.2.2 Chemins minimaux

Pour implémenter la notion de chemin minimal, on utilise une classe abstraite, `MinimalPathMaker` qui construit un ensemble de chemins permettant de lier deux concepts dans une carte. Cette classe est paramétrable via divers attributs et en réimplémentant certaines méthodes non représentées ici.

L’organisation générale des classes définissant la notion de chemin minimal est représentée dans la diagramme UML de la figure 6.7.

Les attributs de classe de `MinimalPathMaker` (`DIRECT_ONLY`, `CLASSIC`, `UNIQUE_INFLUENCE`, `UNIQUE_CYCLE`) correspondent à différentes définitions d’un

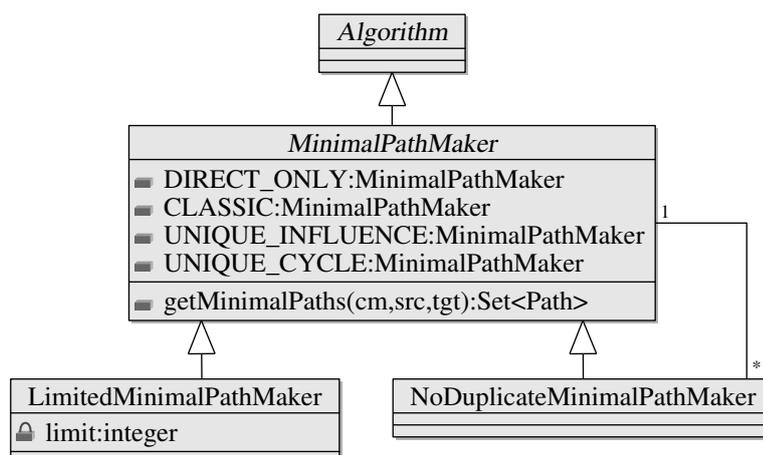


FIGURE 6.7 – Diagramme de classes des chemins minimaux.

chemin minimal (respectivement les définitions 2.24, 2.2, 2.26 et 2.28).

La classe `LimitedMinimalPathMaker` correspond à la définition des chemins minimaux limités (définition 2.25). L'attribut `limit` correspond au paramètre K .

La classe `NoDuplicateMinimalPathMaker` correspond à la définition des chemins minimaux ignorant l'ordre des influences (définition 2.29).

6.2.3 Algorithmes d'influences

Différents algorithmes d'influences existent et chacun de ces algorithmes peuvent être classés en différentes catégories, telles que les algorithmes de calcul de l'influence propagée sur un chemin, de l'influence propagée d'un concept sur un autre ou encore de l'influence taxonomique. Pour représenter les algorithmes choisis par un utilisateur, on définit une classe qui représente un gestionnaire d'influences. Ce gestionnaire d'influences est tout simplement une instance de la classe `InfluencesManager`, non représentée ici, qui réunit au sein d'une même structure un algorithme de calcul des chemins minimaux et un algorithme appartenant à chacune des catégories présentées précédemment.

La hiérarchie de classes des algorithmes de calcul d'influence est représentée dans le diagramme UML de la figure 6.8.

Tous les algorithmes de calcul d'influence doivent implémenter l'interface `InfluenceAlgorithm`. Tout comme `Algorithm`, cette interface est essentiellement sémantique. Puisqu'elle manipule des valeurs et en renvoie une, c'est une fonction de valeurs. Elle impose cependant de fixer l'arité des ensemble de valeurs à 1. Les algorithmes de calcul d'influence sont donc définis sur un unique ensemble de valeurs, typiquement celui de la carte cognitive.

Le premier algorithme de calcul d'influence est représenté par l'interface `PropagatedInfluencePath`. Celle-ci représente tous les algorithmes de calcul de l'influence propagée sur un chemin. Les paramètres de la méthode qui y est définie sont une carte cognitive `cm` et un chemin `p` de cette carte.

Le deuxième algorithme de calcul d'influence est représenté par l'interface

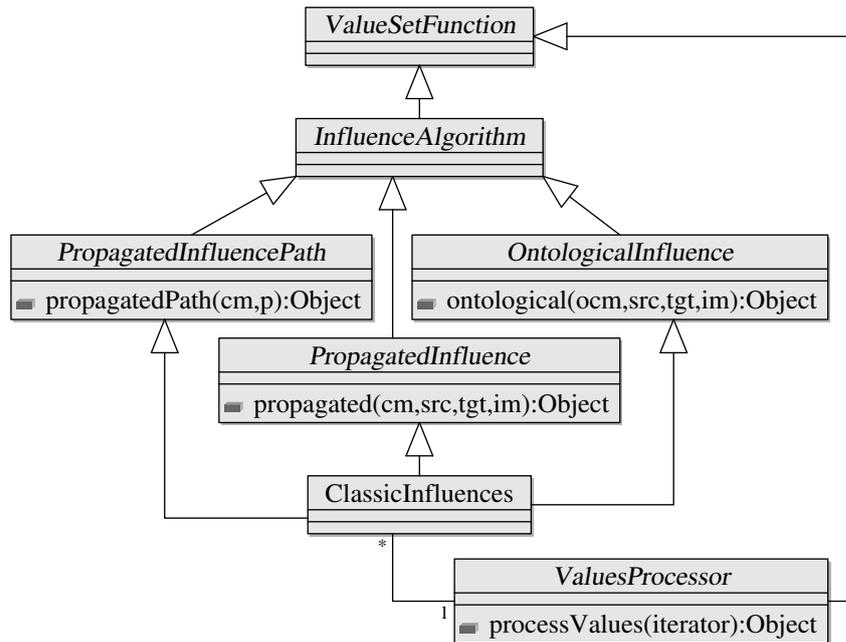


FIGURE 6.8 – Diagramme de classes des algorithmes de calcul d'influence.

PropagatedInfluence. Celle-ci représente tous les algorithmes de calcul de l'influence propagée d'un concept sur un autre. Les paramètres de la méthode qui y est définie sont une carte cognitive **cm**, un concept-source **src**, un concept-destination **tgt** et un gestionnaire d'influences **im**. Celui-ci nous permet d'appeler le **PropagatedInfluencePath** choisi par l'utilisateur.

Enfin, le troisième algorithme de calcul d'influence est représenté par l'interface **OntologicalInfluence**. Celle-ci représente tous les algorithmes de calcul de l'influence taxonomique. Nous avons à nouveau conservé l'ancienne dénomination. Les paramètres de la méthode qui y est définie sont quasiment identiques à ceux de la méthode de la classe **PropagatedInfluence**, excepté **ocm** qui représente une carte cognitive taxonomique. Ceci permet d'accéder à une taxonomie, indispensable pour le calcul. Le gestionnaire d'influence permet d'accéder au **PropagatedInfluence** choisi par l'utilisateur. L'instance de **PropagatedInfluence** accède à son tour au **PropagatedInfluencePath** de ce même gestionnaire d'influences. Un gestionnaire d'influences permet donc aussi de simplifier l'écriture du code en diminuant le nombre de paramètres de certaines méthodes.

Diverses classes implémentent ces interfaces. Ainsi, l'interface **ClassicInfluences** représente les définitions classiques du calcul des influences. Comme nous l'avons dans le chapitre 2, les définitions pour les différents ensembles de valeurs sont très proches. L'accès aux valeurs est identique pour toutes ces définitions, seule la manière de manipuler ces valeurs change. C'est l'idée de cette classe : elle implémente uniquement le parcours d'une carte pour en extraire les valeurs. Le traitement des valeurs est laissé à un autre objet, une instance de l'interface **ValuesProcessor**. Cette interface représente un processeur de valeurs : un algorithme qui effectue un calcul et renvoie une valeur en mani-

pulant différentes autres valeurs. Toutes ces valeurs doivent appartenir au même ensemble. Par conséquent, c'est une fonction sur un ensemble de valeurs d'arité 1. La classe `ClassicInfluences` implémente les trois algorithmes de calcul d'influence en faisant appel à ce processeur de valeurs. Par exemple, lors d'un appel à `propagatedPath(cm,p)` dans une instance de `ClassicInfluences`, un itérateur sur les valeurs des influences de ce chemin est envoyé en argument à la méthode `processValues(iterator)` de l'objet `ValuesProcessor` lié à l'instance de `ClassicInfluences`. Cette méthode manipule alors ces valeurs une à une pour en renvoyer une autre, qui est à son tour renvoyée par `propagatedPath(cm,p)`. La même opération est effectuée pour les autres calculs d'influence. Ainsi, `propagated(cm,src,tgt,im)` fait appel au `MinimalPathMaker` du gestionnaire d'influences pour construire l'ensemble des chemins minimaux. Pour chacun de ces chemins minimaux, leur influence propagée calculée par le `PropagatedInfluencePath` du gestionnaire d'influences est envoyée au processeur de valeurs. Dans les faits, la méthode `processValues(iterator)` prend également en paramètre un ensemble de valeurs, parce que le processeur de valeurs a besoin d'informations sur les ensembles de valeurs, par exemple en vue d'optimisations. Il existe ainsi des processeurs de valeurs pour les opérateurs min et max, pour la multiplication, la moyenne mais également pour les opérateurs \wedge (définition 2.30) et \vee (définition 2.31), entre autres.

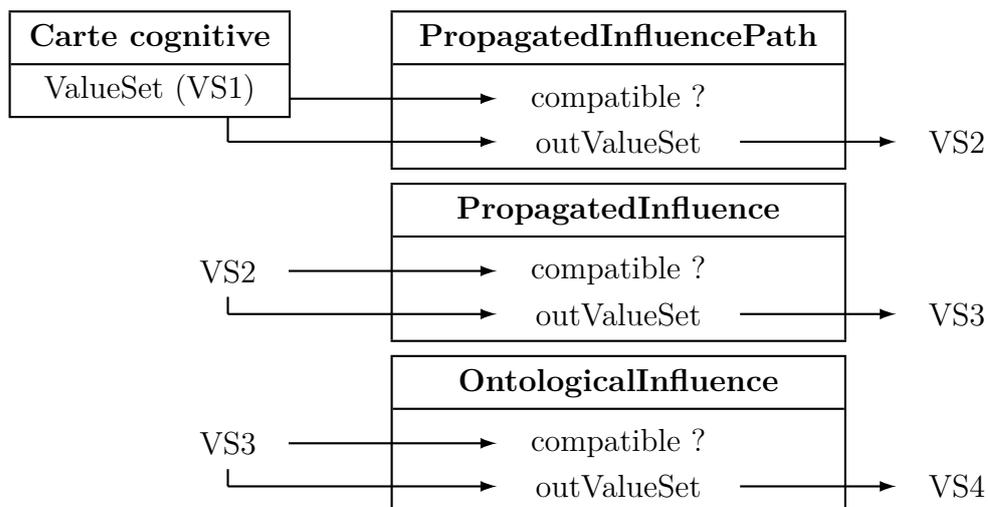


FIGURE 6.9 – Vérification de la compatibilité des algorithmes de calcul d'influence.

On rappelle qu'un objet `InfluencesManager` représente les définitions de calcul d'influence choisies par un utilisateur. Pour s'assurer que les différents algorithmes choisis sont bien compatibles entre eux, on utilise les méthodes définies au sein de l'interface `ValueSetFunction`. Le comportement de cette vérification est représenté sur la figure 6.9. Une carte cognitive est définie sur un ensemble de valeurs ($VS1$). Si on prend l'exemple de la définition classique du calcul des influences pour l'ensemble de valeurs $\{+, -\}$ (voir section 2.4.2), $VS1$ est l'ensemble $\{+, -\}$. La première étape consiste à vérifier que la définition du calcul de l'influence propagée sur un chemin peut manipuler les valeurs de cette carte

cognitive. Pour cela, on vérifie que $VS1$ est compatible avec cet algorithme. Si c'est le cas, alors on demande à l'algorithme à quel ensemble de valeur appartient la valeur qu'il calcule en fonction de $VS1$. Ceci nous donne un nouvel ensemble de valeurs, $VS2$. Dans le cas de notre exemple, après application de l'opérateur \wedge , les valeurs appartiennent toujours à $\{+, -\}$, ce qui nous donne donc $VS2$. On vérifie alors si $VS2$ est compatible avec l'algorithme de calcul de l'influence propagée d'un concept sur un autre. Si c'est le cas, on demande quel est l'ensemble de valeurs de sortie, ce qui nous donne un nouvel ensemble $VS3$. Dans le cas de notre exemple, $VS3$ est l'ensemble $\{+, 0, -, ?\}$. On renouvelle l'opération pour l'influence taxonomique. Dans le cas de notre exemple, $VS4$ est l'ensemble $\{+, \oplus, 0, \ominus, -, ?\}$. $VS4$ correspond donc au codomaine de l'influence taxonomique.

6.2.4 Influence propagée probabiliste

Le package `model.influences.bayesian` est un sous-package de `model.influences`. Il est dédié à la représentation de l'influence propagée probabiliste décrite dans la section 5.2.2. L'organisation de ses classes et interfaces est représenté dans le diagramme UML de la figure 6.10.

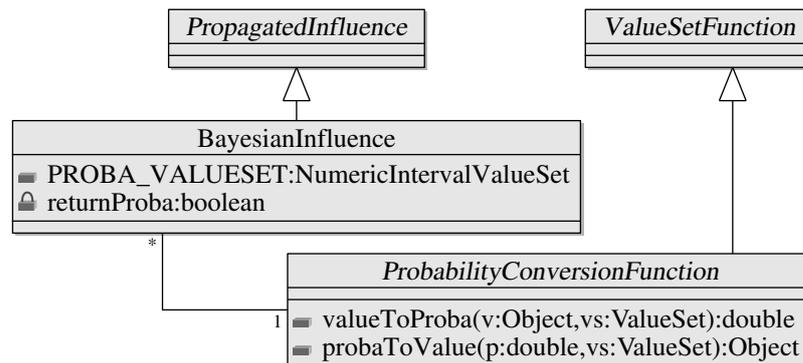


FIGURE 6.10 – Diagramme de classes du package `model.influences.bayesian`.

La classe `ProbabilityConversionFunction` représente à la fois une fonction de conversion et sa fonction inverse (définition 5.20). C'est également une fonction définie sur un ensemble de valeurs. La méthode `valueToProba(v, vs)` permet de convertir la valeur v appartenant à l'ensemble de valeurs vs en une valeur de l'intervalle $[-0.5; 0.5]$. La méthode `probaToValue(p, vs)` permet de convertir la valeur p appartenant à l'intervalle $[-0.5; 0.5]$ en une valeur de l'ensemble de valeurs vs .

La classe `BayesianInfluence` représente l'influence propagée probabiliste (définition 5.24). Celle-ci est définie comme étant une influence propagée d'un concept sur un autre et utilisant une fonction de conversion. L'attribut `returnProba` est un paramètre de l'algorithme qui permet de renvoyer une valeur de probabilité plutôt qu'une valeur appartenant à l'ensemble de valeurs de la carte cognitive. Ainsi, pour l'influence propagée de c_1 sur c_2 , elle renverrait $\mathcal{I}'_{\mathbb{P}}(c_1, c_2) + 0.5$. Une telle valeur appartient donc à l'intervalle $[0; 1]$. L'attribut de classe `PROBA_VALUESET` est un

ensemble de valeurs qui représente cet intervalle. L'ensemble de valeurs de sortie de l'influence propagée probabiliste dépend donc du paramètre `returnProba`.

6.3 Validation

Le package `model.validation` contient les différentes classes liées à la notion de validation d'une carte cognitive, présentée dans le chapitre 3. Il a été développé différemment des définitions présentées dans ce chapitre.

Pour présenter les différentes classes de package, nous expliquons tout d'abord les différences entre les définitions et l'implémentation dans la section 6.3.1. Puis, nous présentons comment est implémentée la notion de critère de qualité dans la section 6.3.2. Enfin, nous présentons dans la section 6.3.3 comment est implémentée la gestion des conflits pour chacun des critères de qualité.

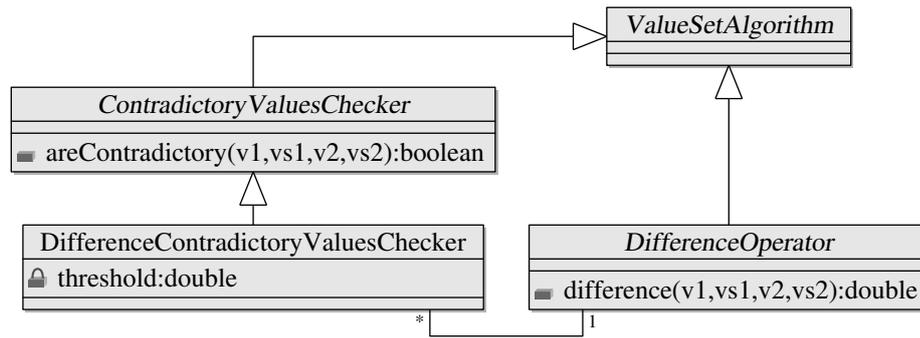
6.3.1 Approche de l'implémentation

Dans le chapitre consacré à la validation, nous avons défini de différentes manières les mêmes critères, pour leur ajouter de la souplesse ou bien pour les adapter à d'autres ensembles de valeurs. Leur structure était identique : seule la manière de comparer deux valeurs changeait. Ainsi, si on considère le critère de cohérence, il a été défini originellement avec une égalité stricte (définition 3.7), puis avec la notion de valeurs compatibles (définition 3.10) et enfin avec la notion d'intervalles compatibles (définition 3.16). Dans toutes ces définitions, la manière d'accéder aux valeurs reste la même puisqu'on compare toujours une influence taxonomique à la valeur de la contrainte. Notre implémentation a été conçue de telle manière que l'accès aux valeurs n'est codé qu'une seule fois : seule la façon de comparer les valeurs change.

Pour représenter la manière de comparer deux valeurs, nous définissons une interface `ContradictoryValuesChecker`. Cette interface définit une méthode chargée de comparer deux valeurs pour indiquer si elles sont ou non contradictoires. Ainsi, lorsque cette comparaison est une égalité, les deux valeurs sont contradictoires si elles sont différentes. Cette méthode prend en paramètres deux valeurs ainsi que leurs ensembles de valeurs respectifs. La méthode renvoie donc vrai ssi les deux valeurs passées en paramètres sont différentes. C'est donc une fonction de valeurs d'arité 2. Ainsi, la notion de valeurs compatibles pour $\{+, -\}$ (définition 3.9) ne peut s'appliquer qu'à des valeurs de l'ensemble $\{+, \oplus, 0, \ominus, -, ?\}$. De la même manière, la notion d'intervalles compatibles (définition 3.15) ne peut s'appliquer qu'à des valeurs qui sont des intervalles.

La différence utilisée pour comparer des valeurs numériques (définition 3.12) ou des valeurs d'ensembles finis totalement ordonnés (définition 3.17) peut elle aussi être vue comme étant une instance de `ContradictoryValuesChecker`. Le diagramme de classes UML de la figure 6.11 représente les liens entre les différentes classes et interfaces représentant cette comparaison de valeurs.

L'interface `DifferenceOperator` représente une différence entre deux valeurs appartenant chacune à un ensemble de valeurs. Comme pour l'interface

FIGURE 6.11 – Diagramme de classes liées à `ContradictoryValuesChecker`.

`ContradictoryValuesChecker`, elle dispose d'une méthode définie sur deux valeurs permettant de calculer leur différence. C'est donc aussi un algorithme qui s'applique à deux ensembles de valeurs. La notion de seuil (définition 3.11) n'apparaît pas ici : on se contente simplement de renvoyer un nombre qui représente la différence entre deux valeurs. Ainsi, pour l'ensemble de valeurs $[-1; 1]$, on renvoie la différence numérique entre les deux valeurs passées en paramètre.

La classe `DifferenceContradictoryValuesChecker` est basée sur la notion de valeurs compatibles selon un seuil (définitions 3.12 et 3.17). Elle représente cependant plutôt la notion de valeurs *incompatibles* selon un seuil, puisqu'on cherche à savoir si deux valeurs sont contradictoires. Elle implémente donc logiquement l'interface `ContradictoryValuesChecker`. Le calcul de la différence est délégué à une instance de l'interface `DifferenceOperator`. L'attribut `threshold` représente le seuil. Pour savoir si deux valeurs sont compatibles, on calcule leur différence grâce au `DifferenceOperator`. Si le résultat de cette différence est inférieur ou égal au seuil en valeur absolue, alors les valeurs sont compatibles. Le domaine de cette fonction est le même que celui de l'opérateur de différence associé.

6.3.2 Critères de qualité

Le diagramme de classes UML de la figure 6.12 représente les liens entre les différentes classes et interfaces représentant des critères de qualité.

L'interface `QualityCriterion` représente un critère de qualité pour une carte cognitive. Sémantiquement, les critères de qualité doivent utiliser une instance de `ContradictoryValuesChecker` et comparent par conséquent deux valeurs. Cette interface introduit deux méthodes, chacune s'appliquant à l'une de ces deux valeurs. Leur intérêt est de savoir à quel ensemble ces deux valeurs appartiennent. Si on prend l'exemple du critère de propreté (définition 3.2) on compare une valeur d'influence propagée d'un concept sur un autre à une valeur d'influence. Pour l'ensemble de valeurs $\{+, -\}$, on compare donc une valeur de l'ensemble $\{+, 0, -, ?\}$ à une valeur de l'ensemble $\{+, -\}$. Cette interface introduit également une méthode dédiée aux conflits, qui est présentée dans la section suivante.

L'interface `Verification` représente un critère de vérification. Ce type de critère nécessite deux méthodes : une pour vérifier si le critère est validé ou non

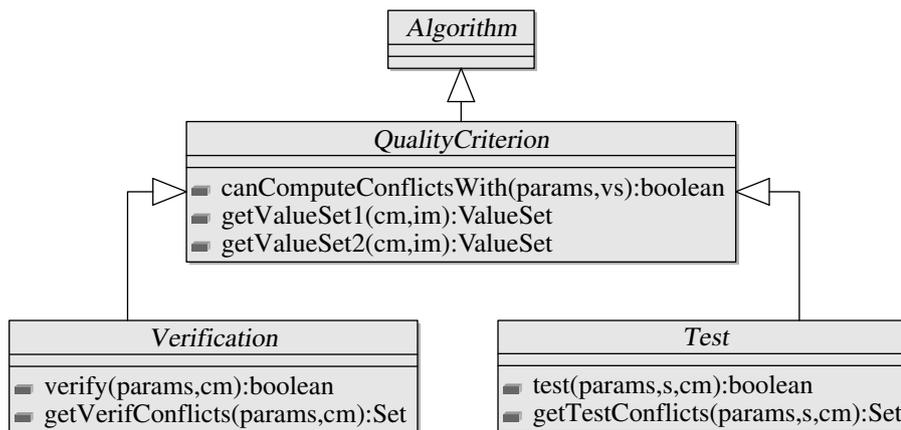


FIGURE 6.12 – Diagramme de classes des critères de qualité.

et l'autre pour construire l'ensemble des conflits associé à ce critère. Puisque ce type de critère ne nécessite pas d'information externe, il s'applique directement sur une carte cognitive, ici le paramètre `cm`. Le paramètre `params` représente différents paramètres pour la vérification. C'est en réalité une instance de la classe `QualityCriterionParams` qui, à la manière de la classe `InfluencesManager` (section 6.2.3), permet de simplifier l'écriture du code. Elle réunit un gestionnaire d'influences, une instance de `ContradictoryValuesChecker` et éventuellement une spécification, pour les critères de test. Les critères de propreté (définition 3.2) et de non-ambiguïté (définition 3.4) implémentent donc cette interface.

La spécification accessible par les paramètres `params` permet de savoir à quelle taxonomie appartiennent les concepts de la contrainte ainsi que l'ensemble de valeurs de sa valeur. Elle est donc définie sur une taxonomie de concepts et un ensemble de valeurs. Elle dispose de méthodes permettant d'ajouter ou de retirer des contraintes. À chaque ajout, on vérifie que la contrainte est compatible avec la spécification, c'est-à-dire que ses concepts sont ordonnés par la taxonomie et que sa valeur appartient à l'ensemble de valeurs.

L'interface `Test` est finalement très semblable à l'interface `Verification`. Ses méthodes nécessitent cependant un paramètre supplémentaire, `s`, qui représente une contrainte. Une contrainte est définie comme étant une influence. On note que la carte cognitive en paramètre n'est pas une carte taxonomique. En réalité, puisque la taxonomie est accessible via les paramètres via la spécification, on construit une carte taxonomique en unifiant cette taxonomie et la carte cognitive. Définir le test uniquement pour une contrainte nous permet d'éviter de le faire pour une spécification puisque la définition de la validité d'une spécification est toujours la même une fois qu'on connaît le résultat du test des contraintes, et ce quelque soit le critère de test. Le critère de cohérence avec une contrainte (définition 3.7) implémente donc l'interface `Test`.

6.3.3 Gestion des conflits

Avant de construire un ensemble de conflits pour un critère donné, il faut tout d'abord s'assurer qu'on peut bien calculer cet ensemble. En effet, comme expliqué dans le chapitre 3, la notion de conflit est intrinsèquement liée, entre autres, à l'ensemble de valeurs sur lequel est définie la carte cognitive. La définition d'un conflit varie donc en fonction de ses paramètres. Certains paramétrages sont même incompatibles avec la construction de conflits. Pour toutes ces raisons, la méthode `canComputeConflictsWith(params,vs)` de l'interface `QualityCriterion` doit être appelée avant toute construction d'un ensemble de conflits. Cette méthode indique en effet si on peut construire un ensemble de conflits en fonction de l'instance de `ContradictoryValuesChecker`, de l'ensemble de valeurs de la carte cognitive ainsi que des définitions de calcul d'influence utilisés.

Les ensembles des conflits de chacun des critères de qualité peuvent être construits via leurs méthodes dédiées. Les ensembles construits par ces méthodes peuvent correspondre à différentes définitions selon le paramétrage considéré. Les ensembles de conflits calculés doivent être des ensembles de conflits minimaux. Pour construire simplement de tels ensembles, on peut utiliser la classe `MinimalConflictSet` qui construit automatiquement un ensemble de conflits minimaux. À chaque fois qu'un conflit est ajouté à cet ensemble, il le refuse si l'un de ses conflits est englobé par ce conflit ou bien supprime tous ses conflits englobant ce conflit. Cette classe contient également une méthode de classe qui construit l'ensemble des diagnostics à partir d'un ensemble de conflits minimaux.

6.4 Synthèse

Le package `model.merge` contient les différentes classes liées à la notion de synthèse de cartes cognitives, présentée dans le chapitre 4.

Pour décrire ce package, nous présentons tout d'abord la façon dont ont été implémentées les notions de bases de la synthèse, telles que les préférences ou l'ensemble univoque de cartes attribuées dans la section 6.4.1. Ensuite, nous présentons l'implémentation des différentes notions permettant la synthèse des concepts, des influences et des valeurs dans la section 6.4.2. Enfin, nous présentons l'algorithme permettant la synthèse de cartes cognitives et utilisant ces différentes notions dans la section 6.4.3.

6.4.1 Préférences et cartes attribuées

Nous présentons dans un premier temps comment nous avons implémenté la notion de préférence. Nous présentons dans un second temps les classes utilisées pour représenter un ensemble de cartes.

Préférences

La notion de préférence est représentée par la classe `Preference`. Elle doit représenter une relation d'ordre partiel sur des éléments. Pour l'implémenter, nous nous

sommes basés sur sa représentation sous forme stratifiée (définition 4.7). Pour ordonner les éléments, on construit l'ordre de préférence strate par strate ou on place un élément dans une strate identifiée par son index. Diverses méthodes permettent de s'assurer qu'un élément n'apparaît pas deux fois et qu'aucune strate n'est vide.

Cartes attribuées

Le diagramme de classes UML de la figure 6.13 représente les principales classes liées à la synthèse de cartes cognitives et leurs liens.

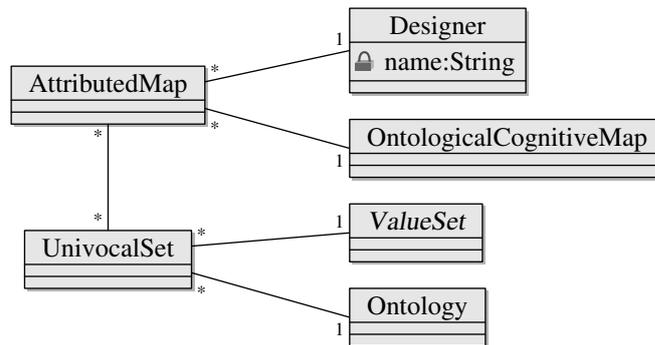


FIGURE 6.13 – Diagramme de classes des principales classes liées à la synthèse de cartes cognitives.

La classe `Designer` représente le designer d'une carte cognitive. Cette classe ne renseigne que le nom de ce designer.

La classe `AttributedMap` représente une carte attribuée. Elle est définie comme étant l'association d'une carte cognitive taxonomique et de son designer. Les concepts de la carte cognitive taxonomique ne sont pas nécessairement élémentaires. La carte cognitive taxonomique représente donc ici l'association d'une carte cognitive et d'une taxonomie.

La classe `UnivocalSet` représente un ensemble univoque de cartes attribuées. Cette classe est définie sur un ensemble de valeurs et sur une taxonomie. Elle est liée à plusieurs cartes attribuées. Cette classe vérifie, à chaque ajout d'une carte attribuée, que celle-ci est définie sur la même taxonomie et le même ensemble de valeurs que les autres cartes. On s'assure également qu'un designer n'est pas associé à plus d'une carte.

6.4.2 Synthèse de concepts, d'influences et de valeurs

Pour la synthèse de cartes cognitives, l'approche de l'implémentation est différente de celle des définitions de la section 4.2. Nous décrivons dans un premier temps les différences entre les deux approches, puis nous présentons comment nous avons implémenté les différents algorithmes pour synthétiser les concepts, influences et valeurs.

Approche de l'implémentation

L'approche de la définition de la synthèse des concepts (définition 4.11) est ensembliste : on construit l'ensemble des concepts synthétisés. Ensuite, la définition de la synthèse des influences (définition 4.13) utilise la notion de représentation (définition 4.12) pour indiquer les influences que représente chaque influence synthétisée. De ces influences synthétisées, on construit l'ensemble des valeurs d'influences extraites (définition 4.15) pour la synthèse des valeurs (définition 4.16).

L'approche est différente pour l'implémentation. Celle-ci a en effet été effectuée avant la version finale des définitions présentées dans ce mémoire de thèse.

L'approche de l'implémentation de la synthèse des concepts est fonctionnelle : la synthèse de concepts consiste ici à construire une fonction qui associe à chaque concept synthétisé les concepts qu'il représente. La notion de représentation pour les concepts est plus ou moins présente dans le chapitre 4. Elle est cependant limitée à la comparabilité : un concept en représente un autre s'ils sont comparables. Il n'y a pas réellement de raisons de définir autrement cette relation. La définir au sein d'une fonction a cependant deux avantages : cela permet d'une part d'éviter une redondance du code et d'autre part, cela permet de prévoir d'éventuelles évolutions futures. Dans les faits, cette fonction de représentation est implémentée par une classe **Assignment**. Cette classe dispose d'une méthode qui permet d'associer un concept représentant à un concept représenté. Elle assure en outre que chaque concept se représente lui-même. Elle dispose de diverses méthodes permettant d'accéder à l'ensemble des concepts représentés ou encore au concept qui représente un concept donné.

L'implémentation de la synthèse des influences utilise cet assignment pour construire l'ensemble des influences synthétisées. Là encore, l'implémentation est différente des définitions pour la synthèse des influences. En effet, on construit ici directement les valeurs d'influences extraites. Ceci évite d'avoir à accéder aux influences représentées par une influence synthétisée. Les valeurs d'influences extraites sont représentées par une instance de la classe **ExtractedInfluences**. Cette classe dispose d'une méthode qui permet de lier un concept synthétisé cause, un concept synthétisé effet, un designer et l'influence originale. Lier ces quatre informations permet à la fois de construire une valeur d'influence extraite mais également une influence synthétisée en associant le concept-cause et le concept-effet. Cette classe utilise plusieurs tables dont l'organisation est présentée dans la figure 6.14. À partir d'un couple de concepts représentant une influence synthétisée, on accède aux influences représentées, c'est-à-dire les influences originales. On accède à ces influences à partir de leurs designers respectifs. À partir d'un designer, on accède donc à l'ensemble des influences qu'il a conçues et qui sont représentées par le couple de concepts précédemment choisi.

L'implémentation de la synthèse des influences consiste alors à effectuer le lien entre un assignment et des valeurs d'influences extraites. L'opérateur de synthèse est plus souple que celui défini dans le chapitre 4. Avec les définitions, toutes les influences qui en représentaient une autre étaient nécessairement des influences synthétisées. Ici, on choisit directement les influences qui seront présentes dans la carte. Pour savoir quelles influences elles représentent, on utilise l'assignment. Chaque influence doit toutefois représenter au moins une autre influence. La plu-

		...
(c _{s1} , c _{s2})	d ₁	Influence(c ₁ , c ₂ , α)
		Influence(c ₃ , c ₄ , β)
	...	
	d ₆	Influence(c ₅ , c ₆ , γ)
		...
		Influence(c ₇ , c ₈ , δ)
		...

FIGURE 6.14 – Schéma d'organisation de la classe `ExtractedInfluences`.

part du temps, cet opérateur se résume en un parcours des influences de toutes les cartes. Si on choisit de synthétiser l'influence parcourue, alors on construit une valeur d'influence extraite avec cette influence, le designer de la carte et les concepts représentant les extrémités de l'influence.

La synthèse des valeurs consiste simplement à utiliser les valeurs d'influences extraites pour calculer une valeur synthétisée. L'implémentation de la synthèse des valeurs est donc quasiment identique à sa définition.

Algorithmes de synthèse

Les interfaces représentant les différents algorithmes liés à la synthèse (synthèse des concepts, des influences et des valeurs) et leurs liens sont présentés dans le diagramme UML de la figure 6.15.

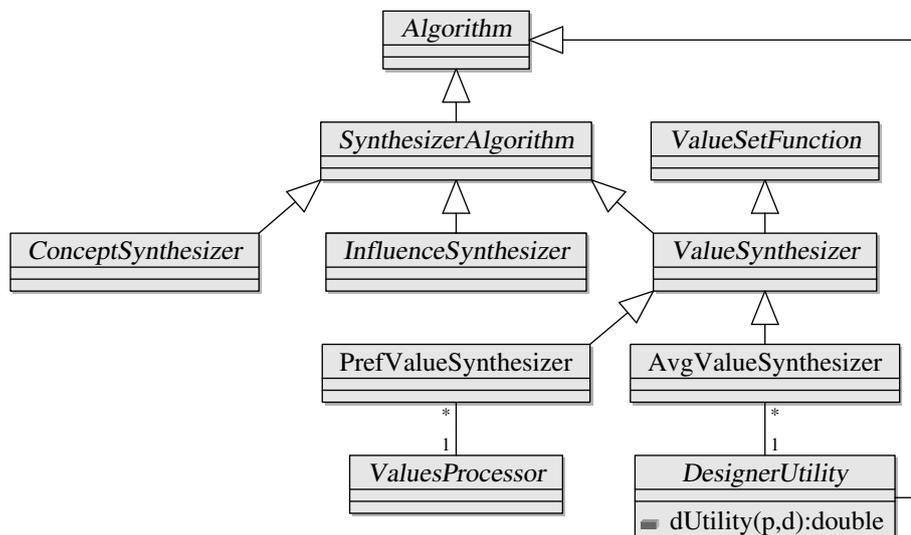


FIGURE 6.15 – Diagramme de classes des algorithmes de synthèse.

L'interface `SynthesizerAlgorithm` est l'interface-mère de tous les algorithmes qui synthétisent des concepts, des influences ou des valeurs. Cette interface est purement sémantique puisqu'elle ne nécessite de définir aucune méthode et ne contient aucun attribut.

L'interface `ConceptSynthesizer` représente un algorithme synthétisant des concepts. Cette interface définit une méthode qui construit un assignement à partir d'un ensemble univoque et de préférences sur les designers de cet ensemble. Ainsi, pour implémenter la définition 4.11, on déclare une classe implémentant cette interface. En plus de construire un ensemble de concepts sélectionnés, on construit également un assignement, qui est renvoyé par la méthode. On fait de même pour les autres définitions de la synthèse de concepts, telles que les définitions 4.24 ou encore 4.25.

L'interface `InfluenceSynthesizer` représente un algorithme synthétisant des influences. Cette interface définit une méthode qui construit des valeurs d'influences extraites à partir d'un ensemble univoque, de préférences sur les designers ainsi que d'un assignement. Ainsi, pour implémenter la définition 4.13, on déclare une classe implémentant cette interface. On construit directement les valeurs d'influences extraites en utilisant l'assignement. On fait de même pour les autres définitions de la représentation permettant la synthèse d'influences, telles que les définitions 4.26 ou encore 4.27.

L'interface `ValueSynthesizer` représente un algorithme synthétisant des valeurs. Puisqu'elle manipule les valeurs des cartes d'un ensemble univoque définies toutes sur le même ensemble, c'est une fonction d'arité 1 sur cet ensemble de valeurs. Cette interface définit une méthode qui calcule une valeur synthétisée à partir d'un ensemble univoque, de préférences sur les designers ainsi que de valeurs d'influences extraites pour un concept-cause et un concept-effet donnés. Cette interface est implémentée de différentes manières.

La classe `AvgValueSynthesizer` représente la définition 4.16 et implémente donc l'interface `ValueSynthesizer`. Elle est cependant plus paramétrable que la définition originale. En effet, on peut choisir la fonction d'utilité (ou fonction de pondération) qui va donner un poids à chaque designer. Celle-ci n'est donc pas nécessairement définie comme étant l'index du designer dans l'ordre de préférence. Cette fonction d'utilité est représentée par l'interface `DesignerUtility`, un algorithme. Cette interface définit une méthode qui doit renvoyer un nombre réel en fonction d'un ordre de préférence et d'un designer ordonné par cette préférence. La classe `AvgValueSynthesizer` contient également divers champs permettant de paramétrer le calcul de la moyenne pondérée, comme par exemple pour regrouper les valeurs fournies par un même designer en en faisant la moyenne.

La classe `PrefValueSynthesizer` représente la définition 4.29 et implémente donc l'interface `ValueSynthesizer`. Elle est également plus paramétrable. En effet, cet opérateur peut s'appliquer à divers ensembles de valeurs puisqu'il utilise un processeur de valeurs. Ce processeur de valeurs est utilisé pour combiner les valeurs renseignées par des designers sur la même strate selon l'ordre de préférence. On peut réutiliser les mêmes processeurs que pour le calcul d'influences.

Pour construire une carte synthétisée non-valorisée (définition 4.14), on utilise la même algorithme que pour la construction d'une carte synthétisée valorisée (définition 4.17). Pour construire une telle carte, on utilise alors un opérateur de synthèse de valeurs qui renvoie toujours une non-valeur (le champ `NO_VALUE` de l'interface `ValueSet`) et ce quels que soient ses paramètres. Cet opérateur implémente donc l'interface `ValueSynthesizer`.

6.4.3 Synthèse de cartes cognitives

À partir de toutes ces classes et interfaces, on peut construire une carte synthétisée. Pour cela, on construit tout d'abord un objet instance de la classe `Synthesizer` qui, à la manière de `InfluencesManager` et de `QualityCriterionParams`, réunit un `ConceptSynthesizer`, un `InfluenceSynthesizer` et un `ValueSynthesizer`. Cette classe contient également une méthode de classe qui permet de construire une carte cognitive synthétisée à partir d'une instance de `Synthesizer`, d'un ensemble univoque et de préférences sur les designers de cet ensemble. L'algorithme utilisé pour construire une carte synthétisée est présenté à la figure 6.16.

Algorithme : Construction d'une carte cognitive synthétisée

Entrées : s : un `Synthesizer`,

u : un ensemble univoque de cartes attribuées,

p : des préférences sur les designers de u

Sortie : une carte cognitive synthétisée CS

```

1  $a \leftarrow s.\text{ConceptSynthesizer}.\text{synthesize}(u, p)$ 
2  $EI \leftarrow s.\text{InfluenceSynthesizer}.\text{synthesize}(u, p, a)$ 
3  $outVS \leftarrow s.\text{ValueSynthesizer}.\text{getOutValueSet}(u.\text{getValueSet}())$ 
4  $CS \leftarrow \text{CognitiveMap}(outVS)$ 
5 pour chaque  $c \in a.\text{getSelectedConcepts}()$  faire
6   |  $c.\text{addConcept}(c)$ 
7 fin
8 pour chaque  $ei \in EI$  faire
9   |  $(src, tgt) \leftarrow ei.\text{getSynthesizedInfluence}()$ 
10  |  $\{(d, OI)\} \leftarrow ei.\text{getOriginalInfluences}()$ 
11  |  $v \leftarrow s.\text{ValueSynthesizer}.\text{synthesize}(u, p, (src, tgt), \{(d, OI)\})$ 
12  |  $i \leftarrow \text{Influence}(src, tgt, v)$ 
13  |  $cm.\text{addInfluence}(i)$ 
14 fin
15 retourner  $CS$ 

```

FIGURE 6.16 – Algorithme pour construire une carte cognitive synthétisée.

Cet algorithme construit pas à pas une carte cognitive. Tout d'abord, on calcule un assignement avec l'opérateur de synthèse de concepts. Ensuite, on calcule les valeurs d'influences extraites à partir de cet assignement avec l'opérateur de synthèse des influences. Enfin, on utilise l'opérateur de synthèse de valeurs pour récupérer l'ensemble de valeurs auquel appartiennent les différentes valeurs produits par cet opérateur. Cet ensemble de valeurs est l'ensemble sur lequel est définie la carte cognitive synthétisée. On construit donc une ébauche de carte cognitive synthétisée à la ligne 4, définie sur cet ensemble de valeurs. Il faut à présent ajouter à cette carte ses concepts et ses influences.

Dans un premier temps, on ajoute à la carte chacun des concepts sélectionnés de l'assignement (ligne 5). Dans un second temps, on fait de même pour les influences (ligne 13). Le procédé est cependant plus compliqué. On parcourt les

ensembles clé-valeur de la table de l'objet représentant les valeurs d'influences extraites (ligne 8). La clé représente une valeur d'influence extraite représentée par un couple de concepts, l'un étant la source et l'autre la destination, qu'on récupère à la ligne 9. On récupère également la valeur, c'est-à-dire une table liant des designers aux influences originales qu'ils ont conçues à la ligne 10. Ces deux composantes sont passées en paramètre à l'opérateur de synthèse des valeurs à la ligne 11. On récupère ainsi une valeur qui nous sert à construire une influence synthétisée à partir du concept-source et du concept-destination. Cette influence est ensuite ajoutée à la carte synthétisée (ligne 13).

Enfin, on renvoie la carte synthétisée construite (ligne 15).

6.5 Entrées/Sorties et interface graphique

Deux packages sont voués à l'interaction entre l'utilisateur du logiciel et les différentes classes métier présentées précédemment. Il s'agit du package `io`, dédié à tout ce qui touche aux entrées/sorties, présenté dans la section 6.5.1 et du package `ui`, dédié, lui, à l'interface graphique, présenté dans la section 6.5.2.

6.5.1 Entrées/Sorties

Le package `io` regroupe différentes classes liées aux entrées/sorties. Celles-ci assurent diverses fonctionnalités. La plupart ont en commun d'être dédiées à une communication facilitée entre les classes présentées précédemment et un utilisateur ou d'autres outils informatiques. Ainsi, certaines sont liées à l'affichage d'informations, notamment des valeurs. On peut par exemple définir un certain formatage pour les nombres, par exemple pour choisir combien de chiffres après la virgule afficher. D'autres sont liées à l'affichage des différents algorithmes utilisés. Ces classes permettent en outre de lister les différents algorithmes proposés par le logiciel. Enfin, il est possible d'exporter les cartes cognitives réalisées par le logiciel sous la forme de fichiers XML.

Algorithmes et descriptions d'algorithmes

Les descriptions d'algorithmes permettent de donner une étiquette textuelle à un algorithme de manière à pouvoir être identifié facilement par un utilisateur. De plus, on définit, pour chaque type d'algorithme, une liste permettant au logiciel de récupérer l'intégralité des algorithmes disponibles pour une tâche donnée. L'organisation des classes permettant un tel procédé est présentée dans le diagramme UML de la figure 6.17.

La classe `AlgorithmDescription` représente une description d'algorithme. Elle associe donc un algorithme à un texte descriptif, permettant de le représenter. Par exemple, l'algorithme implémentant la différence pour l'ensemble de valeurs $[-1; 1]$ (voir section 6.3.1) est identifié par la chaîne de caractères "`i1 - i2`".

La classe `AlgorithmDescriptions` représente un ensemble de descriptions d'algorithmes, tous étant instances du même type d'algorithme. Elle dispose par conséquent de diverses méthodes permettant d'ajouter ou de retirer des descriptions

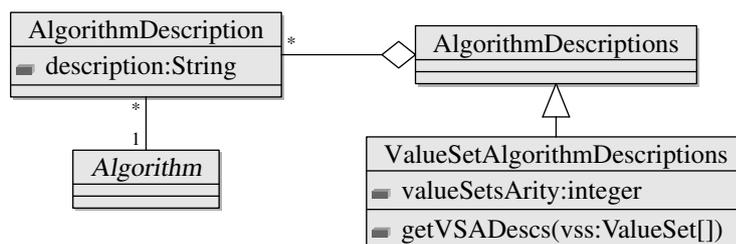


FIGURE 6.17 – Diagramme de classes des descriptions d’algorithmes.

d’algorithmes. Avant chaque ajout, cette classe vérifie qu’on peut bien ajouter la description en question, par exemple pour éviter les doublons.

La classe **ValueSetAlgorithmDescriptions** représente un ensemble de descriptions d’algorithmes sur des ensembles de valeurs. La distinction avec les autres ensembles de descriptions d’algorithmes classiques permet d’ajouter des méthodes intéressantes. Tout d’abord, un attribut spécifie l’arité des ensembles de valeurs de l’algorithme. Il est utilisé lors de l’ajout d’une description d’algorithme, pour s’assurer que cet algorithme a la même arité que celui des autres algorithmes décrits. L’intérêt de cette classe réside cependant dans la méthode `getVSADescs(vss: ValueSet[])`. Celle-ci renvoie en effet uniquement les algorithmes qui sont compatibles avec les ensembles de valeurs spécifiés.

Il existe différentes instances de ces classes, regroupées au sein de classes suivant leur utilité. Ainsi, la classe **InfluenceDescriptions** regroupe les descriptions d’algorithmes liés au calcul d’influence. Elle instancie donc un **AlgorithmDescriptions** pour le calcul des ensembles de chemins minimaux et quatre **ValueSetAlgorithmDescriptions** pour le calcul de l’influence propagée sur un chemin, de l’influence propagée d’un concept sur un autre, de l’influence taxonomique et enfin pour les fonctions de conversion, utilisées pour le calcul de l’influence propagée probabiliste. La classe **ValidationDescriptions** regroupe les descriptions d’algorithmes liés à la validation. Elle instancie deux **ValueSetAlgorithmDescriptions** pour les différentes instances de **ContradictoryValuesChecker** et pour les opérateurs de différence. Enfin, la classe **SynthesizerDescriptions** regroupe les descriptions d’algorithmes liés à la synthèse. Elle instancie deux **AlgorithmDescriptions** pour la synthèse des concepts et des influences et un **ValueSetAlgorithmDescriptions** pour la synthèse des valeurs.

Export de cartes cognitives dans un fichier

Diverses méthodes permettent de sauvegarder ou de lire les cartes cognitives créées avec notre logiciel dans un fichier. Un tel fichier stocke en réalité un ensemble univoque. Même si un utilisateur construit seul une carte unique, cela revient à créer un ensemble univoque ne contenant qu’une seule carte. Le seul inconvénient de ce choix d’implémentation est que l’utilisateur doit nécessairement définir une taxonomie. Par conséquent, s’il souhaite ajouter un concept à sa carte cognitive, il doit préalablement ajouter ce concept à la taxonomie.

Pour construire de tels fichiers, nous avons développé un format de fichier basé sur XML [W3C, 2008]. Nous avons fait ce choix car il existe de nombreuses bibliothèques permettant de manipuler facilement de tels fichiers. Nous n'avons pas défini de DTD par manque de temps. Néanmoins, les fichiers générés doivent respecter une certaine grammaire qui est décrite en partie dans cette section.

Tout d'abord, on doit définir l'ensemble de valeurs sur lequel est défini l'ensemble univoque et donc toute carte cognitive de cet ensemble. Pour cela, on crée un élément `valeurs` qui doit nécessairement spécifier un `type`. Ainsi, la figure 6.18 montre comment déclarer l'ensemble de valeurs $\{+, -\}$ en spécifiant le type `plusminus`. La figure 6.19 montre comment déclarer l'ensemble $[-1; 1]$ en spécifiant le type `intervalle`. Avec un tel type, on doit également renseigner les valeurs minimale et maximale qui doivent nécessairement être numériques. La figure 6.20 montre comment déclarer l'ensemble $\{nul < faible < moyen < fort\}$ en spécifiant le type `enumerate`. Cette dernière déclaration permet en réalité de construire une instance d'un ensemble fini de valeurs. À l'intérieur de l'élément `valeurs`, on spécifie les différentes valeurs de cet ensemble par ordre décroissant, grâce à différents éléments `valeur`. Par ailleurs, on doit également spécifier le type des valeurs de l'ensemble : `string`, comme sur la figure, indique que ce sont des chaînes de caractères, `integer` indique que ce sont des valeurs numériques.

```
<valeurs type="plusminus" />
```

FIGURE 6.18 – Déclaration de l'ensemble de valeurs $\{+, -\}$.

```
<valeurs type="intervalle" min="-1" max="1" />
```

FIGURE 6.19 – Déclaration de l'ensemble de valeurs $[-1; 1]$.

```
<valeurs type="enumerate" valueType="string">
  <valeur>fort</valeur>
  <valeur>moyen</valeur>
  <valeur>faible</valeur>
  <valeur>nul</valeur>
</valeurs>
```

FIGURE 6.20 – Déclaration de l'ensemble de valeurs $\{nul < faible < moyen < fort\}$.

Le fichier doit ensuite déclarer une taxonomie, c'est-à-dire les concepts sur lesquels sont définies les différentes cartes ainsi que les relations d'héritage entre ces concepts. Pour cela, on utilise un élément `ontologie`, l'ancienne dénomination ayant là encore été conservée. Cet élément permet de construire une arborescence d'éléments `concept`, représentant ainsi les liens d'héritage entre les différents concepts. La figure 6.21 représente ainsi la déclaration de la taxonomie de l'exemple 4.2. Chaque concept dispose d'un identifiant généré par le programme

et d'un label représentant un texte descriptif. Cet identifiant est nécessaire car le label n'est pas nécessairement unique. Chaque concept dispose également de coordonnées permettant de l'afficher à l'endroit prévu par l'utilisateur. On note que la taxonomie dispose elle aussi d'un identifiant. Or, nous n'utilisons qu'une seule taxonomie. L'identifiant a en réalité été créé pour d'éventuelles évolutions futures du modèle où les designers utilisent différentes taxonomies pour la construction de leur carte.

```
<ontologie id="o1">
  <concept id="e19" label="Vegetation" x="527" y="130" />
  <concept id="e18" label="Foudre" x="458" y="130" />
  <concept id="e17" label="Secheresse" x="361" y="130" />
  <concept id="e12" label="Feu" x="159" y="100">
    <concept id="e16" label="Contre-feu" x="206" y="130" />
    <concept id="e13" label="Incendie" x="100" y="130">
      <concept id="e15" label="Incendie humain" x="142" y="160" />
      <concept id="e14" label="Incendie naturel" x="18" y="160" />
    </concept>
  </concept>
  <concept id="e7" label="Meteo" x="492" y="10">
    <concept id="e11" label="Chaleur" x="532" y="40" />
    <concept id="e8" label="Mauvais temps" x="418" y="40">
      <concept id="e9" label="Pluie" x="420" y="70" />
      <concept id="e10" label="Orage" x="467" y="70" />
    </concept>
  </concept>
  <concept id="e1" label="Humain" x="156" y="10">
    <concept id="e6" label="Pyromane" x="263" y="40" />
    <concept id="e3" label="Forces d'intervention" x="104" y="40" />
    <concept id="e5" label="Policier" x="187" y="70" />
    <concept id="e4" label="Pompier" x="108" y="70" />
  </concept>
  <concept id="e2" label="Campeur" x="21" y="40" />
</concept>
</ontologie>
```

FIGURE 6.21 – Déclaration d'une taxonomie de concepts.

Le fichier doit enfin déclarer le contenu de l'ensemble univoque, c'est-à-dire les différentes cartes qui le composent. Pour cela, on utilise un élément `cartes`, défini sur la taxonomie précédemment déclarée, et contenant de multiples éléments `carte`, chacun représentant la carte d'un designer. La figure 6.22 représente la déclaration d'une partie de l'ensemble univoque de l'exemple 4.2, et notamment la carte cognitive de l'exemple 4.1. Chaque élément `carte` doit représenter son designer. Pour cela, on utilise un élément `designer` qui renseigne son nom ainsi qu'une couleur représentée par un code hexadécimal afin de pouvoir l'identifier clairement dans le logiciel. Ensuite, on déclare les concepts utilisés par cette carte. Les concepts ayant déjà été définis par la taxonomie, indiquer leur identifiant suffit. Chaque concept fournit également des coordonnées, qui représentent cette fois leurs coordonnées dans la carte. Enfin, on déclare les différentes influences entre ces concepts. Chaque influence doit fournir son concept-cause et son concept-effet, représentés par leurs identifiants respectifs, ainsi qu'une valeur. Cette valeur doit bien évidemment appartenir à l'ensemble de valeurs défini auparavant.

```

<cartes ontologie="o1">
  ...
  <carte>
    <designer nom="Ministre" couleur="#666666" />
    <concept id="e4" x="10" y="110" />
    <concept id="e2" x="110" y="10" />
    <concept id="e16" x="110" y="110" />
    <concept id="e13" x="210" y="60" />
    <concept id="e6" x="110" y="60" />
    <concept id="e5" x="10" y="10" />
    <influence from="e16" to="e13" valeur="-0.5" />
    <influence from="e4" to="e16" valeur="0.4" />
    <influence from="e6" to="e13" valeur="1" />
    <influence from="e2" to="e13" valeur="0.3" />
    <influence from="e5" to="e6" valeur="-0.9" />
    <influence from="e5" to="e2" valeur="-0.1" />
  </carte>
  ...
</cartes>

```

FIGURE 6.22 – Déclaration de cartes cognitives.

Export de réseaux bayésiens

Dans le chapitre 5 et notamment dans la section 5.2, nous avons défini une méthode de traduction pour représenter une carte cognitive en un réseau bayésien. Nous avons également développé un système d'export d'une carte cognitive en un fichier représentant un réseau bayésien.

Le format de fichier est celui utilisé par le logiciel *Belief and Decision Networks*². Ce logiciel permet de créer sous forme graphique des réseaux bayésiens et les tables de probabilité de chacun des sommets. En outre, il a l'avantage d'être gratuit et est facilement utilisable. Malheureusement, il ne permet pas de représenter des réseaux bayésiens causaux.

Pour que la représentation d'une carte cognitive en un réseau bayésien causal soit toute de même réalisable, nous avons utilisé le fait que l'application de l'intervention sur un sommet revienne à séparer ce sommet de ses parents (voir section 5.1.1). Nous avons conçu un algorithme qui, pour chaque concept de la carte cognitive, construit dans un fichier qui lui est dédié un réseau bayésien dans lequel le sommet le représentant n'a pas de parent. En choisissant le bon fichier, on peut donc calculer l'influence de ce concept sur les autres concepts de la carte. La génération du réseau bayésien est en outre paramétrable par une fonction de conversion.

²<http://aispace.org/bayes/>

6.5.2 Interface graphique

Le package `ui` contient les différentes classes et interfaces relatives à l'interface graphique. Nous n'allons pas détailler le contenu de ce package puisqu'il ajoute essentiellement une représentation graphique à des notions déjà présentées. Les différents composants sont illustrés de manière plus visuelle dans la section suivante, qui présente quelques cas d'utilisation. Dans cette section, nous présentons donc uniquement l'organisation générale du package.

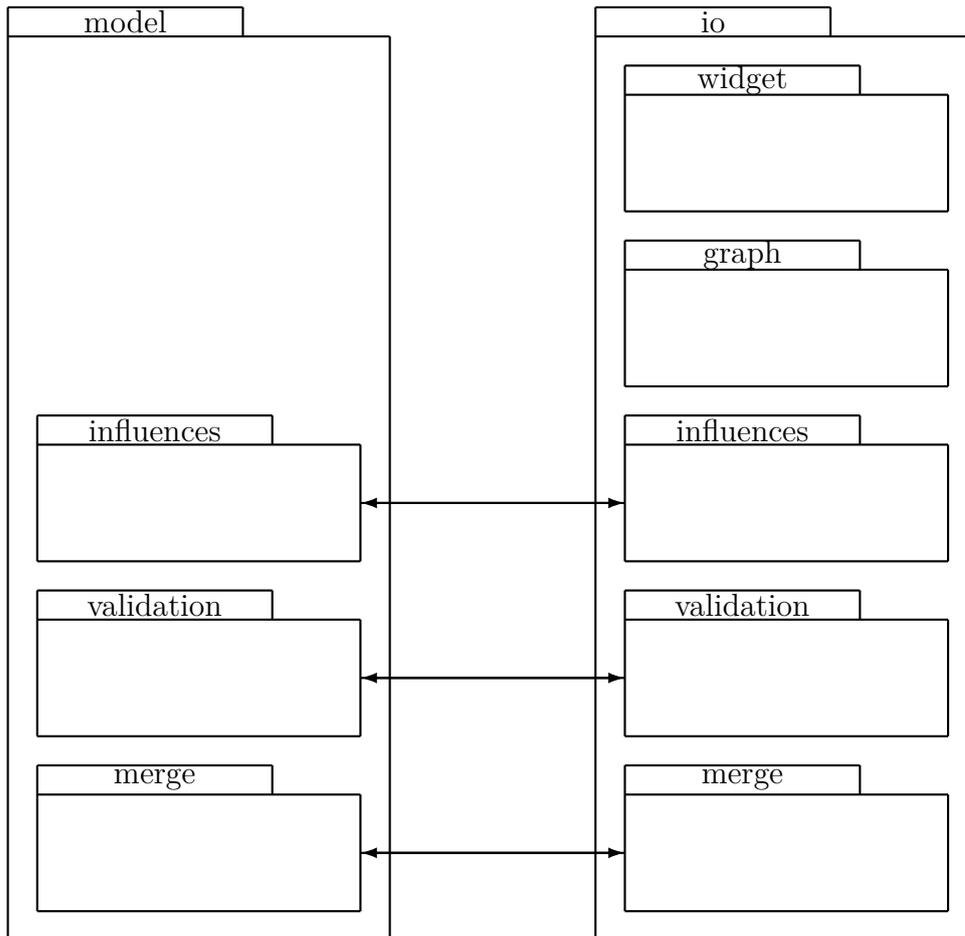


FIGURE 6.23 – Organisation du package `io` et ses liens avec le package `model`.

La figure 6.23 représente l'organisation générale du package `io` et ses divers sous-packages. On peut voir qu'à chaque sous-package du package `model` correspond un sous-package du package `io`. Cela permet une meilleure organisation du code dans lequel les classes dédiées à une tâche particulière sont regroupées au même endroit.

Le package `io.widget` contient différents composants utilisés fréquemment par les autres packages. Ainsi, ce package dispose d'un composant `ValueField` permettant à l'utilisateur de rentrer une valeur. Selon l'ensemble de valeurs considéré, ce composant affiche une liste déroulante ou un champ de texte.

Le package `io.graph` contient différentes classes permettant d'afficher les cartes

cognitives et les taxonomies sous la forme d'un graphe. Pour représenter ces graphes, nous avons utilisé la bibliothèque *jgraph*³.

Les autres packages définissent simplement des composants permettant d'afficher les différentes notions auxquelles ils sont liées. Ainsi, le package `io.influences` définit des listes déroulantes affichant les différents algorithmes pour le calcul des influences dans une carte cognitive. Le package `io.validation` définit par exemple des composants permettant de sélectionner une certaine instance de `ContradictoryValuesChecker` pour un critère de qualité. Le package `io.merge` définit entre autres un composant permettant de construire des préférences sur un ensemble de designers.

6.6 Cas d'utilisation

Dans cette section, nous montrons comment utiliser le logiciel que nous avons développé via quelques cas d'utilisations. Nous présentons tout d'abord comment construire une carte cognitive et un ensemble univoque dans la section 6.6.1. Puis, nous montrons comment calculer des influences sur une carte dans la section 6.6.2. Ensuite, nous détaillons comment valider cette même carte dans la section 6.6.3. Enfin, nous montrons comment procéder à une synthèse des différentes cartes cognitives dans la section 6.6.4.

6.6.1 Construction des cartes cognitives

Avant la création de nouvelles cartes cognitives, le logiciel demande tout d'abord sur quel ensemble de valeurs celles-ci doivent être définies. L'écran présenté à la figure 6.24 s'affiche alors. Comme on peut le voir, différents choix d'ensembles de valeurs sont proposés à l'utilisateur. Ici, nous allons construire l'ensemble univoque de l'exemple 4.2. Nous choisissons donc l'intervalle de valeurs $[-1; 1]$.

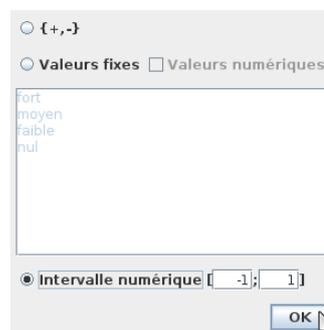


FIGURE 6.24 – Choix de l'ensemble de valeurs.

Une fois l'ensemble de valeurs sélectionné, la fenêtre principale de l'application s'ouvre sur l'écran présenté à la figure 6.25. La zone **1** est dédiée à la construction de la taxonomie. La zone **2** est dédiée à la construction d'une carte cognitive. La taxonomie doit nécessairement être construite avec la carte cognitive car tout

³<http://www.jgraph.com>

concept de la carte cognitive doit être présent dans la taxonomie. La zone **3** représente la liste des différents designers de l'ensemble univoque. Cliquer sur l'un de ces designers permet d'afficher sa carte attribuée. Le menu déroulant **4** liste les différentes opérations possibles sur les cartes qui sont présentées dans les sections suivantes. On note qu'il y a une entrée *Vue* dans le menu des opérations. Elle permet la construction d'une échelle pour construire une vue d'une carte cognitive, conformément au modèle définition par L. Chauvin (voir section 2.3). Puisque cette notion n'est pas issue de notre travail de thèse, nous ne décrivons pas son implémentation.

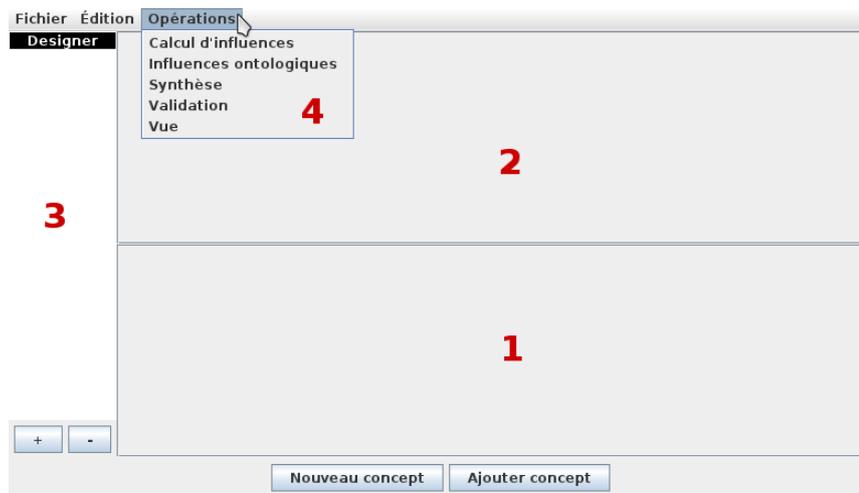


FIGURE 6.25 – Fenêtre principale de l'application.

Construction d'une taxonomie

Nous présentons dans cette section comment construire la taxonomie de la figure 4.2. La construction d'une taxonomie commence par la définition de nouveaux concepts. On clique tout d'abord sur le bouton *Nouveau concept*. Une fenêtre de dialogue s'ouvre invitant l'utilisateur à entrer le texte descriptif du nouveau concept. Un nouveau concept est alors créé dans la taxonomie. La figure 6.26 illustre l'ajout d'un concept *Policier* dans la taxonomie.

Une fois les concepts de la taxonomie définis, il faut spécifier les liens d'héritage entre ces concepts. Pour cela, on sélectionne le centre du concept spécialisé et on trace un arc vers le concept plus général, comme présenté à la figure 6.27. Le système interdit certains liens d'héritage s'ils entraînent des incohérences, par exemple si cela crée un circuit dans la taxonomie.

Construction d'une carte cognitive

Une fois la taxonomie construite, on peut construire la carte cognitive définie sur les concepts de cette taxonomie. La première étape consiste à sélectionner les concepts de la taxonomie pour les ajouter dans la zone de construction de la carte cognitive. Pour cela, on sélectionne le concept qu'on veut ajouter depuis la taxonomie, puis on clique sur le bouton *Ajouter concept*. Le concept est alors ajouté dans la zone

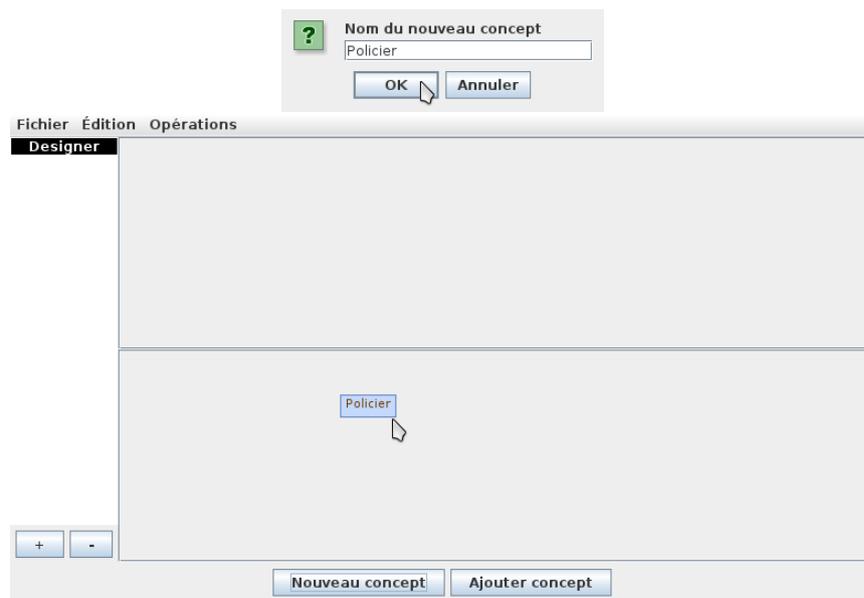


FIGURE 6.26 – Ajout d’un nouveau concept dans la taxonomie.

de construction de la carte cognitive, aux mêmes coordonnées que dans la zone de construction de la taxonomie. La figure 6.28 illustre l’ajout d’un concept *Policier* dans la carte cognitive. Il faut noter qu’on ne peut pas ajouter un concept de la taxonomie dans la carte cognitive si celui-ci est comparable avec un concept déjà présent dans cette carte.

L’ajout d’une influence se fait de manière similaire à l’ajout de liens d’héritage dans la taxonomie, en traçant un arc. Par défaut, les influences n’ont pas de valeur, comme spécifié dans la section 6.1.2. Pour donner une valeur à l’influence, il faut la taper au clavier en sélectionnant le milieu de l’arc, comme illustré sur la figure 6.29. Si la valeur donnée par l’utilisateur n’appartient à l’ensemble de valeurs sur lequel la carte est définie, la valeur précédente de l’influence est restaurée.

Construction d’un ensemble univoque

Si on cherche à réaliser une synthèse de cartes cognitives, il faut construire plusieurs cartes et renseigner des informations sur leurs designers respectifs. Pour ajouter un nouveau designer, il suffit de cliquer sur le bouton + situé sous la liste des designers. Ceci permet d’ajouter un nouveau designer en ouvrant une fenêtre de dialogue invitant l’utilisateur à entrer des informations sur celui-ci. On peut ainsi renseigner son nom ainsi qu’une couleur pour mieux l’identifier, comme présenté sur la figure 6.30. Pour ce nouveau designer, une nouvelle carte attribuée est automatiquement créée. Celle-ci est pour l’instant vide mais est définie sur la même taxonomie et le même ensemble de valeurs que les autres cartes. Le bouton - permet, lui, de supprimer un designer de la liste et par conséquent sa carte attribuée.

Une fois les cartes attribuées de chacun de designers terminées, l’ensemble univoque de cartes attribuées est défini dans son intégralité (figure 6.31).

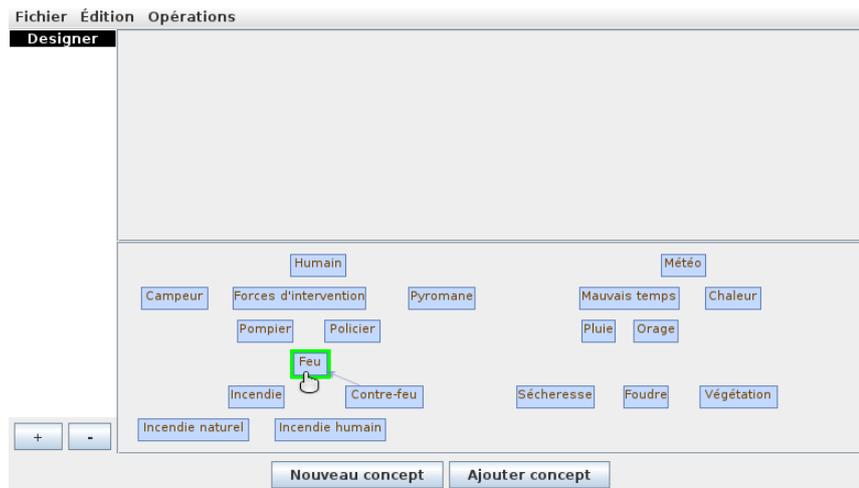


FIGURE 6.27 – Ajout d'un lien d'héritage entre deux concepts de la taxonomie.

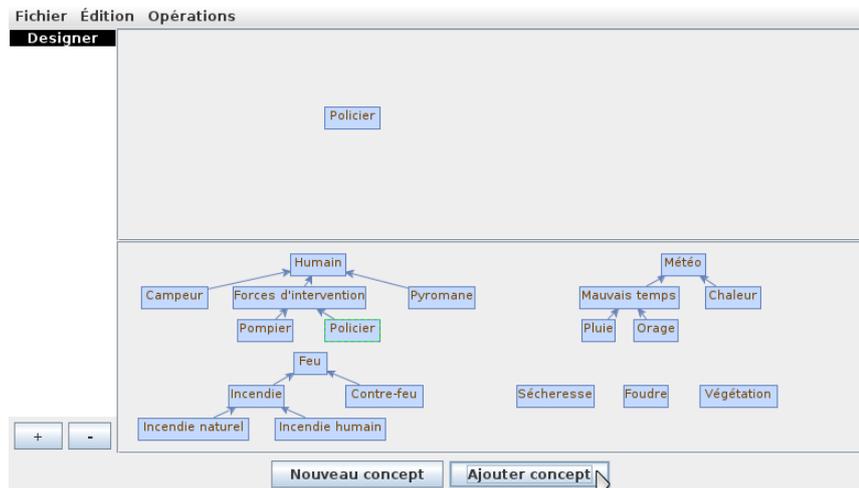


FIGURE 6.28 – Ajout d'un nouveau concept dans la carte cognitive.

6.6.2 Calcul d'influence

Le calcul de l'influence propagée d'un concept sur un autre et le calcul de l'influence taxonomique sont deux opérations distinctes.

Calcul de l'influence propagée d'un concept sur un autre

On accède à la fenêtre de calcul de l'influence propagée d'un concept sur un autre via le menu des opérations. Le calcul se fait sur la carte couramment sélectionnée.

Pour effectuer le calcul, on sélectionne tout d'abord les différents opérateurs qu'on veut appliquer. La figure 6.32 présente l'écran de sélection des algorithmes pour le calcul de l'influence propagée. La carte cognitive est celle de l'exemple 2.1. On a le choix entre deux types d'influences propagées : l'influence classique, qui correspond aux définitions présentées dans le chapitre 2 et une influence propagée probabiliste, ici identifiée sous le terme d'*influence bayésienne*, présentée au chapitre 5.

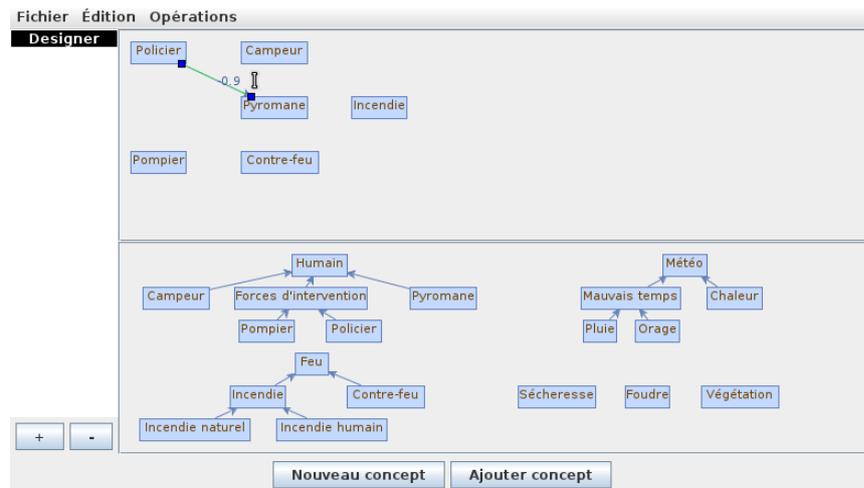


FIGURE 6.29 – Ajout d’une nouvelle influence dans la carte cognitive.

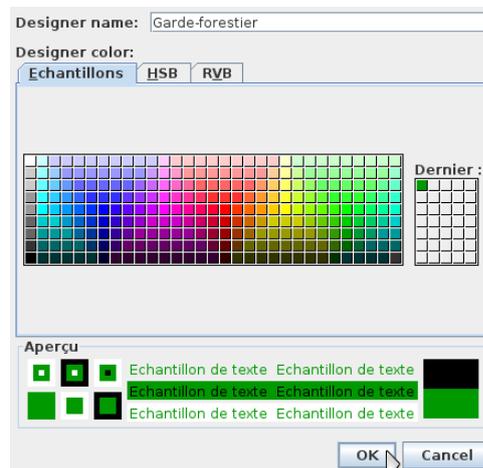


FIGURE 6.30 – Ajout d’un nouveau designer et d’une nouvelle carte attribuée dans l’ensemble univoque.

Pour paramétrer l’influence classique, on sélectionne tout d’abord la définition d’un chemin minimal qu’on souhaite appliquer, puis le processeur de valeurs (présenté dans la section 6.2.3) pour l’influence propagée sur un chemin et enfin le processeur de valeurs pour l’influence propagée d’un concept sur un autre. La liste des processeurs de valeurs pour l’influence propagée sur un chemin contient l’ensemble des processeurs de valeurs qui sont compatibles avec l’ensemble de valeurs de la carte cognitive. Lorsqu’un processeur de valeurs est choisi, la liste des processeurs de valeurs pour l’influence propagée d’un concept sur un autre est mise à jour pour ne contenir que les processeurs de valeurs compatibles avec l’ensemble de sortie de ce processeur, comme présenté dans la figure 6.9.

Pour paramétrer l’influence propagée probabiliste, on doit sélectionner la fonction de conversion nécessaire au calcul. On peut également indiquer si on souhaite que cet opérateur renvoie une probabilité ou non.

On choisit ensuite les concepts concernés par le calcul de l’influence. Pour cela,

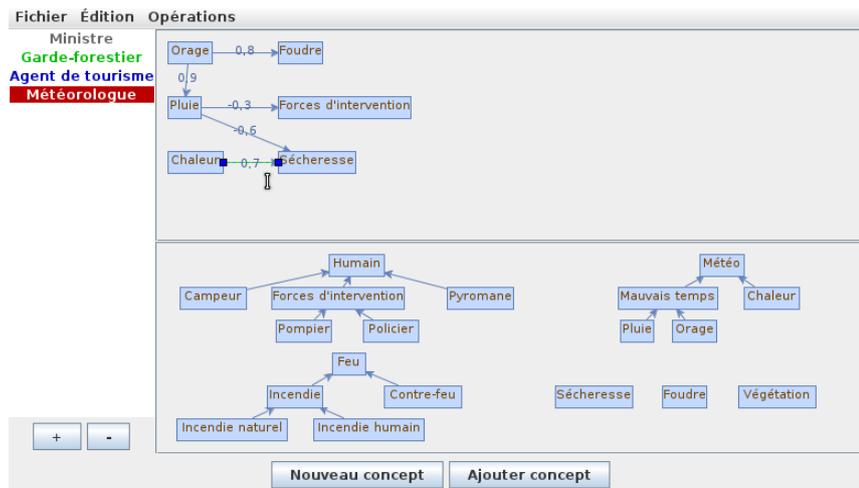


FIGURE 6.31 – Un ensemble univoque de cartes attribuées.

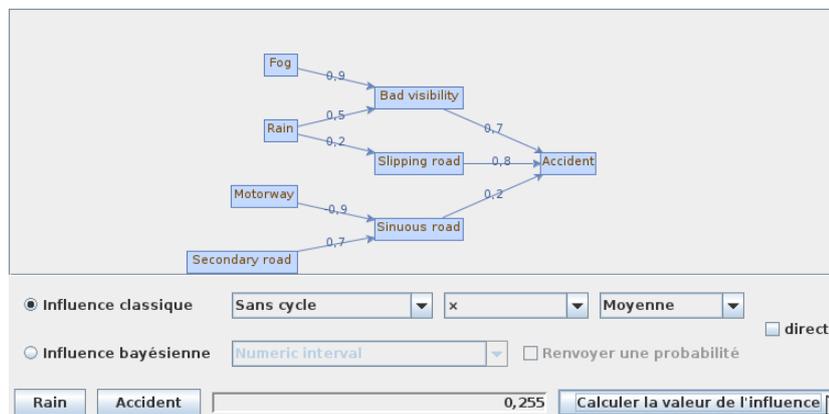


FIGURE 6.32 – Calcul de l'influence propagée d'un concept sur un autre.

l'utilisateur doit considérer les deux boutons situés en bas à gauche. Le bouton de gauche représente le concept-cause tandis que le bouton de droite représente le concept-effet. Pour sélectionner un concept, l'utilisateur doit cliquer sur de ces deux boutons, puis sur un concept dans la carte cognitive. Une fois la même opération effectuée avec l'autre bouton, le calcul de l'influence peut se faire en cliquant sur le bouton *Calculer la valeur de l'influence*.

Calcul de l'influence taxonomique

On accède à la fenêtre de calcul de l'influence taxonomique également via le menu des opérations.

La manière de procéder est assez similaire. La principale différence réside dans le fait qu'on sélectionne cette fois les concepts de la taxonomie et pas uniquement ceux de la carte cognitive, comme présenté à la figure 6.33.

Pour paramétrer ce calcul, on sélectionne tout d'abord l'influence propagée d'un concept sur un autre, exactement présenté précédemment. La différence est qu'il faut en plus sélectionner l'influence taxonomique. Celle-ci est simplement

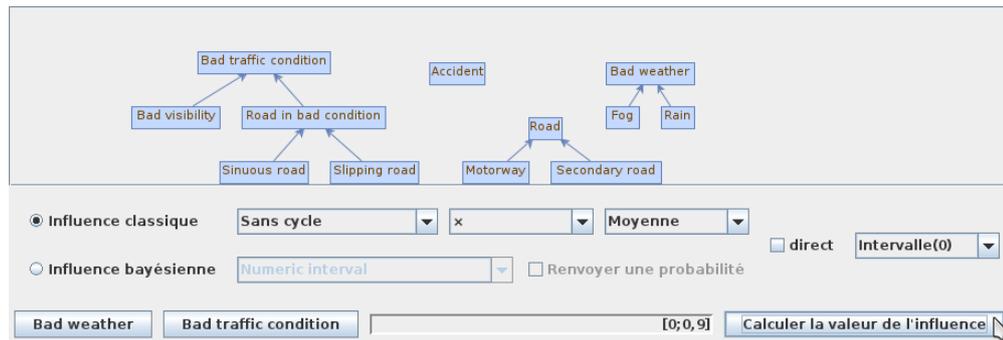


FIGURE 6.33 – Calcul de l'influence propagée taxonomique.

définie par un processeur de valeurs, qu'on choisit de la même manière que ceux de l'influence classique. Là aussi, les processeurs proposés sont uniquement ceux compatibles avec l'ensemble de sortie de l'influence propagée d'un concept sur un autre.

6.6.3 Validation d'une carte cognitive

On accède à la fenêtre de validation d'une carte cognitive via le menu des applications.

La figure 6.34 illustre la fenêtre présentée à l'utilisateur pour valider une carte cognitive. La carte cognitive en cours de validation est celle de l'exemple 2.1. La partie *Influences* permet de sélectionner les opérateurs de calcul d'influence, notamment d'influence propagée d'un concept sur un autre et d'influence taxonomique. Tant que les opérateurs ne sont pas choisis, les boutons de validation des critères sont grisés de manière à désactiver la validation de la carte cognitive. À noter que si on choisit une influence propagée probabiliste, puisque celle-ci ne définit pas d'influence propagée sur un chemin, le critère de non-ambiguïté reste désactivé.

Chacun des critères de qualité permet de sélectionner une instance de *ContradictoryValuesChecker*. À gauche, on peut choisir cet opérateur parmi une liste ne nécessitant pas de paramètre. Ainsi, pour l'ensemble de valeurs $\{+, -\}$, on a le choix entre l'égalité stricte et la notion de valeurs compatibles. À droite, on peut choisir un opérateur de différence ainsi qu'un seuil. Ces deux composantes permettent de définir la notion de valeurs compatibles selon un seuil. Tout à droite se trouve un composant indiquant si le critère a été validé ou non. S'il est vert, le critère est validé. S'il est rouge, il ne l'est pas. Une absence de couleur signifie que la validation du critère n'a pas encore été effectuée.

Pour le critère de test de cohérence, on doit définir en plus une spécification de contraintes, représentée par le tableau situé tout en bas. L'ajout d'une contrainte se fait via une nouvelle fenêtre (figure 6.35) permettant de choisir les deux concepts de cette contrainte parmi les concepts de la taxonomie. Ce choix s'effectue de la même manière que pour choisir les concepts cause et effet lors du calcul d'une influence. On choisit aussi la valeur de la contrainte. Dans le tableau représentant la spécification, la case tout à droite de chaque ligne indique si la contrainte sur

Influences

Influence classique Sans cycle A v direct OK

Influence bayésienne {+, @, 0, e, ~, ?} Renvoyer une probabilité

Propreté

= Différence : Seuil : 0 Valider ■

Non-ambiguïté

= Différence : Seuil : 0 Valider ■

Cohérence

Compatible Différence : Fixed Seuil : 0 Valider ■

+ M -

Source	Destination	Valeur
Bad traffic condition	Accident	+ ■
Bad weather	Bad traffic condition	+ ■
Road	Sinuous road	- ■

FIGURE 6.34 – Validation d'une carte cognitive.

cette ligne a été ou non validée avec le même code couleur que celui du composant présenté précédemment.

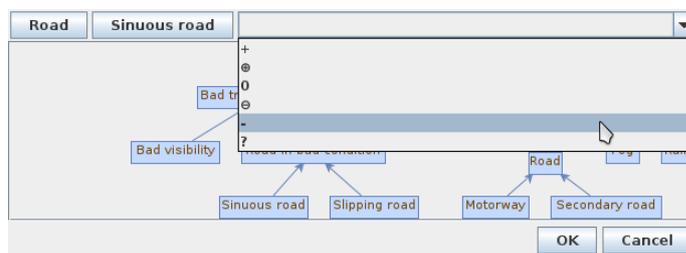


FIGURE 6.35 – Création d'une contrainte de la spécification.

6.6.4 Synthèse de cartes cognitives

On accède à la fenêtre de synthèse de cartes cognitives via le menu des applications.

La figure 6.36 présente la fenêtre qui s'affiche à l'utilisateur. Celui-ci est invité à entrer des préférences sur les designers des différentes cartes ainsi que les différents algorithmes pour la synthèse. Les trois listes déroulantes présentent respectivement

les algorithmes de synthèse des concepts, de synthèse des influences et de synthèse des valeurs.

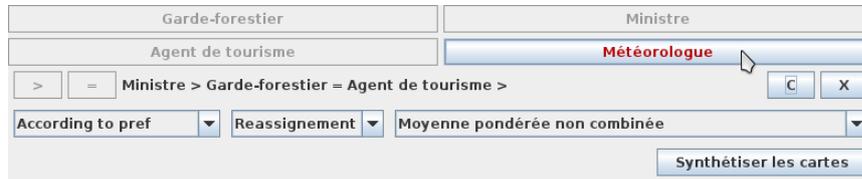


FIGURE 6.36 – Configuration de la synthèse des cartes cognitives.

En appuyant sur le bouton *Synthétiser les cartes*, une nouvelle fenêtre s’ouvre affichant la carte synthétisée. La carte ressemble à ce qui est présenté à la figure 6.37. Les concepts sont en réalité placés selon leurs coordonnées dans la taxonomie. Idéalement, des coordonnées devraient être calculées automatiquement pour un affichage optimal. En réorganisant manuellement les concepts comme présenté à la figure 6.38, on retrouve un affichage similaire à celui de la figure 4.6. Dans l’idéal, la carte synthétisée devrait être ajoutée dans la fenêtre principale du logiciel de manière à être manipulée comme n’importe quelle autre carte cognitive.

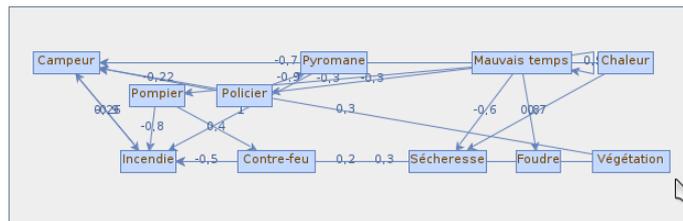


FIGURE 6.37 – Affichage de la carte synthétisée.

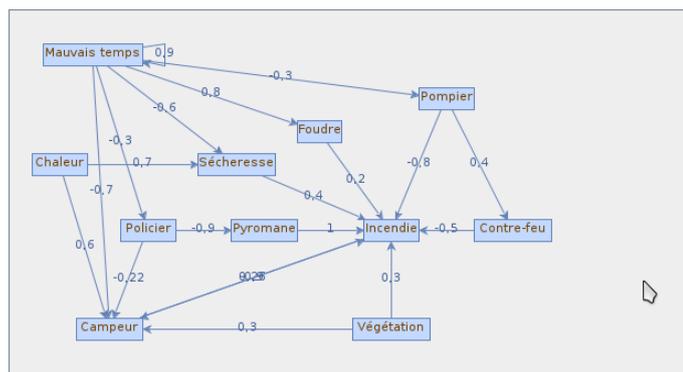


FIGURE 6.38 – Meilleur affichage de la carte synthétisée.

Conclusion

L’implémentation des différentes notions que nous avons introduites de notre thèse nous a permis de les tester sur des exemples. De plus, le fait d’implémenter toutes

ces notions au sein du même logiciel nous ont permis de nous assurer qu'elles étaient compatibles entre elles.

Par ailleurs, le fait d'avoir un logiciel unique pour tous les modèles de cartes existants nous a encouragés à produire un code le plus générique possible. Ceci a également permis de nous interroger sur les caractéristiques que devaient avoir ces modèles pour que les différentes notions définies puissent s'y appliquer.

Nous avons également écrit la documentation des classes et méthodes pour le package `model`, qui constitue le cœur de notre travail de thèse en vue d'une réutilisation future du code par d'autres personnes.

Conclusion

Notre travail de thèse permet d'aider un designer dans la construction et la manipulation d'une carte cognitive selon trois axes. D'abord, la validation permet à un designer de s'assurer de la qualité de la carte cognitive qu'il a construite grâce à différents critères de qualité. Lorsque la carte est invalide, la construction d'un conflit ou d'un diagnostic lui permet de corriger aisément la carte de manière à ce qu'elle respecte le critère voulu. Nous proposons des critères spécifiques aux cartes cognitives qui prennent en compte leurs particularités, ce qui n'a, à notre connaissance, jamais été fait. Nous avons proposé une méthode pour adapter les différents critères définis selon les valeurs utilisées par la carte cognitive. La validation permet également de mettre en avant les ambiguïtés dans une carte cognitive dans le but de provoquer une discussion entre différents intervenants lors d'une séance de brainstorming. Ensuite, la synthèse regroupe les connaissances de divers designers au sein d'une carte unique représentant un consensus entre les différents points de vue. Cette synthèse utilise un ordre de préférence sur les différents designers de manière à indiquer que, en cas d'information contradictoire, on préfère l'information fournie par un designer plutôt que celle fournie par un autre. La synthèse permet donc une manipulation d'une carte cognitive par plusieurs designers ou utilisateurs. Les travaux précédents sur la synthèse de cartes cognitives [Poignonec, 2006; Christiansen, 2011] ne permettaient pas de spécifier une telle préférence. La synthèse est personnalisable selon les besoins de l'utilisateur en proposant différents opérateurs de synthèse, que ce soit pour les concepts, les influences ou les valeurs. Enfin, l'étude de la sémantique attribuée à une carte cognitive que nous avons proposée permet au designer d'une carte de choisir le modèle qui lui correspond le mieux. Elle lui permet aussi de voir si les opérateurs qu'il utilise pour calculer des influences sont pertinents. Le modèle des cartes cognitives probabilistes indique également quelle sémantique porte une carte cognitive dont les valeurs sont apparentées à des probabilités.

Les différentes notions que nous avons introduites sont toutes applicables au modèle de base des cartes cognitives. En effet, ces différentes notions étendent ce modèle en liant une carte cognitive classique à d'autres informations telles qu'une taxonomie, des critères de qualité ou bien des préférences sur les designers des différentes cartes. Ces extensions n'interfèrent pas entre elles et peuvent donc être appliquées en même temps sur une même carte sans souci d'adaptation et sans craindre d'éventuels problèmes. Ceci nous a permis de développer un outil logiciel unique, qui permet d'appliquer ces notions à une carte ou à un ensemble de cartes dans une interface unifiée. Nous l'avons ainsi utilisé pour tester ces notions sur des cartes cognitives du monde de la pêche [Christiansen, 2011].

Notre thèse ouvre des perspectives qui mériteraient d'être étudiées. Quatre perspectives sont évoquées.

L'utilisation de notre travail dans le cadre de réels projets est une première perspective. Nous avons appliqué chacune des notions sur les exemples introduits dans les différents chapitres ainsi que sur quelques cartes issues du monde réel. Si nous en avons tiré des conclusions qui nous ont semblé pertinentes, nous n'avons pas eu accès aux avis de réels experts pour les confirmer. Ceux-ci pourraient fournir des critiques constructives sur nos définitions, ce qui nous permettrait de les améliorer pour qu'elles soient mieux adaptées aux problèmes que ces experts cherchent à modéliser. Nous sommes entrés en contact avec des biologistes et des géographes qui étaient intéressés par notre approche, notamment en ce qui concerne la synthèse de cartes cognitives. Ils cherchent à effectuer une enquête auprès de pêcheurs pour étudier les conséquences de l'installation d'éoliennes en mer sur la pêche. Notre logiciel a convaincu les collègues de la pertinence de notre approche. Nous avons répondu conjointement à un appel à projet pour réaliser cette enquête. En cas d'acceptation, les travaux menés dans le cadre de ce projet se baseraient sur les notions proposées dans cette thèse et la partie développement reprendrait notre outil logiciel.

Une deuxième perspective consiste à tirer profit des nos travaux de validation pour ceux de synthèse. On rappelle d'un côté que la validation permet de mettre en avant les ambiguïtés qui peuvent exister au sein d'une carte. Définir des conflits et des diagnostics permet à un designer de corriger aisément la carte pour résoudre cette ambiguïté. On rappelle d'un autre côté que la synthèse permet de construire une carte unique à partir de plusieurs cartes en utilisant des préférences pour gérer les informations contradictoires d'une carte à l'autre. On peut voir un rapprochement entre la notion d'ambiguïté et d'information contradictoire. Ainsi, on peut imaginer définir des critères de qualité non pas applicables à une carte unique comme ceux que nous avons présentés mais applicables à un ensemble de cartes, dans le but de les synthétiser. Disposer de critères exprimés sur un ensemble de cartes permet de mettre en avant les informations contradictoires d'une carte à l'autre et d'aider les designers à adapter leurs cartes pour corriger ces contradictions. Une fois les cartes corrigées et les informations contradictoires supprimées, la synthèse des cartes s'en retrouve facilitée. Une telle approche permet de plus de s'assurer qu'aucune information essentielle n'est supprimée lors de la synthèse puisque celle-ci est entièrement supervisée par les designers.

Une troisième perspective porte sur la sémantique des cartes cognitives. Notre discussion sur la sémantique peut en effet être étendue à une réflexion sur la pertinence des critères de qualité et des opérateurs de synthèse pour une sémantique donnée. Ceci permettrait de s'assurer que les notions de validation et de synthèse sont bien adaptées à la sémantique de la carte. En effet, la validation est chargée entre autres de vérifier qu'une carte cognitive ne contient pas d'ambiguïtés. Prendre en compte la sémantique attribuée aux concepts, aux influences et aux valeurs permet de définir des critères de qualité chargés de s'assurer que cette sémantique est bien respectée. Les opérateurs utilisés lors de la synthèse seraient également plus pertinents si on prenait en compte cette sémantique.

De cette discussion, nous pouvons également définir une quatrième perspective

portant sur l'utilisation des QPN. Le modèle des cartes cognitives probabilistes que nous avons défini se limite à une catégorie de problèmes très particuliers. En effet, de par la sémantique de ce modèle, les systèmes qu'il est capable de représenter doivent respecter strictement certaines conditions. Par exemple, les concepts d'une telle carte cognitive doivent être nécessairement quantifiables, ce qui limite nécessairement leur degré d'expressivité. Pour cette raison, utiliser des valeurs de probabilités directement sur les arcs d'une carte cognitive nous semble une mauvaise approche. Spécifier des contraintes sur les probabilités comme le fait le modèle des QPN nous semble donc plus pertinent. Une réflexion sur la manière d'adapter la validation et la synthèse au modèle des QPN serait utile.

Annexes

A Validation de cartes du monde de la pêche

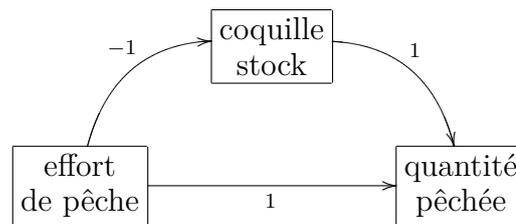
Nous avons testé les critères de qualité que nous avons définis dans le chapitre 3 sur des cartes cognitives du monde de la pêche issues d'un mémoire d'agronomie [Christiansen, 2011]. Ce mémoire a été rédigé par Gwen Christiansen et présente diverses cartes cognitives réalisées par différents pêcheurs dans le but d'étudier les influences de divers concepts sur la pêche à la coquille Saint-Jacques. Il présente également des cartes cognitives représentant une synthèse des informations d'autres cartes pour en étudier les divergences. Nous présentons ici deux cartes cognitives réalisées par deux pêcheurs différents et appliquons sur ces cartes nos critères de qualité.

Validation d'une première carte cognitive

La carte cognitive de la figure A.1 est une carte issue du mémoire. Elle représente la vision d'un pêcheur sur les influences de différents concepts sur la pêche à la coquille Saint-Jacques. Les couleurs ainsi que la forme des concepts (rond, carré ou losange) font référence à des notions qui ne nous intéressent pas ici. Les couleurs des influences servent uniquement à mettre en évidence les influences positives et négatives. Comme on peut le voir sur cette carte, il y a beaucoup d'influences directes et les concepts s'influencent les uns les autres généralement sur un seul chemin.

L'ensemble de valeurs sur lequel est définie la carte cognitive est $\{-3, -2, -1, 0, 1, 2, 3\}$. Ces valeurs permettent de quantifier la force de l'influence, à la manière de $\{nul < faible < moyen < fort\}$. Par soucis de simplification, nous modifions la carte pour qu'elle soit définie sur l'ensemble de valeurs $\{-1, -\frac{2}{3}, -\frac{1}{3}, 0, \frac{1}{3}, \frac{2}{3}, 3\}$. Cet ensemble de valeurs est sémantiquement quasiment identique à l'ensemble $\{-3, -2, -1, 0, 1, 2, 3\}$. Nous effectuons cette modification pour appliquer simplement les opérateurs de calcul d'influence utilisés par les cartes cognitives définies sur $[-1; 1]$, présentés dans le chapitre 2.

Le couple de concepts qui nous intéresse est (*effort de pêche*, *quantité pêchée*). La figure A.2 représente un détail de la figure A.1 quand laquelle seules les influences liant les concepts de ce couple sont représentées. Nous avons également adapté les valeurs, comme expliqué précédemment. On voit qu'il y a deux chemins permettant de lier ces deux concepts. Intuitivement, il y a une ambiguïté entre les valeurs des influences sur les différents chemins. C'est d'ailleurs cette ambiguïté qui est indiquée par G. Christiansen sur la carte de la figure A.1 par une flèche

FIGURE A.2 – Influences de *effort de pêche* sur *quantité pêchée*.

Application de nos critères sur la carte

En appliquant le critère de propreté (définition 3.1) sur le couple de concepts (*effort de pêche*, *quantité pêchée*), on se rend compte qu'il n'est pas propre. En effet, le calcul de l'influence propagée de *effort de pêche* sur *quantité pêchée* nous donne 0. Or, l'influence directe de *effort de pêche* vers *quantité pêchée* est de 1. Ces deux valeurs étant différentes, on en déduit que le couple de concepts n'est pas propre.

On est donc également sûr que ce couple de concepts est ambigu. On peut le vérifier par le calcul pour s'en assurer. L'influence propagée sur le chemin *effort de pêche* \rightarrow *coquille stock* \rightarrow *quantité pêchée* est de -1 . L'influence propagée sur le chemin *effort de pêche* \rightarrow *quantité pêchée* est de 1. Ces deux valeurs sont donc différentes de l'influence propagée entre ces deux concepts, qui est de 0. Le couple de concepts est donc bien ambigu.

Adaptation de nos critères pour correspondre au formalisme de la carte

Si on veut être rigoureux dans l'application de nos critères, il faut être en accord avec la sémantique associée à la carte cognitive. Il faut donc s'assurer d'utiliser le même formalisme que celui utilisé par la carte, notamment en ce qui concerne le calcul des influences. Or, dans de telles cartes, la manière de calculer les influences est différente de celle que nous utilisons pour nos critères de qualité. En effet, on rappelle que les seuls chemins que nous ne considérons pour le calcul de l'influence propagée sont des chemins minimaux. Nous avons défini la notion de chemin minimal comme étant un chemin sans cycle (définition 2.2). Or, dans son mémoire, G. Christiansen précise qu'elle se limite pour les calculs d'influences à des chemins composés d'au plus 3 influences. Les chemins minimaux que nous devons considérer sont donc des chemins limités à 3 influences (définition 2.25). C'est donc cette définition de chemin minimal qu'il faut utiliser pour nos critères de qualité.

L'influence propagée sur un chemin utilisée par G. Christiansen est également différente de celle que nous utilisons couramment. Sa définition étant assez complexe, nous ne l'exprimons pas de manière formelle mais nous en donnons simplement l'idée. Le signe de l'influence propagée sur un chemin est défini comme étant le produit des signes des valeurs des influences qui composent ce chemin. On associe ensuite ce signe à une valeur, qui est la plus petite en valeur absolue des influences qui composent ce chemin. L'influence propagée sur un chemin est alors définie comme étant l'association de ce signe et de cette valeur.

Dans son mémoire, G. Christiansen ne précise pas quel est l'opérateur utilisé pour le calcul de l'influence propagée d'un concept sur un autre. Or, les critères

que nous avons définis en ont besoin. Nous utilisons donc l'opérateur classique pour des valeurs numériques (définition 2.6).

En adaptant les critères de propreté et de non-ambiguïté à tous ces paramètres, on montre à nouveau que le couple (*effort de pêche, quantité pêchée*) est non-propre et ambigu. On peut également appliquer ces critères de la même manière pour d'autres couples de concepts de la carte. Ainsi, on peut facilement montrer que le couple (*amélioration du matériel, quantité pêchée*) est également ambigu. De la même manière, on montre que (*pêche à la paire, quantité pêchée*) et (*effort de pêche, chacun dans son coin, B vs T*) sont ambigus. En revanche, le couple (*amélioration du matériel, chacun dans son coin, B vs T*) n'est pas ambigu : en effet, puisque les chemins minimaux sont limités à 3 influences, aucun chemin minimal entre ces deux concepts ne passe par *coquille stock* et donc par l'influence portant la valeur -3 .

Validation d'une seconde carte cognitive

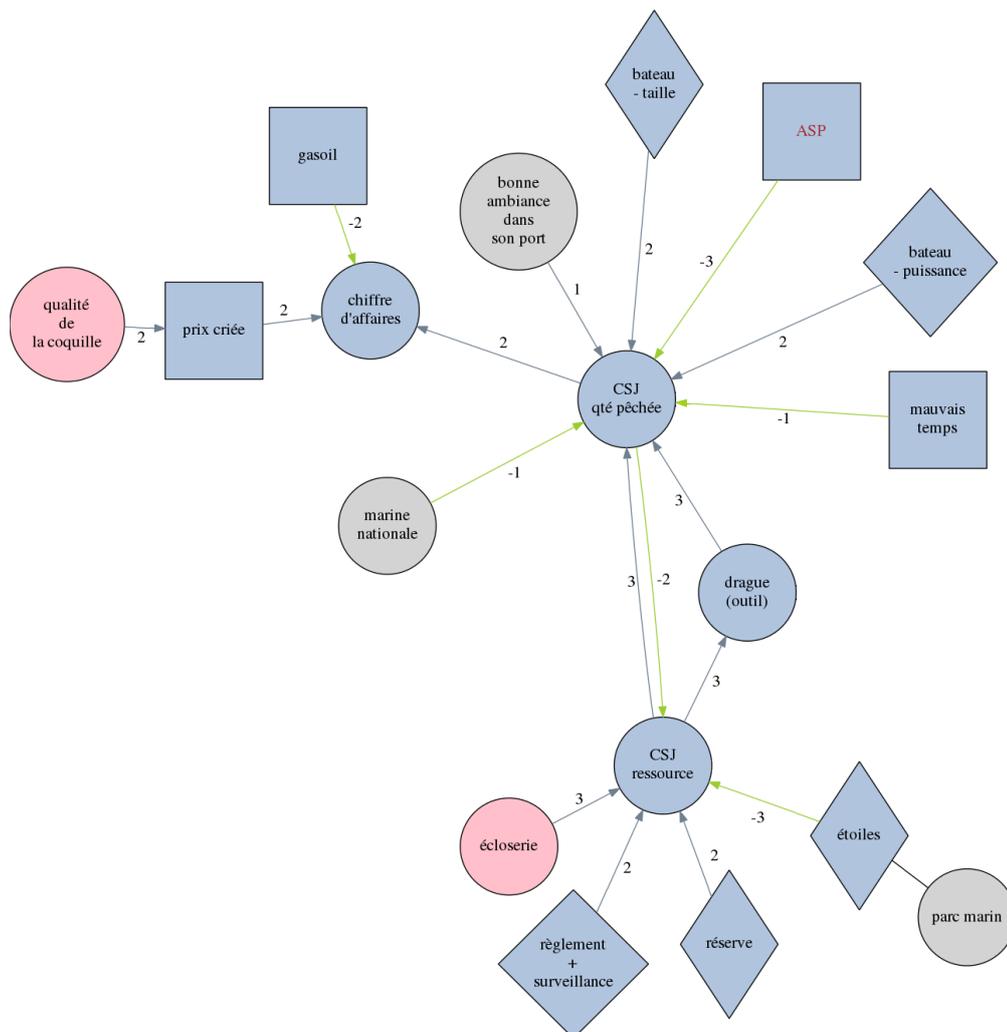


FIGURE A.3 – Carte cognitive d'un autre pêcheur.

Nous avons également appliqué nos critères sur une seconde carte cognitive réalisée par un autre pêcheur, présentée à la figure A.3. Nous l'avons à nouveau redéfinie sur l'ensemble de valeurs $\{-1, -\frac{2}{3}, -\frac{1}{3}, 0, \frac{1}{3}, \frac{2}{3}, 3\}$.

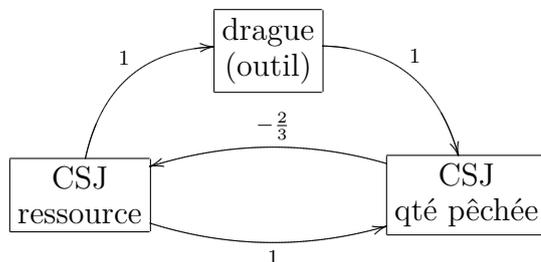


FIGURE A.4 – Influences de *CSJ ressource* sur *CSJ qté pêchée*.

En appliquant nos critères de propreté et de non-ambiguïté avec les opérateurs classiques du calcul de l'influence propagée, on en déduit que cette carte est propre et non-ambiguë. Ainsi, le couple de concepts (*CSJ ressource*, *CSJ qté pêchée*) (figure A.4) est propre et non-ambigu. La valeur $-\frac{2}{3}$ portée par l'influence de *CSJ qté pêchée* vers *CSJ ressource* n'est pas utilisée dans le calcul de l'influence propagée entre les concepts de ce couple puisqu'elle ne fait partie d'aucun chemin minimal de *CSJ ressource* vers *CSJ qté pêchée*. En effet, avec les opérateurs classiques, les chemins minimaux ne doivent comporter aucun cycle.

On adapte à présent nos critères au formalisme de la carte cognitive, comme nous l'avons présenté dans la section précédente. L'influence portant la valeur $-\frac{2}{3}$ fait à présent partie d'un chemin minimal. En effet, si on part du concept *CSJ ressource*, qu'on atteint le concept *CSJ qté pêchée*, puis qu'on revient à *CSJ ressource* en passant par l'influence marquée par $-\frac{2}{3}$, on peut à nouveau rejoindre le concept *CSJ qté pêchée*. Nous avons emprunté un chemin composé d'exactly 3 influences. Il est donc minimal. Pour cette raison, le couple de concepts (*CSJ ressource*, *CSJ qté pêchée*) devient ambigu si on adapte le critère d'ambiguïté au formalisme de la carte. On en déduit par conséquent que ce couple de concepts est également non-propre.

B Preuve du théorème 1

Dans cette section, nous démontrons le théorème 1. Nous cherchons donc à montrer $\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \mathbb{P}(X_{c_2} = + | \text{do}(X_{c_1} = +)) - 0.5$. Pour cela, nous exprimons $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$ sous la forme d'un opérateur récursif. Puis, nous faisons de même pour $\mathbb{P}(X_{c_2} | \text{do}(X_{c_1}))$. Enfin, nous montrons l'équivalence des deux opérateurs récursifs, ce qui permet de déduire la relation entre $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$ et $\mathbb{P}(X_{c_2} | \text{do}(X_{c_1}))$.

Par souci de concision, on note dans cette partie :

- pour un concept c associé à une variable aléatoire X_c , l'état $X_c = +$ par c et l'état $X_c = -$ par \bar{c} ;
- pour deux concepts c_1 et c_2 , $\mathbb{P}(c_1 \cap c_2)$ par $\mathbb{P}(c_1 c_2)$;
- pour une influence (c_1, c_2) , $\mathcal{F}(\text{label}((c_1, c_2)))$ par α_{c_1, c_2} ;
- pour un concept c , l'ensemble des vecteurs représentant les configurations possibles des états des parents de c dans un réseau bayésien par $\vec{\mathcal{C}}(c)$ (ainsi, dans l'exemple 5.14, on a $\vec{\mathcal{C}}(S) = \{(J, V), (J, \bar{V}), (\bar{J}, V), (\bar{J}, \bar{V})\}$ ¹ ;
- pour un concept c , l'ensemble des vecteurs représentant les configurations possibles des états des parents de c privés de la valeur associée à c' par $\vec{\mathcal{C}}_{c'}(c)$ (ainsi, dans l'exemple 5.14, on a $\vec{\mathcal{C}}_J(S) = \{(V), (\bar{V})\}$).

Remarque. On peut facilement montrer qu'on a, pour n'importe quel concept c , $\sum_{X \in \vec{\mathcal{C}}(c)} \mathbb{P}(X) = 1$. De la même manière, on a, pour n'importe quel autre concept c' , $\sum_{X \in \vec{\mathcal{C}}(c)} \mathbb{P}(X c') = \mathbb{P}(c')$.

Expression de $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$ sous forme récursive

On rappelle que :

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \begin{cases} 0.5 & \text{si } c_1 = c_2 \\ \sum_{P \in \mathcal{P}_{c_1, c_2}} \text{part}(P) \times \mathcal{I}\mathcal{P}_{\mathbb{P}}(P) & \text{sinon} \end{cases}$$

Nous exprimons l'opérateur $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$ sous forme récursive. Tout d'abord, nous considérons que $c_1 = c_2$. Puis, nous considérons que $c_1 \neq c_2$ et qu'il n'y a aucun chemin de c_1 vers c_2 , c'est-à-dire que $\mathcal{P}_{c_1, c_2} = \emptyset$. Enfin, nous considérons que $c_1 \neq c_2$ et qu'il y a au moins un chemin de c_1 vers c_2 , c'est-à-dire que $\mathcal{P}_{c_1, c_2} \neq \emptyset$.

Cas où $c_1 = c_2$

Si $c_1 = c_2$, on a, par définition, $\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = 0.5$.

¹Cette notion ne doit pas être confondue avec $\mathcal{C}(c)$, qui représente l'ensemble des parents du concept c .

Cas où $c_1 \neq c_2$ et $\mathcal{P}_{c_1, c_2} = \emptyset$

Si $c_1 \neq c_2$ et $\mathcal{P}_{c_1, c_2} = \emptyset$, on a trivialement $\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = 0$.

Cas où $c_1 \neq c_2$ et $\mathcal{P}_{c_1, c_2} \neq \emptyset$

Si $c_1 \neq c_2$, en utilisant les définitions de la part d'un chemin (définition 5.22) et de l'influence propagée probabiliste sur un chemin (définition 5.21), on a :

$$\begin{aligned} \mathcal{I}_{\mathbb{P}}'(c_1, c_2) &= \sum_{P \in \mathcal{P}_{c_1, c_2}} \text{part}(P) \times \mathcal{I}\mathcal{P}_{\mathbb{P}}(P) \\ &= \sum_{P \in \mathcal{P}_{c_1, c_2}} \left(\prod_{i=1}^k \frac{1}{|\mathcal{C}(u_i)|} \right) \times \left(\frac{1}{2} \times \prod_{i=0}^{k-1} 2 \times \mathcal{F}(\text{label}((u_i, u_{i+1}))) \right) \\ &= \sum_{P \in \mathcal{P}_{c_1, c_2}} \frac{1}{2} \times \left(\prod_{i=0}^{k-1} \frac{1}{|\mathcal{C}(u_{i+1})|} \right) \times \left(\prod_{i=0}^{k-1} 2 \times \alpha_{u_i, u_{i+1}} \right) \\ &= \frac{1}{2} \times \sum_{P \in \mathcal{P}_{c_1, c_2}} \prod_{i=0}^{k-1} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \end{aligned}$$

où le nombre k et les concepts u_i et u_{i+1} font référence au chemin P correspondant.

Nous avons donc l'expression complète de $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$:

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \frac{1}{2} \times \sum_{P \in \mathcal{P}_{c_1, c_2}} \prod_{i=0}^{k-1} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|}$$

Nous réécrivons à présent cette expression pour la mettre sous forme récursive.

La proposition B.1 nous permet de créer une quasi-partition de l'ensemble des chemins minimaux selon les parents du concept-destination. Pour cette proposition, nous réutilisons la notion d'ensemble des chemins minimaux passant par un parent (définition 5.15). Nous parlons de quasi-partition car nous n'avons pas l'assurance que chaque ensemble de chemins minimaux passant par un parent est vide.

Proposition B.1. *Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts.*

1. $\bigcup_{c'_2 \in \mathcal{C}(c_2)} \mathcal{P}_{c_1, c'_2, c_2} = \mathcal{P}_{c_1, c_2}$
2. $\forall c'_2, c''_2 \in \mathcal{C}(c_2), c'_2 \neq c''_2 \Rightarrow \mathcal{P}_{c_1, c'_2, c_2} \cap \mathcal{P}_{c_1, c''_2, c_2} = \emptyset$

Démonstration.

1. (a) Soit $c'_2 \in \mathcal{C}(c_2)$. Si $\mathcal{P}_{c_1, c'_2, c_2} = \emptyset$, alors trivialement $\mathcal{P}_{c_1, c'_2, c_2} \subseteq \mathcal{P}_{c_1, c_2}$. Si $\mathcal{P}_{c_1, c'_2, c_2} \neq \emptyset$, soit $P \in \mathcal{P}_{c_1, c'_2, c_2}$. D'après la définition 5.15, $P \in \mathcal{P}_{c_1, c_2}$. D'où $\mathcal{P}_{c_1, c'_2, c_2} \subseteq \mathcal{P}_{c_1, c_2}$.
Donc $\bigcup_{c'_2 \in \mathcal{C}(c_2)} \mathcal{P}_{c_1, c'_2, c_2} \subseteq \mathcal{P}_{c_1, c_2}$.
- (b) Soit $P \in \mathcal{P}_{c_1, c_2}$. Soit (c'_2, c_2) la dernière influence de P . Par définition, $(c'_2, c_2) \in A$ et, par conséquent, $c'_2 \in \mathcal{C}(c_2)$. On a donc $P \in \mathcal{P}_{c_1, c'_2, c_2}$, d'où $P \in \bigcup_{c'_2 \in \mathcal{C}(c_2)} \mathcal{P}_{c_1, c'_2, c_2}$.
Donc $\bigcup_{c'_2 \in \mathcal{C}(c_2)} \mathcal{P}_{c_1, c'_2, c_2} \supseteq \mathcal{P}_{c_1, c_2}$. □

2. Soient $c'_2, c''_2 \in \mathcal{C}(c_2)$ tels que $c'_2 \neq c''_2$. Soit $P \in \mathcal{P}_{c_1, c'_2, c_2}$. (c'_2, c_2) est donc la dernière influence de P . Or, puisque $c'_2 \neq c''_2$, $(c'_2, c_2) \neq (c''_2, c_2)$. (c''_2, c_2) n'est donc pas la dernière influence de P . Par conséquent, $P \notin \mathcal{P}_{c_1, c''_2, c_2}$. En prenant $P' \in \mathcal{P}_{c_1, c''_2, c_2}$, on montre facilement de la même manière que $P' \notin \mathcal{P}_{c_1, c'_2, c_2}$.
D'où $\mathcal{P}_{c_1, c'_2, c_2} \cap \mathcal{P}_{c_1, c''_2, c_2} = \emptyset$. \square

Grâce à la proposition B.1, en raisonnant sur chacun des parents de c_2 , on peut réécrire :

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \frac{1}{2} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \sum_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \prod_{i=0}^{k-1} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|}$$

Et, en extrayant le dernier membre du produit :

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \frac{1}{2} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \sum_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \left(\frac{2 \times \alpha_{u_{k-1}, u_k}}{|\mathcal{C}(u_k)|} \times \prod_{i=0}^{k-2} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right)$$

On sait que $u_k = c_2$ et, d'après la définition 5.15, que $u_{k-1} = c'_2$. On a donc :

$$\mathcal{I}_{\mathbb{P}}'(c_1, c_2) = \frac{1}{2} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \sum_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \left(\frac{2 \times \alpha_{c'_2, c_2}}{|\mathcal{C}(c_2)|} \times \prod_{i=0}^{k-2} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right)$$

La proposition B.2 nous assure que le nombre de parents de c_2 n'est pas 0.

Proposition B.2. *Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts.*

$$\mathcal{P}_{c_1, c_2} \neq \emptyset \Rightarrow \mathcal{C}(c_2) \neq \emptyset$$

Démonstration. On suppose $\mathcal{P}_{c_1, c_2} \neq \emptyset$. Soit $P \in \mathcal{P}_{c_1, c_2}$ un chemin de c_1 vers c_2 de longueur k . On considère le concept u_{k-1} de P . D'après la définition d'un chemin (définition 2.1), $(u_{k-1}, c_2) \in A$. Donc, d'après la définition des parents d'un concept (définition 2.4), $u_{k-1} \in \mathcal{C}(c_2)$. D'où $\mathcal{C}(c_2) \neq \emptyset$. \square

Par la proposition B.2, puisque $\mathcal{P}_{c_1, c_2} \neq \emptyset$, on a $\mathcal{C}(c_2) \neq \emptyset$. On a donc l'assurance que $|\mathcal{C}(c_2)| \neq 0$. On a alors :

$$\begin{aligned} \mathcal{I}_{\mathbb{P}}'(c_1, c_2) &= \frac{1}{2} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\frac{2}{|\mathcal{C}(c_2)|} \times \alpha_{c'_2, c_2} \times \sum_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \prod_{i=0}^{k-2} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right) \\ &= \frac{1}{2} \times \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \sum_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \prod_{i=0}^{k-2} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right) \end{aligned}$$

La proposition B.3 nous permet de construire l'ensemble des chemins minimaux de c_1 vers un parent de c_2 à partir des chemins minimaux de c_1 vers c_2 passant par ce parent.

Proposition B.3. *Soit $CM = (C, A, \text{label})$ une carte cognitive. Soient $c_1, c_2 \in C$ deux concepts. Soit $c'_2 \in \mathcal{C}(c_2)$ un concept parent de c_2 .*

$$\bigcup_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \bigcup_{i=0}^{k-2} (u_i, u_{i+1}) = \bigcup_{P \in \mathcal{P}_{c_1, c_2}} \bigcup_{i=0}^{k-1} (u_i, u_{i+1})$$

où le nombre k et les concepts u_i et u_{i+1} font référence au chemin P correspondant.

Démonstration.

1. Soit $P \in \mathcal{P}_{c_1, c'_2, c_2}$ un chemin de longueur k avec des concepts u_i . On définit un chemin P' de longueur $k' = k - 1$ basé sur P avec des concepts u'_i tel que $\forall i \in [0; k']$, $u_i = u'_i$. Puisque $P \in \mathcal{P}_{c_1, c'_2, c_2}$, $c'_2 = u_{k-1} = u'_{k'}$. Donc, par définition, P' est un chemin de c_1 vers c'_2 . De plus, comme P' est un chemin dans un graphe sans circuit, P' est minimal (proposition 2.1) et donc $P' \in \mathcal{P}_{c_1, c'_2}$. Soit $i \in [0; k' - 1]$. (u_i, u_{i+1}) est une influence de P et (u'_i, u'_{i+1}) est une influence de P' . Or, puisque $\forall i \in [0; k']$, $u_i = u'_i$, $(u'_i, u'_{i+1}) = (u_i, u_{i+1})$ est une influence de P' .

$$\text{D'où } \bigcup_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \bigcup_{i=0}^{k-2} (u_i, u_{i+1}) \subseteq \bigcup_{P \in \mathcal{P}_{c_1, c'_2}} \bigcup_{i=0}^{k-1} (u_i, u_{i+1}).$$

2. Soit $P' \in \mathcal{P}_{c_1, c'_2}$ un chemin de longueur k' avec des concepts u'_i . On a donc $u'_{k'} = c'_2$. Puisque $c'_2 \in \mathcal{C}(c_2)$, $(c'_2, c_2) \in A$. On définit un chemin P de longueur $k = k' + 1$ basé sur P' avec des concepts u_i tel que $\forall i \in [0; k - 1]$, $u'_i = u_i$ et $u_k = c_2$. Donc, par définition, P est un chemin de c_1 vers c_2 . De plus, comme P est un chemin dans un graphe sans circuit, P est minimal et donc $P \in \mathcal{P}_{c_1, c_2}$. Soit $i \in [0; k - 2]$. (u'_i, u'_{i+1}) est une influence de P' et (u_i, u_{i+1}) est une influence de P . Or, puisque $\forall i \in [0; k - 1]$, $u'_i = u_i$, $(u_i, u_{i+1}) = (u'_i, u'_{i+1})$ est une influence de P' .

$$\text{D'où } \bigcup_{P \in \mathcal{P}_{c_1, c'_2, c_2}} \bigcup_{i=0}^{k-2} (u_i, u_{i+1}) \supseteq \bigcup_{P \in \mathcal{P}_{c_1, c'_2}} \bigcup_{i=0}^{k-1} (u_i, u_{i+1}).$$

□

En utilisant la proposition B.3, on peut simplifier l'expression et en déduire la récursivité :

$$\begin{aligned} \mathcal{I}_{\mathbb{P}}'(c_1, c_2) &= \frac{1}{2} \times \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \sum_{P \in \mathcal{P}_{c_1, c'_2}} \prod_{i=0}^{k-1} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right) \\ &= \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\frac{1}{2} \times \alpha_{c'_2, c_2} \times \sum_{P \in \mathcal{P}_{c_1, c'_2}} \prod_{i=0}^{k-1} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right) \\ &= \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\frac{1}{2} \times \sum_{P \in \mathcal{P}_{c_1, c'_2}} \prod_{i=0}^{k-1} \frac{2 \times \alpha_{u_i, u_{i+1}}}{|\mathcal{C}(u_{i+1})|} \right) \right) \\ &= \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathcal{I}_{\mathbb{P}}'(c_1, c'_2) \right) \end{aligned}$$

Expression récursive de $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$

Dans le cas où $c_1 \neq c_2$ et où $\mathcal{P}_{c_1, c_2} \neq \emptyset$, la définition de $\mathcal{I}_{\mathbb{P}}'(c_1, c_2)$ est récursive selon les parents de c_2 . Puisque la carte est sans circuit, en appliquant cette définition, on finit forcément par atteindre un sommet sans parent. Si ce parent est c_1 , alors on a $\mathcal{I}_{\mathbb{P}}'(c_1, c_1) = 0.5$. Si ce parent est un autre concept c , alors $\mathcal{P}_{c_1, c} = \emptyset$ et donc $\mathcal{I}_{\mathbb{P}}'(c_1, c) = 0$. Ces deux situations sont des cas d'arrêt : la récursivité s'arrête donc

forcément. On a finalement :

$$\mathcal{I}_{\mathbb{P}'}(c_1, c_2) = \begin{cases} 0.5 & \text{si } c_1 = c_2 \\ 0 & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathcal{I}_{\mathbb{P}'}(c_1, c'_2) \right) & \text{sinon} \end{cases}$$

Expression de $\mathbb{P}(c_2 | \text{do}(c_1))$ sous forme récursive

Nous exprimons à présent la loi de probabilité conditionnelle $\mathbb{P}(c_2 | \text{do}(c_1))$ dans le réseau bayésien causal représentant la carte cognitive sous forme récursive. On rappelle que l'application de $\text{do}(c_1)$ dans un tel réseau consiste à couper les branches arrivant vers c_1 . On se place dans un tel réseau. Par conséquent, c_1 devient un sommet sans parent. Si c_1 est un parent de c_2 , on a donc $\mathbb{P}(c_2 | \text{do}(c_1)) = \mathbb{P}(c_2 | c_1)$; si c_1 n'est pas un parent de c_2 , on a $\mathbb{P}(c_2 | \text{do}(c_1)) = \mathbb{P}(c_2)$.

Tout d'abord, nous considérons que $c_1 = c_2$. Puis, nous considérons que $c_1 \neq c_2$ et que c_1 n'est pas un parent de c_2 , c'est-à-dire que $\mathcal{P}_{c_1, c_2} = \emptyset$. Ensuite, nous considérons que $c_1 \neq c_2$ et que c_1 est un parent non-direct de c_2 , c'est-à-dire que $\mathcal{P}_{c_1, c_2} \neq \emptyset$ et $c_1 \notin \mathcal{C}(c_2)$. Enfin, nous considérons que $c_1 \neq c_2$ et que c_1 est un parent direct de c_2 , c'est-à-dire que $\mathcal{P}_{c_1, c_2} \neq \emptyset$ et $c_1 \in \mathcal{C}(c_2)$.

On pose dès à présent, pour deux concepts c_1 et c_2 :

$$\mathbb{P}'(c_2, c_1) = \mathbb{P}(c_2 | c_1) - 0.5$$

Nous aurons besoin de cet opérateur par la suite car c'est lui qui sera exprimé sous forme récursive.

Cas où $c_1 = c_2$

Si $c_1 = c_2$, alors $\mathbb{P}(c_2 | c_1) = \mathbb{P}(c_1 | c_1) = 1$. D'où :

$$\mathbb{P}'(c_2, c_1) = \mathbb{P}(c_2 | c_1) - 0.5 = 1 - 0.5 = 0.5$$

Cas où c_1 n'est pas un parent de c_2

Si c_1 n'est pas un parent de c_2 , puisque c_1 est un sommet sans parent, le seul moyen pour que ces deux concepts soient liés est qu'ils aient au moins un enfant en commun c' . Dans une telle situation, c_1 et c_2 sont d-séparés par c' . Par conséquent, c_1 et c_2 sont indépendants puisque la valeur de c' n'est pas connue. Si un tel enfant n'existe pas, alors c_1 et c_2 ne sont pas liés dans le graphe et sont donc trivialement indépendants. On est donc sûr que c_1 et c_2 sont indépendants. D'où $\mathbb{P}(c_2 | c_1) = \mathbb{P}(c_2) = 0.5$ (définition 5.25). Finalement :

$$\mathbb{P}'(c_2, c_1) = \mathbb{P}(c_2 | c_1) - 0.5 = 0.5 - 0.5 = 0$$

Cas où c_1 est un parent non-direct de c_2

Nous supposons que c_1 est un parent non-direct de c_2 .

On a alors :

$$\begin{aligned}\mathbb{P}(c_2|c_1) &= \frac{\mathbb{P}(c_2 c_1)}{\mathbb{P}(c_1)} \\ &= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \mathbb{P}(c_2 X c_1) \\ &= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(c_2|X c_1) \times \mathbb{P}(X c_1) \right)\end{aligned}$$

Grâce à la notion de d-séparation, puisque $c_1 \notin \mathcal{C}(c_2)$, c_2 est indépendant de c_1 sachant l'état de tous les parents de c_2 . On peut simplifier l'expression de $\mathbb{P}(c_2|c_1)$:

$$\mathbb{P}(c_2|c_1) = \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(c_2|X) \times \mathbb{P}(X c_1) \right)$$

En utilisant la définition 5.27, on peut développer la somme selon les configurations possibles des différents parents c'_2 de c_2 X en utilisant $c(c'_2) = \begin{cases} \alpha_{c'_2, c_2} & \text{si } c'_2 = + \\ -\alpha_{c'_2, c_2} & \text{si } c'_2 = - \end{cases}$. Par ailleurs, par la proposition B.2, puisque $\mathcal{P}_{c_1, c_2} \neq \emptyset$, on a $\mathcal{C}(c_2) \neq \emptyset$. On a donc l'assurance que $|\mathcal{C}(c_2)| \neq 0$.

$$\begin{aligned}& \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(c_2|X) \times \mathbb{P}(X c_1) \right) \\ &= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\left(0.5 + \frac{1}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \times \mathbb{P}(X c_1) \right) \\ &= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\left(0.5 \times \mathbb{P}(X c_1) \right) + \left(\frac{1}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \text{ de } X} c(c'_2) \times \mathbb{P}(X c_1) \right) \right) \\ &= \left(\frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(0.5 \times \mathbb{P}(X c_1) \right) \right) + \left(\frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\frac{\mathbb{P}(X c_1)}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right) \\ &= \left(\frac{0.5}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \mathbb{P}(X c_1) \right) + \left(\frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right) \\ &= \left(\frac{0.5}{\mathbb{P}(c_1)} \times \mathbb{P}(c_1) \right) + \left(\frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right) \\ &= 0.5 + \left(\frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right)\end{aligned}$$

On a défini plus haut $\mathbb{P}'(c_2, c_1) = \mathbb{P}(c_2|c_1) - 0.5$. D'où :

$$\mathbb{P}'(c_2, c_1) = \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right)$$

On développe à présent la somme sur les différentes configurations possibles des parents de c_2 en une somme sur les parents de c_2 . L'idée est de regrouper les termes selon si l'état du parent de c_2 est + ou -. On considère dans un premier temps un seul concept $c'_2 \in \mathcal{C}(c_2)$. L'ensemble des vecteurs peut en effet être partitionné

selon si $c'_2 = +$ ou si $c'_2 = -$:

$$\sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(Xc_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) = \begin{cases} \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \left(\sum_{c''_2 \text{ de } X} c(c''_2) + \alpha_{c'_2, c_2} \right) \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \left(\mathbb{P}(\bar{c}'_2 X c_1) \times \left(\sum_{c''_2 \text{ de } X} c(c''_2) - \alpha_{c'_2, c_2} \right) \right) \end{cases}$$

On peut développer cette expression en l'écrivant sous la forme de 4 sommes :

$$= \begin{cases} \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) + \mathbb{P}(c'_2 X c_1) \times \alpha_{c'_2, c_2} \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \left(\mathbb{P}(\bar{c}'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) - \mathbb{P}(\bar{c}'_2 X c_1) \times \alpha_{c'_2, c_2} \right) \end{cases}$$

$$= \begin{cases} \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) + \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \alpha_{c'_2, c_2} \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \left(\mathbb{P}(\bar{c}'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) - \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \left(\mathbb{P}(\bar{c}'_2 X c_1) \times \alpha_{c'_2, c_2} \right) \end{cases}$$

On regroupe ensuite les expressions semblables :

$$= \begin{cases} \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \alpha_{c'_2, c_2} \right) - \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \left(\mathbb{P}(\bar{c}'_2 X c_1) \times \alpha_{c'_2, c_2} \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) + \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \left(\mathbb{P}(\bar{c}'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) \end{cases}$$

$$= \begin{cases} \left(\alpha_{c'_2, c_2} \times \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \mathbb{P}(c'_2 X c_1) \right) - \left(\alpha_{c'_2, c_2} \times \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \mathbb{P}(\bar{c}'_2 X c_1) \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(c'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) + \mathbb{P}(\bar{c}'_2 X c_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) \end{cases}$$

$$= \begin{cases} \alpha_{c'_2, c_2} \times \left(\sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \mathbb{P}(c'_2 X c_1) - \sum_{X \in \vec{\mathcal{C}}_{\bar{c}'_2}(c_2)} \mathbb{P}(\bar{c}'_2 X c_1) \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\left(\mathbb{P}(c'_2 X c_1) + \mathbb{P}(\bar{c}'_2 X c_1) \right) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) \end{cases}$$

$$= \begin{cases} \alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 c_1) - \mathbb{P}(\bar{c}'_2 c_1) \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{c'_2}(c_2)} \left(\mathbb{P}(Xc_1) \times \sum_{c''_2 \text{ de } X} c(c''_2) \right) \end{cases}$$

On a donc l'expression d'un parent de c_2 additionné à une somme sur les configurations possibles des autres parents. Cette somme correspond à la somme originale à laquelle on a retiré ce parent de c_2 . La récursivité apparaît alors. On a donc :

$$\sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(Xc_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) = \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2c_1) - \mathbb{P}(\overline{c'_2}c_1) \right) \right)$$

On modifie alors l'expression de $\mathbb{P}'(c_2, c_1)$:

$$\begin{aligned} \mathbb{P}'(c_2, c_1) &= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2c_1) - \mathbb{P}(\overline{c'_2}c_1) \right) \right) \\ &= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2|c_1) \times \mathbb{P}(c_1) - \mathbb{P}(\overline{c'_2}|c_1) \times \mathbb{P}(c_1) \right) \right) \\ &= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathbb{P}(c_1) \times \left(\mathbb{P}(c'_2|c_1) - \mathbb{P}(\overline{c'_2}|c_1) \right) \right) \\ &= \frac{\mathbb{P}(c_1)}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2|c_1) - \mathbb{P}(\overline{c'_2}|c_1) \right) \right) \\ &= \frac{1}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2|c_1) - (1 - \mathbb{P}(c'_2|c_1)) \right) \right) \\ &= \frac{1}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times (2 \times \mathbb{P}(c'_2|c_1) - 1) \right) \\ &= \frac{1}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times 2 \times \left(\mathbb{P}(c'_2|c_1) - 0.5 \right) \right) \\ &= \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2|c_1) - 0.5 \right) \right) \end{aligned}$$

Or, $\mathbb{P}'(c'_2, c_1) = \mathbb{P}(c'_2|c_1) - 0.5$. D'où :

$$\mathbb{P}'(c_2, c_1) = \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathbb{P}'(c'_2, c_1) \right)$$

$\mathbb{P}'(c_2, c_1)$ est donc récursif dans le cas où $c_1 \neq c_2$ et quand c_1 est un parent non-direct de c_2 . Il faut maintenant vérifier son comportement dans le cas où c_1 est un parent direct de c_2 .

Cas où c_1 est un parent direct de c_2

Nous supposons à présent que c_1 est un parent direct c_2 , c'est-à-dire que $c_1 \in \mathcal{C}(c_2)$.

On a cette fois-ci :

$$\mathbb{P}(c_2|c_1) = \frac{\mathbb{P}(c_2c_1)}{\mathbb{P}(c_1)} = \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \mathbb{P}(c_2Xc_1)$$

La seule différence avec l'expression du cas précédent est qu'on connaît l'état d'un parent de c_2 : c_1 . On peut alors reprendre le développement de $\mathbb{P}(c_2|c_1)$ de manière

similaire au cas précédent :

$$\begin{aligned}
\mathbb{P}(c_2|c_1) &= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \mathbb{P}(c_2 X c_1) \\
&= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(c_2|X c_1) \times \mathbb{P}(X c_1) \right) \\
&= \frac{1}{\mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\left(0.5 + \frac{1}{|\mathcal{C}(c_2)|} \times \left(\alpha_{c_1, c_2} + \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right) \times \mathbb{P}(X c_1) \right)
\end{aligned}$$

Selon les hypothèses, on a $c_1 \in \mathcal{C}(c_2)$ et donc $|\mathcal{C}(c_2)| \neq 0$. En faisant un développement similaire à celui du cas précédent, on obtient :

$$\mathbb{P}(c_2|c_1) = 0.5 + \left(\frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \left(\alpha_{c_1, c_2} + \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right) \right)$$

Et, en utilisant à nouveau le fait que $\mathbb{P}'(c_2, c_1) = \mathbb{P}(c_2|c_1) - 0.5$:

$$\begin{aligned}
\mathbb{P}'(c_2, c_1) &= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \left(\alpha_{c_1, c_2} + \sum_{c'_2 \text{ de } X} c(c'_2) \right) \right) \\
&= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \alpha_{c_1, c_2} + \mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \\
&= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \left\{ \begin{array}{l} \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \alpha_{c_1, c_2} \right) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \end{array} \right\} \\
&= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \left\{ \begin{array}{l} \alpha_{c_1, c_2} \times \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \mathbb{P}(X c_1) \\ + \\ \sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) \end{array} \right\}
\end{aligned}$$

En développant l'expression $\sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right)$ de la même manière que l'expression $\sum_{X \in \vec{\mathcal{C}}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right)$ du cas précédent, on en déduit qu'on a :

$$\sum_{X \in \vec{\mathcal{C}}_{c_1}(c_2)} \left(\mathbb{P}(X c_1) \times \sum_{c'_2 \text{ de } X} c(c'_2) \right) = \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 c_1) - \mathbb{P}(\overline{c'_2 c_1}) \right) \right)$$

L'expression de $\mathbb{P}'(c_2, c_1)$ devient donc :

$$\begin{aligned}
\mathbb{P}'(c_2, c_1) &= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \left\{ \begin{array}{l} \alpha_{c_1, c_2} \times \sum_{X \in \overline{\mathcal{C}}_{c_1}(c_2)} \mathbb{P}(X c_1) \\ + \\ \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 c_1) - \mathbb{P}(\overline{c'_2} c_1) \right) \right) \end{array} \right\} \\
&= \frac{1}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \left\{ \begin{array}{l} \alpha_{c_1, c_2} \times \mathbb{P}(c_1) \\ + \\ \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 | c_1) \times \mathbb{P}(c_1) - \mathbb{P}(\overline{c'_2} | c_1) \times \mathbb{P}(c_1) \right) \right) \end{array} \right\} \\
&= \frac{\mathbb{P}(c_1)}{|\mathcal{C}(c_2)| \times \mathbb{P}(c_1)} \times \left\{ \begin{array}{l} \alpha_{c_1, c_2} \\ + \\ \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 | c_1) - \mathbb{P}(\overline{c'_2} | c_1) \right) \right) \end{array} \right\} \\
&= \frac{1}{|\mathcal{C}(c_2)|} \times \left(\alpha_{c_1, c_2} + \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(2 \times \alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 | c_1) - 0.5 \right) \right) \right) \\
&= \frac{2}{|\mathcal{C}(c_2)|} \times \left(\alpha_{c_1, c_2} \times 0.5 + \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(\alpha_{c'_2, c_2} \times \left(\mathbb{P}(c'_2 | c_1) - 0.5 \right) \right) \right)
\end{aligned}$$

Or, là encore, $\mathbb{P}'(c'_2, c_1) = \mathbb{P}(c'_2 | c_1) - 0.5$. De plus, $\mathbb{P}'(c_1, c_1) = \mathbb{P}(c_1 | c_1) - 0.5 = 1 - 0.5 = 0.5$. D'où :

$$\begin{aligned}
\mathbb{P}'(c_2, c_1) &= \frac{2}{|\mathcal{C}(c_2)|} \times \left(\alpha_{c_1, c_2} \times \mathbb{P}'(c_1, c_1) + \sum_{\substack{c'_2 \in \mathcal{C}(c_2) \\ c'_2 \neq c_1}} \left(\alpha_{c'_2, c_2} \times \mathbb{P}'(c'_2, c_1) \right) \right) \\
&= \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathbb{P}'(c'_2, c_1) \right)
\end{aligned}$$

$\mathbb{P}'(c_2, c_1)$ est donc également récursif dans le cas où $c_1 \neq c_2$ et quand c_1 est un parent direct de c_2 . De plus, son expression est équivalente au cas où c_1 est un parent non-direct de c_2 . c_1 n'est un parent (direct ou non-direct) de c_2 que s'il existe au moins un chemin de c_1 vers c_2 . On peut donc dire que si $\mathcal{P}_{c_1, c_2} \neq \emptyset$:

$$\mathbb{P}'(c_2, c_1) = \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathbb{P}'(c'_2, c_1) \right)$$

Expression récursive de $\mathbb{P}'(c_2, c_1)$

Dans le cas où $\mathcal{P}_{c_1, c_2} \neq \emptyset$, la définition de $\mathbb{P}'(c_2, c_1)$ est récursive selon les parents de c_2 . Puisque la carte est sans circuit, en appliquant cette définition, on finit forcément par atteindre un sommet sans parent. Si ce parent est c_1 , alors on a $\mathbb{P}'(c_1, c_1) = 0.5$. Si ce parent est un autre concept c , alors $\mathcal{P}_{c_1, c} = \emptyset$ et donc

$\mathbb{P}'(c, c_1) = 0$. Ces deux situations sont des cas d'arrêt : la récursivité s'arrête donc forcément. On a donc finalement :

$$\mathbb{P}'(c_2, c_1) = \begin{cases} 0.5 & \text{si } c_1 = c_2 \\ 0 & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathbb{P}'(c'_2, c_1) \right) & \text{sinon} \end{cases}$$

Équivalence des formes récursives

On rappelle qu'on a démontré :

$$\mathcal{I}_{\mathbb{P}'}(c_1, c_2) = \begin{cases} 0.5 & \text{si } c_1 = c_2 \\ 0 & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathcal{I}_{\mathbb{P}'}(c_1, c'_2) \right) & \text{sinon} \end{cases}$$

$$\mathbb{P}'(c_2, c_1) = \begin{cases} 0.5 & \text{si } c_1 = c_2 \\ 0 & \text{si } \mathcal{P}_{c_1, c_2} = \emptyset \\ \frac{2}{|\mathcal{C}(c_2)|} \times \sum_{c'_2 \in \mathcal{C}(c_2)} \left(\alpha_{c'_2, c_2} \times \mathbb{P}'(c'_2, c_1) \right) & \text{sinon} \end{cases}$$

$\mathcal{I}_{\mathbb{P}'}(c_1, c_2)$ et $\mathbb{P}'(c_2, c_1)$ sont donc trivialement équivalents. Finalement, on a :

$$\mathcal{I}_{\mathbb{P}'}(c_1, c_2) = \mathbb{P}'(c_2, c_1) = \mathbb{P}(c_2|c_1) - 0.5$$

Cette égalité est valable pour une carte cognitive et un réseau bayésien représentant cette carte construit de telle manière que le concept c_1 n'a plus de parent. Puisque le calcul d'une probabilité conditionnelle sachant c_1 dans un tel réseau est équivalent à l'application de l'opérateur $\text{do}(c_1)$ dans un réseau bayésien causal, on peut en conclure que $\mathcal{I}_{\mathbb{P}'}(c_1, c_2) = \mathbb{P}(c_2|\text{do}(c_1)) - 0.5$ pour une carte cognitive et un réseau bayésien causal la représentant. Le théorème 1 est donc démontré. \square

Table des matières

Introduction	7
1 Cartes sémantiques	13
1.1 Carte mentale	14
1.1.1 Présentation	14
1.1.2 Représentation	15
1.1.3 Édition et exemple illustratif	16
1.1.4 Discussion	18
1.2 Carte conceptuelle	20
1.2.1 Présentation	20
1.2.2 Représentation	21
1.2.3 Édition et exemple illustratif	22
1.2.4 Discussion	23
1.3 Graphe conceptuel	25
1.3.1 Présentation	26
1.3.2 Représentation	26
1.3.3 Édition et exemple illustratif	28
1.3.4 Discussion	30
1.4 Carte cognitive	31
1.4.1 Présentation	32
1.4.2 Représentation	32
1.4.3 Édition et exemple illustratif	34
1.4.4 Discussion	35
1.5 Comparaison	35
1.5.1 Tableau de comparaison	36
1.5.2 Influences	38
2 Carte cognitive	43
2.1 Définition du calcul de l'influence propagée	44
2.1.1 Chemins minimaux	45
2.1.2 Influence propagée sur un chemin	47
2.1.3 Influence propagée d'un concept sur un autre	48
2.2 Carte cognitive taxonomique	48
2.2.1 Taxonomie	48
2.2.2 Influence taxonomique	51
2.3 Vue d'une carte cognitive taxonomique	52

2.3.1	Échelle	52
2.3.2	Vue pour une échelle	54
2.3.3	Vues adaptées	55
2.4	Paramétrage	57
2.4.1	Chemins minimaux	58
2.4.2	Définitions des opérateurs d'influence pour $\{+, -\}$	61
2.4.3	Définitions des opérateurs d'influence pour $\{nul < faible < moyen < fort\}$	64
3	Validation d'une carte cognitive	67
3.1	Vérification	68
3.1.1	Propreté	69
3.1.2	Non-ambiguïté	70
3.2	Test	72
3.2.1	Spécification	72
3.2.2	Cohérence	74
3.2.3	Compatibilité	75
3.3	Paramétrage	77
3.3.1	Critères de qualité pour $[-1; 1]$	78
3.3.2	Critères de qualité pour $\{nul < faible < moyen < fort\}$	83
3.4	Conflit	84
3.4.1	Conflit de propreté	85
3.4.2	Conflit d'ambiguïté	87
3.4.3	Conflits de cohérence et de compatibilité	91
3.4.4	Construction des diagnostics	92
4	Synthèse de cartes cognitives	97
4.1	Cartes attribuées et préférences	99
4.1.1	Cartes attribuées	99
4.1.2	Préférences	101
4.2	Opérateurs de synthèse	103
4.2.1	Concepts synthétisés	104
4.2.2	Influences synthétisées	107
4.2.3	Valeurs synthétisées	110
4.3	Vue d'une carte synthétisée	111
4.3.1	Vue d'une carte synthétisée	112
4.3.2	Vue adaptée à plusieurs profils et préférences	113
4.4	Personnalisation	118
4.4.1	Synthèse des concepts	118
4.4.2	Synthèse des influences	120
4.4.3	Synthèse des valeurs	122
5	Discussion sur les valeurs	129
5.1	Interprétation des valeurs	130
5.1.1	Réseaux probabilistes	130
5.1.2	Cartes cognitives floues	138
5.2	Carte cognitive probabiliste	146

5.2.1	Sémantique des concepts et des influences	146
5.2.2	Influence propagée probabiliste	151
5.2.3	Codage d'une carte cognitive probabiliste en un réseau bayésien	155
5.3	Sémantique des concepts et des influences	164
5.3.1	Cartes cognitives floues	165
5.3.2	Cartes cognitives floues dynamiques	165
5.3.3	Qualitative Probabilistic Networks (QPN)	166
5.3.4	Cartes cognitives probabilistes	167
6	Système	169
6.1	Cartes cognitives	170
6.1.1	Cartes cognitives et taxonomie	170
6.1.2	Ensembles de valeurs	172
6.2	Calcul d'influences	175
6.2.1	Algorithmes	175
6.2.2	Chemins minimaux	176
6.2.3	Algorithmes d'influences	177
6.2.4	Influence propagée probabiliste	180
6.3	Validation	181
6.3.1	Approche de l'implémentation	181
6.3.2	Critères de qualité	182
6.3.3	Gestion des conflits	184
6.4	Synthèse	184
6.4.1	Préférences et cartes attribuées	184
6.4.2	Synthèse de concepts, d'influences et de valeurs	185
6.4.3	Synthèse de cartes cognitives	189
6.5	Entrées/Sorties et interface graphique	190
6.5.1	Entrées/Sorties	190
6.5.2	Interface graphique	195
6.6	Cas d'utilisation	196
6.6.1	Construction des cartes cognitives	196
6.6.2	Calcul d'influence	199
6.6.3	Validation d'une carte cognitive	202
6.6.4	Synthèse de cartes cognitives	203
	Conclusion	207
	Annexes	211
A	Validation de cartes du monde de la pêche	211
B	Preuve du théorème 1	216
	Table des matières	227
	Table des figures	231
	Bibliographie	235

Publications personnelles

247

Table des figures

1.1	Carte mentale décrivant Internet.	18
1.2	Carte conceptuelle décrivant Internet.	22
1.3	Ordonnancement pour une carte conceptuelle.	23
1.4	Confusion sur l'arité de la relation.	23
1.5	Confusion entre relation binaire et n-aire.	24
1.6	Un même concept apparaît deux fois.	24
1.7	Confusion entre les relations.	25
1.8	Réification des relations.	25
1.9	Graphe conceptuel représentant diverses informations sur Internet. .	29
1.10	Ensemble des types de concepts et de types de relations du graphe conceptuel de la figure 1.9.	29
1.11	Signatures des types de relations.	30
1.12	Marqueurs individuels.	30
1.13	Graphe-requête pour le graphe conceptuel de la figure 1.9.	31
1.14	Graphe-image issu du graphe-requête de la figure 1.13 pour le graphe conceptuel de la figure 1.9.	31
1.15	Carte cognitive représentant l'influence du filtrage d'Internet sur le régime politique en place.	34
1.16	Comparaison des modèles de cartes sémantiques.	36
1.17	Carte mentale représentant des relations d'influence.	39
1.18	Carte conceptuelle représentant des relations d'influence sous forme ternaire.	39
1.19	Autre carte conceptuelle représentant des relations d'influence sous forme binaire.	40
1.20	Extrait du support d'un graphe conceptuel figurant des relations d'influence ternaires.	40
1.21	Graphe conceptuel représentant des relations d'influence ternaires. .	40
1.22	Extrait du support d'un graphe conceptuel figurant des relations d'influence binaires.	41
1.23	Graphe conceptuel représentant des relations d'influence binaires. .	41
2.1	La carte cognitive $CM1$ définie sur $[-1; 1]$	45
2.2	La taxonomie de concepts $T1$	49
2.3	Concepts élémentaires de $T1$	51
2.4	Vue de TMI pour l'échelle C_3	55
2.5	Vue de TMI pour le profil P_m	56

2.6	Vue de $TM1$ pour les profils P_m et P_r	57
2.7	Influences reliant un concept A à un concept F	58
2.8	La carte cognitive $CM2$ définie sur $\{+, -\}$	62
2.9	La carte cognitive $CM3$ définie sur $\{nul < faible < moyen < fort\}$	64
3.1	Couple de concepts non-propre.	69
3.2	La carte cognitive $CM4$, non-propre et ambiguë.	70
3.3	Couple de concepts ambigu.	71
3.4	Rappel de la taxonomie $T1$ (exemple 2.6).	73
3.5	Rappel de la carte cognitive $CM2$ (exemple 2.25).	74
3.6	La carte cognitive $CM5$	80
3.7	Rappel de la carte cognitive $CM1$ (exemple 2.1).	82
4.1	La carte cognitive attribuée CA_P construite par le designer <i>Préfet</i> définie sur l'ensemble de valeurs $[-1; 1]$	100
4.2	La taxonomie $T2$	100
4.3	L'ensemble univoque de cartes attribuées U	101
4.4	L'ensemble des concepts synthétisés de U selon \succeq	106
4.5	Carte synthétisée non-valorisée construite à partir de U selon \succeq	109
4.6	Carte synthétisée valorisée construite à partir de U selon \succeq	112
4.7	La taxonomie synthétisée $T2b$	113
4.8	Vue de la carte synthétisée.	114
4.9	Influences présentes dans au moins une des cartes attribuées.	120
4.10	Influences présentes dans les cartes attribuées des designers dont l'indice dans l'ordre de préférence est plus grand que ou égal à 2.	121
4.11	Influences présentes dans au moins 2 cartes attribuées différentes.	122
4.12	L'ensemble univoque de cartes attribuées U' défini sur $\{+, -\}$	125
4.13	Carte synthétisée valorisée définie sur $\{+, -, ?\}$ construite à partir de U' selon \succeq	126
4.14	Carte synthétisée valorisée figurant les intervalles des valeurs des différents designers construite à partir de U selon \succeq	127
5.1	Une carte cognitive floue.	139
5.2	Modification de la carte cognitive floue de la figure 5.1 pour le calcul des influences.	141
5.3	Une carte cognitive simple.	147
5.4	Une carte cognitive définie sur l'ensemble de valeurs $[-1; 1]$	151
5.5	Illustration du calcul des parts de deux différents chemins de A vers B	153
5.6	Une carte cognitive figurant un concept avec deux parents.	157
5.7	Une carte cognitive figurant un concept avec plusieurs parents.	159
6.1	Organisation générale des packages.	170
6.2	Diagramme de classes du package <code>model</code>	171
6.3	Diagramme des principales classes du package <code>model.valuesets</code>	172
6.4	Diagramme de classes des ensembles de valeurs liés à $\{+, -\}$	174
6.5	Diagramme de classes des ensembles de valeurs numériques.	175

6.6	Diagramme de classes des interfaces d'algorithmes.	176
6.7	Diagramme de classes des chemins minimaux.	177
6.8	Diagramme de classes des algorithmes de calcul d'influence.	178
6.9	Vérification de la compatibilité des algorithmes de calcul d'influence.	179
6.10	Diagramme de classes du package <code>model.influences.bayesian</code>	180
6.11	Diagramme de classes liées à <code>ContradictoryValuesChecker</code>	182
6.12	Diagramme de classes des critères de qualité.	183
6.13	Diagramme de classes des principales classes liées à la synthèse de cartes cognitives.	185
6.14	Schéma d'organisation de la classe <code>ExtractedInfluences</code>	187
6.15	Diagramme de classes des algorithmes de synthèse.	187
6.16	Algorithme pour construire une carte cognitive synthétisée.	189
6.17	Diagramme de classes des descriptions d'algorithmes.	191
6.18	Déclaration de l'ensemble de valeurs $\{+, -\}$	192
6.19	Déclaration de l'ensemble de valeurs $[-1; 1]$	192
6.20	Déclaration de l'ensemble de valeurs $\{nul < faible < moyen < fort\}$	192
6.21	Déclaration d'une taxonomie de concepts.	193
6.22	Déclaration de cartes cognitives.	194
6.23	Organisation du package <code>io</code> et ses liens avec le package <code>model</code>	195
6.24	Choix de l'ensemble de valeurs.	196
6.25	Fenêtre principale de l'application.	197
6.26	Ajout d'un nouveau concept dans la taxonomie.	198
6.27	Ajout d'un lien d'héritage entre deux concepts de la taxonomie.	199
6.28	Ajout d'un nouveau concept dans la carte cognitive.	199
6.29	Ajout d'une nouvelle influence dans la carte cognitive.	200
6.30	Ajout d'un nouveau designer et d'une nouvelle carte attribuée dans l'ensemble univoque.	200
6.31	Un ensemble univoque de cartes attribuées.	201
6.32	Calcul de l'influence propagée d'un concept sur un autre.	201
6.33	Calcul de l'influence propagée taxonomique.	202
6.34	Validation d'une carte cognitive.	203
6.35	Création d'une contrainte de la spécification.	203
6.36	Configuration de la synthèse des cartes cognitives.	204
6.37	Affichage de la carte synthétisée.	204
6.38	Meilleur affichage de la carte synthétisée.	204
A.1	Carte cognitive d'un pêcheur.	212
A.2	Influences de <i>effort de pêche</i> sur <i>quantité pêchée</i>	213
A.3	Carte cognitive d'un autre pêcheur.	214
A.4	Influences de <i>CSJ ressource</i> sur <i>CSJ qté pêchée</i>	215

Bibliographie

- [Aiello, 1990] LUGIA C. AIELLO, éditeur (1990). *Proceedings of the 9th European Conference on Artificial Intelligence (ECAI 1990), August 8-10, 1990, Stockholm, Sweden*. Pitman Publishing. 235
- [Axelrod, 1976] ROBERT M. AXELROD (1976). *Structure of decision : the cognitive maps of political elites*. Princeton, N.J. 7, 13, 32, 33, 47, 61
- [Ayel et Laurent, 1991] MARC AYEL et JEAN-PIERRE LAURENT (1991). *Validation, verification and test of knowledge-based systems*. John Wiley & Sons, Inc., New York, NY, USA. 67
- [Baader et al., 2003] FRANZ BAADER, DIEGO CALVANESE, DEBORAH L. MCGUINNESS, DANIELE NARDI et PETER F. PATEL-SCHNEIDER, éditeurs (2003). *The Description Logic Handbook : Theory, Implementation, and Applications*. Cambridge University Press. 7
- [Baget, 2007] JEAN-FRANÇOIS BAGET (2007). A Datatype Extension for Simple Conceptual Graphs and Conceptual Graphs Rules. *In* [Priss et al., 2007], pages 83–96. 40
- [Baget et al., 2009] JEAN-FRANÇOIS BAGET, MICHEL LECLÈRE, MARIE-LAURE MUGNIER et ERIC SALVAT (2009). Extending Decidable Cases for Rules with Existential Variables. *In* [Boutillier, 2009], pages 677–682. 31
- [Bajcsy, 1993] RUZENA BAJCSY, éditeur (1993). *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), August 28 - September 3, 1993, Chambéry, France*. Morgan Kaufmann. 238
- [Bandler et Kohout, 1980] WYLLIS BANDLER et LADISLAV L. KOHOUT (1980). Fuzzy power sets and fuzzy implication operators. *Fuzzy Sets and Systems*, 4:13–30. 140
- [Beauvieux et Dague, 1990] ALAIN BEAUVIEUX et PHILIPPE DAGUE (1990). A General Consistency (Checking and Restoring) Engine for Knowledge Bases. *In* [Aiello, 1990], pages 77–82. 11
- [Beel et Gipp, 2010] JÖRAN BEEL et BELA GIPP (2010). Link Analysis in Mind Maps : A New Approach To Determine Document Relatedness. *In* [ICUIMC, 2010]. 20

- [Ben Messaoud, 2012] MONTASSAR BEN MESSAOUD (2012). *SemCaDo : une approche pour la découverte de connaissances fortuites et l'évolution ontologique*. Thèse de doctorat, LARODEC – Institut Supérieur de Gestion de Tunis, Tunisie. 164
- [Ben Messaoud *et al.*, 2009] MONTASSAR BEN MESSAOUD, PHILIPPE LERAY et NAHLA BEN AMOR (2009). Integrating Ontological Knowledge for Iterative Causal Discovery and Visualization. *In* [Sossai et Chemello, 2009], pages 168–179. 164
- [Ben Yaghlane *et al.*, 2003] BOUTHEINA BEN YAGHLANE, PHILIPPE SMETS et KHALED MELLOULI (2003). Directed Evidential Networks with Conditional Belief Functions. *In* [Nielsen et Zhang, 2003], pages 291–305. 168
- [Benferhat *et al.*, 1999] SALEM BENFERHAT, DIDIER DUBOIS, LAURENT GARCIA et HENRI PRADE (1999). Possibilistic logic bases and possibilistic graphs. *In* [Laskey et Prade, 1999], pages 57–64. 168
- [Benferhat *et al.*, 2002] SALEM BENFERHAT, DIDIER DUBOIS, SOUHILA KACI et HENRI PRADE (2002). Bipolar Possibilistic Representations. *In* [Darwiche et Friedman, 2002], pages 45–52. 127
- [Benferhat *et al.*, 2001] SALEM BENFERHAT, DIDIER DUBOIS et HENRI PRADE (2001). Towards a Possibilistic Logic Handling of Preferences. *Applied Intelligence*, 14(3):303–317. 127
- [Bouali *et al.*, 1997] FATMA BOUALI, STÉPHANE LOISEAU et MARIE-CHRISTINE ROUSSET (1997). Revision of Rule Bases. *In* [Vanthienen et van Harmelen, 1997], pages 193–203. 68
- [Boukhris *et al.*, 2011] IMEN BOUKHRIS, ZIED ELOUEDI et SALEM BENFERHAT (2011). Modeling Interventions Using Belief Causal Networks. *In* [Murray et McCarthy, 2011]. 168
- [Boutilier, 2009] CRAIG BOUTILIER, éditeur (2009). *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09), July 11-17, 2009, Pasadena, California, USA*. 235
- [Boutilier *et al.*, 2004] CRAIG BOUTILIER, RONEN I. BRAFMAN, CARMEL DOMSHLAK, HOLGER H. HOOS et DAVID POOLE (2004). CP-nets : A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research (JAIR)*, 21:135–191. 127
- [Boutilier *et al.*, 1999] CRAIG BOUTILIER, RONEN I. BRAFMAN, HOLGER H. HOOS et DAVID POOLE (1999). Reasoning With Conditional Ceteris Paribus Preference Statements. *In* [Laskey et Prade, 1999], pages 71–80. 127

- [Brewka *et al.*, 2006] GERHARD BREWKA, SILVIA CORADESCHI, ANNA PERINI et PAOLO TRAVERSO, éditeurs (2006). *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006), August 29 - September 1, 2006, Riva del Garda, Italy*, volume 141 de *Frontiers in Artificial Intelligence and Applications*. IOS Press. 245
- [Buzan, 2003] TONY BUZAN (2003). *Use your head*. BBC Active. 14
- [Buzan et Buzan, 2003] TONY BUZAN et BARRY BUZAN (2003). *The Mind Map Book*. BBC Active. 7, 13, 14
- [Cañas et Novak, 2009] ALBERTO J. CAÑAS et JOSEPH DONALD NOVAK (2009). What is a concept map? Rapport technique, Florida Institute for Human and Machine Cognition. 25
- [Celik *et al.*, 2005] FILIZ D. CELIK, UYGAR OZESMI et ASUMAN AKDOGAN (2005). Participatory Ecosystem Management Planning at Tuzla Lake (Turkey) Using Fuzzy Cognitive Mapping. *eprint arXiv :q-bio/0510015*. 8, 32
- [Charniak et McDermott, 1985] EUGENE CHARNIAK et DREW MCDERMOTT (1985). *Introduction to Artificial Intelligence*. Addison-Wesley, Reading MA. 133
- [Chauvin, 2009] LIONEL CHAUVIN (2009). *Modèles de cartes cognitives étendues aux notions de contexte et d'échelle*. Thèse de doctorat, LERIA, Université d'Angers, France. 10, 43, 44, 52
- [Chauvin *et al.*, 2011] LIONEL CHAUVIN, DAVID GENEST, AYMERIC LE DORZE et STÉPHANE LOISEAU (2011). Cartes cognitives : une exploitation à base d'échelle, vue et profil. In [Khenchaf et Poncelet, 2011], pages 503–514. 35, 48
- [Chauvin *et al.*, 2008] LIONEL CHAUVIN, DAVID GENEST et STÉPHANE LOISEAU (2008). Les cartes cognitives hiérarchiques. In [Guillet et Trousse, 2008], pages 91–102. 63
- [Chauvin *et al.*, 2009] LIONEL CHAUVIN, DAVID GENEST et STÉPHANE LOISEAU (2009). Ontological Cognitive Map. *International Journal on Artificial Intelligence Tools*, 18(5):697–716. 43
- [Cheah *et al.*, 2007] WOUI PING CHEAH, KYOUNG-YUN KIM, HYUNG-JEONG YANG, SOOK-YOUNG CHOI et HYUNG-JAE LEE (2007). A manufacturing-environmental model using Bayesian belief networks for assembly design decision support. In [Okuno et Ali, 2007], pages 374–383. 147, 148, 158
- [Chein et Mugnier, 1992] MICHEL CHEIN et MARIE-LAURE MUGNIER (1992). Conceptual Graphs : Fundamental Notions. *Revue d'Intelligence Artificielle (RIA)*, 6(4):365–406. 26, 28

- [Chickering, 1996] DAVID MAXWELL CHICKERING (1996). Learning Bayesian Networks is NP-Complete. *In Learning from Data : Artificial Intelligence and Statistics V*, pages 121–130. Springer-Verlag. 131
- [Christensen, 1997] RONALD CHRISTENSEN (1997). *Log-Linear Models and Logistic Regression*. Springer. 168
- [Christiansen, 2011] GWEN CHRISTIANSEN (2011). Modélisation du savoir portée par les acteurs d'un système : application aux pêcheurs à la coquille en rade de Brest. Mémoire de Master, Agrocampus Ouest, Saint-Brieuc, France. 58, 95, 98, 169, 175, 207, 211
- [Console *et al.*, 1993] LUCA CONSOLE, GERHARD FRIEDRICH et DANIELE THESEIDER DUPRÉ (1993). Model-Based Diagnosis Meets Error Diagnosis in Logic Programs. *In* [Bajcsy, 1993], pages 1494–1501. 11
- [Coste-Marquis *et al.*, 2004] SYLVIE COSTE-MARQUIS, JÉRÔME LANG, PAOLO LIBERATORE et PIERRE MARQUIS (2004). Expressive Power and Succinctness of Propositional Languages for Preference Representation. *In* [Dubois *et al.*, 2004], pages 203–212. 127
- [Darwiche et Friedman, 2002] ADNAN DARWICHE et NIR FRIEDMAN, éditeurs (2002). *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI'02) August 1-4, 2002, Edmonton, Alberta, Canada*. Morgan Kaufmann. 236, 243
- [Das, 2004] BALARAM DAS (2004). Generating Conditional Probabilities for Bayesian Networks : Easing the Knowledge Acquisition Problem. *CoRR*, cs.AI/0411034. 158
- [de Kleer et Williams, 1987] JOHAN DE KLEER et BRIAN C. WILLIAMS (1987). Diagnosing Multiple Faults. *Artificial Intelligence*, 32(1):97–130. 95
- [Dickerson et Bart, 1994] JULIE A. DICKERSON et KOSKO BART (1994). Virtual Worlds as Fuzzy Cognitive Maps. *Presence*, 3(2):73–89. 32, 144
- [Domshlak *et al.*, 2003] CARMEL DOMSHLAK, FRANCESCA ROSSI, KRISTEN BRENT VENABLE et TOBY WALSH (2003). Reasoning about soft constraints and conditional preferences : complexity results and approximation techniques. *In* [Gottlob et Walsh, 2003], pages 215–220. 127
- [Doyle et Ford, 1999] JAMES K. DOYLE et DAVID N. FORD (1999). Mental models concepts revisited : some clarifications and a reply to Lane. *System Dynamics Review*, 15(4):411–415. 32
- [Druzdzel et Henrion, 1993] MAREK J. DRUZDZEL et MAX HENRION (1993). Efficient reasoning in qualitative probabilistic networks. *In* [Fikes et Lehnert, 1993], pages 548–553. 137
- [Dubois et Prade, 1988] DIDIER DUBOIS et HENRI PRADE (1988). *Possibility theory*. Plenum Press, New-York. 35, 168

- [Dubois et Prade, 2001] DIDIER DUBOIS et HENRI PRADE (2001). Possibility Theory, Probability Theory and Multiple-Valued Logics : A Clarification. *Annals of Mathematics and Artificial Intelligence*, 32(1-4):35–66. 168
- [Dubois *et al.*, 2004] DIDIER DUBOIS, CHRISTOPHER A. WELTY et MARY-ANNE WILLIAMS, éditeurs (2004). *Proceedings of the 9th International Conference on Principles of Knowledge Representation and Reasoning (KR2004), June 2-5, 2004, Whistler, Canada*. AAAI Press. 238
- [Dung, 1995] PHAN MINH DUNG (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357. 42
- [Dupin de Saint-Cyr, 1996] FLORENCE DUPIN DE SAINT-CYR (1996). *Gestion de l'évolutif et de l'incertain en logiques pondérées*. Thèse de doctorat, IRIT – Université Paul Sabatier, Toulouse, France. 111
- [Farrand *et al.*, 2002] PAUL FARRAND, FEARZANA HUSSAIN et ENID HENNESSY (2002). The efficacy of the ‘mind map’ study technique. *Medical Education*, 36:426–431. 15
- [Fikes et Lehnert, 1993] RICHARD FIKES et WENDY G. LEHNERT, éditeurs (1993). *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI 1993), July 11-15, 1993, Washington, DC, USA*. AAAI Press / The MIT Press. 238
- [FUZZ-IEEE, 1992] FUZZ-IEEE (1992). *Proceedings of the 1st IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 1992), March 8-12, 1992, San Diego, California, USA*. IEEE Xplore. 240
- [FUZZ-IEEE, 2006] FUZZ-IEEE (2006). *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2006), July 16-21, 2006, Vancouver, British Columbia, Canada*. IEEE Xplore. 244
- [Genest et Loiseau, 2007] DAVID GENEST et STÉPHANE LOISEAU (2007). Modélisation, classification et propagation dans des réseaux d'influence. *Technique et Science Informatiques*, 26(3-4):471–496. 65
- [Goldszmidt et Pearl, 1992] MOISÉS GOLDSZMIDT et JUDEA PEARL (1992). Rank-based Systems : A Simple Approach to Belief Revision, Belief Update, and Reasoning about Evidence and Actions. In [Nebel *et al.*, 1992], pages 661–672. 134
- [Gottlob et Walsh, 2003] GEORG GOTTLÖB et TOBY WALSH, éditeurs (2003). *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03), August 9-15, 2003, Acapulco, Mexico*. Morgan Kaufmann. 238
- [Grabisch, 1995] MICHEL GRABISCH (1995). Fuzzy integral in multicriteria decision making. *Fuzzy Sets and Systems*, 69:279–298. 126

- [Grabisch, 1996] MICHEL GRABISCH (1996). The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Research*, 89:445–456. 126
- [Greiner *et al.*, 1989] RUSSELL GREINER, BARBARA A. SMITH et RALPH W. WILKERSON (1989). A Correction to the Algorithm in Reiter’s Theory of Diagnosis. *Artificial Intelligence*, 41(1):79–88. 95
- [Guillet et Trousse, 2008] FABRICE GUILLET et BRIGITTE TROUSSE, éditeurs (2008). *Actes des 8èmes journées Extraction et Gestion des Connaissances (EGC’2008), 29 janvier - 1er février, 2008, Sophia-Antipolis, France*, volume RNTI-E-11 de *Revue des Nouvelles Technologies de l’Information*. Cépaduès-Éditions. 237
- [Gérard *et al.*, 2007] ROMAIN GÉRARD, SOUHILA KACI et HENRI PRADE (2007). Ranking Alternatives on the Basis of Generic Constraints and Examples - A Possibilistic Approach. In [Veloso, 2007], pages 393–398. 127
- [Hagiwara, 1992] MASAFUMI HAGIWARA (1992). Extended Fuzzy Cognitive Maps. In [FUZZ-IEEE, 1992], pages 795–801. 146
- [Hansson, 1996] SVEN OVE HANSSON (1996). What is Ceteris Paribus Preference? *Journal of Philosophical Logic*, 25(3):307–332. 127
- [ICTAI, 2012] ICTAI (2012). *Proceedings of the 24th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2012), November 7-9, 2012, Athens, Greece*. IEEE. 243
- [ICUIMC, 2010] ICUIMC (2010). *Proceedings of the 4th International Conference on Ubiquitous Information Management and Communication (ICUIMC 2010), January 14-15, 2010, Suwon, Korea*. ACM. 235
- [ISO/IEC JTC1/SC34, 2007] ISO/IEC JTC1/SC34 (2007). The Topic Maps reference model. <http://www.isotopicmaps.org/tmrm/>. 42
- [ISO/IEC JTC1/SC34, 2008] ISO/IEC JTC1/SC34 (2008). Topic Maps - Data model. <http://www.isotopicmaps.org/sam/sam-model/>. 42
- [Kaci, 2009] SOUHILA KACI (2009). Working with Preferences : less is more. Habilitation à diriger des recherches, CRIS – IUT de Lens, Université d’Artois. 11, 97, 101
- [Karp, 1972] RICHARD M. KARP (1972). Reducibility among combinatorial problems. In R. E. MILLER et J. W. THATCHER, éditeurs : *Complexity of Computer Computations*, pages 85–103. Plenum Press. 95
- [Khenchaf et Poncelet, 2011] ALI KHENCHAF et PASCAL PONCELET, éditeurs (2011). *Actes des 11èmes journées Extraction et Gestion des Connaissances (EGC’2011), 25-29 janvier, 2011, Brest, France*, volume RNTI-E-20 de *Revue des Nouvelles Technologies de l’Information*. Hermann-Éditions. 237

- [Kjaerulff et Madsen, 2008] UFFE B. KJAERULFF et ANDERS L. MADSEN (2008). *Bayesian Networks and Influence Diagrams : A Guide to Construction and Analysis*. Information Science and Statistics. Springer-Verlag New York. 32
- [Klement *et al.*, 2000] ERICH PETER KLEMENT, RADKO MESIAR et ENDRE PAP (2000). *Triangular Norms*. Trends in logic, Studia logica library. Springer. 142
- [Kosko, 1986] BART KOSKO (1986). Fuzzy cognitive maps. *International Journal of Man-Machines Studies*, 24:65–75. 32, 35, 64, 129, 138, 149
- [Laskey et Prade, 1999] KATHRYN B. LASKEY et HENRI PRADE, éditeurs (1999). *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'99), July 30 - August 1, 1999, Stockholm, Sweden*. Morgan Kaufmann. 236, 243
- [Lehmann, 1992] FRITZ LEHMANN (1992). *Semantic Networks in Artificial Intelligence*. Pergamon Press. 7, 25, 26
- [Lemmer et Gossink, 2004] JOHN F. LEMMER et DON E. GOSSINK (2004). Recursive noisy or - a rule for estimating complex probabilistic interactions. *Transactions on Systems, Man, Cybernetics Part B*, 34(6):2252–2261. 158
- [Levi et Tetlock, 1980] ARIEL LEVI et PHILIP E. TETLOCK (1980). A Cognitive Analysis of Japan's 1941 Decision for War. *The Journal of Conflict Resolution*, 24(2):195–211. 8, 32
- [Liskov et Wing, 1999] BARBARA H. LISKOV et JEANNETTE M. WING (1999). Behavioral Subtyping Using Invariants and Constraints. Rapport technique, School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213, USA. 105, 119
- [Liu et Zhang, 2003] ZHI-QIANG LIU et JIAN YING ZHANG (2003). Interrogating the structure of fuzzy cognitive maps. *Soft Computing*, 7(3):148–153. 129, 144
- [Loiseau, 1997] STÉPHANE LOISEAU (1997). Checking and restoring the consistency of knowledge database. *In Encyclopedia of Computer Science and Technology*, volume 36, pages 15–33. Marcel Dekker, Inc. 11
- [Louçã, 2000] JORGE A. LOUÇÃ (2000). *Cartographie Cognitive, Réflexion Stratégique et Interaction Distribuée : une Approche Multi-Agent*. Thèse de doctorat, Université Paris IX Dauphine, France. 11
- [Mugnier et Chein, 1996] MARIE-LAURE MUGNIER et MICHEL CHEIN (1996). Représenter des connaissances et raisonner avec des graphes. *Revue d'Intelligence Artificielle (RIA)*, 10(1):7–56. 26, 27

- [Murray et McCarthy, 2011] R. CHARLES MURRAY et PHILIP M. MCCARTHY, éditeurs (2011). *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2011), May 18-20, 2011, Palm Beach, Florida, USA*. AAAI Press. 236
- [Nadkarni et Shenoy, 2001] SUCHETA NADKARNI et PRAKASH P. SHENOY (2001). A Bayesian network approach to making inferences in causal maps. *European Journal of Operational Research*, 128(3):479–498. 161
- [Nadkarni et Shenoy, 2004] SUCHETA NADKARNI et PRAKASH P. SHENOY (2004). A causal mapping approach to constructing Bayesian networks. *Decision Support Systems*, 38(2):259–281. 134, 161
- [Naïm *et al.*, 2004] PATRICK NAÏM, PIERRE-HENRI WUILLEMIN, PHILIPPE LERAY, OLIVIER POURRET et ANNA BECKER (2004). *Réseaux bayésiens*. Eyrolles, Paris. 131
- [Nebel *et al.*, 1992] BERNHARD NEBEL, CHARLES RICH et WILLIAM R. SWARTOUT, éditeurs (1992). *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92). October 25-29, 1992, Cambridge, MA, USA*. Morgan Kaufmann. 239
- [Nielsen et Zhang, 2003] THOMAS D. NIELSEN et NEVIN LIANWEN ZHANG, éditeurs (2003). *Proceedings of the 7th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2003) July 2-5, 2003, Aalborg, Denmark*, volume 2711 de *Lecture Notes in Computer Science*. Springer. 236, 243
- [Novak et Cañas, 2006] JOSEPH DONALD NOVAK et ALBERTO J. CAÑAS (2006). The Theory Underlying Concept Maps and How to Construct Them, Technical Report IHMC CmapTools. Rapport technique, Florida Institute for Human and Machine Cognition. 7, 13, 20
- [Okuno et Ali, 2007] HIROSHI G. OKUNO et MOONIS ALI, éditeurs (2007). *New Trends in Applied Artificial Intelligence, Proceedings of the 20th international conference on Industrial, engineering, and other applications of applied intelligent systems (IEA/AIE 2007), June 26-29, 2007, Kyoto, Japan*, volume 4570 de *Lecture Notes in Computer Science*. Springer. 237
- [Pearl, 1988] JUDEA PEARL (1988). *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 7, 129, 130, 134
- [Pearl, 1993] JUDEA PEARL (1993). [Bayesian Analysis in Expert Systems] : Comment : Graphical Models, Causality and Intervention. *Statistical Science*, 8(3):266–269. 134
- [Pearl, 2000] JUDEA PEARL (2000). *Causality : models, reasoning, and inference*. Cambridge University Press, New York, NY, USA. 133

- [Pearl, 2005] JUDEA PEARL (2005). Influence Diagrams - Historical and Personal Perspectives. *Decision Analysis*, 2(4):232–234. 32
- [Pearl, 2009] JUDEA PEARL (2009). *Causality : Models, Reasoning and Inference*. Cambridge University Press, New York, NY, USA, 2nd édition. 7, 129, 130, 133
- [Poignonec, 2006] DENIS POIGNONEC (2006). *Apport de la combinaison cartographie cognitive/ontologie dans la compréhension de la perception du fonctionnement d'un écosystème récifo-lagonaire de Nouvelle-Calédonie par les acteurs locaux*. Thèse de doctorat, ENSA Rennes, France. 8, 11, 32, 98, 207
- [Priss et al., 2007] UTA PRISS, SIMON POLOVINA et RICHARD HILL, éditeurs (2007). *Conceptual Structures : Knowledge Architectures for Smart Applications, Proceedings of the 15th International Conference on Conceptual Structures (ICCS 2007), July 22-27, 2007, Sheffield, UK*, volume 4604 de *Lecture Notes in Computer Science*. Springer. 235
- [Reiter, 1987] RAYMOND REITER (1987). A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32(1):57–95. 11, 68, 84, 92, 95
- [Renooij et al., 2003] SILJA RENOUIJ, SIMON PARSONS et PAULINE PARDIECK (2003). Using Kappas as Indicators of Strength in Qualitative Probabilistic Networks. In [Nielsen et Zhang, 2003], pages 87–99. 168
- [Renooij et van der Gaag, 1999] SILJA RENOUIJ et LINDA C. VAN DER GAAG (1999). Enhancing QPNs for Trade-off Resolution. In [Laskey et Prade, 1999], pages 559–566. 168
- [Renooij et van der Gaag, 2002] SILJA RENOUIJ et LINDA C. VAN DER GAAG (2002). From Qualitative to Quantitative Probabilistic Networks. In [Darwiche et Friedman, 2002], pages 422–429. 168
- [Rudolph et al., 2009] SEBASTIAN RUDOLPH, FRITHJOF DAU et SERGEI O. KUZNETSOV, éditeurs (2009). *Conceptual Structures : Leveraging Semantic Technologies, Proceedings of the 17th International Conference on Conceptual Structures (ICCS 2009), July 26-31, 2009, Moscow, Russia*, volume 5662 de *Lecture Notes in Computer Science*. Springer. 40
- [Satur et Liu, 1999] RICHARD SATUR et ZHI-QIANG LIU (1999). A Contextual Fuzzy Cognitive Map Framework for Geographic Information Systems. *IEEE Transactions on Fuzzy Systems*, 7(5):481–494. 32, 146
- [Sedki et Bonneau de Beaufort, 2012a] KARIMA SEDKI et LOUIS BONNEAU DE BEAUFORT (2012a). Cognitive Maps and Bayesian Networks for Knowledge Representation and Reasoning. In [ICTAI, 2012], pages 1035–1040. 149, 158
- [Sedki et Bonneau de Beaufort, 2012b] KARIMA SEDKI et LOUIS BONNEAU DE BEAUFORT (2012b). D'une carte causale à un réseau bayésien causal. *Actes des Journées francophones sur les réseaux bayésiens*. 158

- [Shachter, 1986] ROSS D. SHACHTER (1986). Evaluating Influence Diagrams. *Operations Research*, 33(6). 137
- [Shachter *et al.*, 1990] ROSS D. SHACHTER, TOD S. LEVITT, LAVEEN N. KANAL et JOHN F. LEMMER, éditeurs (1990). *Proceedings of the 4th Annual Conference on Uncertainty in Artificial Intelligence (UAI'88), July 10-12, 1988, Minneapolis, MN, USA*. North-Holland. 244, 245
- [Shafer, 1976] GLENN SHAFER (1976). *A Mathematical Theory of Evidence*. Princeton University Press. 168
- [Song *et al.*, 2006] HENG-JIE SONG, ZHI-QI SHEN, CHUN-YAN MIAO, ZHI-QIANG LIU et YUAN MIAO (2006). Probabilistic Fuzzy Cognitive Map. In [FUZZ-IEEE, 2006], pages 1221–1228. 146
- [Sossai et Chemello, 2009] CLAUDIO SOSSAI et GAETANO CHEMELLO, éditeurs (2009). *Proceedings of the 10th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2009) July 1-3, 2009, Verona, Italy*, volume 5590 de *Lecture Notes in Computer Science*. Springer. 236
- [Sowa, 1976] JOHN F. SOWA (1976). Conceptual Graphs for a Data Base Interface. *IBM Journal of Research and Development*, 20(4):336–357. 7, 13, 26
- [Sowa, 1984] JOHN F. SOWA (1984). *Conceptual Structures : Information Processing in Mind and Machine*. Addison Wesley. 7, 13, 26
- [Spirtes *et al.*, 2001] PETER SPIRITES, CLARK GLYMOUR et RICHARD SCHEINES (2001). *Causation, prediction, and search*, volume 1. MIT Press. 134
- [Spohn, 1988] WOLFGANG SPOHN (1988). A general non-probabilistic theory of inductive reasoning. In [Shachter *et al.*, 1990], pages 149–158. 168
- [Tolman, 1948] EDWARD C. TOLMAN (1948). Cognitive maps in rats and men. *The Psychological Review*, 55(4):189–208. 8, 32
- [Tricot, 2006] CHRISTOPHE TRICOT (2006). *La cartographie sémantique : des connaissances à la carte*. Thèse de doctorat, Université de Savoie, France. 7
- [van Gelder, 2013] TIM VAN GELDER (2013). Argument mapping. In HAL PASHLER, éditeur : *The Encyclopedia of the Mind*. Sage. 42
- [Vanthienen et van Harmelen, 1997] JAN VANTHIENEN et FRANK VAN HARMELLEN, éditeurs (1997). *Proceedings of the 4th European Symposium on the Validation and Verification of Knowledge-Based Systems (EUROVAV'97), June 26-28, 1997, Leuven, Belgium*. Katholieke Universiteit Leuven, Belgium. 236

- [Veloso, 2007] MANUELA M. VELOSO, éditeur (2007). *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, January 6-12, 2007, Hyderabad, India. 240
- [Verma et Pearl, 1988] THOMAS VERMA et JUDEA PEARL (1988). Causal networks : semantics and expressiveness. In [Shachter *et al.*, 1990], pages 69–78. 131
- [W3C, 2004a] W3C (2004a). Resource Description Framework (RDF). <http://www.w3.org/RDF/>. 42
- [W3C, 2004b] W3C (2004b). Web Ontology Language (OWL). <http://www.w3.org/2004/OWL/>. 42
- [W3C, 2008] W3C (2008). eXtensible Markup Language (XML). <http://www.w3.org/XML/>. 42, 192
- [Wellman, 1990] MICHAEL P. WELLMAN (1990). Fundamental Concepts of Qualitative Probabilistic Networks. *Artificial Intelligence*, 44(3):257–303. 130, 134, 137
- [Wellman, 1994] MICHAEL P. WELLMAN (1994). Inference in cognitive maps. 137
- [Wilson, 2006] NIC WILSON (2006). An Efficient Upper Approximation for Conditional Preference. In [Brewka *et al.*, 2006], pages 472–476. 127
- [Wotawa, 2001] FRANZ WOTAWA (2001). A variant of Reiter’s hitting-set algorithm. *Information Processing Letters*, 79(1):45–51. 95
- [Zadeh, 1965] LOTFI A. ZADEH (1965). Fuzzy Sets. *Information and Control*, 8(3):338–353. 12, 35, 138
- [Zhou *et al.*, 2003] SANMING ZHOU, JIAN YING ZHANG et ZHI-QIANG LIU (2003). Quotient FCMs – A Decomposition Theory for Fuzzy Cognitive Maps. *IEEE Transactions on Fuzzy Systems*, 11(5):593–604. 32

Publications personnelles

Chapitres de livres

- [1] LIONEL CHAUVIN, DAVID GENEST, AYMERIC LE DORZE et STÉPHANE LOISEAU (2013). User Centered Cognitive Maps. *In* FABRICE GUILLET, BRUNO PINAUD, GILLES VENTURINI et DJAMEL A. ZIGHED, éditeurs : *Advances in Knowledge and Discovery Management*, volume 471 de *Studies in Computational Intelligence*, pages 203–220. Springer.

Conférences internationales avec comité de sélection

- [2] AYMERIC LE DORZE, LIONEL CHAUVIN, LAURENT GARCIA, DAVID GENEST et STÉPHANE LOISEAU (2012). Views and Synthesis of Cognitive Maps. *In* ALLAN RAMSAY et GENNADY AGRE, éditeurs : *AIMSA 2012*, volume 7557 de *Lecture Notes in Computer Science*, pages 119–124. Springer.

Conférences francophones avec comité de sélection

- [3] LIONEL CHAUVIN, DAVID GENEST, AYMERIC LE DORZE et STÉPHANE LOISEAU (2011). Cartes cognitives : une exploitation à base d'échelle, vue et profil. *In* ALI KHENCHAF et PASCAL PONCELET, éditeurs : *EGC'2011*, volume RNTI-E-20 de *Revue des Nouvelles Technologies de l'Information*, pages 503–514. Hermann-Éditions. Prix du meilleur article académique.
- [4] AYMERIC LE DORZE, LAURENT GARCIA, DAVID GENEST et STÉPHANE LOISEAU (2012). Cartes cognitives synthétisées. *In* *Actes de la 18ème conférence Reconnaissance des Formes et Intelligence Artificielle (RFIA 2012)*, pages 66–72.
- [5] AYMERIC LE DORZE, LAURENT GARCIA, DAVID GENEST et STÉPHANE LOISEAU (2013). Validation d'une carte cognitive. *In* DJAMEL A. ZIGHED et GILLES PONCELET, éditeurs : *EGC'2013*, volume RNTI-E-24 de *Revue des Nouvelles Technologies de l'Information*, pages 91–96. Hermann-Éditions.

Ateliers francophones

- [6] AYMERIC LE DORZE, LAURENT GARCIA, DAVID GENEST et STÉPHANE LOISEAU (2011). Modèles interactifs de cartes pour l'aide à la prise de décision. *In « Interagir avec des représentations formelles », Atelier de la conférence IHM 2011.*

- [7] AYMERIC LE DORZE, BÉATRICE DUVAL, LAURENT GARCIA, DAVID GENEST, PHILIPPE LERAY et STÉPHANE LOISEAU (2013). Cartes cognitives probabilistes. *In Actes des 7èmes Journées de l'Intelligence Artificielle Fondamentale (JIAF 2013).*

Thèse de Doctorat

Aymeric LE DORZE

Validation, synthèse et paramétrage des cartes cognitives

Validation, synthesis, and settings of cognitive maps

Résumé

Les cartes cognitives sont un modèle graphique qui permet de représenter visuellement des influences entre des concepts. Ce modèle est utilisé pour l'aide à la prise de décision. Il dispose pour cela d'une opération, appelée influence propagée, qui calcule l'influence globale de n'importe quel concept de la carte sur n'importe quel autre.

Notre première contribution consiste à proposer différents critères de qualité pour valider une carte. Ces critères permettent de s'assurer que l'influence propagée d'un concept sur un autre n'est pas ambiguë. Notre seconde contribution est de synthétiser différentes cartes cognitives produites par différents designers en une carte unique. Nous utilisons pour cela une taxonomie de concepts qui exprime les liens entre les concepts utilisés dans les différentes cartes et un ordre de préférence sur ces designers pour gérer les informations contradictoires. Notre troisième contribution consiste à étudier la sémantique qu'on peut donner à une carte cognitive en comparant ce modèle à d'autres modèles de représentation des connaissances. De cette étude, nous définissons le modèle des cartes cognitives probabilistes dans lequel les valeurs des influences sont apparentées à des probabilités. Nous avons également développé un outil logiciel implémentant les différentes notions décrites précédemment.

Mots clés

carte sémantique, carte cognitive, validation, synthèse, réseau bayésien.

Abstract

A cognitive map is a graphical model that represents influences between concepts. This model is used in decision-making. It provides an operation, called propagated influence, that computes the global influence of any concept of the map on any other one.

Our first contribution is to provide some quality criteria in order to validate a map. These criteria verify if the propagated influence of a concept on another one is not ambiguous. Our second contribution is to synthesize a set of cognitive maps provided by different designers into a unique map. To do so, we use a taxonomy of concepts that expresses relations between the concepts of the different maps and a preference order on these designers to manage contradictory information. Our third contribution is the study of the semantics of a cognitive map by comparing this model to other knowledge representation models. In that study, we define the probabilistic cognitive map model such that the influence values are close to probabilities. We have also developed a software that implements the notions previously described.

Key Words

semantic map, cognitive map, validation, synthesis, bayesian network.