Une approche d'intelligence collective pour la conception d'un system d'aide à la décision appliqué à l'économie circulaire

Mickaël Bettinelli

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Informatique**

Arrêté ministériel : 25 mai 2016

Présentée par

Mickaël BETTINELLI

Thèse dirigée par **Michel OCCELLO**, Professeur à l'université Grenoble Alpes,
codirigée par **Daniel BRISSAUD**, Professeur INP, Grenoble INP,
et co-encadrée par **Damien GENTHIAL**, Maître de conférence à l'université Grenoble Alpes
préparée au sein du **Laboratoire de Conception et d'Intégration des Systèmes**
dans **l'École Doctorale Mathématiques, Sciences et technologies de l'information, Informatique**

Une approche d'intelligence collective pour la conception d'un système d'aide à la décision appliqué à l'économie circulaire

A collective intelligence approach for the conception of a decision support system applied to the circular economy

Thèse soutenue publiquement le **10 décembre 2021**, devant le jury composé de :

Madame MARIE-PIERRE GLEIZES
PROFESSEUR DES UNIVERSITÉS, UNIVERSITÉ TOULOUSE 3 - PAUL SABATIER, Rapporteur

Monsieur OLIVIER BOISSIER
PROFESSEUR DE L'INSTITUT MINES-TÉLÉCOM, ÉCOLE DES MINES DE SAINT-ÉTIENNE, Rapporteur

Monsieur JEAN-PAUL JAMONT
PROFESSEUR DES UNIVERSITÉS, UNIVERSITÉ GRENOBLE ALPES, Président

Madame SALIMA HASSAS
PROFESSEUR DES UNIVERSITÉS, UNIVERSITÉ LYON 1 - CLAUDE BERNARD, Examinateur

Monsieur EMMANUEL ADAM
MAÎTRE DE CONFÉRENCE, UNIVERSITÉ POLYTECHNIQUE DES HAUTS-DE-FRANCE, Examinateur

À mes parents,

Remerciements

Ces trois dernières années sont passées très vite. Je n'imaginais pas que cette expérience serait aussi enrichissante au moment de me lancer dedans. Le doctorat nécessite d'apprendre et de développer de nombreuses compétences, pour la plupart que j'étais loin de maîtriser à mon arrivée. Je me sens particulièrement chanceux d'avoir bénéficié d'un encadrement à l'écoute de mes interrogations et très diplomate. Je tiens à remercier Michel Occello, Damien Genthial et Daniel Brissaud pour leur temps, leurs conseils et l'encadrement chaleureux qu'ils m'ont prodigués tout ce temps. Je profite de cette page pour leur manifester ma gratitude pour m'avoir permis de faire un stage de master avec eux puis de m'avoir offert l'opportunité de continuer en doctorat. J'espère que nous aurons l'occasion de travailler ensemble à nouveau.

Je tiens également à remercier les membres du jury qui ont accepté de participer à la soutenance et plus particulièrement les rapporteurs qui m'ont fait l'honneur de rapporter sur ce travail. J'espère qu'il a été aussi intéressant à lire qu'il a été à faire. Merci à Marie-Pierre Gleizes, Olivier Boissier, Jean-Paul Jamont, Emmanuel Adam et Salima Hassas pour leurs remarques constructives sur le manuscrit et lors de la soutenance. Merci aux membres du jury et aux encadrants qui l'ont pu de s'être déplacés jusqu'à Valence pour participer à la soutenance en ces temps de pandémie. Votre présence fut fortement appréciée.

La réalisation de ce travail aurait été plus complexe sans les membres du laboratoire LCIS. Merci à Clément Raïevsky pour ses conseils et son aide, tant au niveau scientifique que pour l'enseignement, et pour sa bonne humeur quasiment constante. Un grand merci à Jean-Paul Jamont pour son intérêt pour mon travail, pour ses relectures, ses conseils et sa gentillesse. Le coronavirus et le télétravail qu'il a engendré ne m'ont pas permis de participer autant que je l'aurais souhaité à la vie du laboratoire, mais je garderai de bons souvenirs de quelques pauses café et surtout des sorties d'équipe en plein air que nous avons faites ensemble. Merci à André Lagrèze pour l'organisation de ces sorties. Merci aussi à Caroline Palisse et Carole Seyvet pour l'aide qu'elles m'ont apportée dans les diverses phases administratives qui jonchent le travail de thèse.

Une pensée toute particulière va aux anciens et aux nouveaux amis rencontrés, soit au laboratoire, soit dans la vie courante. Je remercie Eskandar Kouicem pour son aide durant la thèse (les rappels de dates butoir), pour toutes nos sorties et nos discussions. Merci également à Karem Hafsi et à Arthur Baudet, nos sorties au restaurant me manqueront.

Mes divers loisirs m'ont permis de décompresser après les longues journées de rédaction, je tiens à remercier Omer Sharon et Audrey Manoha pour tous les bons moments passés ensemble. J'espère vous revoir au détour d'une bière ou d'un mur d'escalade dans le futur. Il est difficile de faire une liste exhaustive des personnes rencontrées mais ces quelques personnes garderont une place particulière dans ma mémoire : Rahul, Hakim, Jean-Baptiste, Zain, Omar. Merci à Chloé Damas pour son soutien, sa gentillesse, et sa présence durant ces dernières années. J'espère que tes visites à Valence ne te manqueront pas trop. Je remercie aussi mes amis de plus longue date que j'ai malheureusement moins vus ces dernières années étant sur Valence. Merci à vous de m'avoir supporté tout ce temps : Antoine Roux, Mathilde Boltenhagen, Arthur Aubret, Théo Jaunet, Miguel Solinas, Maximilien Vogt, Claire Cartron, Erwan Guilloux, Corentin Maisonneuve, Jocelyn

Caraman, Charles Vuillermin, Vincent Aumenier et Rémi Gérard. Enfin, une pensée pour le petit chat dont l'origine demeure inconnue qui a observé l'avancement de ce manuscrit depuis ses premières lignes jusqu'à ces remerciements allongé sur mon canapé.

Pour terminer ces remerciements, je souhaite adresser ma plus profonde gratitude à mes parents qui m'ont toujours soutenu et poussé dans les études. Je ne serais très certainement pas allé aussi loin sans vous.

Table des matières

1 Introduction

Sommaire

Les systèmes industriels sont aujourd'hui pour la majorité dans un modèle économique *prendre–faire–consommer–disposer* dans lequel nous jetons nos produits usagés du quotidien, parfois car ils ne sont plus fonctionnels, mais parfois simplement pour les remplacer par des modèles plus récents. Les matériaux les composants sont inutilement transformés en déchets et l'énergie utilisée pour leur fabrication est gaspillée. Le recyclage apporte une solution partielle à ce problème en permettant de transformer certains de nos déchets en matériaux réutilisables comme nous le faisons par exemple pour les batteries de véhicules électriques. Cependant ce procédé est coûteux et ne permet pas toujours de réutiliser tous les matériaux. Pour répondre aux problèmes de ce modèle économique, des stratégies de réutilisation ont été conçues afin de réintégrer des matériaux et des composants encore fonctionnels dans des produits neufs. L'économie circulaire est un modèle économique intégrant ces stratégies de réutilisation afin de maximiser la durée d'utilisation de ce que nous fabriquons. Elle peut être définie comme un modèle économique dans lequel est remplacé le concept de fin de vie d'un produit par des processus de réutilisation et de recyclage qui ont pour objectif un développement durable [Kirchherr2017]. Cependant, celle-ci est encore difficilement envisageable à grande échelle en raison de contraintes législatives empêchant les acteurs de pleinement réutiliser les composants et matériaux usagés mais aussi en raison d'une méconnaissance de ce modèle qui induit une difficulté à réorganiser la chaîne logistique, à adapter les processus de production ou encore à spécifier des exigences de qualité pour la fabrication des produits. En outre, les consommateurs sont encore très peu conscients de l'existence de ce modèle économique et ne sont donc pas particulièrement intéressés par ce type de produits [Kirchherr2018].

Mais les systèmes industriels classiques doivent faire face aux principaux défis de l'industrie du futur : pénurie de ressources, efficacité énergétique, attentes en matière de personnalisation, qualité des emplois requise par la quatrième révolution industrielle, *etc*. Dans le cadre de l'IDEX Grenoblois, le projet Circular est un projet financé par l'Agence Nationale de la Recherche qui a pour objectif de montrer que les systèmes industriels circulaires agiles donneront aux entreprises la capacité de satisfaire ces exigences tout en proposant de nouveaux produits à haute valeur ajoutée en minimisant l'utilisation des ressources matérielles et énergétiques. L'objectif de ce projet est de faire évoluer le modèle économique classique et actuel *prendre–faire–consommer–disposer* vers un modèle économique circulaire *prendre–faire–consommer–réutiliser*. Les principaux obstacles scientifiques à ce modèle sont l'agilité des systèmes de production, la collaboration homme / machine, la mise en œuvre des solutions numériques et la conception et la mise en place des nouvelles organisations nécessaires.

Devant la grande quantité de produits arrivés en bout de cycle de vie chaque jour, l'étape *réutiliser* est loin d'être évidente à mettre en place. En effet, les produits en fin de vie sont souvent abîmés et nécessitent des réparations (quand cela est possible) pour être réintroduits sur le marché. Mais même une fois les produits démontés et leurs composants nettoyés et réparés, leur grande diversité et leur grand nombre rend la conception et la production de nouveaux produits très complexe pour des opérateurs humains. Pour résoudre ce problème, le projet Circular propose la mise en place d'un système d'aide à la décision permettant de guider les utilisateurs dans la conception de nouveaux produits à partir de composants utilisés par le passé. Ce système d'aide à la décision sera le fil rouge dont nous nous servirons tout au long de ce manuscrit pour présenter notre contribution.

1.1 Motivations

Dans le contexte du projet Circular, notre objectif est l'élaboration d'un système d'aide à la décision capable de concevoir des produits grâce à des composants usés et une spécification de la part d'un concepteur humain (figure 1.1). Le but de ce système est de rendre possible l'économie circulaire à travers l'usage de stratégies de réutilisation. Plus précisément, nous nous concentrons sur les stratégies de *remanufacturing* et de *repurposing*. La première consiste en la réutilisation de composants usés pour une application similaire à leur objectif initial, par exemple récupérer des composants d'un lave-linge pour construire un nouvel appareil du même modèle. La seconde consiste en leur réutilisation et leur adaptation de manière plus large dans des applications pour lesquelles ils n'avaient pas été pensés à l'origine. Par exemple réutiliser le moteur d'un lave-linge pour la fabrication d'un sèche-linge de marque différente.

Le système d'aide à la décision doit être capable de gérer les caractéristiques liées aux composants usés afin de les mettre en relation pour concevoir un produit. Une caractéristique ne décrit pas uniquement le physique d'un composant mais peut aussi décrire son état logistique, par exemple la localisation de son entrepôt, son prix ou ses frais de port. Ainsi certaines caractéristiques sont sujettes au changement. Le prix d'un composant peut par exemple être modifié en fonction de son ancienneté ou des stocks disponibles. Mais l'état du composant peut aussi être changé lorsque celui-ci est transformé pour subvenir à d'autres besoins dans le cas d'une stratégie de repurposing. Un exemple bien connu dans le domaine du repurposing est celui de l'adaptation et la réutilisation de batteries usées de véhicules électriques pour le stockage d'énergies renouvelables irrégulières comme l'énergie solaire ou éolienne. La modification des caractéristiques des composants peut avoir lieu à n'importe quel moment. Le système d'aide à la décision doit alors être être suffisamment dynamique pour prendre en compte les changements d'état des composants utilisés quant bien même un utilisateur serait en train de s'en servir pour concevoir un produit. En outre, l'utilisateur peut lui aussi décider de modifier sa requête à n'importe quel moment ce qui demande au système la capacité de s'adapter dynamiquement au changement d'objectif.

Les composants sont introduits dans le système par des fournisseurs. L'un d'eux peut mettre à disposition du système d'aide à la décision des composants nouvellement arrivés dans son inventaire modifiant l'inventaire disponible au système d'aide à la décision. Celui-ci doit être capable de s'adapter dynamiquement à la modification des pièces en inventaire. De la même manière, lorsqu'un utilisateur décide d'acheter les composants d'un produit conçu par le système, ceux-ci doivent sortir du système pour ne pas être choisis à nouveau par d'autres utilisateurs. Le système doit donc être ouvert et s'adapter à l'entrée et la sortie de composants.

Enfin, ce système d'aide à la décision doit négocier avec l'utilisateur pour l'aider à concevoir les produits dont il a besoin. Supposons que l'utilisateur décide de concevoir un produit électroménager. Si la solution répond parfaitement à son besoin, les composants peuvent simplement être réservés et plus tard assemblés par les divers acteurs industriels. Mais si elle présente des différences avec la demande, il est nécessaire à l'utilisateur de comprendre pourquoi la solution que lui retourne le système est telle qu'elle est afin d'adapter sa demande et peut être de trouver des solutions plus adéquates. Cela peut consister en une brève description de la raison rendant le système incapable de proposer une solution plus pertinente, par exemple l'absence des composants nécessaires ou l'inadéquation de certaines de leurs caractéristiques au besoin utilisateur.

La figure 1.1 présente une vue du système d'aide à la décision où les fournisseurs fournissent des composants au système ainsi que les connaissances sur la façon dont ils s'assemblent. Un client peut interagir avec le système en lui demandant de concevoir un produit. Le système d'aide à la décision doit alors répondre le mieux possible au besoin en fonction des composants disponibles. Par exemple le schéma montre que le système est composé d'un ensemble de vis, système de refroidissement et cellules de batteries qui peuvent produire une batterie électrique. Si l'utilisateur lui demande de concevoir une batterie pour une véhicule électrique, le système lui proposerait une conception avec ces trois pièces mais serait incapable de lui proposer une solution complète puisque certains composants de batterie sont manquants.

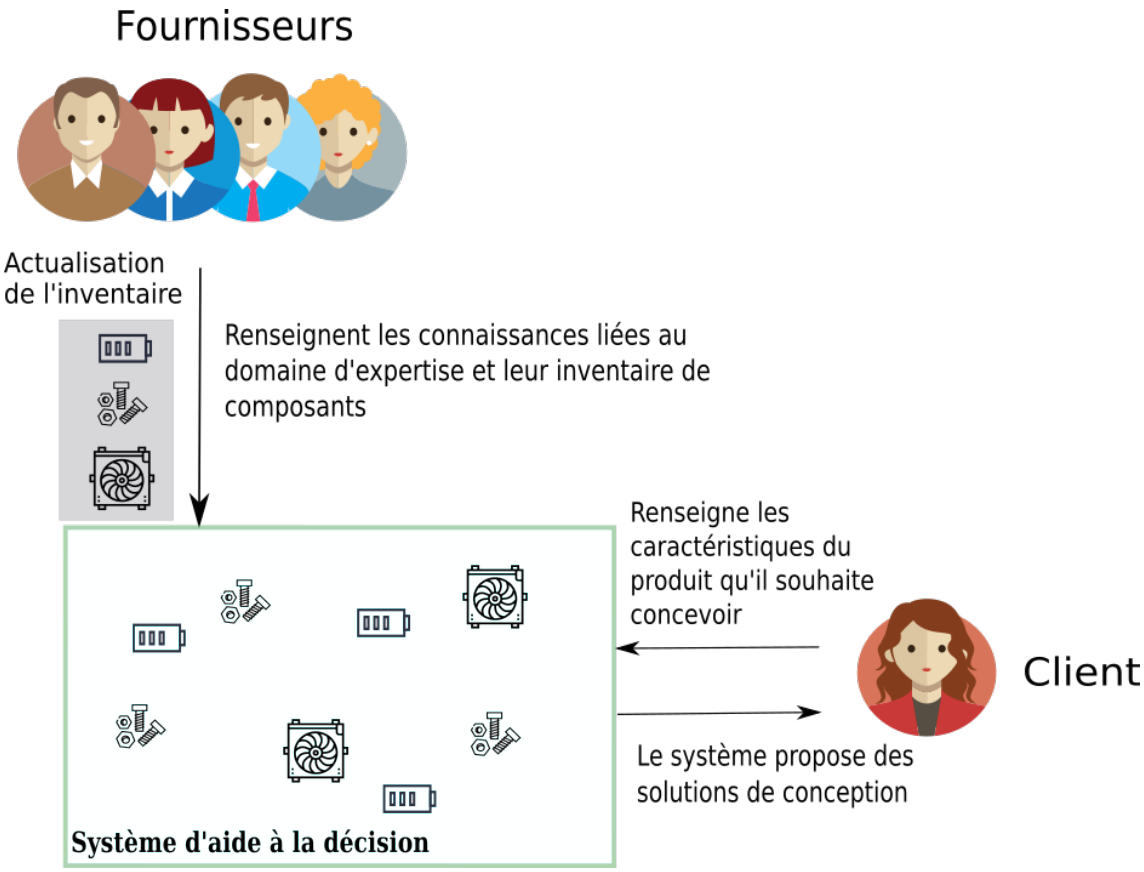


Figure 1.1 – Vision du système d'aide à la décision.

Pour résumer, le système d'aide à la décision devra, à partir d'un ensemble de composants usés, essayer de concevoir des produits en les regroupant en fonction de leurs caractéristiques et du désir d'un utilisateur. Cette problématique peut être rapprochée d'un problème de génération de structures de coalitions.

1.2 Problématique

Comme nous l'avons vu, le système d'aide à la décision manipulera de nombreux composants et devra les utiliser pour concevoir des produits. Le problème que nous traitons est donc celui du regroupement de composants pour la génération de conceptions de produits guidée par une demande utilisateur. Bien que le contexte de ce travail soit orienté vers la conception de ce système d'aide à la décision, notre problématique est du point de vue informatique celui d'un problème de formation de coalitions. Plus spécifiquement, il s'agit d'un problème de génération de structures de coalitions (CSG) dans lequel le système va devoir générer des partitionnements de coalitions composés de pièces physiques où chaque coalition doit ressembler le plus possible à ce qui est attendu par l'utilisateur. Comme nous le verrons dans le chapitre suivant, ce problème est très répandu dans la littérature et varie selon les domaines et leurs spécificités. Le contexte du système d'aide à la décision force cependant la mise en place de certaines variations de cette problématique pour la rendre plus réaliste et adaptée au cas d'application. La majorité des problèmes de CSG partent du principe que l'ensemble des entités (souvent appelé des agents) à partitionner est fixe. Dans notre cas, nous traiterons d'un problème de CSG variable et ouvert pour que des composants puissent être retirés ou ajoutés au système à tout moment, mais aussi qu'ils puissent être modifiés au besoin. En outre, contrairement à de nombreux problèmes de CSG, nous supposons que nos partitions peuvent se chevaucher. Autrement dit que les conceptions de produits formées par le système d'aide à la décision puissent utiliser des composants en commun du moment que ces composants n'ont pas été réservés par d'autres utilisateurs.

Pour résumer, notre problématique est celle d'un problème de génération de structures de coalitions chevauchantes avec un ensemble ouvert d'agents variables. Dans la section suivante, nous présentons notre positionnement par rapport à ce problème et sa littérature.

1.3 Positionnement

Les approches classiques sont peu adaptées à un problème de CSG ouvert et variable. Comme nous le verrons dans le chapitre suivant, nombre d'entre elles sont très efficaces pour trouver des solutions optimales dans des systèmes composés de peu d'agents ou encore pour trouver des solutions approximatives sur des systèmes à grande échelle. Il n'existe cependant pas à notre connaissance de méthodes pour répondre à notre problème de CSG ouvert, variable avec un partitionnement chevauchant des coalitions. Pour cause, les méthodes exactes sont souvent très chronophages et ne permettent pas de s'adapter à un système changeant et les approches permettant un passage à l'échelle reposent souvent sur des méthodes peu adaptées à l'ouverture du système ou à la modification de ses agents. La plupart de ces approches ou méthodes ont pour parti pris que les données utilisées sont immuables occultant alors cette contrainte pourtant réaliste qu'est la variabilité. Leur principal souci est alors de trouver les coalitions optimales parmi un ensemble d'agents fixe. Contrairement à ces méthodes, la grande variabilité de notre système ne nous permet pas de considérer un ensemble d'agents fixe

. Notre objectif n'est donc pas de trouver la structure de coalitions optimale à partir d'un ensemble invariable d'agents, mais plutôt de faire émerger des coalitions de manière incrémentale dans un système changeant.

Nous nous tournons naturellement vers une méthode de résolution de problèmes incrémentale régie par une optimisation continue de l'interaction entre les éléments locaux : les systèmes multi-agents. Dans les chapitres suivants, nous proposerons l'utilisation d'une approche multi-agents et justifierons les avantages apportés par ce paradigme dans le cadre d'un système d'aide à la décision ouvert, variable et nécessitant de l'explicabilité.

1.4 Thèse défendue

Nous défendons dans ce manuscrit l'intérêt de l'utilisation d'un paradigme multi-agents pour la résolution d'un problème de génération de structures de coalitions variables. Pour cela, ce travail propose une architecture d'agent s'inspirant de mécanismes sociaux pour la formation de coalitions dans le cadre d'un système d'aide à la décision pour le remanufacturing et le repurposing. Du point de vue de l'intelligence artificielle et du génie logiciel, la spécificité de cette architecture est sa capacité à créer des structures de coalitions chevauchantes tout en s'adaptant à un environnement ouvert et changeant. Mais du point de vue du génie industriel la particularité de l'architecture est aussi sa capacité à prendre en compte des contraintes réalistes liées au domaine du remanufacturing ainsi que d'être adaptée à un outil d'aide à la décision pour la conception de produits.

1.5 Structure du manuscrit

Ce manuscrit est divisé en huit chapitres. L'introduction a présenté les motivations de notre étude, notre problématique et notre positionnement. Le chapitre 2 aborde des approches permettant de résoudre plusieurs variantes du problème de formation de coalitions et dresse une comparaison de ces approches afin de vérifier leur adéquation avec notre problématique. Nous verrons qu'en plus d'être étudiés à travers de nombreuses méthodes, les problèmes de formation de coalitions possèdent une multitude de variantes. Outre la présentation de l'état de l'art, le chapitre 2 introduit et justifie l'utilisation du paradigme multi-agents pour la résolution de notre problématique. Le chapitre 3 fait le lien entre notre problématique de formation de coalitions et la formation de groupes en sociologie. Elle présente notre source d'inspiration, *les Dynamiques de Groupes*, et présente les grandes théories à l'origine de la formation de groupes chez l'être humain pour en extraire des facteurs utilisables dans un système artificiel et une architecture d'agent. Le chapitre 4 définit les besoins de notre architecture en fonction des propriétés requises par le système d'aide à la décision auquel elle doit s'adapter et passe en revue les diverses classifications et architectures

existantes pour en sélectionner une sur laquelle nous appuyer pour répondre à notre problématique. Le chapitre 5 montre les limites de l'architecture choisie et présente nos ajouts et notre contribution : l'architecture nommée ABSG (Attraction Based Structure Generation), permettant la génération de coalitions dans le cadre d'un système ouvert et variable. Le chapitre 6 introduit les enjeux et problématiques futures des acteurs industriels du remanufacturing ainsi qu'un cas d'étude sur les appareils électroménagers construit grâce à une collaboration industrielle. Ce cas d'étude est utilisé dans le chapitre 7 dans lequel nous évaluons notre approche et notre architecture d'agent sur la base de métriques définies dans ce même chapitre.

2

Vers le problème de formation de coalitions

Sommaire

2.1 Propriétés requises

Le système d'aide à la décision doit concevoir des produits dans un contexte dynamique dans lequel il doit s'adapter aux types de composants manipulés, à l'utilisateur avec lequel il interagit et à un flux entrant et sortant de composants. Pour se faire, son fonctionnement repose sur quatre propriétés lui permettant de travailler de manière dynamique quel que soit le domaine d'étude. Nous les présentons dans cette partie.

2.1.1 Exprimabilité

Le système d'aide à la décision doit être capable de concevoir des produits de différentes natures sans modifier son fonctionnement tout en étant capable de modéliser des composants possédant un large éventail de spécificités. Premièrement, cela nécessite de s'affranchir d'un domaine d'expertise spécifique et d'être capable de s'adapter à n'importe quel type de composant. Par exemple, les problématiques sous-jacentes à la fabrication d'un vélo sont différentes de celles d'une batterie de véhicule électrique. La première a comme principale préoccupation la mécanique des roues, de la chaîne et du changement de vitesses alors que la seconde se concentre sur les propriétés chimiques et électriques du produit. Deuxièmement, il est nécessaire que le système puisse décrire tous les composants quels que soient leurs attributs, leurs fonctions et leur complexité. Le système d'aide à la décision doit aussi bien être capable de décrire les caractéristiques physiques d'un composant que son emplacement dans un produit ou les modifications qui lui sont applicables. Il doit aussi pouvoir en exprimer un grand nombre afin de modéliser les composants les plus riches en attributs. Par exemple, le système doit pouvoir utiliser des composants tels qu'une batterie électrique possédant des dizaines de caractéristiques physiques comme : les dimensions, le voltage, l'ampérage, le type des réactions chimiques utilisées, le type de système de refroidissement, les formes des cellules, le nombre de cellules que peut gérer le système de management de l'énergie, *etc* tout en possédant la capacité d'exprimer comment assembler les composants pour former cette batterie (*e.g.* dans quel type de voiture la placer et avec quelles pièces du véhicule la raccorder). Enfin, il faut que le système puisse exprimer des règles de modification – aussi appelées *transformations* – des produits afin qu'un composant puisse être réutilisé plus facilement dans son application d'origine (*e.g* modèle différent d'un même appareil), mais aussi être réutilisé pour une nouvelle application (*e.g.* un autre type d'appareil que l'application d'origine). Nous pouvons illustrer le concept de *transformation* avec l'exemple des batteries de véhicules électriques qui peuvent être remanufacturées et réutilisées pour le stockage d'énergies renouvelables irrégulières.

2.1.2 Dynamicité

Les composants que devra traiter le système d'aide à la décision ont un large éventail d'attributs. Certains sont physiques, plus ou moins stables dans le temps, d'autres sont logistiques et donc beaucoup plus sujets à être modifiés. Par exemple des composants peuvent changer de lieu de stockage et leur prix ou leur frais de port peuvent être modifiés en fonction de la distance du composant ou de sa rareté. Il est nécessaire d'actualiser les informations des composants dans le système pour que les décisions prises par celui-ci et par l'utilisateur demeurent pertinentes. Le

système d'aide à la décision doit donc être capable de s'adapter à un environnement variable rendant nécessaire une capacité d'ajustement de son comportement de manière dynamique pour faire face aux imprévus.

2.1.3 Ouverture

Un fournisseur de composants peut vouloir ajouter ou retirer ses pièces du système d'aide à la décision. Celui-ci doit donc être ouvert pour pouvoir prendre en compte les modifications de disponibilité des pièces à tout moment. Il doit pouvoir s'adapter à ces modifications, même lorsqu'il est en cours d'utilisation. Par exemple, si une cellule de batterie électrique est ajoutée quelques instants après qu'un utilisateur demande au système de lui concevoir une batterie, la nouvelle pièce doit être prise en compte par le système. De la même manière si un utilisateur achète une batterie, les composants utilisés doivent être retirés du système et des propositions faites aux autres utilisateurs afin de remplacer ces composants devenus indisponibles et les empêcher d'être utilisés par plusieurs personnes.

2.1.4 Explicabilité

Lorsque le système d'aide à la décision conçoit un produit, il peut ne pas toujours avoir tous les composants nécessaires à sa disposition. Si son objectif était de concevoir un vélo il pourrait ne pas trouver deux roues de mêmes dimensions. Pour déterminer quelles conceptions sont les meilleures, l'utilisateur a besoin d'une métrique pour évaluer leur qualité. Le système doit être capable d'associer des valeurs de qualité aux conceptions qu'il réalise afin de proposer les meilleures à l'utilisateur. Dans la même idée, si les conceptions proposées sont incomplètes le système d'aide à la décision doit pouvoir négocier avec l'utilisateur en proposant des composants capables de remplacer les pièces manquantes. Ce dernier doit être capable de comprendre quels facteurs impactent négativement la qualité d'une conception pour pouvoir la compléter avec des composants plus adéquats ou simplement modifier sa requête d'origine pour prendre en compte l'incapacité du système d'aide à la décision à trouver une pièce. La capacité du système à expliquer ses résultats est une fonctionnalité centrale de l'outil d'aide à la décision permettant la négociation avec les utilisateurs pour améliorer l'aide apportée dans le processus de conception de produits.

2.2 La formation de coalitions

Le système d'aide à la décision travaille avec un grand nombre de composants. Nous avons besoin de les regrouper en coalitions afin de former les produits demandés par l'utilisateur. Chaque solution du système est une coalition et représente une conception du produit demandé par l'utilisateur. Ces conceptions peuvent ne pas ressembler exactement à ce qu'a demandé l'utilisateur si les composants présents dans le système ne permettent pas de concevoir le produit demandé. L'objectif du système est de répondre au mieux à la demande de l'utilisateur en formant une multitude de coalitions parmi lesquelles ce dernier peut sélectionner la solution qu'il souhaite mettre en

œuvre. Ce problème peut être rapproché d'un problème de génération de structures de coalitions (CSG) où chacune de nos conceptions de produits est une coalition faisant partie de la structure de coalitions. De manière générale, une structure de coalitions est un ensemble de coalitions, chevauchantes ou non, composées d'individus. Lorsque les coalitions sont chevauchantes, plusieurs individus peuvent participer à plusieurs coalitions. Celles-ci ont deux principales caractéristiques :

1. Elles sont générées pour atteindre un objectif et sont de courte durée. Dans notre problème, les coalitions sont créées pour ressembler le plus possible à une demande et elles peuvent être détruites ou modifiées si la demande vient à changer ;
2. Leur organisation est souvent plate. Il n'existe pas de hiérarchie parmi les membres de la coalition.

Le système d'aide à la décision devant être dynamique et ouvert, notre problème de formation de coalitions dépend d'un ensemble de composants variables en nombre, en caractéristiques et en dynamiques. L'objectif n'est pas de former des coalitions optimales à partir d'un ensemble fixe d'éléments mais plutôt de faire émerger de nouvelles structures à partir d'un nombre variable de composants. Le système d'aide à la décision pouvant présenter plusieurs conceptions de produits, nous nous plaçons dans un problème de formation de coalitions chevauchantes. Les conceptions peuvent être en conflit les unes avec les autres lorsqu'elles utilisent les mêmes composants en quantité limité. Ces conflits peuvent nécessiter l'intervention de l'utilisateur et le comportement du système doit donc pouvoir être explicable.

Nous traitons donc un problème de génération de structures de coalitions chevauchantes ouvert et variable. De nombreuses approches se penchent sur ce problème. Voici les plus courantes :

— le clustering : approche se concentrant sur le partitionnement d'éléments en groupes ;

— l'intelligence distribuée : approche visant à coordonner des entités dans un environnement situé pour les regrouper selon certaines spécifités ;

— le choix social informatique : approche à la croisée de l'économie et de l'informatique qui s'intéresse à la conception de méthodes de prise de décision collective ;

— la théorie des jeux coopératifs : approche où le processus de formation de coalitions est fait à partir d'un ensemble de joueurs possédant des préférences sur ces coalitions ;

— la théorie des enchères : approche voyant le processus de formation de coalition comme une enchère où les coalitions sont formées des meilleurs enchérisseurs ;

— la programmation dynamique : approche visant à trouver une solution exacte au problème de CSG en cherchant des stratégies de réduction de l'espace de recherche ;

— les systèmes multi-agents : approche qui a pour spécificité de distribuer le problème sur plusieurs nœuds appelés *agents* pour en faire émerger une solution ;

La partie suivante fait un état de l'art de ces approches et discute de leurs propriétés.

2.3 Approches communes

2.3.1 Clustering

Le clustering est une approche dans laquelle l'objectif est de trier un ensemble de données en sous-ensembles sur la base d'un critère de similarité. Les algorithmes de clustering peuvent être divisés en deux catégories : les méthodes de partitionnement et les méthodes hiérarchiques. La première permet de générer un partitionnement de l'ensemble de données original alors que la seconde génère des partitionnements arrangés sous formes de couches dans lequel chaque couche est un partitionnement de la couche précédente. K-Means [MacQueen1967] est un des algorithmes de partitionnement les plus connus. Il partitionne un ensemble de données en essayant de minimiser la distance entre les points de chaque partition. Bien que très utilisé, il a néanmoins comme principal désavantage de devoir connaître le nombre de partitions à créer avant son exécution ce qui le rend difficilement utilisable pour une application sur le problème de formation de coalitions. SLINK [Sibson1973] est un algorithme hiérarchique utilisant une matrice de distances pour construire des clusters. Une matrice de distances indiquant la distance entre chaque donnée est construite à chaque itération de l'algorithme. À chaque étape, les éléments les plus proches dans la matrice sont fusionnés en un cluster. L'algorithme réitère ces opérations jusqu'à n'avoir plus qu'un seul cluster englobant toutes les données. On peut visualiser la formation de ces clusters sous forme d'un dendrogramme représentant un arbre. On peut ensuite couper le dendrogramme à la hauteur souhaitée pour obtenir la répartition des données en clusters. Cette méthode ne nécessite pas de connaître à l'avance le nombre de clusters à former ce qui la rend plus facilement applicable au problème de formation de coalition.

Le clustering est donc une tâche très proche de la génération de structures de coalitions (GSC). La principale différence porte sur l'objectif de la création de ces groupes. Les coalitions sont par nature orientées par un objectif, elles sont dynamiques et leur existence est de courte durée. En revanche, les sous-ensembles de données construits par un algorithme de partitionnement n'ont pas vocation à la dynamicité. Les données sont le plus souvent fixes et les groupes ne sont pas construits pour répondre à un objectif mais simplement de manière à regrouper des données similaires.

Cependant, étant donné la ressemblance de ces approches, certain travaux essaient de faire le pont entre le partitionnement de données et la formation de coalitions. Farinelli *et al.* [Farinelli2017] proposent un algorithme de formation de coalition basé sur une approche de partitionnement hiérarchique. Cet algorithme, nommé Coalition Link (C-LINK), permet de résoudre un problème de formation de coalitions à grande échelle (jusqu'à 2700 agents) de manière non optimale en un temps raisonnable. La fonction d'association utilisée est basée sur le gain, défini comme la différence entre la valeur d'une coalition composée des agents impliqués et de leurs valeurs individuelles. Les agents forment ou rejoignent des coalitions uniquement si leur gain est positif. Les résultats montrent que la méthode est efficace lorsque les agents sont fortement connectés les uns aux autres mais qu'elle devient peu performante comparé à des algorithmes *branch and bound* lorsque les agents sont peu connectés entre eux. Ce résultat s'explique par le fait que lorsque les agents ont peu de connexions les uns avec les autres, les algorithmes *branch and bound* travaillent sur un plus petit espace de recherche et peuvent alors trouver de meilleures

solutions. De plus, la méthode étant basée sur un algorithme de clustering hiérarchique, elle rend les membres d'une coalition incapables de réévaluer leur gain à rester dans la coalition. Si un agent devait entrer dans le système ou s'il devait être modifié, une approche de clustering hiérarchique ne permettrait pas de prendre en compte le nouvel état du système pour adapter les coalitions. Cette méthode ne peut donc pas être utilisée dans un système ouvert ni être utilisée avec des agents variables.

Le clustering hiérarchique se basant uniquement sur une matrice de distance pour former des coalitions, il répond au besoin d'explicabilité du système d'aide à la décision. En effet, un observateur extérieur pourrait comprendre pourquoi deux agents ont fusionné en un cluster en regardant la matrice de distances. Cependant, dans ces travaux les agents sont vus et traités comme des données immuables et les coalitions qui en résultent, bien que dépendantes d'un objectif, sont construites pour être définitives. Cette approche ne permet pas aux agents d'être variables ni au système d'être ouvert.

2.3.2 Intelligence distribuée

L'intelligence distribuée se concentre sur la coordination dans l'espace de multiples entités, appelées agents ou particules, sans contrôle centralisé. Les agents permettent via leur comportement le calcul distribué d'une solution à un problème de coordination. La coordination des agents revêt différentes formes, l'objectif peut être la formation d'essaims stables dans lesquels la position de chaque agent n'est pas connue à l'avance [Bettinelli2020b] et converge vers une position finale, ou alors la mise en place par les agents de formations préconçues par le concepteur du système. Par exemple, dans [Gazi2005], Veysel Gazi utilise une méthode de contrôle non linéaire pour faire en sorte que les agents de l'essaim forment une figure géométrique. Dans le cadre du problème de formation de coalitions, nous nous intéressons plus particulièrement à la formation d'essaims qui peuvent être assimilés à des coalitions. L'approche la plus courante consiste à analyser la formation et la stabilité d'un unique essaim constitué de quelques dizaines d'individus homogènes. Les particules sont dites homogènes lorsque celles-ci sont identiques les unes aux autres et que rien ne permet de les distinguer. Au contraire, elles sont dites hétérogènes lorsqu'elles ont leurs propres caractéristiques modifiant leur comportement. Par exemple [Shi2011] utilise des agents homogènes pour former un unique essaim. Un champ de potentiel, défini sous forme d'une fonction d'attraction/répulsion régit les interactions que les agents ont entre eux et permet la stabilisation de l'essaim. Mais l'intelligence distribuée se concentre aussi sur la formation de plusieurs essaims constitués d'agents hétérogènes. Ces travaux ne parlent pas de formation de coalitions mais du concept de ségrégation. La ségrégation est souvent utilisée dans les essaims robotiques. Son objectif est de maintenir des robots en groupes en fonction de leurs fonctionnalités et de leurs objectifs. Ce concept est donc très proche de la formation de coalitions sans chevauchement.

Kumar *et al.* présentent une méthode parvenant à doter les agents d'un essaim d'un comportement ségrégatif [Kumar2010]. De la même manière que Shi *et al.* dans [Shi2011], leur méthode utilise un champ de potentiel appliqué sur chaque agent permettant de les agréger en plusieurs essaims stables. Cependant Kumar *et al.* s'inspirent de la ségrégation de cellules biologiques pour que les agents de leur système soient capables de se diviser en plusieurs essaims. Pour cela, les agents sont dotés d'un type, par exemple A et B. La fonction de potentiel utilisée par les agents

prend en compte la distance entre les types des agents tel que, par exemple, $d^{AA} = d^{BB}$ mais $d^{BB} < d^{AB}$. Cette distance permet ensuite d'adapter la portée du champ d'attraction et de répulsion en fonction du type des agents dont on calcule le mouvement.

Santos *et al.* étendent le travail de Kumar *et al.* pour faire de la ségrégation à plus grande échelle avec 150 agents et 15 types [Santos2014]. Les auteurs notent que la fonction de potentiel de [Kumar2010] peut faire tendre les agents vers un minimum local les empêchant parfois d'atteindre leur groupe. Ils modifient l'équation de Kumar *et al.* en y ajoutant un terme quadratique permettant de supprimer le minimum local. Sa suppression permet ensuite aux agents de continuer leur mouvement vers leur groupe.

Inácio *et al.* utilisent une stratégie inspirée de l'optimisation par essaims particulaires (PSO) pour la ségrégation d'agents hétérogènes [Inácio2019]. De la même manière que dans [Kumar2010], les agents possèdent un type leur servant à identifier les individus avec lesquels former un essaim. Mais contrairement à l'étude de Kumar *et al.*, les agents ont une portée de capteurs limitée. Ils ne peuvent communiquer ou détecter un agent se trouvant hors d'une certaine portée. Le déplacement des agents dépend de trois éléments : 1. leur vitesse à l'étape précédente pour calculer l'inertie, 2. la détection des agents de même type se trouvant à leur portée, 3. une mémoire enregistrant les emplacements des sous groupes qu'ils ont détectés et qui possèdent un type différent du leur. Cette mémoire permet aux agents de se transmettre des informations sur les positions des essaims en cours de formation et de diminuer le temps de convergence. Ces trois paramètres sont agrégés et donnés en paramètre à l'algorithme ORCA [Van Den Berg2011] pour calculer leur mouvement. ORCA est un algorithme permettant à une multitude d'agents mobiles de se déplacer dans un espace en évitant les collisions. Cette stratégie est très similaire à [Kumar2010] mais montre qu'il est possible de l'utiliser pour un passage à plus grande échelle avec jusqu'à 150 agents et 15 coalitions.

Pour résumer, la ségrégation d'agents hétérogènes consiste principalement à spatialement séparer en plusieurs d'essaims un ensemble d'agents dont le type est connu à l'avance. Les agents obéissent à une loi de contrôle commune mais les dynamiques sont individuelles rendant cette méthode adaptée à une application nécessitant de la dynamicité. Cependant, elle semble limitée si l'on souhaite créer une multitude d'essaims sur la base de multiples critères. En effet, un type unique ne serait pas suffisant pour décrire les agents d'une application réaliste de formation de coalitions dans laquelle ils sont caractérisés par de nombreux critères. Bien que similaire, notre problème de formation de coalitions ne consiste pas à trier des agents selon un type mais selon un ensemble de caractéristiques. Nos agents sont beaucoup plus riches, ils possèdent une multitude de caractéristiques et de désirs et chaque agent doit adapter ses dynamiques en fonction des spécificités des autres agents. Contrairement à un type d'agent qui permet d'identifier rapidement les membres d'un même groupe, le grand nombre de caractéristiques associées à nos agents ne permettrait pas au concepteur du système de déterminer à l'avance les groupes formés par le système.

2.3.3 Choix social informatique

Le choix social est une discipline à la frontière entre l'économie et l'informatique. Elle s'intéresse à la conception et l'analyse de méthodes de prise de décisions collectives. Dans le choix

social informatique, le problème de formation de coalitions est aussi référé comme un problème d'assortiment (*matching*) dans lequel on peut affecter des agents à d'autres agents mais aussi des agents à des ressources. Cette seconde alternative est très similaire au problème d'allocation de tâches [Anshelevich2021].

Le problème de l'assortiment est introduit par David Gale et Lloyd Shapley [Gale1962]. Dans ce travail les auteurs proposent un algorithme de mariages stables dans lequel l'objectif est de regrouper des hommes et des femmes deux à deux de manière à respecter au mieux les préférences de tous. Le problème est étendu par le problème hôpitaux/résidents [Manlove2008] [Brandt2016]. Il est similaire à celui des mariages stables. Cependant, ce n'est plus un problème liant deux éléments de deux ensembles, mais un élément d'un ensemble à plusieurs éléments d'un autre ensemble. Dans le contexte de l'affectation de jeunes médecins dans les hôpitaux, les grands hôpitaux ont souvent le plus de candidatures. Les médecins étant difficilement classables, il est nécessaire de trouver un moyen pour gérer les affectations. Soit l'ensemble des jeunes médecins $R = \{r_1, .., r_n\}$ et celui des hôpitaux $H = \{h1, .., h_m\}$. Chaque hôpital h_j a une capacité $c_j > 0$ à accueillir des médecins. Il existe un ensemble $E \subseteq R \times H$ de paires médecin-hôpital acceptables. Soit $m = |E|$. Chaque médecin $r_i \in R$ a un ensemble d'hôpitaux préférés $A(r_i) = \{h_j \in H : (r_i, h_j) \in E\}$. Réciproquement, chaque hôpital $h_j \in H$ à un ensemble de médecins préférés $A(h_j) = \{r_i \in R : (r_i, h_j) \in E\}$. Toutes les entités $e_k \in R \cup H$ ont une liste de préférences dans laquelle elles ordonnent $A(e_k)$. L'ensemble des assignations de e_k dans M est noté $M(e_k)$. L'objectif est de trouver un ensemble $M(e_k)$ stable où chaque médecin a été assigné à un hôpital. $M(e_k)$ est dit stable lorsque tous les médecins sont assignés à un hôpital de façon à ce que r_i ne préfère pas un hôpital h_j à $M(r_i)$ et qu'un hôpital h_j ne préfère pas r_i à $M(h_j)$. Ce problème peut être rapproché à celui de la formation de coalition où les hôpitaux seraient des coalitions d'un nombre d'individus prédéfini. Contrairement au problème classique de formation de coalitions, ce ne sont pas les membres qui veulent s'associer à d'autres membres mais les groupes qui possèdent des préférences sur les futurs membres.

Delorme *et al.* étudient une extension du problème hôpitaux/résidents dans laquelle les médecins peuvent postuler en couple à des hôpitaux [Delorme2020]. Cette contrainte rapproche le problème hôpitaux/résidents du problème de formation de coalitions tel que nous l'avons défini. Il permet aux médecins de décider de rejoindre une organisation sous condition d'être avec une personne spécifique. En outre, l'extension permet aux hôpitaux et aux résidents d'inclure des égalités dans leur liste de préférence. Dans leur travail, Delorme *et al.* étendent la définition de la stabilité pour leur cas d'étude de façon à intégrer la notion de couple. Ils utilisent ensuite une méthode d'optimisation linéaire en nombres entiers pour trouver les meilleures assignations possibles. Les auteurs ont testé leur méthode sur un cas à grande échelle avec 750 médecins et 50 hôpitaux. Ils montrent entre autre que moins il y a de couples parmi les médecins, plus leur méthode à tendance à trouver l'ensemble d'assignations optimal.

Une autre version du problème, appelée travailleurs/entreprises, consiste à rendre possible la multiple assignation des individus à des organisations. Chaque organisation possède des membres faisant aussi partie d'autres organisations. Le problème se rapproche alors d'un problème de formation de coalitions avec chevauchement des coalitions [Klijn2014].

Pour conclure, les différentes versions du problème de formation de coalitions de cette discipline montrent quelques différences avec celle sur laquelle nous travaillons. Dans notre version, les groupes sont construits autour des besoins des composants et non d'une coalition comme c'est

le cas dans le problème hôpitaux/résidents ou travailleurs/entreprises. Cependant deux versions du problème d'assortiment peuvent paraître intéressantes pour notre problème : 1. la version étendue prenant en compte les couples et 2. la version travailleurs/entreprises. La première a l'avantage de prendre en compte les relations inter-médecins pour la formation de groupes. La seconde présente l'avantage de pouvoir réutiliser les travailleurs dans plusieurs coalitions. Du point de vue de notre problème, cette contrainte est intéressante puisqu'elle permettrait la réutilisation de composants dans plusieurs groupes. De plus, cette approche permet l'explicabilité des solutions. Il serait facile à un observateur extérieur de vérifier les coalitions formées en s'assurant que les préférences de tous les individus sont respectées.

2.3.4 Théorie des jeux coopératifs

Dans la théorie des jeux coopératifs, un jeu hédonique est un jeu qui modélise la formation de coalitions dans lequel les joueurs ont des préférences sur les groupes auxquels ils appartiennent. Les coalitions sont des groupes disjoints et la fonction de valeur utilisée pour déterminer le score d'une coalition est dite super-additive. Cela signifie que l'union de deux coalitions disjointes est toujours supérieure ou égale à la somme des valeurs des coalitions séparées. Une autre spécificité des jeux hédoniques est que le niveau de satisfaction des joueurs dépend des membres de la coalition dans laquelle ils se trouvent. En outre, les joueurs ne portent pas attention à la structure des coalitions autres que la leur. L'objectif des joueurs est de former des coalitions stables, c'est-à-dire dans lesquelles aucun des joueurs ne préfère un autre joueur à ceux présents dans sa propre coalition, en fonction des préférences individuelles de chacun. Cet équilibre est appelé équilibre de Nash et défini comme un état stable dans lequel les joueurs n'ont aucun intérêt à changer leur stratégie. Cependant, certaines situations font qu'il est impossible de trouver une configuration stable des coalitions. Par exemple, dans un système composé de 3 agents, A, B et C. Si A souhaite être avec B, B avec C, et C avec A et que les joueurs ne souhaitent pas être tous ensemble dans la même coalition, aucune coalition ne peut être stable. C'est pour cette raison qu'une autre forme de stabilité, plus faible, a été réalisée. Dans cette définition de la stabilité on ignore la préférence d'un membre de la coalition (pour des membres extérieurs au groupe) si les autres membres ne souhaitent pas son départ de la coalition. Par exemple, le partitionnement en coalitions $\{\{A\},\{B,C\}\}$ est dit contractuellement et individuellement stable car C voudrait rejoindre A mais que B pose un veto sur son départ.

Dans les jeux coopératifs classiques les joueurs forment le plus souvent des coalitions disjointes en coopérant uniquement avec les membres de leur coalition. Dans [Saad2010], les auteurs proposent un algorithme d'allocation de tâches pour résoudre un problème dans lequel des agents doivent collecter des données auprès de services. Ces services peuvent par exemple fonctionner sur des appareils embarqués. Chaque service récupère des informations sur son milieu et attend qu'un agent vienne les récupérer pour les transmettre à un serveur. L'objectif est de former des coalitions d'agents et de services afin de collecter leurs données en fonction des préférences des joueurs. Les agents peuvent servir de relais pour assurer la transmission des données ou de collecteurs qui se déplacent de service en service pour récupérer les informations. Les joueurs (agents et services) forment des coalitions en fonction de leurs préférences pour les autres agents ou services. L'algorithme de formation de coalitions crée un partitionnement initial. À partir de celui-ci,

les joueurs peuvent quitter et rejoindre d'autres coalitions en fonction de leurs préférences jusqu'à ce que le partitionnement converge vers un équilibre de Nash. Afin de gérer des changements dans l'environnement comme le déploiement d'un nouveau service ou la suppression d'un ancien, l'algorithme recalcule à partir de zéro une nouvelle structure de coalitions régulièrement.

Dans certaines situations les joueurs peuvent avoir besoin de participer à plusieurs coalitions [Bennis2013]. [Wang2016] se concentre sur la formation de coalitions avec chevauchement. Ils présentent une formalisation pour modéliser et analyser des communications dans un scénario d'allocation de ressources radio. Dans le cas où des agents peuvent participer à plusieurs coalitions, ceux-ci n'ont plus besoin de quitter leur ancien groupe pour en rejoindre un nouveau. La notion de stabilité doit donc être adaptée. Pour cette raison, les auteurs proposent une nouvelle notion de la stabilité dans laquelle les joueurs quittant une coalition peuvent garder leur récompense si les membres restants ne la quittent pas non plus. La résolution du problème se fait en essayant de maximiser le bien-être des joueurs. Les coalitions étant super-additives, les joueurs ont un plus grand bien-être à se regrouper en coalitions. Ce dernier est défini comme la somme des bénéfices des coalitions auxquelles un joueur participe. L'algorithme des auteurs calcule pour chaque structure de coalitions stables la valeur maximale que les joueurs peuvent générer. La complexité de cet algorithme est $O(N^s)$, S étant le nombre de mouvements possibles que les joueurs peuvent faire (sortir ou rentrer dans une coalition) et N le nombre de joueurs.

Les travaux de formation de coalitions dans les jeux coopératifs se concentrent sur la recherche d'un équilibre lors de la création des coalitions. L'équilibre est atteint lorsque les préférences des membres des coalitions sont satisfaites. L'approche des jeux coopératifs présente des similitudes avec celle du choix social, les deux essaient de trouver un partitionnement stable de coalitions en fonction des préférences de chaque individu. Pourtant les problématiques de formation de coalitions sont différentes. Celle du choix social essaie d'associer des individus à des coalitions tandis que celle des jeux coopératifs essaie de former des coalitions en regroupant les individus entre eux. Les préférences ne sont pas d'un individu à un groupe mais entre les individus. Cependant, l'approche des jeux coopératifs a le même inconvénient que le choix social, elle est peu adaptée à un cas applicatif changeant. Bien que les auteurs de [Saad2010] prennent en considération la possibilité d'un environnement ouvert, leurs agents ne se réorganisent pas depuis la dernière structure de coalitions trouvée. Le système recherche les coalitions comme s'il le faisait pour la première fois ce qui pourrait être inadapté à un passage à l'échelle.

2.3.5 Théorie des enchères

En théorie des enchères le problème le plus proche de celui de la formation de coalition est celui de la détermination du gagnant. Ce problème se pose dans de nombreux types d'enchères comme les enchères séquentielles, parallèles ou combinatoires. Les enchères séquentielles sont des enchères dans lesquelles les enchérisseurs proposent des prix pour l'achat d'un produit et les produits sont proposés à la vente les uns après les autres. Il suffit de chercher le prix le plus important sur chaque produit pour en déterminer le gagnant. Les enchères parallèles ont lieu simultanément sur plusieurs produits. Enfin, les enchères combinatoires se produisent sur des combinaisons de produits que les enchérisseurs choisissent. C'est sur ce dernier type d'enchère que le problème de la détermination du gagnant se rapproche le plus d'un problème de formation de coalitions.

Dans ce cas la résolution du problème est plus complexe puisque l'objectif est de maximiser les profits du commissaire-priseur tout en faisant en sorte que tous les produits soient vendus à un unique enchérisseur.

[Vig2006] présente RACHNA, une architecture destinée à résoudre le problème d'affectation de tâches dans un système multi-robot. La méthode se base sur un algorithme modifié, originellement utilisé pour le problème de détermination du gagnant dans les enchères combinatoires à multiples unités. La multiplicité des unités est une variante des enchères combinatoires où les produits vendus peuvent être présents en plusieurs exemplaires. L'architecture multi-robots comporte deux types d'agents, l'agent service et l'agent tâche. Les agents tâches possèdent un score d'utilité. Ils enchérissent sur les agents services afin de se les voir affecter. Une tâche complexe peut nécessiter plusieurs agents *services* pour être réalisée. Par exemple, pousser un objet nécessiterait un service de localisation de l'objet et un service pour le pousser. La méthode a pour but d'attribuer des services à des tâches en maximisant l'utilité globale des affectations pour le système multi-robot.

Takalloo *et al.* proposent un algorithme de partage de la possession de voitures autonomes basé sur les enchères combinatoires [Takalloo2020]. Le problème traité est similaire à celui de [Vig2006]. Cependant dans ce travail il est impossible aux enchérisseurs d'acheter plusieurs fois le même produit. Dans le travail de Takallo *et al.*, les produits achetables sont des plages horaires que les propriétaires d'une voiture autonome se partagent. L'objectif de l'algorithme de résolution du problème de la détermination du gagnant est de partager les horaires entre les propriétaires en maximisant leur bien être. Un conflit apparaît lorsque les propriétaires essaient de réserver les même plages horaires. Dans ce cas, il est résolu à l'aide d'un logiciel résolveur de problèmes d'optimisation linéaire en nombres entiers.

Kayal *et al.* développent une méthode d'approximation inspirée des enchères combinatoires dans le cadre de l'informatique géodistribuée [Kayal2019]. Cette branche de l'informatique consiste à exploiter des applications et des infrastructures de traitement. Ce travail porte sur le problème d'affectation de tâches dans lequel il faut attribuer des applications à des nœuds de calculs. Les nœuds possèdent des capacités limitées de calcul les empêchant de faire fonctionner toutes les applications. De plus, les applications peuvent interagir entre elles en s'envoyant différents volumes de messages. Si elles sont exécutées par le même noeud, deux applications peuvent interagir sans utiliser de bande passante. L'objectif de cette méthode est de minimiser la consommation d'énergie et l'utilisation de la bande passante des nœuds de calculs. Leur algorithme, nommé Distributed Fog Service Placement (DFSP), imite le processus d'enchère combinatoire. Les nœuds de calculs enchérissent sur les applications puis celles-ci sont affectées aux nœuds ayant proposé les meilleures enchères.

Contrairement à notre problématique, le problème de formation de coalition traité par la théorie des enchères est un problème d'affectation de tâches. L'objectif de ce problème est d'attribuer un ensemble de tâches à un ensemble d'agents ce qui revient à regrouper des tâches en un nombre prédéfini de coalitions que sont les agents. Notre problématique de formation de coalition consiste à générer un nombre indéfini de coalitions dont la principale préoccupation est la synergie de leurs membres. Au même titre que le choix social, le problème soulevé par la théorie des enchères se tourne plus vers la satisfaction des contraintes imposées par les coalitions (que sont les enchérisseurs) que par cette synergie inter-membres. En outre, ces méthodes se montrent inadaptées à

gérer un système ouvert et dynamique. Par exemple, on ne pourrait ajouter un nouvel enchérisseur à la fin d'une enchère une fois que tous les produits ont été assignés. Un produit ne pourrait pas non plus être modifié durant une enchère sans devoir annuler toutes les enchères précédentes.

2.3.6 Programmation dynamique

Les algorithmes de programmation dynamique sont des algorithmes créant des coalitions optimales. Le problème de formation de coalitions est traité par les CSP (Complete Set Partitioning) dans lesquels l'objectif est de trouver, à partir d'un ensemble d'entités fini, un ensemble de coalitions dont la somme des scores est maximisée.

Le premier algorithme de programmation dynamique pour un problème de partitionnement d'un ensemble a été proposé par [Yeh1986]. L'objectif de ce problème est d'agréger des entités de manière à ce que chacune d'entre elles soit exactement dans un groupe tout en minimisant le coût des agrégations. On peut visualiser l'ensemble des solutions à ce problème comme un graphe dans lequel chaque nœud est un ensemble de coalitions possédant une valeur et chaque liaison indique que l'on peut passer d'un nœud à l'autre en effectuant uniquement une modification (fusion ou séparation d'une coalition par exemple) sur les solutions liées. L'algorithme de Yeh, nommé DP, évalue toutes les divisions de coalitions possibles afin de déterminer quelles coalitions valent le coût d'être séparées. Par exemple, si des entités $a = 10$, $b = 20$ et que leur agrégation est $\{a,b\} = 40$, alors la séparation de a et b n'est pas intéressante. L'algorithme se déplace ensuite de nœud en nœud dans le graphe en partant du nœud qui contient une unique coalition contenant toutes les entités. Le choix du prochain nœud se fait en choisissant les meilleures transitions possibles jusqu'à trouver la solution optimale. Cet algorithme a une complexité $O(3^n)$, n étant le nombre d'entités à traiter, ce qui le rend inutilisable dans une application à grande échelle. De plus, la superposition des coalitions ne rentre pas dans le cadre de ce travail. Si elle était prise en compte, elle augmenterait significativement l'espace de recherche et rendrait cet algorithme encore plus difficilement exploitable.

Michalak *et al.* [Michalak2010] proposent un algorithme distribué exact et *anytime* appelé D-IP pour la résolution du problème de génération de structures de coalitions. Ce travail est réalisé en combinant deux algorithmes de la littérature : IP [Rahwan2009], algorithme exact et *anytime* pour résoudre le CSG calculant les frontières de l'espace de recherche afin de le réduire (*prune*) et limiter sa taille. IP a une complexité plus élevée que celle de DP, $O(n^n)$, mais présente tout de même un temps moyen de résolution du problème de partitionnement plus court grâce à sa meilleure capacité à réduire les espaces de recherche infructueux. DCVC [Rahwan2007] est un algorithme exact et *anytime* distribuant le calcul des coalitions sur les agents du système. D-IP intègre le mécanisme de réduction de l'espace de recherche de IP dans DCVC afin de le rendre distribué. L'algorithme montre une réduction du temps de traitement du problème de formation de coalitions par rapport à IP mais ne permet pas l'augmentation de l'échelle des systèmes. Malgré une augmentation de la vitesse de traitement, D-IP a besoin de plusieurs heures pour résoudre le problème de GSC de manière optimale avec 28 agents. L'avantage que D-IP a par rapport à d'autres algorithmes de programmation dynamique est sa capacité à être anytime et trouver des solutions, même non optimales, en un temps raisonnable.

Michalak *et al.* [Michalak2016] présentent un algorithme exact et anytime appelé ODP-IP

pour la résolution du problème de CSG. ODP-IP est l'algorithme exact le plus rapide à ce jour pour résoudre un problème CSG. Il s'appuie sur les algorithmes DP et IP. ODP-IP développe une nouvelle représentation de l'espace de recherche permettant à DP et IP de fonctionner ensemble. Une nouvelle version de DP est créée, appellée IDP, capable de chercher des solutions dans un espace de recherche dans un graphe de partitions en nombres entiers. IP est modifié pour accélérer la recherche de solutions et pour pouvoir chercher dans plusieurs sous-espaces de recherche simultanément. Le mélange de ces deux algorithmes permet à ODP-IP de possèder les propriétés d'IDP et d'IP, il est anytime, possède la même complexité que DP et est en moyenne aussi rapide que le plus rapide des deux algorithmes. ODP-IP a donc une complexité de $O(3^n)$, n étant le nombre d'agents à traiter. Le temps d'exécution d'ODP-IP est comparé à celui d'ODP et celui d'IP sur plusieurs distributions de valeurs avec un nombre d'agents allant jusqu'à 25. Quel que soit la distribution et le nombre d'agents, ODP-IP trouve une solution optimale en un temps inférieur aux deux autres algorithmes. De plus, ODP-IP trouve souvent des solutions de très bonne qualité (environ 90% de l'optimal) dans les 10 premiers pourcents de son temps d'exécution total sur une expérience de 25 agents. Cependant, Rahwan *et al.* n'évaluent pas la qualité des coalitions en fonction du temps d'exécution sur des systèmes de différentes tailles. Il semble probable que sur une expérience contenant plus d'agents, ODP-IP soit dans un espace de recherche tellement vaste que même trouver des solutions proches de l'optimal devienne une tâche chronophage. Enfin, pour Changder *et al.* [Changder2020], une des principales limites d'ODP-IP est son parcours de l'espace de recherche. Les auteurs montrent que dans de nombreux cas, ODP-IP cherche des solutions deux fois dans le même espace à cause des méthodes très différentes de fonctionnement sur lesquelles se basent IDP et IP.

Changder *et al.* [Changder2020] proposent l'algorithme hybride, nommé ODSS, pour la résolution du problème de CSG. ODSS s'inspire d'ODP-IP et essaie de limiter le chevauchement produit lors du parcours de l'espace de recherche. Pour cela, ODSS divise l'espace de recherche en deux sous-espaces disjoints dans lesquels IDP et IP vont travailler indépendamment. Lorsque l'un des deux espaces de recherche a entièrement été parcouru, l'algorithme qui lui a été associé va aider l'autre à terminer le parcours de son espace. ODSS s'arrête et retourne la solution optimale au moment où les deux espaces de recherches ont entièrement été parcourus. Puisque ODSS se base sur les algorithmes IDP et IP, sa complexité dans le pire cas est le minimum entre les complexités de ces deux algorithmes, à savoir $O(3^n)$. L'objectif de ce travail étant de combler les limitations d'ODP-IP, les évaluations d'ODSS comparent les vitesses d'exécution de ces deux algorithmes en utilisant différentes distributions de valeurs associées aux coalitions. Les évaluations montrent qu'ODSS est plus rapide qu'ODP-IP sur de nombreuses distributions. Ce résultat peut s'expliquer par la suppression du chevauchement des espaces de recherche associés à IDP et IP et par une meilleure efficacité de leur algorithme de réduction de l'espace de recherche. Cependant ODP-IP reste légèrement plus rapide sur certaines distributions de valeurs. Bien que les auteurs ne commentent pas ce résultat, il se pourrait qu'il soit dû à l'algorithme de division de l'espace de recherche prenant du temps à disjoindre deux sous-espaces qui le sont plus ou moins déjà.

Bien que cette approche fournisse des résultats optimaux, elle supporte très mal le passage à l'échelle. De plus, elle ne répond pas au besoin d'explicabilité du système d'aide à la décision. Un observateur extérieur pourrait difficilement vérifier l'optimalité de la structure de coalitions choisie par l'algorithme de programmation dynamique. Pour cela, il serait obligé de parcourir par

lui-même l'arbre de recherche et s'assurer qu'aucune autre structure ne possède une meilleure valeur.

2.3.7 Synthèse

La majorité de ces méthodes cherchent à trouver une solution optimale à leur version du problème de formation de coalitions dans des contextes statiques où les données utilisées sont fixes durant la résolution. Pour la plupart d'entre elles, un changement des données du problème nécessiterait que leur méthode recalcule entièrement les structures de coalitions. Elles sont très peu dynamiques et ne supportent pas l'ouverture. Par exemple, le changement de valeur d'une seule coalition dans les méthodes de programmation dynamique entraînerait la modification de l'espace de recherche et obligerait l'algorithme à chercher la solution optimale dans un tout nouvel arbre. Le temps d'exécution de ces algorithmes étant exponentiel sur le nombre d'agents, cette méthode n'est pas adaptée à un système d'aide à la décision variable et fonctionnant à grande échelle.

La théorie des jeux, le choix social et la théorie des enchères présentent le même problème : une seule modification de l'entrée remet en question toutes les coalitions formées par leurs méthodes. En outre, les problématiques de formation de coalitions de la théorie des enchères et du choix social diffèrent de celles que nous avons défini. En effet, les problèmes de détermination du gagnant et de la recherche de mariages stables ressemblent plus à un problème d'affectation de tâches que de génération de structures de coalitions. La principale différence étant que les coalitions du CSG doivent se construire autour des besoins des autres membres plutôt qu'autour des préférences définies par les groupes. Au contraire, la problématique de CSG est proche de celle de la ségrégation dans les travaux d'intelligence distribuée. De plus, cette approche répond aux besoins de dynamicité du système d'aide à la décision. En effet, la modification du type d'un agent ne remettrait pas en question la structure de toutes les coalitions. L'agent en question s'éloignerait étape par étape de son ancienne coalition jusqu'à trouver la nouvelle. Cependant, comme écrit dans la partie sur l'intelligence distribuée, les agents sont vus comme des données. Au même titre que pour une grande partie des approches observées la notion d'agent est très proche de celle de la donnée. Dans ces disciplines un agent est presque uniquement vu comme un identifiant auquel est associé une valeur lui permettant de chercher une coalition minimisant ou maximisant un score. Ce manque d'exprimabilité rend difficile la modélisation d'un objet plus complexe. Notre système d'aide à la décision doit former des groupes de composants décrivables par des caractéristiques physiques (*e.g.* poids, forme, prix, *etc.*) mais aussi par des règles fonctionnelles (*e.g.* quels sont les composants assemblables ou quelles sont les transformations applicables à un composant). De plus, le besoin d'explicabilité du système d'aide à la décision est rarement comblé par ces approches.

Le choix social, le clustering et les jeux coopératifs sont les trois approches les plus transparentes de cette revue de littérature. Contrairement aux autres, la construction des coalitions se fait à partir d'un critère (*e.g.* la distance et la préférence) permettant de comprendre le choix des membres d'une coalition. D'autres méthodes comme la programmation dynamique et la théorie des enchères modélisent un état global du système et essayent d'y trouver les coalitions optimales en parcourant l'espace de recherche. Il est alors compliqué de déterminer pourquoi une solution est meilleure qu'une autre sans retracer le chemin dans l'espace de recherche ce qui rend le processus

de formation de coalitions beaucoup plus difficile à comprendre.

2.4 L'approche multi-agents

La formation de coalitions peut être vu comme un sous-domaine de l'auto-organisation dans les systèmes multi-agents. C'est une problématique très étudiée de la discipline [Di Marzo Serugendo2005]. Un système multi-agent est un système composé d'agents. Selon l'approche du problème à traiter un agent peut adopter plusieurs formes. Une première définition est celle d'un script capable d'exécuter des actions sur son environnement et de communiquer avec d'autres agents. Elle est qualifiée de *faible* par Jacques Ferber dans [Ferber1997]. La seconde, qualifiée de *forte* par Ferber, est celle d'une entité plus abstraite capable de communiquer avec d'autres agents, possédant des ressources et un objectif propre, une représentation partielle des autres agents du système et un comportement visant à satisfaire ses objectifs. Les agents adoptant la définition qualifiée de *forte* sont dit cognitifs. Le chapitre 4 étudie plus en détail ce type d'agents. Bien que d'autres définitions existent et sont utilisées par les autres disciplines, les travaux du domaine multi-agent se concentrent sur celles-ci. Ces travaux se concentrent souvent sur l'élaboration de méthodes plus dynamiques et plus robustes pour la résolution de problèmes d'optimisation. Elles parviennent souvent mieux à gérer l'incertitude liée à l'évolution des données d'un problème que les méthodes de programmation dynamique, de théorie des jeux ou de la théorie des enchères. En contrepartie leur capacité à trouver des solutions optimales sur des problèmes dont les données sont constantes est souvent moins importante que dans ces disciplines.

Dans [Aknine2004], Aknine *et al.* présentent deux méthodes pour former des coalitions. Dans ce travail, les agents sont des agents *faibles* selon la définition de Jacques Ferber. Ils ont la capacité de communiquer avec les autres agents du système mais sont considérés comme des nœuds de calculs plutôt que des entités disposant de leur propre comportement et objectif. Les auteurs présentent un mécanisme de formation de coalitions basé sur l'agrégation de critères de préférences que les agents ont les uns pour les autres. Ces agrégations sont construites avec une intégrale de Choquet et sont appelées des modèles de préférences. Les modèles de préférences peuvent aussi être associés à un agent et concerner un groupe d'agents, ou être associés à un groupe d'agents et concerner un agent. Les modèles de préférences construits sont ensuite utilisés pour trouver quels agents peuvent former des coalitions ensemble. De nombreuses méthodes de formation de coalitions ne garantissent pas la génération de solutions. L'objectif de Aknine *et al.* dans leur travail est de concevoir une méthode de formation de coalitions robuste garantissant la formation de groupes. Pour cela, les agents peuvent négocier via un protocole de communication et changer les critères pris en compte pour former des coalitions lorsque les critères utilisés ne le permettent pas. Une seconde méthode de formation de coalitions est présentée dans le cadre des SMA coopératifs dans laquelle les agents peuvent se partager leur modèle de préférences et coordonner leur activités. L'évaluation montre que leurs méthodes arrivent à former des coalitions dans la plupart des cas sur des expériences allant de 50 à 200 agents. Bien que leur objectif soit atteint dans le

cadre des SMA coopératifs, la qualité des coalitions formées n'est pas évaluée. De plus, les auteurs choisissent de ne faire fonctionner leur méthode qu'avec 7 critères de préférences pour des raisons de temps de calcul ce qui laisse penser que le nombre de critères à un impact important sur la capacité de la méthode à trouver des solutions en un temps raisonnable. Le temps d'exécution en fonction du nombre de critères n'est pas non plus évalué. Ce travail ne se place pas dans un contexte dynamique. Le système multi-agent est fermé et non modifiable.

Ramos *et al.* [Ramos2013] présentent la méthode SACF permettant de construire des coalitions d'agents dans un système ouvert. Dans ce travail, les auteurs utilisent des agents cognitifs. Ceux-ci possèdent une représentation de leur environnement et un comportement tendant à satisfaire leur objectif. SACF est une méthode dynamique basée sur une heuristique centrée agent appliquée à un scénario de smart-grid. Dans ce scénario, les agents doivent former des coalitions afin de fournir de l'énergie à un réseau électrique. Les agents interagissent uniquement avec le voisinage proche permettant d'éviter une explosion combinatoire liée aux problèmes de CSG. Tous les agents étant dans un périmètre proche les uns des autres reconnaissent la présence des autres agents mais aussi la coalition à laquelle ils appartiennent ou non. Pour rejoindre ou créer une coalition, les agents utilisent un protocole de communication. La décision finale est prise grâce à une fonction de valeur permettant d'affecter une valeur à une coalition en fonction de ses membres. Si la coalition à une plus grande valeur avec que sans un agent, l'agent est ajouté à la coalition. Cependant il n'existe aucun moyen pour un agent de quitter une coalition sans quitter le système. Les auteurs évaluent leur méthode jusqu'à 70 agents et montrent qu'elle est robuste à l'entrée et la sortie de nouveaux agents. De plus, leur méthode forme des coalitions de qualité très proche de l'optimal dans les expériences à petite échelle. Bien que leur approche donne de bons résultats, la dynamicité de leur système est limitée. En effet, l'énergie que les agents peuvent apporter au réseau ne peut pas varier pendant l'expérience, les coalitions n'ont donc pas besoin de se réorganiser. En outre, les auteurs évaluent la charge en communication induite par SACF en se concentrant sur le protocole de formation de coalitions et en omettant l'échange de nombreuses informations liées aux caractéristiques des agents : leur position, leur quantité d'énergie disponible, les coalitions auxquelles ils appartiennent, *etc.*

Dans [Ye2013], les auteurs proposent une méthode de formation de coalitions inspirée de SACF capable de gérer un environnement changeant dans lequel des agents cognitifs peuvent modifier leur implication pour leurs coalitions à tout moment. Les agents interagissent dans un réseau de voisinage dans lequel les coalitions sont chevauchantes. Ils peuvent quitter leurs coalitions, en rejoindre une autre ou simplement modifier leur degré d'implication dans leurs coalitions en fonction des autres membres. Pour cela les agents utilisent un protocole de communication utilisé pour négocier la formation des coalitions. Ils peuvent choisir de créer ou de rejoindre une coalition en fonction de leur gain à participer à la coalition. Plus l'implication de l'agent dans une coalition est faible, plus le coût à y rester est élevé. En entrant dans une coalition, un agent passe un contrat pour exécuter une tâche avec son groupe. Lorsque l'implication d'un agent pour une coalition devient trop faible, celui-ci peut briser son contrat et quitter ses coalitions pour en rejoindre d'autres.

Plus récemment dans [Houhamdi2020], Houhmamdi et Athamena présentent le modèle Collaborative Team Construction Model (CTCM). Ce modèle a pour but de créer des équipes à partir d'un ensemble d'agents. Pour cela ils utilisent une technique de raisonnement social permettant aux agents de collecter des informations sur leur voisinage pour définir un objectif et construire

des équipes. Les agents utilisent plusieurs zones de voisinage afin d'augmenter le nombre d'agents avec lesquels interagir. Ce compromis permet de laisser plus de choix aux agents pour créer des équipes tout en limitant le nombre de combinaisons possibles. La spécificité de ce modèle est d'être conçu pour un système ouvert. Les agents peuvent entrer dans un voisinage ou le quitter à tout moment. La construction d'une équipe se base sur l'évaluation de la relation entre les agents. En fonction de l'objectif des agents elle peut être collaborative s'ils partagent le même objectif, conflictuelle si leur objectif est opposé, ou complémentaire si leur objectif n'est ni l'un ni l'autre.

Les travaux du multi-agent se montrent adaptés à gérer un environnement dynamique et ouvert. L'approche étant centrée agent, elle possède une bonne capacité de modélisation des composants de notre système. De plus, elle permet de comprendre les choix de chaque agent dans le processus de formation de coalitions. Par exemple, dans [Ye2013] et [Ramos2013] les agents forment des coalitions en fonction du gain que celles-ci leur apporte. Cette transparence dans la prise de décision des individus rend possible la compréhension par un observateur extérieur des raisons qui ont poussées les agents à former une coalition.

2.5 Discussion

Approches / Propriétés	Exprimabilité	Dynamicité	Ouverture	Explicabilité
Clustering				✓
Intelligence distribuée		✓	✓	
Choix social informatique				✓
Théorie des jeux				✓
Théorie des enchères				
Programmation dynamique				
Système multi-agent	✓	✓	✓	✓

Tableau 2.1 – Synthèse de l'adéquation des principales approches du problème de formation de coalitions aux besoins du système d'aide à la décision. Le motif ✓ signifie qu'une approche répond à un besoin.

La dynamicité du système dépend du comportement intrinsèque des composants et de leur capacité à évoluer avec les données. Il est alors impossible d'identifier un état global du système et d'en déduire un modèle global de résolution. Le multi-agent est une approche adaptée aux problèmes nécessitant de la dynamicité. Contrairement à la majorité des approches utilisées pour le problème de formation de coalitions, elle est l'une des seules à être utilisable dans un cadre dy-

namique dans lequel les données du problème peuvent changer pendant la recherche de solutions. Le système d'aide à la décision doit travailler avec une multitude de composants. Le multi-agent est l'une des seules approches permettant de décomposer de manière adéquate la modélisation du système. En outre, l'utilisation d'agents cognitifs rend possible la modélisation de ces composants sous forme d'entités abstraites. Elle permet au système d'aide à la décision de manipuler de manière générique tout type de composants mais offre aussi une grande capacité de description des composants. En effet, elle peut aussi bien décrire des caractéristiques physiques que fonctionnelles. Par exemple, le physique d'un composant peut être associé à celui d'un agent tandis que ses règles fonctionnelles peuvent être inscrites dans son comportement. De plus, les agents possèdent leur propre comportement et sont capables d'adapter leurs dynamiques en fonction des autres agents du système et de leur objectif individuel. Enfin, contrairement à d'autres méthodes de la littérature, une approche centrée agent rend possible l'explicabilité de leur comportement et des coalitions formées. Par conséquent, l'approche multi-agent utilisée avec des agents cognitifs est privilégiée pour concevoir le système d'aide à la décision. La table 2.1 synthétise l'adéquation qu'ont les approches vues dans cette revue bibliographique aux propriétés nécessaires au système d'aide à la décision.

La dynamique des groupes est une discipline qui étudie comment se forment et se maintiennent les groupes restreints. Le chapitre suivant présente ce domaine et identifie les notions dont nous pourrons nous inspirer pour notre modèle d'agent.

3

Inspiration sociale

Sommaire

Nous voyons les agents du système comme des entités autonomes qui possèdent leur propre objectif, une vision partielle et individuelle de leur environnement, des croyances, des désirs et leur propre comportement. En les définissant de cette manière nous pouvons faire l'analogie entre le système multi-agent et une société d'individus avec laquelle nous souhaitons former des groupes afin de remplir un objectif global. Les sciences humaines et sociales étudient les comportements humains et sont bien placées pour comprendre comment les groupes se construisent. Plus spécifiquement, la *Dynamiques des Groupes* est un domaine qui se concentre sur l'étude de la formation et du maintien des groupes restreints [Forsyth2010]. Un groupe restreint est un groupe à durée limitée dans lequel les individus peuvent entrer ou sortir et qui est souvent créé pour répondre à un objectif. Par exemple, les équipes sportives et les groupes d'études sont des groupes restreints. Ils peuvent se former pour remplir un objectif, jouer un match ou réviser un examen, puis se démantèlent une fois l'objectif réalisé. Cependant, la vie des groupes restreints n'est pas uniquement rythmée par les activités de ses membres mais aussi par leurs interactions sociales. Nous nous intéressons à ce domaine pour essayer d'en extraire les facteurs influant sur la construction des groupes afin de nous en inspirer pour résoudre notre problème de formation de coalitions.

3.1 Théories de la formation des groupes

La dynamique de groupe présente quatre principales théories sur la façon dont se forment les groupes. Ces théories sont très espacées dans le temps puisque les plus vieilles datent des années 50 tandis que la plus récente date de 2013. Bien que ce sujet soit étudié depuis près de 70 ans, la littérature ne privilégie pas une unique théorie pour définir les facteurs de formation de groupes. Comme nous le verrons, elles se complètent et parfois se chevauchent. À la suite de cette partie, nous sélectionnerons les principaux critères permettant à des individus de former des groupes.

3.1.1 La théorie de la propinquité

La propinquité est un facteur de proximité conduisant à l'évolution de l'attraction inter-personnelle, c'est-à-dire l'augmentation ou la diminution de l'appréciation que deux personnes ont l'une pour l'autre. La proximité entre deux individus est originellement proposée comme un facteur favorisant leur attraction réciproque. Certaines études menées à large échelle montrent en effet que les individus proches ont plus tendance à s'apprécier que ceux qui sont éloignés [Segal1974] [Newcomb1960]. Cependant ce résultat est critiquable, il est difficile d'apprécier une personne avec qui on ne peut pas interagir à cause d'une longue distance. Ebbersen *et al.* montrent dans [Ebbesen1976] que la proximité spatiale n'est pas le critère principal favorisant l'attraction, mais que ce sont les interactions entre les individus. Bien qu'il existe une corrélation entre la proximité et la fréquence des interactions, les interactions peuvent être l'élément clef permettant la formation de groupes. Cette hypothèse a été confirmée avec le développement d'internet et des communautés en ligne. Ducheneaut *et al.* étudient les jeux massivement multi-joueurs dans lesquels des individus interagissent fréquemment tout en maintenant une relation longue distance [Ducheneaut2006]. Ils notent que malgré la distance, les nombreuses interactions entre les joueurs leur permettent de

tisser des liens et de former des groupes stables.

3.1.2 La théorie de Homans

La théorie de Homans [Homans1950] sur la formation de groupes est basée sur un modèle composé de trois éléments interconnectés : les interactions, les activités et les émotions. Deux individus s'attirent mutuellement à mesure qu'ils interagissent et partagent des activités. Les relations entre ces éléments induisent un auto-renforcement de l'attraction. Par exemple, plus les individus partagent d'activités, plus ils interagissent et plus ils partagent d'émotions. Et plus ils partagent d'émotions, plus ils sont susceptibles de partager des activités et d'interagir à nouveau. De la même manière que ces facteurs peuvent mener à un cycle de rétroaction positive, la perturbation d'une des trois dimensions est susceptible de perturber les deux autres. Dans cette théorie, les interactions tiennent un rôle primordial permettant aux individus de développer des sentiments les uns pour les autres. Ces sentiments servent ensuite de base à la construction d'un groupe informel. Par exemple, les étudiants d'une même promotion qui s'entraident pour réviser des examens peuvent, lors d'une première session de révision, tous se réunir dans une salle dans le but d'acquérir les connaissances qui leur manquent. Les étudiants interagissent en prodiguant des conseils à leurs camarades ce qui crée des sentiments des étudiants aidés aux étudiants aideurs. En retour, les étudiants aidés peuvent soutenir à leur tour leurs camarades sur d'autres sujets qu'ils maîtrisent mieux. Ces interactions ont lieu dans le cadre d'une activité de groupe et augmentent les sentiments que les étudiants ont les uns pour les autres jusqu'à la création de sous-groupes informels au sein de la promotion. Groupes qui pourront se réunir à l'avenir pour de nouvelles activités (révisions ou non) augmentant ainsi le nombre d'interactions entre les membres.

3.1.3 La théorie de l'échange social

Skinner a travaillé sur l'aspect comportemental des individus [Skinner1953]. D'après lui, les processus psychologiques peuvent façonner leurs réactions. Son travail se base sur deux hypothèses. La première est que les processus psychologiques d'un individu sont trop complexes à évaluer et la seconde qu'il est plus facile d'étudier les comportements induits par ces processus que les processus eux-mêmes. Plus tard, Thibaut et Kelley étendent ces travaux aux groupes et présentent la théorie des échanges sociaux [Thibaut1959]. D'après cette théorie, le ratio bénéfice / coût des interactions entre les individus permet de prédire la formation d'un groupe. Comme dans les travaux de Homans [Homans1974], les individus sont vus comme hédonistes et cherchent à maximiser les gains qu'ils peuvent tirer d'une relation. Par exemple, le bénéfice de la relation d'un doctorant avec ses encadrants est très important puisqu'elle lui permet d'apprendre la méthodologie de recherche en plus d'accéder à un diplôme universitaire. Si le doctorant apprécie son travail, le coût de cette relation est très faible. Au contraire, la relation peut être beaucoup plus coûteuse pour les encadrants lorsque le doctorant leur envoie des documents à relire le vendredi soir. Le principe de maximisation de la récompense et de la minimisation du coût est appelée le *principe minimax*. Finalement, cette théorie ajoute que, les individus ne pouvant maintenir toutes les relations leur étant bénéfiques, ceux-ci privilégient uniquement celles qui leur offrent le meilleur ratio bénéfice / coût.

3.1.4 La théorie de l'équilibre

La théorie de l'équilibre est créée par Fritz Heider [Heider1946] et étendue par Cartwright *et al.* dans [Cartwright1956]. D'après celle-ci, les groupes sont formés en fonction de l'attraction que les individus ont les uns pour les autres ainsi que sur la similarité de leurs attitudes et de leurs valeurs. Les groupes essaient constamment de maintenir un équilibre structurel où les relations entre les membres sont symétriques en terme d'attraction, attitudes et valeurs. Par exemple, deux membres d'un groupe essayent d'accorder leur comportement et l'attraction ressentie l'un pour l'autre doit être réciproque. Un groupe est dit équilibré lorsque toutes les relations inter-membre sont positives, c'est-à-dire que tout le monde s'apprécie. Contre-intuitivement, il est aussi considéré comme équilibré lorsqu'il existe un nombre pair de relations négatives, par exemple si deux membres ne s'apprécient pas. Enfin, un groupe est dit déséquilibré si ses membres ont un nombre impair de relations négatives. Par exemple lorsque l'affection d'un membre n'est pas réciproque. Lorsqu'une relation est déséquilibrée, les deux individus essaient de la rééquilibrer. S'ils n'y parviennent pas, leur relation est dissoute.

3.1.5 Synthèse

Ces quatre théories se chevauchent et se complètent. La théorie de la propinquité utilise originellement la proximité comme facteur de formation de groupes, mais les recherches ont montré que les interactions pouvaient être plus pertinentes pour comprendre comment les groupes se forment. Bien que la théorie de Homans reprenne cet élément dans son modèle et y ajoute les sentiments et les activités, les interactions restent l'élément central et déclencheur de la formation d'un groupe. D'après la théorie de Thibaut et Kelley, les individus recherchent la maximisation de leur profit en priorité. Enfin, d'après la théorie de l'équilibre, les individus essaient de maintenir un équilibre dans leur relation. Celui-ci passe par une symétrie des attitudes et des valeurs mais aussi par une symétrie de l'attraction ressentie pour une personne. En résumé, ces travaux se reposent sur la fréquence des interactions inter-individus ainsi que sur leur gain à maintenir leur relation (le gain sentimental, l'avancement dans une tâche, *etc.*). Les théories de la formation de groupes introduites par Homans, Thibaut, Kelley et Heider utilisent des approches très différentes mais pas nécessairement contradictoires.

La formation de groupes est une combinaison complexe de processus personnels, situationnels et interpersonnels. Les processus interpersonnels sont représentés par le concept de l'attraction que nous détaillons dans la partie suivante. Quant à eux, les processus personnels sont par exemple les traits de personnalité ou les expériences passées de l'individu. Enfin, les processus situationnels peuvent être les besoins émotionnels induits par un ensemble des circonstances dans lesquelles l'individu se trouve. Autrement dit, tout le monde n'est pas prédisposé de la même manière à s'intégrer dans des groupes restreints et tous les moments de la vie d'un individu n'y sont pas propices. Néanmoins, ces deux derniers processus se concentrent sur l'état de l'individu plus que sur ses relations avec ses pairs. Avec l'utilisation de l'approche multi-agent pour la résolution de notre problématique, nous recherchons à faire émerger les coalitions grâce aux interactions inter-agents. C'est pourquoi nous nous concentrons dans la suite de cet état de l'art sur les processus interpersonnels de la formation de groupes et sur l'attraction. La littérature tente d'en identifier les éléments clefs et propose plusieurs *principes de l'attraction*, un ensemble de facteurs permettant

de prédire si deux individus sont prédisposés à être attirés l'un par l'autre.

3.2 Les principes de l'attraction interpersonnelle

L'*attraction interpersonnelle*, aussi appelée dans ce manuscrit *attraction*, est un processus menant à la découverte d'affinités avec autrui. Elle peut être vue comme une métrique de l'amitié. Lorsque l'attraction est faible, les personnes ne s'apprécient pas. Au contraire, lorsqu'elle est élevée, deux individus peuvent aboutir à une relation amicale ou amoureuse. Comme nous l'avons vu dans les théories précédentes. Les individus qui s'apprécient ont plus tendance à former des groupes ensemble. L'attraction peut donc être vue comme une des origines de la formation de groupes. Elle n'est pas uniquement étudiée dans le cadre de la *dynamique des groupes* mais aussi dans les dyades, couples amicaux ou amoureux de deux personnes. Les facteurs de l'attraction sont partiellement issus des différentes théories de la partie précédente. L'ensemble de ces principes permet de déterminer dans quelles conditions des individus ont le plus de chances de s'apprécier.

— **le principe de proximité** : suggère que les individus géographiquement proches ont plus tendances à se regrouper [Newcomb1960]. Ce résultat ressort dans de nombreux travaux [Gieryn2000, Sacerdote2005]. Il peut s'expliquer par deux raisons, premièrement les individus ont tendance à préférer les objets ou les personnes qui leur sont familiers [Bornstein1989] : par exemple les personnes présentes sur leur lieu de travail ou habitant le même quartier. Deuxièmement, comme décrit dans la partie précédente, ce phénomène peut s'expliquer par l'augmentation des possibilités d'interaction lorsque deux personnes vivent ou travaillent dans la même zone géographique.

— **le principe de similarité** : est la tendance à apprécier les individus similaires à soi-même en terme d'attitudes, de valeurs, d'intérêts, de croyances, *etc* [Newcomb1963, Walster1966]. Les groupes sont le plus souvent composés de membres qui partagent des similitudes [Henningsen2013]. Ce sont souvent les membres les plus différents qui quittent le groupe en premier.

— **le principe de complémentarité** : suggère que deux individus dont les qualités se complètent s'attirent [Winch1958]. Ce principe semble être en contradiction avec le principe de similarité. Tout d'abord, la similarité est un facteur plus courant que la complémentarité [Miller2007]. Les individus sont plus souvent avec ceux qui leur sont similaires que complémentaires. Ensuite, la similarité ne joue pas exactement le même rôle. Tandis qu'elle sert de déclencheur dans une relation, la complémentarité sert principalement de consolidateur de l'attraction lorsque les individus se connaissent. Dans ce cas, la complémentarité peut se jouer sur les compétences des individus dans une activité. Par exemple des joueurs de handball peuvent avoir initié leur amitié grâce à leur amour commun pour ce sport et la voir se consolider grâce au talent de leur coéquipier dans un rôle qui n'est pas le leur. Ce principe agit aussi d'un individu vers un groupe lorsque celui-ci remarque l'efficacité de son équipe pour effectuer une tâche [Kristof-Brown2005].

— **le principe de l'élaboration** : suggère que les groupes se forment à partir de deux personnes et qu'ils se développent en acceptant les proches de ces deux membres [Tobin2008]. Par exemple deux colocataires formant une dyade peuvent se présenter leurs amis respectifs. Ainsi, le groupe peut s'agrandir si les proches apprécient l'un des deux colocataires.

— **le principe de réciprocité** : vient de la théorie de l'équilibre. Il suggère que l'attraction tend à être réciproque [Newcomb1979]. Par exemple, lorsqu'une personne reçoit des compliments ou l'admiration d'un individu, celle-ci a tendance a plus l'apprécier. Au contraire, lorsqu'une personne semble ne pas en aimer une autre, cette dernière a tendance à la rejeter en retour.

— **le principe du minimax** : est basé sur la théorie de l'échange social [Henningsen2013]. Comme décrit plus en détail dans la partie précédente, ce principe suggère que les individus sont attirés par ceux qui leur offrent le meilleur ratio bénéfice / coût. Cette évaluation de la relation peut aussi se faire d'un individu à un groupe. Dans ce cas, l'individu évalue les bénéfices et les coûts apportés par le groupe [Moreland1993].

— **le principe de l'attractivité physique** : concerne principalement les dyades. D'après ce principe, les individus physiquement attirants sont plus enclins à provoquer un sentiment d'attraction chez les autres personnes [Henningsen2013]. Cependant l'attractivité physique n'influe sur l'attraction interpersonnelle qu'au début de la relation. Pour que la relation se développe, il est nécessaire qu'au moins l'un des principes précédents agisse sur les individus. L'attractivité physique peut amorcer la formation de groupe via le principe de l'élaboration. Par exemple, deux personnes initialement physiquement attirées l'une par l'autre peuvent former une dyade à partir de laquelle un groupe se développera s'ils se sentent suffisamment similaires ou que leur relation leur apporte un bénéfice mutuel.

Les études de *dynamiques des groupes* tentent de trouver les facteurs expliquant le mieux le processus de formation de groupes. Mais lorsque ceux-ci sont formés, de nouveaux facteurs entrent en jeu. Ils permettent de déterminer les conditions dans lesquelles le groupe peut maintenir son unité. Dans la partie suivante, nous nous intéressons plus particulièrement au concept de cohésion.

3.3 La cohésion de groupe

La cohésion est le concept déterminant permettant de prédire si un groupe peut maintenir son unité ou non. Elle a la faculté d'impacter positivement ou négativement le groupe ; une bonne cohésion a pour conséquence d'améliorer les performances du groupe dans sa tâche et de réduire la paresse sociale de ses membres [Casey-Campbell2009]. Au contraire une mauvaise cohésion peut avoir des conséquences néfastes sur la performance du groupe et les relations des membres menant le groupe à se dissocier. De nombreuses études tentent de la définir et de comprendre les facteurs qui l'impactent.

3.3.1 Définitions

Bien que la cohésion de groupe soit très étudiée dans la littérature, elle reste un concept vaste qui a été défini de nombreuses manières au fil des années. Il n'existe aujourd'hui pas de consensus sur la définition de la cohésion de groupe.

Définition 3.3.1 (L'attraction inter-membres)**.** Lott & Lott définissent la cohésion au sein des groupes restreints par l'attraction interpersonnelle que les membres ont entre eux [Lott1965]. La cohésion du groupe correspond alors à une agrégation de l'attraction que tous les membres ont les uns pour les autres.

Définition 3.3.2 (L'attraction des membres pour le groupe)**.** Nixon suggère que la cohésion de groupe est une agrégation de l'attraction que chaque membre a pour le groupe [Nixon1979]. Dans ce cas, le groupe n'est plus vu à travers les individus qui le compose mais plutôt comme un tout.

Définition 3.3.3 (La morale et l'appartenance)**.** D'après Bollen & Hoyle, la cohésion possède deux dimensions : le sentiment d'appartenance des membres au groupe et leur moral [Bollen1990]. Le moral représente la réponse émotionnelle que provoque l'appartenance à un groupe chez ses membres. Il peut par exemple se mesurer en demandant aux individus leur enthousiasme à faire partie d'un groupe.

Définition 3.3.4 (Les forces sociales maintenant les membres dans le groupe)**.** Festinger définit la cohésion comme la somme des forces qui agissent sur tous les membres et les maintiennent dans le groupe. Ces forces peuvent par exemple être l'attraction que les individus ont pour le groupe, l'attraction qu'ils ont entre eux, le prestige du groupe ou l'activité du groupe [Festinger1950].

Définition 3.3.5 (Le travail d'équipe et la confiance)**.** Siebold définit la cohésion comme la confiance entre les membres du groupes et leur capacité à réaliser des tâches ensemble [Siebold2007].

Bien que la définition de Festinger soit la plus ancienne [Festinger1950], elle se veut plus holistique que les précédentes en englobant nombre de leurs facteurs. En contrepartie elle semble également plus difficile à opérationnaliser de part l'étendue des concepts qu'elle prend en compte. Au contraire, les définitions de Nixon [Nixon1979] et Lott & Lott [Lott1965] sont beaucoup plus restreintes mais également plus évidentes à mesurer.

D'après la *dynamique de groupe*, l'attraction est un des concepts à l'origine de la formation des groupes. Les principes de l'attraction se révèlent alors être une source d'inspiration intéressante pour les systèmes artificiels dans lesquels les agents ont pour objectif de former des coalitions. Par extension, les définitions se prêtant le mieux à la cohésion pour les systèmes artificiels usant de l'attraction comme source d'inspiration sont sans doute celles de Lott & Lott et Nixon qui voient la cohésion de groupe comme une agrégation de l'attraction que les membres ressentent pour leurs accointances ou leur groupe. Cependant, comme l'explique la partie suivante, la cohésion est un concept multi-dimensionnel qui n'est pas uniquement défini par l'attraction.

3.3.2 Les dimensions de la cohésion

Les premiers modèles, fondés sur les définitions de la cohésion de Festinger, Lott & Lott et Nixon, voient la cohésion comme un concept unidimensionnel basé sur les forces que les membres

d'un groupe ont pour leur groupe. Ces forces peuvent être représentées par l'attraction que les membres ont les uns pour les autres ou par la similarité de leurs croyances ou attitudes. Mais le concept de *force* de Festinger n'a pas été opérationnalisé rendant difficile l'évaluation de la cohésion de groupe. Au contraire, les définitions de Nixon et Lott & Lott sont basées sur *l'attraction interpersonnelle* et *l'attraction pour le groupe* qui sont plus facilement opérationnalisables car se référant à une notion beaucoup moins vaste et plus détaillée, notamment via les facteurs de l'attraction vus dans la partie précédente. Cependant ces définitions se basent sur les individus et estompent le concept de groupe et les dynamiques qui lui sont propres. On se rend compte en lisant les définitions de la cohésion que de nombreux facteurs peuvent entrer en jeu pour maintenir l'unité d'un groupe. À partir de 1960 arrivent les modèles de la cohésion de groupe fondés sur plusieurs dimensions. Les modèles décrivant la cohésion de groupe sont tout aussi nombreux que les définitions du concept. On peut par exemple noter celui de Mikalachki présenté dans [Mikalachki1969] et de Carron *et al.* dans [Carron1985] qui se basent sur les dimensions sociales et d'activité de la cohésion. L'étude de Carron *et al.* a été reprise et adaptée plus tard par les français Heuzé et Fontayne dans le but de définir une mesure de la cohésion chez les athlètes francophones [Heuzé2002]. L'étude de la cohésion est réalisée à l'aide d'un questionnaire rempli par les membres d'un groupe. Il collecte des informations à propos des ressentis des individus sur les autres membres et sur leur groupe. L'étude francophone a donc eu besoin de traduire le questionnaire originellement anglophone pour le réutiliser dans des équipes sportives françaises. Puisque ce questionnaire a été traduit et adapté en français, il n'est pas tout à fait similaire à l'original mais les résultats confirment le modèle de Carron et ses collègues. Dans ces modèles multi-dimensionnels le groupe n'est plus uniquement considéré comme un ensemble d'individus mais plutôt comme une entité qui possède un but propre et ses propres dynamiques. La littérature ne montre pas de consensus sur une liste exhaustive des dimensions de la cohésion. Nous présentons donc ici les deux dimensions qui sont le plus souvent relevées par ces travaux, la cohésion opératoire et la cohésion sociale. La cohésion opératoire représente le degré d'implication des membres du groupe dans le travail d'équipe afin de réaliser des objectifs communs. Cette dimension est liée à la performance du groupe face à la tâche qu'il doit réaliser. Plus la cohésion opératoire est forte, plus le groupe performe dans son activité.

La *cohésion opératoire* ressentie par les membres du groupe s'attache à la fois aux autres membres (*e.g.* si un individu apprécie le travail d'un autre membre de l'équipe), mais aussi au groupe vu comme un tout (*e.g.* si la manière dont le groupe réalise une tâche est convenable pour un membre) [Back1951]. La *cohésion sociale* représente les forces sociales qui maintiennent l'unité du groupe. La cohésion sociale est à l'origine du groupe et de sa capacité à réaliser une tâche. En effet, les membres ont besoin d'un minimum de socialisation avant de commencer à travailler ensemble. De la même manière que pour la cohésion opératoire, la cohésion sociale se décline en deux sous-parties. La première représente l'attraction interpersonnelle que les membres du groupe ont les uns pour les autres. L'attraction interpersonnelle se fonde dans ce cas sur les facteurs vus dans la partie précédente. Par exemple, un membre A peut apprécier un membre B pour ses valeurs morales. La seconde représente l'attraction qu'un individu a pour son groupe. Par exemple, un membre A peut apprécier son groupe pour la bonne humeur qui s'en dégage.

La cohésion opératoire et la cohésion sociale ont un impact l'une sur l'autre. Si l'une devient trop forte, elle impacte négativement l'autre. Par exemple, une forte cohésion opératoire entraîne

une bonne réalisation des tâches mais aussi de la pression sur les membres du groupe rendant leurs relations plus compliquées et réduisant leur cohésion sociale. Au contraire, une forte cohésion sociale peut avoir un impact négatif sur la cohésion opératoire. Les membres peuvent hésiter à faire remarquer les erreurs de leurs coéquipiers ce qui réduit les performances du groupe [Rovio2009]. Les études sur les athlètes montrent que l'optimalité se trouve dans une cohésion opératoire et sociale modérée [Hardy2005].

La figure 3.1 représente la cohésion telle que présentée par *Carron et al.* dans [Carron2003]. Nous retrouvons dans ce schéma les deux types de cohésion introduits par *Lott & Lott* [Lott1965] et *Back* [Back1951] ; la cohésion sociale et la cohésion opératoire. Les deux se mélangeant aux dimensions de l'attraction individuelle et de l'intégration du groupe permettent de définir quatre dimensions de la cohésion de groupe :

— *l'attraction individuelle sociale* : représente la satisfaction des attentes que le groupe apporte aux individus du point de vue social (interactions avec les autres, productivité, objectifs, *etc.*). Cette dimension représente l'ensemble des sentiments individuels à l'égard du groupe [McLeod2013] ;

— *l'attraction individuelle opératoire* : représente la satisfaction des attentes que le groupe apporte aux individus du point de vue opératoire (productivité du groupe, atteinte des objectifs, *etc.*). Comme pour la dimension précédente, cette dimension représente l'ensemble des sentiments individuels à l'égard du groupe ;

— *l'intégration sociale du groupe* : représente la façon dont les individus perçoivent le groupe du point de vue social (*e.g.* la similarité inter-membre, leur proximité, la perception du degré d'unité du groupe *etc.*) [Buton2006] ;

— *l'intégration opératoire du groupe* : représente la façon dont les individus perçoivent le groupe du point de vue opératoire (*e.g.* la satisfaction des activités menées en groupe) [McLeod2013].

3.3.3 Synthèse

Cette partie a présenté les principales définitions et modélisations de la cohésion de groupe. Celles-ci ont le plus souvent été proposées avant les années 2000 et pourtant aucune de ces définitions ni de ces modélisations ne font encore consensus. Certain travaux ont pourtant tenté de mieux définir les dimensions de la cohésion [Dion2000] mais face à la difficulté de contrôler expérimentalement le poids relatif des différentes forces, les recherches ont tendance à utiliser des définitions unidimensionnelles comme l'attraction individuelle ou les forces opératoires délaissant alors le statut du groupe [McLeod2013].

Dans la partie suivante, nous aborderons l'aspect social dans les systèmes multi-agents et verrons la manière dont les systèmes artificiels décentralisés s'inspirent de la sociologie.

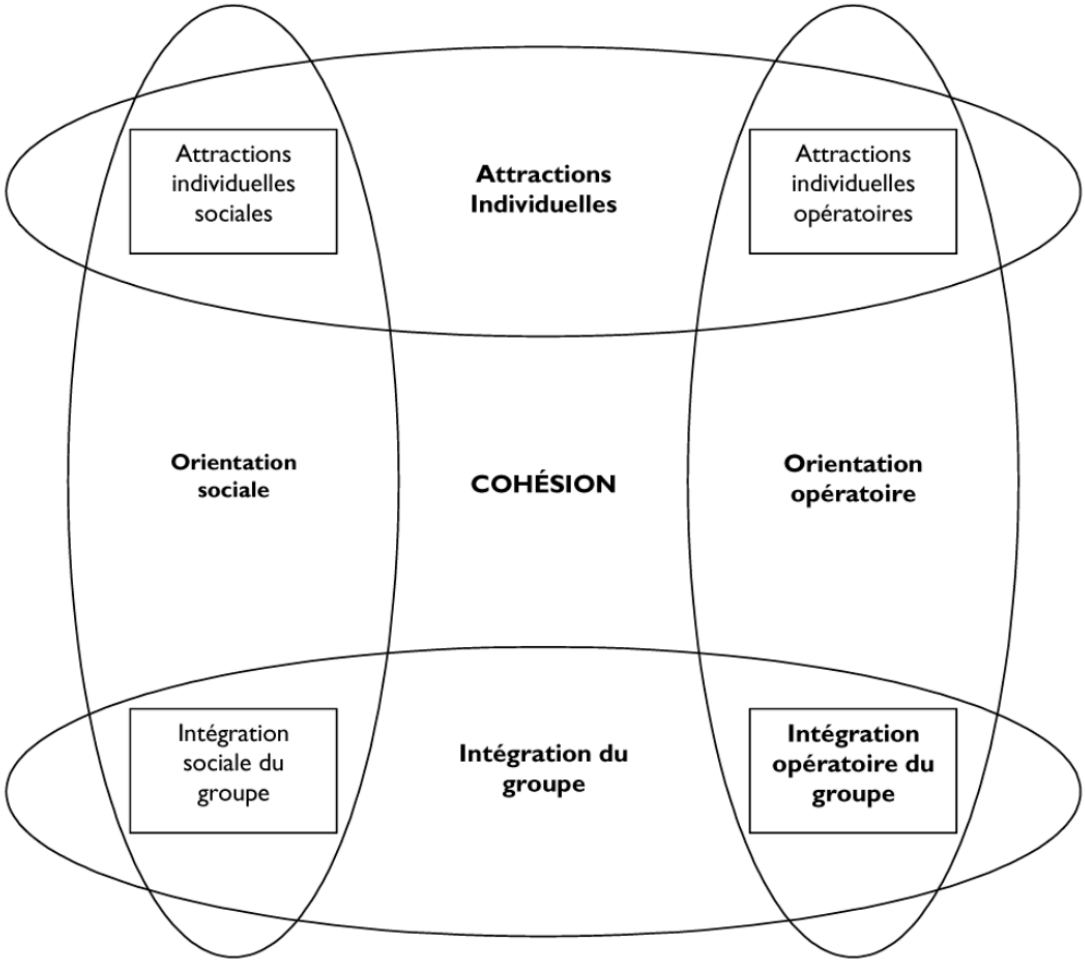


Figure 3.1 – Représentation schématique de la cohésion selon Carron et al. [Carron2003]

3.4 Le multi-agent et le social

Il existe peu de travaux s'inspirant des comportements sociaux humains où l'intelligence n'est pas uniquement la propriété d'un individu mais émerge de leurs interactions. La théorie du choix social s'en rapproche en essayant d'analyser comment les combinaisons d'opinions individuelles peuvent amener à une décision unique sur le plan collectif. Cependant cette théorie se concentre sur des théories économiques, mathématiques et politiques plus que sur l'humain en lui-même comme le font les sciences humaines et sociales.

Les comportements sociaux ne sont pas propres à l'être humain. Il sont par exemple observables chez certaines espèces d'oiseaux migrateurs qui volent en groupe. Un comportement social désigne toutes les interactions inter-individus qui peuvent aider à la résolution d'un objectif. L'intelligence artificielle s'en inspire largement pour résoudre ses problématiques [Bettinelli2020a]. Certain algorithmes d'optimisation s'inspirent par exemple des capacités d'auto-organisation des colonies de fourmis. Certains auteurs comme Cristiano Castelfranchi voient l'intelligence comme un phénomène social modélisable grâce à des capacités cognitives telles que la planification ou la délibération inter-agent pour la réalisation d'un objectif. Dans notre travail, nous essayons de résoudre une problématique à travers la construction d'une intelligence collective grâce aux inter-

actions d'une multitude d'agents. Cette partie présente des approches inspirées de la sociologie permettant ce type de comportements.

3.4.1 La théorie de l'activité

La théorie de l'activité provient originellement de la psychologie et a été proposée par Lev Vygotsky [Vygotsky1978]. Les travaux de ce domaine essaient de comprendre les activités humaines en tant que phénomène systémique et socialement situé. Elle ne considère pas uniquement l'activité d'un individu mais de multiples dimensions en relation avec l'individu telles que [Chung2019] :

— les outils à sa disposition ;

— ses objectifs ;

— ses interactions sociales et sa façon de répartir le travail avec les autres ;

— son environnement (sa communauté par exemple) ;

— les lois, conventions et normes s'appliquant à l'individu et à sa société.

L'activité humaine y est décrite par cet ensemble de facteurs interdépendants tous en relation directe avec l'activité des individus. Elle est présente dans de nombreux types d'applications comme l'apprentissage mobile [Fulantelli2015], les jeux sérieux pour l'éducation [Plass2015], l'apprentissage collaboratif multi-agent [Hanna2012], la coordination multi-agent [Ricci2002], *etc*. Cette théorie est utilisée dans les systèmes multi-agents comme un moyen de coordonner et faire coopérer les agents. Dans [Ricci2002], Ricci *et al.* s'en inspirent pour la conception d'un framework permettant aux agents de coordonner leurs actions à travers la co-construction de lois, contraintes et normes. Le système multi-agent est construit sur trois niveaux d'abstraction. Premièrement, la co-construction est le niveau où les agents raisonnent à propos de la construction de leurs interactions pour la réalisation de leurs tâches. La coopération est le niveau où les agents planifient les actions à mener pour mener à bien les interactions et le niveau coordination est celui où les agents s'assurent de pouvoir exécuter ces actions. Ces niveaux permettent aux agents de se coordonner tout en appliquant les conventions qu'ils partagent.

La théorie de l'activité étant centrée autour des tâches à réaliser, elle serait une inspiration adéquate pour un problème de *task allocation* où les agents doivent se répartir dans des activités. Bien que notre problématique soit aussi un problème de coordination multi-agent, le comportement de nos agents est guidé par un désir social, celui de former une coalition avec les accointances qu'ils jugent attirantes et non de se coordonner autour d'une activité à réaliser. La notion d'activité étant peu utile à nos agents la théorie de l'activité semble moins pertinente que celle de la dynamique des groupes comme source d'inspiration.

3.4.2 La confiance

L'usage de la confiance dans les systèmes multi-agents est issu du besoin des agents de compter sur des ressources ou des services fournis par d'autres agents du système. Le problème se pose particulièrement lorsque le système est ouvert et que des agents inconnus peuvent entrer dans celui-ci. Les agents peuvent alors être exposés au danger de se faire utiliser par les autres ce qui

peut remettre en question le fonctionnement du système. Les agents utilisent la confiance pour évaluer les agents du système en observant leur comportement et pour déterminer s'ils peuvent coopérer avec des agents inconnus [Premarathne2020]. L'un des objectifs est, pour les agents, de former des coalitions dans lesquelles ils pourront coopérer avec les autres membres à long terme. Quatre types de modèles coexistent [Yu2013] :

— les modèles qui évaluent la confiance directement : les interactions passées avec un agent sont évaluées pour déterminer s'il est possible de lui faire confiance dans le futur. De nombreuses approches ont été mises au point dont certaines se basent sur des algorithmes d'apprentissage pour permettre aux agents de s'adapter à un environnement changeant [Wang2020] ;

— les modèles qui évaluent la confiance indirectement : dans des systèmes à plus grande échelle où il est difficile d'avoir suffisamment interagi avec un agent pour évaluer directement la confiance que l'on peut lui accorder, il est possible de l'évaluer de manière indirecte à travers les témoignages des autres agents ;

— les modèles sociaux-cognitifs : ils sont basés sur des états mentaux de la confiance comme les croyances et les objectifs. Ces modèles ne se fondent pas sur l'ensemble des dimensions psychologiques de la confiance mais uniquement sur des dimensions explicites et conscientes [Falcone2009]. Par exemple, la confiance qu'accorde un agent A à un agent B peut être basée sur le fait qu'ils aient des objectifs en commun ;

— les modèles qui évaluent la confiance à travers une structure organisationnelle : ils nécessitent une structure au sein du système multi-agent dans laquelle un parti tiers de confiance sert de superviseur lorsque des agents souhaitent interagir ensemble.

La confiance permet la coopération au sein d'un système multi-agent. Dans une certaine mesure, son objectif est proche de celui de la théorie de l'activité qui tente de définir un cadre de travail propice à la coopération. L'approche de la confiance est cependant plus tournée vers la sécurité en essayant de déterminer les agents avec lesquels il est possible de travailler sans se faire exploiter. La confiance peut être abordée dans les questions de cybersécurité [Singh2015] tandis que la théorie de l'activité est plus adaptée aux question d'efficacité dans la résolution d'un objectif dans les systèmes multi-agents.

3.4.3 Les normes sociales

Les normes sont définies comme "*le langage qu'une société parle, l'incarnation de ses valeurs dans certaines situations sociales*" [Bicchieri2005] [Baldoni2018]. Elles spécifient des règles de conduite tacites fondées sur des croyances ou habitudes au sein d'une société. Les normes sont utilisées dans le domaine du multi-agent pour la coopération des agents en leur permettant de maintenir un schéma comportemental bénéfique à la majorité des individus du système. Elles sont fondées sur un ensemble de croyances, de valeurs et de désirs des individus sur lesquels ces derniers tombent d'accord. Les normes sont bénéfiques aux systèmes artificiels puisqu'elles permettent aux agents de prendre des décisions en réduisant leur coût computationnel en se basant sur la norme comportementale de la société d'agents. Une norme peut se catégoriser de plusieurs manières [Morris-Martin2019] :

— les règles : elles représentent les lois imposées par une autorité ;

— les normes sociales : ce sont des conventions qui s'appliquent dans de grands groupes comme une société ;

— les normes morales : des conventions morales appelant à la conscience des individus.

Mais les sociétés changent et il arrive un moment où de nouvelles normes émergent, où certaines s'effacent ou se modifient. En spécifiant le comportement d'un système, son concepteur peut connaître les normes utiles à son fonctionnement. Les normes peuvent alors être prévues et implémentées de manière statique. Mais cette façon de concevoir un système n'est pas viable dans des sociétés dynamiques où les normes peuvent changer. L'accord sur les conventions utilisées par les agents étant tacite, elles peuvent être considérées comme étant émergeantes. Du point de vue des systèmes multi-agents, les travaux tendent à essayer de modéliser les comportements humains au travers d'architectures d'agents cognitives capables de couvrir l'aspect émergent des normes. Par exemple l'architecture EMIL-A [Conte2014] à la capacité de les reconnaître, de les adopter et de s'y conformer. Pour que les agents puissent reconnaître et adopter une norme, ils doivent être être capable d'obtenir des informations sur leurs accointances. Les informations peuvent soit se récolter par une observation du comportement des autres agents soit par une interaction directe et privée.

Les normes sociales sont englobées dans la théorie de l'activité en étant prise en compte comme l'un des facteurs de l'efficacité du travail au même titre que les outils que les agents peuvent manipuler. Même si leur objectif est similaire puisque les normes sociales telles que nous les avons présentées dans cette partie ont pour objectif la coopération des agents, la théorie de l'activité se veut plus vaste en tenant compte d'un plus grand nombre de facteurs sociaux.

3.4.4 Synthèse

Les inspirations sociales présentées dans cette partie sont toutes orientées vers la réalisation d'une activité à travers la coopération multi-agent. Les normes et la théorie de l'activité partagent les conventions et les lois comme centres d'intérêt même si la théorie de l'activité est plus large et étudie également de multiples autres facteurs autour de l'individu et de son objectif. Au contraire les travaux portant sur les normes sont presque exclusivement concentrés sur l'émergence, la reconnaissance et l'adoption des normes par un individu. La confiance est un domaine où les agents tentent de déterminer avec qui travailler en évitant de se faire exploiter par ses accointances. Les travaux sur les normes et sur la confiance ont l'avantage du parti pris d'un système ouvert et dynamique. Les environnements dans lesquels travaillent leurs systèmes multi-agents peuvent être changeants ce qui conduit à des méthodes dynamiques pour résoudre leurs problèmes de coordination et coopération.

Comme énoncé dans les sections précédentes, notre problématique n'étant pas celle de la coordination d'agents pour la réalisation de tâches mais plutôt l'imitation d'une structure d'agents, celles-ci ne paraissent pas adaptée à notre problème de formation de coalitions. Au contraire les dynamiques des groupes traitent de la formation de coalitions en fonction des critères physiques et psychiques des individus. Cette approche semble plus pertinente pour notre système multi-agent où des agents devront se regrouper en fonction des caractéristiques physiques de chacun et de leurs désirs.

3.5 Conclusion

L'attraction interpersonnelle est régie par de nombreux facteurs. Elle nous permet de déterminer dans quelles conditions deux individus ont le plus de chances de s'attirer mutuellement. Dans le cadre des systèmes multi-agents, faire l'analogie entre un individu et un agent nous permet de nous inspirer de ce concept pour la formation de coalitions. De plus, nous pouvons nous inspirer des travaux sur la cohésion de groupe afin de maintenir l'unité des coalitions nouvellement formées. Notre problématique étant de former des coalitions le plus proche possible d'une demande utilisateur et non de réaliser une tâche, nous faisons impasse sur la dimension de la cohésion opératoire pour nous concentrer sur la cohésion sociale. Dans les chapitres suivants nous proposerons un modèle d'agent dont les mécanismes s'inspirent de l'attraction et de la cohésion telle que définie par Lott & Lott dans la définition 3.3.1. Nous verrons la cohésion comme un agrégat de l'attraction interpersonnelle présente entre chaque membre d'une coalition. Le chapitre suivant présente les propriétés nécessaires à l'élaboration d'une architecture d'agent permettant la formation de coalition tout en utilisant une inspiration sociale. Il présente également une revue de littérature des architectures d'agent sur lesquelles se baser pour la construction de notre architecture.

4

Étude des architectures d'agents

Sommaire

Nous avons pris le parti de baser la réponse au problème de l'aide à la décision sur une approche multi-agent pour la formation de coalitions d'inspiration sociale. Choisir un modèle d'agent est donc primordial pour construire le système. Dans ce chapitre, nous étudions les principales propriétés que notre architecture doit posséder afin de satisfaire les besoins du système. Une revue de littérature sur les architectures d'agents existantes est ensuite abordée dans le but de trouver celles qui répondent le mieux à nos besoins. Nous détaillons ensuite le fonctionnement de l'architecture Soar qui nous semble rassembler les propriétés nécessaires pour la conception de notre architecture.

4.1 Besoins pour une architecture

Cette partie identifie quatre capacités dont notre architecture a besoin pour répondre aux propriétés du système d'aide à la décision. Nous les présentons ci-dessous.

4.1.1 Représenter des connaissances

Le système d'aide à la décision doit fonctionner avec tout type de composants et manipuler leurs caractéristiques. Il doit aussi pouvoir manipuler des connaissances sur le domaine d'expertise, notamment sur l'assemblage des composants utilisés. Pour cela, l'architecture utilisée par les agents doit être capable de représenter des connaissances de n'importe quelle discipline de manière suffisamment précise pour prendre en compte la grande variété d'attributs que peuvent posséder les composants. De plus, un agent doit pouvoir représenter n'importe quel type de composant, ses règles fonctionnelles et les contraintes que l'on peut lui appliquer. Par exemple, l'architecture doit aussi bien pouvoir mémoriser les caractéristiques physiques d'un moteur d'appareil électroménager que sa place dans l'appareil ou les transformations physiques qui lui sont applicables pour éventuellement l'adapter à une nouvelle machine. Cependant, la méthode de représentation des connaissances doit être suffisamment générique pour aussi être capable de mémoriser les caractéristiques de composants qui appartiennent à d'autres domaines tels que l'automobile.

4.1.2 Adopter un comportement social

Le système d'aide à la décision devra fonctionner avec un grand nombre de composants. Les agents devront interagir entre eux, obligeant l'architecture à être capable d'adopter un comportement social. Plus précisément, elle doit permettre la communication avec d'autres agents, le traitement des connaissances échangées, le maintien d'une représentation mentale des accointances et l'évaluation de l'intérêt d'un agent pour une coalition ou ses accointances. Par exemple, après la requête d'un utilisateur les agents ont besoin de s'échanger des informations sur leurs caractéristiques. Si l'utilisateur demande un appareil électroménager, ceux-ci doivent s'assurer de communiquer et de former des groupes uniquement avec des agents représentant des composants de ce type d'appareils.

4.1.3 Gérer la dynamicité du système

La modification dynamique des composants et de leur nombre au sein du système d'aide à la décision peut engendrer des modifications sur les produits conçus. Par exemple, lorsque le prix d'un composant *moteur de lave-linge* augmente, toutes les coalitions dans lesquelles il se trouve sont impactées. De la même manière, si un agent entre ou sort du système, ses accointances doivent réadapter leurs coalitions en conséquence. Pour faire face à cette dynamicité, l'architecture doit permettre aux agents de réévaluer à tout moment leur intérêt à être dans une coalition pour s'adapter à une nouvelle situation rapidement. Nous pouvons par exemple imaginer la mise en place d'une métrique permettant aux agents de déterminer de manière dynamique les accointances avec lesquelles ils sont les plus proches. Dans le cas du changement de prix d'un composant, une telle métrique donnerait la possibilité aux agents de se réorganiser de façon à toujours répondre au mieux au besoin utilisateur. En effet, une coalition n'aurait plus nécessairement lieu d'être si le principal critère de l'utilisateur pour la conception du produit était pécunier et si le prix de ses membres était modifié.

4.1.4 Expliquer son comportement

Le système d'aide à la décision possédant un nombre limité de composants, il peut proposer des solutions incomplètes à l'utilisateur. Celui-ci peut alors négocier avec le système en adaptant sa demande afin de concevoir un produit correspondant au mieux à son besoin. Pour se faire, le système doit être capable d'expliquer les raisons sous-jacentes à l'incomplétude des conceptions proposées. Par exemple, dans le cas où un utilisateur veut créer un lave-linge avec un mode de lavage écologique et que le système lui propose comme meilleure solution une machine sans ce type de programme, ce dernier doit pouvoir expliquer les raisons qui l'ont poussé à concevoir le produit de cette façon. De manière plus générale, le système doit pouvoir justifier ses choix de conceptions auprès de l'utilisateur de façon à ce que celui-ci puisse rebondir en cas d'insatisfaction en adaptant sa requête pour le système. Pour que le système d'aide à la décision soit explicable, il est nécessaire que l'architecture d'agent utilisée manipule des connaissances de manière symbolique de façon à ne pas occulter les résultats de ses mécanismes fonctionnels.

4.1.5 Synthèse

Afin de permettre au système d'aide à la décision de travailler sur n'importe quel type de composants, l'architecture doit avoir des capacités de représentation des connaissances. Celles-ci doivent être suffisamment développées pour ne pas brider les utilisateurs lors de la conception de leurs produits. L'architecture doit pouvoir fonctionner dans un environnement variable et ouvert avec une multitude d'agents l'instanciant. Les agents auront donc à prendre des décisions dynamiquement en fonction de l'état courant de leurs accointances et des messages échangés avec elles. Puisque chaque agent du système devra communiquer avec ses pairs, notre architecture doit être dotée de la capacité d'envoyer, recevoir et interpréter des messages. Lorsque les agents du système ont décidé d'une coalition, il est nécessaire que l'utilisateur puisse comprendre les décisions qui ont menées à sa structure. L'architecture doit donc prendre des décisions de manière transparente, par exemple en utilisant une représentation des connaissances symbolique.

La section suivante fait une revue de littérature des classifications d'architectures existantes et met en avant les catégories qui répondent le mieux à nos besoins pour la conception de notre architecture.

4.2 Étude des architectures existantes

Plusieurs façons de catégoriser les architectures d'agents coexistent dans la littérature SMA. Ces catégorisations adoptent des points de vue différents de manière à trier les architectures en fonction de leurs spécificités. Notre démarche étant de nous inspirer de comportements individuels humains pour la conception de notre architecture, nous nous concentrons dans cette section uniquement sur les architectures d'agents. Cette partie va les présenter et nous permettre de situer notre contribution par rapport aux catégories existantes.

4.2.1 Du comportement de l'agent

Cette classification est fondée sur les capacités des agents à raisonner ou agir rapidement. Les catégories d'architectures se déclinent en plusieurs niveaux de capacités de raisonnement allant de la planification des actions futures pour la réalisation d'un objectif à l'exécution d'actions sans réflexion préalable [Kotseruba2018] [Kumova2017].

Délibérative

Les architectures délibératives [Mendoza2021] ont des capacités de représentation de l'environnement des agents. Les agents implémentant cette architecture possèdent un objectif et des capacités qui leurs sont propres. Leur principal atout est leur capacité à être flexibles et autonomes pour les atteindre. Contrairement aux agents réactifs, ils peuvent utiliser leurs connaissances pour agir sur leur environnement. À l'aide des perceptions qu'ils ont de leur environnement, les agents délibérateurs peuvent se le représenter, le mémoriser et utiliser ces connaissances pour construire des plans d'actions. Les architectures délibératives maintiennent une représentation symbolique de leur environnement ce qui leur permet d'inférer des connaissances. Elles permettent l'implémentation d'agents intelligents capables de raisonner. En contrepartie, ces agents sont moins réactifs ce qui implique des prises de décisions plus lentes. Contrairement aux autres catégories de cette classification, l'architecture délibérative est la seule à pouvoir représenter son monde et raisonner dessus. Par conséquent, le coût computationnel des agents qui implémentent l'une de ces architectures est plus important que celui des agents qui implémenteraient des architectures des autres catégories.

Proactive

Les architectures proactives [Perumal2014] ont la capacité de raisonner sur des perceptions. Cela leur permet d'identifier des opportunités et de faire des prédictions afin d'améliorer leur prise de décision. Un agent proactif est un agent dynamique réagissant à des stimulis sur lesquels il

peut raisonner. Contrairement aux agents délibératifs, les agents proactifs disposent d'un objectif pré-établi [Gonçalves2010]. Un agent proactif ne possède pas l'architecture lui permettant de modéliser son environnement ce qui l'empêche de planifier des actions. Pour pallier ce manque, l'environnement peut être modélisé à travers son comportement, lors de son implémentation par le concepteur. Ainsi l'agent ne se construit pas lui même sa propre représentation de son monde, mais il le connaît implicitement à l'avance à travers sa manière de raisonner.

Réactives

Des capacités réactives permettent aux agents de réagir rapidement à des stimuli de l'environnement. Cependant elles ne permettent pas aux agents de se le représenter. Les architectures aux capacités réactives comportent des règles de comportement semblables à des règles si/alors. Une condition sur un stimuli (*e.g.* état courant de l'environnement) peut déclencher une action de la part de l'agent. Des capacités purement réactives ne permettent pas aux agents de mémoriser des informations sur leur environnement et les empêchent donc de prendre en compte des expériences passées pour planifier des actions futures. Un agent seul n'implémentant que des capacités réactive est sans intelligence, cependant son association avec d'autres agents peut mener à l'émergence d'un comportement intelligent. Un célèbre exemple de l'émergence de comportement avec des agents réactifs est celui de la recherche du plus court chemin par une colonie de fourmis. Dorigo *et al.* utilisent ce type d'agents dans [Colorni1991] pour rechercher le plus court chemin dans un graphe. Les agents implémentant une architecture réactive possèdent un coût computationnel faible dû à leur besoin en mémoire réduit et à leur manque de raisonnement.

Hybrides

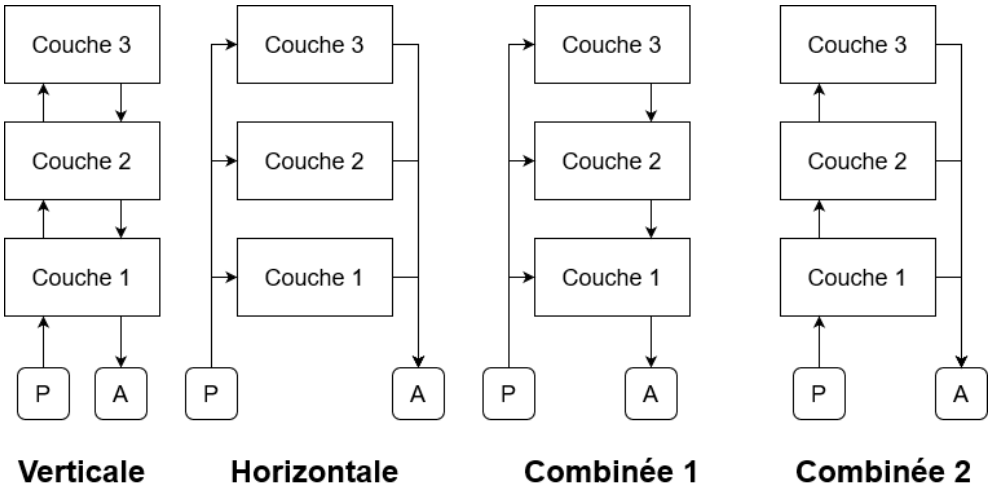


Figure 4.1 – Types d'architectures à couches [Zhang2000]. Le P entrant dans l'architecture est une perception, le A qui en sort est l'action exécutée.

Dans le cadre de la classification comportementaliste, une architecture hybride est une composition des catégories précédentes. Leur objectif est d'outrepasser les inconvénients des architectures de chaque catégorie en utilisant des capacités de chacune d'entre elles. Ces architectures utilisent souvent un système de couches qui possèdent chacune une capacité spécifique.

Une architecture hybride peut être une architecture à couches dans laquelle chaque couche représente une capacité délibérative / proactive / réactive. Il existe 4 types d'architectures hybrides (figure 4.1) :

— les verticales : une perception entre dans une couche N qui peut appeler la couche $N + 1$ en cas de besoin pour la traiter [Zia2009]. Les architectures de ce type peuvent mettre du temps à sélectionner une action lorsque la perception se dirige vers les couches supérieures ;

— les horizontales : ce type d'architecture utilise toutes les couches en simultané. Chaque couche agit comme un agent indépendant. Un module spécifique de l'architecture décide de la couche à utiliser en fonction de la situation [Karavas2015]. L'inconvénient de cette architecture est que chaque couche ne prend pas en compte le résultat des autres ;

— les combinées : elles essayent de combler les inconvénients de ces deux derniers types. La combinée 1 s'inspire de l'architecture à couche *horizontale* et permet la prise en compte des résultats des différentes couches lors de la prise de décision. La combinée 2 s'inspire du type *vertical* et permet une prise de décision plus rapide en sélectionnant une action sans nécessairement devoir attendre le retour des couches les plus hautes [Zhang2000].

Synthèse

La figure 4.2 résume les capacités de chaque catégorie d'architecture. Les architectures positionnées dans la moitié haute (délibérative et proactive) de la figure possèdent une mémoire à long terme et une mémoire à court terme ce qui leur permet des prises de décisions plus réfléchies. Au contraire, les architectures réactives dans la moitié basse ne possède qu'une mémoire à court terme les obligeant à prendre des décisions rapides sur un ensemble de connaissances limité. Les architectures hybrides jouent sur tous les tableaux en s'octroyant les capacités de chacune des catégories précédentes.

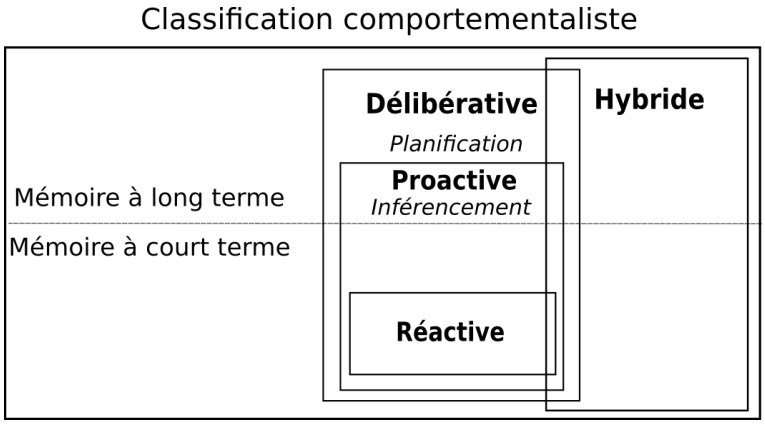


Figure 4.2 – Catégories d'architecture dans une classification par comportement

4.2.2 Des capacités cognitives

Les agents cognitifs ont des capacités de représentation de leur environnement, d'eux-mêmes et des autres agents. Ils possèdent leurs propres objectifs et leur principal atout est leur capacité à être flexibles et autonomes pour les atteindre. Contrairement à des agents réactifs, ils peuvent

utiliser leurs connaissances pour agir sur leur environnement. De la même manière que les agents délibératifs, les agents cognitifs possèdent la capacité de planifier des actions. En outre, ils peuvent être dotés de capacités d'apprentissage leur permettant d'adapter leurs actions à leur environnement. Ce type d'architecture est beaucoup utilisée pour modéliser certains mécanismes de l'esprit humain dans les travaux se trouvant à la frontière entre l'informatique et la psychologie. Les architectures cognitives présentent de fortes similarités avec les architectures délibératives de la classification précédente. Notons que, malgré leurs points communs, les architectures cognitives ont pour objectif de modéliser et imiter la cognition humaine. Les architectures délibératives n'ont pas cette ambition et les agents qui les implémentent tentent d'atteindre leurs objectifs sans essayer d'imiter le comportement humain.

Les architectures cognitives essayent souvent de décomposer des problèmes complexes en sous-problèmes pour en faciliter la résolution. C'est par exemple le cas de l'architecture Soar [Laird1987] s'inspirant des travaux de psychologie de Newell dans [Newell1994]. Cette architecture est réutilisée dans [Van Dang2018]. Les auteurs l'implémentent dans un agent autonome pour une application domotique où l'agent doit effectuer des actions domestiques tout en prêtant attention aux besoins des habitants.

Les architectures cognitives se divisent en trois catégories, les architectures symboliques, émergentes et hybrides. Nous les présentons dans cette partie.

Symboliques

Les architectures symboliques représentent des concepts à l'aide de symboles manipulés à l'aide de règles prédéfinies par leur concepteur. Cette façon de représenter des connaissances et de concevoir un système est commune et utilisée dans une multitude d'architectures EPIC [Hornof1997] MAMID et [Hudlicka2002] Cognet [Zachary1993]. Elle offre la capacité aux agents l'implémentant de manipuler des connaissances et de raisonner dessus. Cependant, ce type d'architecture est peu flexible puisque son fonctionnement est fondé sur un comportement prédéfini par son concepteur. Un évènement non prévu à la conception de l'architecture peut alors la rendre inopérante.

Connexionistes

Ce sont des architectures utilisant des modèles similaires aux réseaux de neurones permettant l'apprentissage des agents qui implémentent ce type d'architecture. Bien que cette méthode fasse perdre en transparence sur le comportement des agents, elle leur permet une meilleure adaptabilité et une meilleure robustesse en cas d'évènements imprévus. En plus de rendre le comportement des agents difficilement compréhensible, le manque de transparence induit également de grandes difficultés à inférer des connaissances à partir de la représentation que les agents se font de leur environnement. Cette catégorie regroupe les architectures CogPrime [Goertzel2013] et Becca [Rohrer2012]

Hybrides

Tout comme les architectures hybrides de la classification précédente, les architectures cognitives hybrides tentent de combiner les avantages des deux catégories précédentes tout en mettant de coté leurs inconvénients. Ainsi, les architectures cognitives hybrides possèdent des modules des architectures symboliques et des architectures émergentes. Il est difficile de spécifier quels sont les modules inspirés ou récupérés des architectures symboliques ou émergentes puisque chaque architecture hybride les combine de manière adaptée à son cas d'application. On retrouve par exemple dans cette catégorie les architectures ACT-R [Anderson1997], Soar [Laird2015] et CLARION [Sun1999].

Synthèse

La figure 4.3 synthétise les propriétés des catégories d'architectures cognitives. La capacité à inférer n'est en réalité pas limité aux architectures symboliques mais c'est lorsque la représentation des connaissances est symbolique que les inférences sont le plus simple à réaliser. Au contraire, le manque de transparence dans la représentation des connaissances des architectures émergentes rend les inférences plus complexes à mettre en œuvre. En outre, toutes les architectures émergentes ne sont pas capables d'inférences. Pour ces raisons, nous associons la capacité à faire des inférences uniquement aux architectures symboliques.

Les architectures hybrides sont quant à elles très intéressantes puisqu'elles permettent d'accéder aux propriétés des deux catégories. La fléxibilité peut aussi être une propriété intéressante dans le cas où les agents d'un SMA doivent évoluer dans un environnement difficilement modélisable par son concepteur. Par exemple lorsque les interactions entre les agents et leur environnement deviennent trop nombreuses pour être prévues lors de la phase de conception du système.

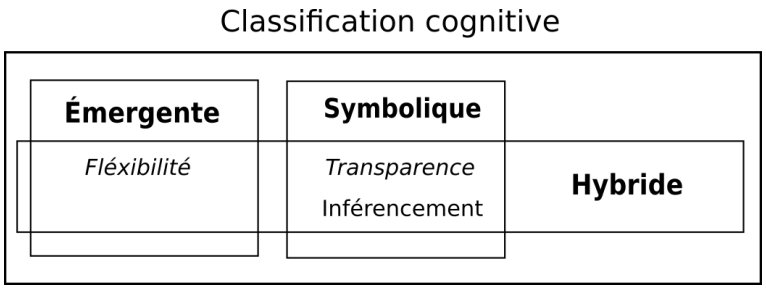


Figure 4.3 – Catégories d'architectures cognitives

4.2.3 De la communication

La collaboration est un mécanisme fondamental des SMA. Dans cette classification, le terme collaboration ne signifie pas nécessairement une entraide inter-agent mais plus un mécanisme de synchronisation leur permettant de se coordonner lors de leurs activités [Hector2005]. En effet, un système multi-agent peut tout aussi bien fonctionner grâce à des agents compétitifs dont l'objectif est de s'assurer l'accès à des ressources communes et limitées plutôt que sur leur capacité à les partager. Il en résulte deux catégories d'architecture : celles qui confèrent des capacités d'interaction inter-agent à travers l'envoi et la réception de messages et celles qui ne sont pas capables de

communiquer.

Communicantes

Les agents implémentant une architecture communicante peuvent interagir en s'envoyant des messages. Ceux-ci permettent aux agents de se synchroniser dans leur activité pour atteindre leur objectif. Selon les agents, la communication peut par exemple prendre la forme d'une activité sociale pour résoudre un problème distribué ou d'une négociation. Les agents communiquent à l'aide d'un langage commun leur permettant de se comprendre. Plusieurs langages ont été développés mais les plus connus sont KQML [Finin1994] et FIPA-ACL [Gan2017].

Non communicantes

Les architectures non communicantes sont à l'exact opposé de la catégorie précédente. Certaines architectures confèrent toutefois aux agents une capacité de coordination indirecte grâce à des mécanismes de stigmergie [Duan2012]. La stigmergie est un mécanisme d'interaction agissant à travers l'environnement des agents. Dans l'exemple des colonies de fourmis, elle opère grâce aux phéromones laissées par les fourmis lors de leur déplacement. Les phéromones sont détectées par les autres fourmis et leur indiquent que l'une des leurs est passée par là peu de temps avant elles.

4.2.4 De l'adaptabilité

Les architectures peuvent être classifiées en fonction de leur capacité à s'adapter à de nouveaux évènements [Kumova2017]. L'adaptabilité représente alors leurs capacités d'apprentissage.

Apprenantes

Les architectures apprenantes offrent aux agents qui les implémentent la capacité d'adapter leur comportement en fonction de leur environnement. Nous pouvons relever 6 types d'apprentissage [Kotseruba2018] :

— le déclaratif : ce type d'apprentissage fonctionne avec la mémoire déclarative. Celle-ci mémorise des faits sur l'environnement de l'agent. Lorsque ce dernier atteint son objectif, l'apprentissage déclaratif permet de mémoriser les actions qui ont été utilisées pour y parvenir ainsi que l'état de la mémoire déclarative avant de les exécuter. Ainsi, lorsque l'agent se retrouve dans la même situation, il est capable de retrouver la séquence d'actions pour parvenir à son but. L'apprentissage déclaratif est défini comme explicite, les connaissances manipulées sont décrites de manière symbolique dans une mémoire à long terme.

— le procédural : il se réfère à l'apprentissage par répétition. En accumulant des exemples de problèmes résolus, un agent est capable de les réutiliser [Franklin2006]. Contrairement à l'apprentissage déclaratif, l'apprentissage procédural est implicite, il utilise des mémoires non symboliques comme des réseaux de neurones ;

— le perceptuel : cet apprentissage traite les informations sensorielles (vue, audition, *etc.*) des agents et est par exemple utilisé pour la cartographie de leur environnement ;

— l'associatif : il fonctionne grâce à l'association de récompenses et de punitions à des états de la mémoire de l'agent permettant d'influencer son comportement [Busoniu2006] ;

— l'apprentissage par amorçage : l'amorçage intervient lorsque l'exposition de l'agent à une perception de l'environnement impacte la manière dont une nouvelle perception proche sera perçue dans le futur. Par exemple la perception d'un contexte spécifique lors de la reconnaissance d'objets sur une image peut permettre une reconnaissance de futurs objets du même contexte plus rapidement [Lane2019].

Quel que soit le type d'apprentissage utilisé, un tel comportement est rendu possible par des mémoires à long terme et se retrouve dans les architectures cognitives et délibératives. Par exemple l'architecture NARS [Slam2015] est capable d'apprentissage par association (apprentissage par renforcement dans son cas) ce qui lui permet de trouver le comportement le plus adapté à l'état de son environnement. Même parmi les architectures cognitives, il est rare de trouver une architecture utilisant tous ces types d'apprentissage. Selon leur objectif, les architectures n'implémentent que quelques unes de ces fonctionnalités.

À couches

Les architectures à couches [Nakashima1998] [Linson2015] peuvent être composées de modules leur conférant une ou plusieurs capacités d'apprentissage abordées dans la catégorie précédente. Contrairement aux architectures apprenantes, une couche dédiée à l'apprentissage est nécessaire pour que celui-ci se fasse ; par nature les architectures à couches ne possèdent pas ces capacités. Dans cette classification, une architecture à couches est une architecture hybride de la classification comportementaliste à laquelle des capacités d'apprentissage ont été ajoutées.

Basées sur les contraintes

Ce type d'architecture permet l'apprentissage des agents mais le limite aux décisions qui ne risquent pas de faire entrer l'agent dans un état critique. On peut voir ce type d'architecture comme un compromis entre les architectures capables d'apprentissage et celles qui ne le sont pas.

Non adaptables

Les architectures non adaptables ne possèdent pas de capacité d'apprentissage. Pour situer ces architectures par rapport aux autres classifications, une architecture non adaptable peut par exemple être réactive car celles-ci ne possèdent pas de mémoire à long terme permettant ce type de comportement.

4.2.5 De l'objectif

Il nous est aussi possible de classifier les architectures selon leur objectif. Cette classification définie peu précisément les architectures qui composent chaque catégorie. Il est tout à fait possible que deux architectures soient très différentes et aient le même objectif. Par exemple une architecture délibérative et une architecture réactive peuvent avoir le même but alors que, comme nous l'avons vu, elles possèdent des capacités très différentes.

Modélisation de la cognition humaine

Certaines architectures comme ACT-R [Anderson1997] ou CLARION [Sun1999] se donnent pour objectif de reproduire la manière de penser de l'être humain. Pour les situer dans les autres classifications, elles sont aussi qualifiées de cognitives (hybrides dans le cas de ACT-R et CLARION). Elles possèdent souvent la capacité de raisonner sur des perceptions similaires aux humains telles que la vision, l'audition, le touché, *etc*. et essaient de reproduire plusieurs fonctionnalités du cerveau comme la mémoire ou l'apprentissage. Par exemple ACT-R possède un modèle d'interprétation de la musique [Chikhaoui2009] capable de simuler l'écoute de la musique. Ces architectures permettent le plus souvent de créer des agents autonomes évoluant en dehors de systèmes multi-agents.

Atteindre l'intelligence générale

L'intelligence générale représente la capacité à résoudre n'importe quel problème. Ces architectures n'ont pas nécessairement pour ambition de reproduire le fonctionnement du cerveau comme celles de la catégorie précédente mais tentent de modéliser un système capable de se représenter n'importe quel type de problème et de le résoudre. Soar [Laird1987] est l'une d'entre elles. Cette architecture utilise une représentation des connaissances symbolique lui permettant de se représenter un grand nombre de situations et de résoudre un large éventail de problèmes.

Développer des systèmes de contrôle intelligents

Ce dernier type d'architecture regroupe celles qui se spécialisent dans la résolution d'un unique type de problème. On peut par exemple y noter la présence de l'architecture cognitive [Agah1997] dont l'objectif est le contrôle de robots dans le cadre d'un travail d'équipe.

4.2.6 Synthèse

Les classifications abordées dans cette partie se chevauchent, il est possible pour une même architecture de se trouver dans plusieurs catégories selon l'objectif du concepteur. Par exemple les architectures délibératives sont très susceptibles d'être également des architectures cognitives.

Dans le cas de notre système d'aide à la décision les agents doivent mémoriser leurs accointances et leurs caractéristiques pour former des coalitions. Par conséquent, notre architecture doit pouvoir représenter des connaissances et les traiter pour prendre des décisions. En outre dans le cadre de notre analogie sociale nos agents doivent être capables d'interagir et négocier avec leurs accointances plutôt que capables de prendre des décisions rapides. Nous avons donc besoin de capacités cognitives qui permettent de créer des agents capables de prendre des décisions à partir de connaissances. Il est nécessaire que les agents possèdent des mémoires à long terme telles qu'une mémoire déclarative pour maintenir une représentation de leurs accointances, mais aussi une mémoire procédurale leur permettant de savoir comment agir sur leur environnement. Une mémoire à court terme peut également leur être nécessaire pour mémoriser de manière temporaire les informations liées à la tâche courante. Par conséquent, une mémoire de travail est requise.

Les agents vont évoluer dans un environnement très variable, mais le système peut être modélisé de façon à ne pas les soumettre à des évènements imprévus nécessitant un apprentissage.

Un agent *A* peut par exemple être dans le groupe d'un agent *B*. Si ce dernier est modifié, alors l'agent *A* pourrait ne plus vouloir être dans son groupe. Cependant la modification de l'agent *B* peut rester intelligible par l'agent *A* qui est capable de réévaluer son attachement. Autrement dit, l'apparition d'évènements imprévus peut être prévu dans la modélisation du comportement des agents. L'apprentissage n'est donc pas une propriété indispensable à notre architecture.

Les agents devant communiquer tout au long de leur activité pour former des groupes, il leur est nécessaire de posséder un module de communication pouvant envoyer et recevoir des messages mais aussi un module permettant leur interprétation faisant le lien entre le module de communication et la mémoire de travail afin de les traiter.

Enfin, notre architecture nécessite une méthode de sélection d'actions lui permettant d'agir sur son environnement. Les architectures délibératives ont la capacité de planifier des séquences d'actions. Étant donné la variabilité de notre système d'aide à la décision, il serait difficile pour nos agents de construire un plan et de l'exécuter sans jamais être interrompu par le changement de l'environnement. C'est pour cette raison que nous proposons une sélection d'actions dynamique basée sur les perceptions et les connaissances courantes de l'agent sur son environnement comme le font les agents proactifs.

Finalement, les propriétés et les modules que nous recherchons appartiennent pour la plupart aux architectures cognitives. La partie suivante détaille plus en profondeur ces catégories d'architectures et les compare afin de trouver l'architecture la plus adaptée à nos besoins.

4.2.7 Du choix de l'architecture

Comme décrit dans la partie précédente, les architectures aux capacités cognitives peuvent être séparées en trois catégories selon leur fonctionnement : 1. émergente (aussi appelée *connexioniste*), 2. symbolique, 3. hybride.

Notre système d'aide à la décision doit être capable de négocier avec l'utilisateur et d'expliquer ses solutions. Pour cela, l'architecture choisie doit prendre des décisions explicables. L'architecture ne peut donc être connexioniste. Comme justifié dans la présentation de nos besoins, notre architecture nécessite la capacité d'expliquer ses choix ou plus précisément de les rendre transparents à l'utilisateur. Les architectures symboliques et hybrides peuvent répondre à ce besoin grâce à leur représentation symbolique des connaissances.

Le système d'aide à la décision requiert une architecture dotée d'une mémoire pour la représentation de l'environnement, d'un comportement social pour les interactions inter-agent, et d'une *heuristique dynamique* de sélection d'actions pour atteindre des objectifs. Les agents ont besoin d'une mémoire à court terme pour traiter des informations liées à leur environnement courant. Cette mémoire est appelée *mémoire de travail*, elle est limitée en capacité et se met à jour rapidement en fonction des perceptions de l'agent sur son environnement. Elle est connectée aux autres mémoires et aux modules de prise de décision. L'architecture nécessite aussi une mémoire à long terme pour se souvenir des informations perçues de l'environnement. Cette mémoire est dite *sémantique*, elle stocke des informations sur l'environnement sous forme de graphes dans lesquels les nœuds représentent des concepts et les liens les relations qui les lient. Elle permet aux agents de se souvenir d'anciennes perceptions et représente ses croyances. Les éléments de la mémoire sémantique peuvent être chargés dans la mémoire de travail puis utilisés par les modules

de décision. Afin de rendre le comportement de l'agent explicable, nous proposons d'utiliser les fonctionnalités des architectures symboliques pour modéliser le comportement de l'agent. Pour cela notre architecture nécessite une *mémoire procédurale* pour mémoriser l'ensemble des règles si/alors représentant les comportements des agents qui l'implémentent.

La table 4.1 compare 12 architectures cognitives symboliques et hybrides selon les besoins énoncés. Les architectures présentant le plus de similarités avec nos préconisations sont les architecture NARS, Novamente [Goertzel2007], Soar, CLARION et ACT-R. Novamente n'est plus maintenue depuis 2008 et a été peu utilisée dans la littérature, s'en servir dans le système d'aide à la décision compliquerait sa conception. De la même manière, bien que NARS soit toujours maintenue et utilisée par la communauté, cette architecture est bien moins utilisée que Soar et est moins bien documentée [Kotseruba2018]. CLARION, Soar et ACT-R sont toutes les trois maintenues et très utilisées dans la littérature. CLARION et Soar s'inspirent des travaux de psychologie ce qui rend leur utilisation plus pertinente par rapport à notre inspiration des sciences sociales. En outre, ACT-R est développé en Lisp. L'utilisation de ce langage diminuerait l'intéropérabilité du système d'aide à la décision et pourrait rendre plus complexes ses interactions avec son écosystème (communication avec l'utilisateur, communication avec le système d'information mis à jour par les fournisseurs de composants). Soar a été développée dans les principaux langages actuels (C++, Java et Python), sa documentation est détaillée et de nombreux tutoriels de prise en main sont proposés par le groupe de recherche qui la maintient. Pour ces raisons, nous utilisons l'architecture Soar comme base pour la conception de notre architecture d'agent.

4.3 L'architecture Soar

Soar est une architecture cognitive hybride possédant une mémoire symbolique. Elle est développée et maintenue par un groupe de recherche de l'université du Michigan depuis 1987. Soar fournit les fondations d'un comportement d'intelligence général aux systèmes l'implémentant. Elle est composée de plusieurs modules lui permettant de raisonner, d'apprendre à agir en fonction de son environnement et d'inférer des connaissances sur ce dernier. Sa structure est illustrée dans la figure 4.4 montrant les liens entre les différentes mémoires et l'intervention des mécanismes d'apprentissage sur celles-ci. Nous présentons ces modules plus en détail dans les parties suivantes.

4.3.1 Mémoires

Mémoire de production (*production memory*). Soar représente son comportement comme un ensemble de règles si/alors. La partie "si" est la *condition* dans laquelle la règle est déclenchable et la partie "alors" est l'*action* effectuée lorsqu'elle est déclenchée. Lorsque l'action est exécutée elle peut modifier l'état courant de la mémoire de travail.

Mémoire de travail (*working memory*). La mémoire de travail peut être représentée comme un graphe dans lequel les nœuds sont des Working Memory Elements (WME). Chaque WME représente un bloc d'information à propos d'un objet. Un objet peut être décrit par un ou plusieurs

Architectures	Domaine d'utilisation	Mémoire procédurale	Mémoire déclarative	Mémoire de travail	Sélection d'action dynamique
ACT-R [Anderson1997]	Expériences psychologiques	✓	✓	✓	✓
CELTS [Faghihi2011]	HRI/HCI	✓	✓	✓	
CLARION [Sun1999]	Expériences psychologiques	✓	✓	✓	✓
CoJack [Ritter2012]	Agent virtuel	✓		✓	✓
DIARC [Schermerhorn2006]	Robotique	✓		✓	✓
NARS [Slam2015]	Divers	✓	✓	✓	✓
Novamente [Goertzel2007]	Agent Virtuel	✓	✓	✓	✓
Pogamut [Gemrot2010]	Clustering	✓		✓	
RCS [Albus2005]	Robotique	✓		✓	
Sigma [Pynadath2014]	NLP	✓	✓	✓	
Soar [Laird1987]	Divers	✓	✓	✓	✓
Touring Machine [Ferguson1992]	Agent Virtuel	✓		✓	

Tableau 4.1 – Comparaison des architectures en fonction des besoins du système

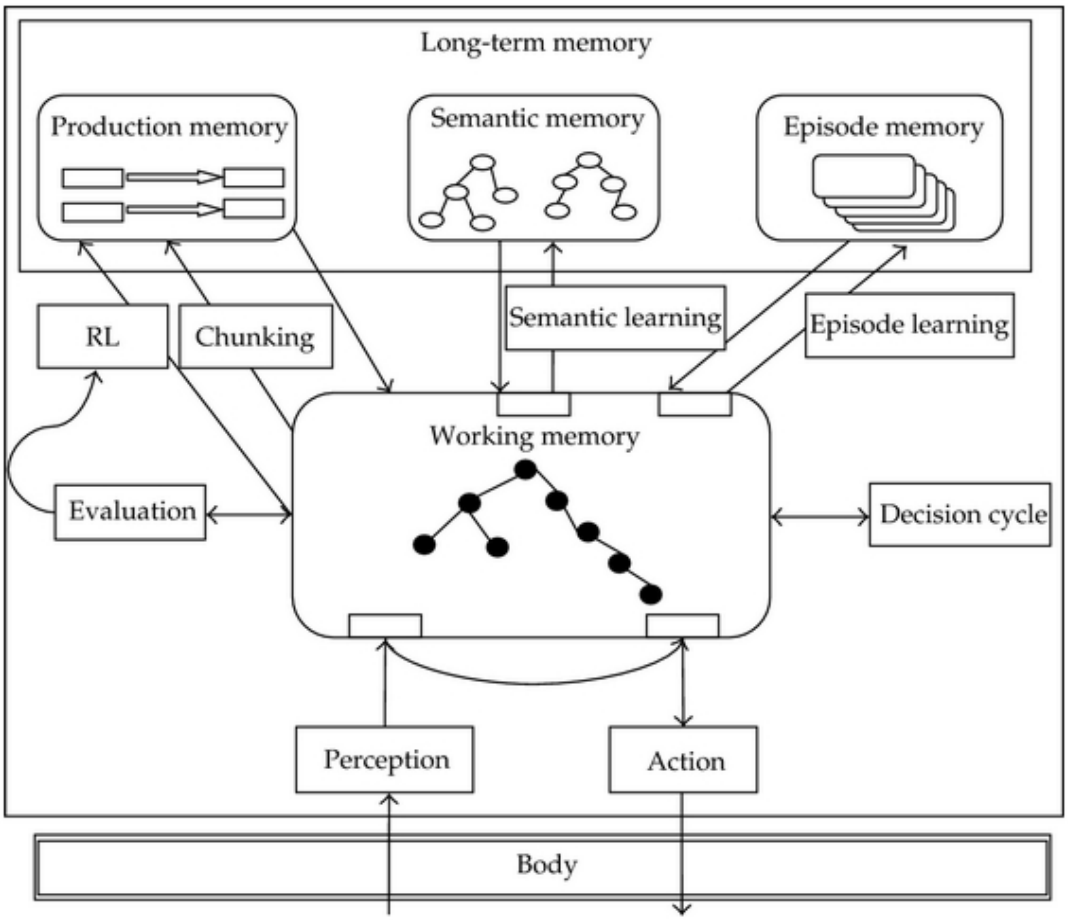


Figure 4.4 – Structure de Soar

WME. Prenons par exemple le monde des blocs. Celui-ci se compose d'un environnement dans lequel des cubes possédant un identifiant peuvent être empilés sur une table (figure 4.5). Dans cet exemple les WME pourraient être : "*B1* est un bloc" et "*B1* est appelé *A*". *B1* est un identifiant et l'ensemble des WME y faisant référence sont représentés par un seul objet dans la mémoire de travail. Chaque WME décrit un attribut de cet identifiant et chaque attribut possède une valeur. Les objets peuvent être connectés les uns avec les autres comme dans la figure 4.6. La mémoire de travail décrit aussi bien les objets que les opérations que l'ont peut leur appliquer. Par exemple dans la figure 4.6, l'opérateur *O4* indique qu'il est possible de bouger le bloc *B2* sur le bloc *B1*.

Les WMEs de la mémoire de travail sont issus de 4 sources :

— les perceptions venant de l'environnement : par exemple la détection d'un élément de l'environnement ;

— les différentes mémoires : par exemple lorsque l'agent charge un élément de la mémoire sémantique dans sa mémoire de travail ;

— l'architecture : par exemple en raison des mécanismes de résolution de blocage lorsque Soar ne peut sélectionner la prochaine action ou sur l'ajout des actions applicables issues de la mémoire de production dans la mémoire de travail ;

— les traitements effectués par l'agent : l'agent peut créer de nouveaux éléments en exécu-

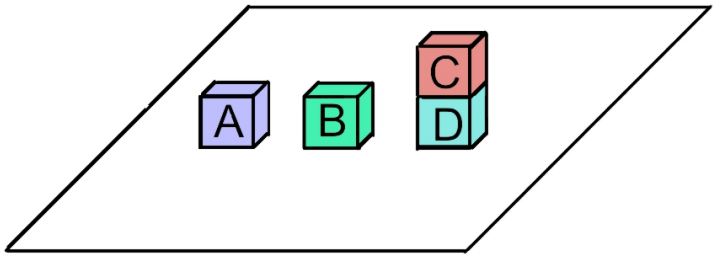


Figure 4.5 – Monde des blocs

tant une action après avoir chargé les règles de production exécutables dans sa mémoire de travail.

Mémoire sémantique (semantic memory). La mémoire sémantique représente des faits généraux à propos de l'environnement. Cette mémoire structure ses connaissances de la même manière que la mémoire de travail. Elle est composée d'une structure en graphe composé d'éléments symboliques structurés en un tuple *identifiant, attribut, valeur*. Cependant il existe deux différences entre les deux mémoires. Le premier est que la mémoire sémantique ne supporte que les attributs constants (pas les identifiants). Contrairement à la mémoire de travail qui utilise des identifiants qui peuvent être renommés à chaque cycle de décision, les attributs utilisés par la mémoire sémantique sont fixes. Le second est que le graphe de la mémoire sémantique peut être composé de sous-graphes qui ne sont pas connectés les uns aux autres.

Mémoire épisodique (episodic memoy). Cette mémoire représente la mémoire des expériences passées de l'agent. Un mécanisme de stockage enregistre les expériences passées dans un ordre chronologique. Ces expériences sont composées d'une action et de son résultat et sont appelées *épisodes*. L'agent Soar peut chercher des épisodes dans sa mémoire et en extraire des informations et régularités qui peuvent ne pas avoir été remarquées et qui, combinées, peuvent améliorer les performances sur de futures tâches. Cette mémoire est liée à la mémoire de travail et fonctionne en augmentant les WME et en permettant à l'agent de savoir par expérience les résultats de ses actions quand l'état a déjà été rencontré.

4.3.2 Cycle de décision

Le cycle de décision de Soar se décompose en 5 étapes :

1. Entrée : récupération des nouvelles données sensorielles dans la mémoire de travail ;
2. Élaboration : recherche de règles déclenchables pour la situation courante. Soar ajoute et retire des WME de sa mémoire de travail au fur et à mesure qu'il recherche les règles déclenchables. La modification de l'état de sa mémoire de travail peut impliquer la rétractation de règles dont la *condition* ne correspond plus à son nouvel état. Soar continue la recherche de règles déclenchables tant que sa mémoire de travail ne s'est pas stabilisée ;
3. Décision : une règle est sélectionnée en fonction des préférences de l'agent. Les préférences correspondent à un niveau de priorité qui peut soit être renseigné par le concepteur de l'agent, soit apprises par renforcement par l'agent lui-même. Soar permet une grande expressivité de préférences, à titre d'exemple elles peuvent prendre les valeurs *acceptable*,

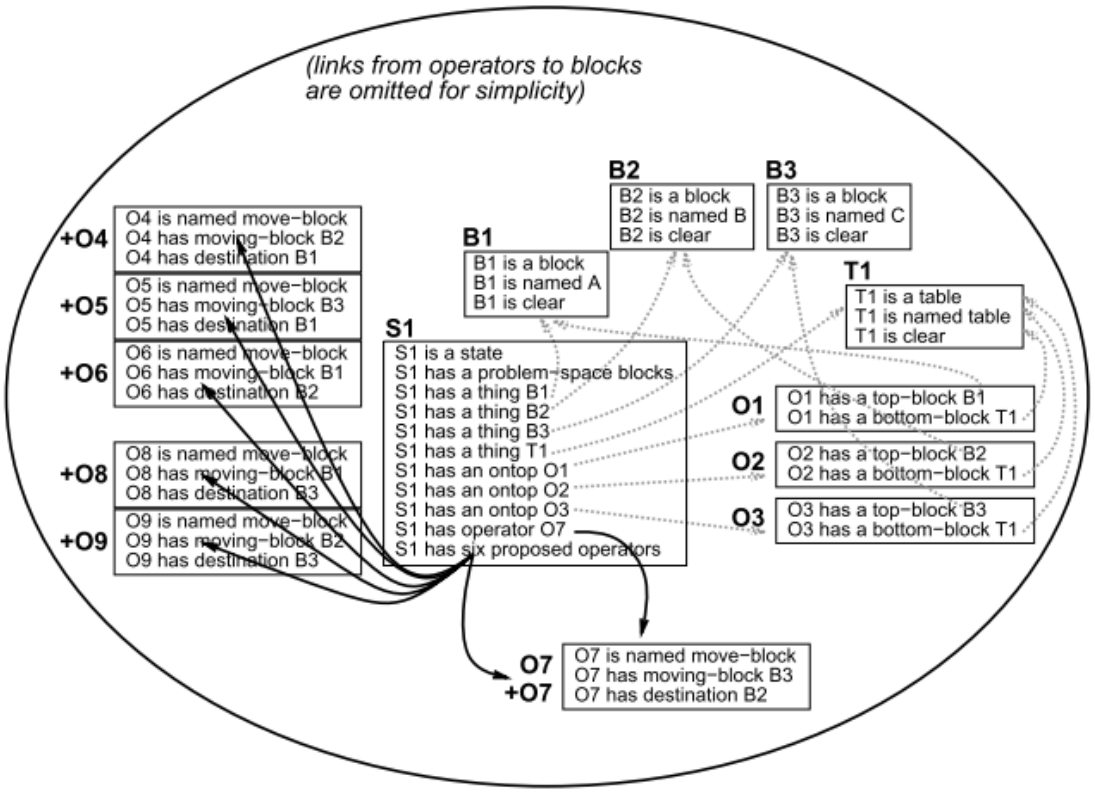


Figure 4.6 – Vue abstraite de la mémoire de travail de Soar [Laird2015]

better, *best*, *reject*, *worse*, *worst*, *etc*. Dans le cas où plusieurs règles ne peuvent être départagées grâce au système de préférences (par exemple si toutes les règles sont définies comme *acceptable*), Soar se tourne vers une fonction nommée *indifferent-selection* dans laquelle le concepteur est censé avoir indiqué à l'agent une stratégie de sélection de règle pour sortir de l'indécision ;

4. Application : la règle choisie est exécutée ;

5. Sortie : l'agent exécute une action sur l'environnement.

Il peut arriver lors des ces étapes que Soar soit incapable de sélectionner des actions (par exemple, aucune règle adaptée à l'état courant ou aucune règle disponible). Dans ce cas, Soar tombe dans une impasse et crée un sous-état au problème qu'il essaye de résoudre. Différents types d'impasses existent et sont gérées par Soar. Tous ces types sont gérés par un mécanisme commun qui est la création d'un nouvel état appelé sous-état (*substate*) dans lequel l'objectif est de résoudre l'impasse. Il est possible que des sous-états génèrent eux-mêmes d'autres sous-états si leur objectif mène aussi à une impasse. Soar peut donc créer et gérer une pile d'états. Ainsi, l'état situé au-dessus d'un sous-état est appelé super-état (*superstate*) et le super-état le plus haut de la pile est appelé le *top-level-state*. Lorsque Soar trouve une solution pour sortir d'une impasse, il la retient grâce à un mécanisme d'apprentissage appelé *Chunking* afin de pouvoir la réutiliser plus tard.

4.3.3 Chunking

Le chunking est un mécanisme d'apprentissage créant des éléments appelés *Chunk* et permettant de retrouver des solutions à des problèmes résolus dans le passé. Lorsque l'impasse d'un problème est résolu, Soar enregistre la solution dans un chunk et le réutilise à chaque fois que le problème est de nouveau rencontré. Le chunk est composé d'une *condition* et d'une *action*. La *condition* correspond aux éléments de l'état de l'impasse qui a été résolue, l'*action* représente les éléments ou les actions qui ont été choisies dans la mémoire de travail au moment de résoudre l'impasse.

4.3.4 Apprentissage par renforcement

Soar est capable d'apprentissage associatif. Cet apprentissage est implémenté sous la forme d'un algorithme d'apprentissage par renforcement (*reinforcement learning*) et plus précisément de l'algorithme *Q-Learning*. Celui-ci permet d'influencer le comportement de l'agent pour la sélection d'actions. Soar utilise cet algorithme pour associer des *Q-valeurs* (valeurs numérique) à des règles de la mémoire de production. Une Q-valeur représente la préférence qu'un agent a pour une règle. Plus la Q-valeur est élevée, plus forte est la préférence. Le fonctionnement classique des règles décrit dans la partie sur le cycle de décision se substitue donc lorsque l'apprentissage associatif de Soar est activé.

Bien que le concepteur de l'agent n'ait plus à affecter lui-même les préférences de chaque règle de la mémoire de production, il doit tout de même associer des récompenses et des punitions à certains états de l'agent pour permettre l'apprentissage. La récompense permet à l'agent d'identifier les états de la mémoire de travail qu'il veut atteindre et la punition ceux que l'on souhaite qu'il évite. Une fois les récompenses et les punitions associées à ces états, l'agent agit de manière à retrouver les récompenses et éviter les punitions.

4.4 Conclusion

Les caractéristiques de Soar répondent partiellement à notre besoin. L'architecture est capable de représenter des connaissances de manière symbolique grâce à ses différentes mémoires. La représentation du comportement des agents sous forme de règles si/alors permet de mieux comprendre leur état au moment d'exécuter une action et facilite l'explicabilité de leur prise de décision. La connexion entre la mémoire de travail et les perceptions de l'environnement permettent aux agents de réagir rapidement face à la dynamicité du système d'aide à la décision. Cependant, Soar ne possède que des capacités de communication limitées ne nous permettant pas de faire interagir des agents sur la base d'un même langage. En effet, les agents Soar sont souvent utilisés comme des agents autonomes évoluant seuls dans leur environnement et ne possèdent donc pas de moyen de communication inter-agent autre qu'en laissant des traces dans leur environnement. En outre, Soar ne possède pas de module permettant à un agent d'évaluer ses préférences ou l'attraction ressentie pour ses accointances ce qui empêche d'instancier un comportement de formation

de coalitions. Le prochain chapitre présente l'architecture ABSG et les modules ajoutés à Soar afin de répondre au besoin social qu'ont les agents du système d'aide à la décision.

5

ABSG : une architecture socio-inspirée

Sommaire

Dans ce chapitre nous présentons le cœur de la contribution de ce travail : l'architecture ABSG. ABSG est fondée sur l'architecture Soar implémentant un module social qui permet aux agents de communiquer et d'évaluer leurs accointances. Les parties suivantes présentent les modules utilisés par ABSG et expliquent leur fonctionnement. Dans les chapitres suivants, nous utiliserons comme synonymes les termes coalitions et groupes.

5.1 Définition du système multi-agent

Pour rappel, le système d'aide à la décision a pour objectif la conception de produits en fonction d'une demande utilisateur. Ce dernier spécifie le produit idéal qu'il souhaiterait posséder et le renseigne au système qui essayera de le reproduire le plus fidèlement possible en fonction des composants à sa disposition. Cette demande utilisateur est donc l'objectif du système d'aide à la décision mais aussi l'objectif des agents de notre système. Du point de vue du système multi-agent, ces objectifs représentent les désirs de nos agents et seront nommés comme tels dans la suite de ce manuscrit. Par exemple dans le cas d'un lave-linge, un agent *tambour* désire être associé à un agent *cuve*. Les désirs peuvent être plus ou moins détaillés ; ils peuvent représenter l'envie d'un agent d'être assemblé avec un type spécifique de composant, mais aussi décrire de manière précise les caractéristiques du composant nécessaire. Notre objectif étant l'émergence de coalitions dans un système très variable plus que celui de la recherche des coalitions optimales parmi un ensemble fixe d'agents, ces derniers n'ont pas besoin d'une vision de l'objectif global et ne connaissent par conséquent que leur objectif individuel. Par exemple, dans le cadre de la conception d'un lave-linge, les agents *tambour* n'ont pas connaissance des désirs des agents *moteur*, ils ne connaissent que les spécifications des tambours. Les désirs sont associés à des caractéristiques, celles qui décrivent les agents de notre système. Elles représentent la description des composants renseignée par leur fournisseur. Les caractéristiques permettent aux agents de savoir qui remplit leur désirs ou non. Pour la même raison que pour les désirs, les agents n'ont pas une vision des caractéristiques de tous leurs pairs mais ils peuvent se les communiquer à travers des messages.

Pour résoudre notre problème de formation de coalitions, nous considérons un ensemble d'agents ABSG dans un système multi-agent ouvert où les agents peuvent entrer ou sortir à tout moment. Ils ont leurs propres caractéristiques et leurs propres désirs également modifiables au cours de l'exécution du système. Les agents sont égoïstes, ils essaient d'assouvir leurs propres désirs et ne sont pas conscients de l'objectif global du système multi-agent. Afin de les satisfaire, les agents doivent former des coalitions avec ceux qui possèdent les caractéristiques les plus proches de leurs désirs. Pour cela, ils évaluent leurs accointances et recherchent celles qui répondent le mieux à leurs besoins. Ils peuvent interagir les uns avec les autres pour s'échanger des informations mais aussi pour former ou rejoindre des coalitions. Les agents qui entrent dans le système n'ont pas connaissance des caractéristiques et désirs des autres agents. Cette méconnaissance les rend initialement incapables de déterminer avec quels agents former des coalitions. C'est pourquoi il leur est nécessaire de pouvoir s'échanger des connaissances sur leurs caractéristiques et leurs désirs. Ces informations leur permettent ensuite de s'évaluer et trouver les meilleures accointances

avec qui former des coalitions.

La partie suivante décrit les besoins de notre architecture afin que les agents ABSG soient capables de résoudre notre problème de formation de coalitions à travers leurs interactions.

5.2 Définition des besoins

Comme énoncé dans le chapitre précédent, l'architecture Soar comble certains de nos besoins grâce aux divers modules lui conférant des capacités cognitives. Cependant, elle est originellement conçue pour des agents autonomes évoluant seuls dans leur environnement et non pour des agents communiquants pour la formation de coalitions. Par conséquent, les agents Soar ne possèdent pas leur propre langage de communication et ne possèdent pas de protocole de communication leur permettant de synchroniser leurs actions. En outre, Soar n'est pas adaptée au fonctionnement d'agents sociaux et se révèle incapable, de part sa conception, d'évaluer l'attirance que les agents qui l'implémentent ont pour leurs accointances. Enfin, Soar n'est pas adaptée à notre contexte applicatif qu'est le remanufacturing pour l'économie circulaire. Elle n'est en l'état pas intégrable à l'écosystème d'un système d'aide à la décision dont l'objectif est de concevoir des produits en interaction avec un opérateur humain. Nous allons voir comment adapter Soar pour atteindre une architecture pleinement adaptée à nos besoins.

5.2.1 Les interactions

Les interactions sont en partie représentées par l'ensemble des messages échangés entre les agents du système. En fonction des applications les interactions peuvent avoir lieu dans un environnement situé dans lequel les agents laissent des traces pouvant être interprétées par ceux qui les croiseront. Soar a originellement choisi un mode de communication analogue, appelé un tableau noir, dans lequel les agents Soar peuvent échanger des informations en les ajoutant dans un espace commun accessible à tous. Tout d'abord, pour que les messages soient correctement interprétés, les agents doivent parler le même langage. Ensuite, pour que l'ensemble des messages échangés puissent être vus comme une conversation – un enchaînement cohérent de messages –, ils doivent suivre un protocole permettant aux agents de savoir comment répondre en fonction des messages précédents. Par exemple, à la réception d'une demande d'affiliation à une coalition, l'absence de message de retour n'est pas considérée comme une réponse par l'émetteur. Enfin, la conversation menée doit avoir un objectif, dans notre cas former des coalitions mais aussi synchroniser les actions des agents pour leur permettre de maintenir une représentation correcte de leur coalition. Par exemple il faut éviter que plusieurs membres d'une coalition ajoutent un nouveau membre ou la quittent au même instant, rendant alors difficile pour les nouveaux agents de savoir qui est dans la coalition.

L'architecture ABSG nécessite d'une part le choix d'un langage de communication permettant aux agents de se comprendre et d'autre part la conception d'un protocole d'interaction pour coordonner leurs actions et former des coalitions.

5.2.2 L'évaluation sociale

Les agents Soar n'ont aucun moyen d'évaluer ce qu'est une accointance attirante ou non. Soar n'intègre aucune métrique permettant cette évaluation et ne définit pas ce qui peut être considéré comme un accointance intéressante. Ce manque d'évaluation conduit les agents Soar à agir indépendamment les uns avec les autres ce qui est peu adapté à notre problématique.

L'architecture ABSG doit doter les agents qui l'implémentent de la capacité à évaluer leurs pairs à travers une métrique d'attirance. En outre, à l'aide de cette métrique, elle doit définir ce qui peut être considéré comme une accointance intéressante avec laquelle former une coalition et pouvoir les différencier de celles qui ne le sont pas.

5.2.3 L'adaptation au contexte applicatif

Soar est une architecture qui possède de bonnes capacités de représentation des connaissances grâce à sa mémoire sémantique. Malgré cela, les processus et les contraintes liés au génie industriel ne sont pas uniquement des connaissances à stocker en mémoire, ce sont des mécanismes qui influencent l'évaluation de l'attirance des accointances et leur comportement. L'architecture Soar ne se concentrant pas sur les problématiques du génie industriel et de l'économie circulaire, elle n'implémente pas ces mécanismes. Par exemple, Soar n'est pas conçue pour déterminer à quel moment l'un des composants d'un lave-linge doit être modifié pour être intégré dans un nouvel appareil.

L'architecture ABSG doit être adaptée à son contexte applicatif en intégrant les mécanismes spécifiques liés à la conception de produit.

5.2.4 Synthèse

Pour résumer, l'architecture ABSG nécessite l'apport de trois capacités permettant aux agents ABSG de résoudre notre problème de formation de coalitions, mais aussi de fonctionner dans un système d'aide à la décision pour la conception de produits dans l'économie circulaire :

— permettre les interactions entre les agents ABSG ;

— évaluer des accointances ;

— intégrer les mécanismes liés au remanufacturing.

Les besoins de l'architecture ayant été identifiés, nous définissons le fonctionnement général du système multi-agent avant de présenter plus en détail les apports de notre architecture. Dans la prochaine partie, nous présenterons l'architecture d'un agent ABSG.

5.3 Architecture

La figure 5.1 présente une vue d'ensemble de l'architecture d'agent ABSG. Le modèle est composé de trois parties : (1) les mémoires, (2) le module social et (3) le module de décision. Les

sections 5.3.1, 5.3.2 et 5.3.3 justifient et détaillent le fonctionnement de ces modules. Comme décrit dans la définition des besoins, les agents ABSG fonctionnent à plusieurs et ont besoin d'un langage commun pour se comprendre et d'un protocole d'interaction pour communiquer. Le module de communication permet de répondre à ce besoin. Sur la figure, les actions représentent les messages qu'un agent peut envoyer. Les perceptions sont les messages reçus des autres agents. La partie 5.3.4 explique plus en détail comment les agents interagissent.

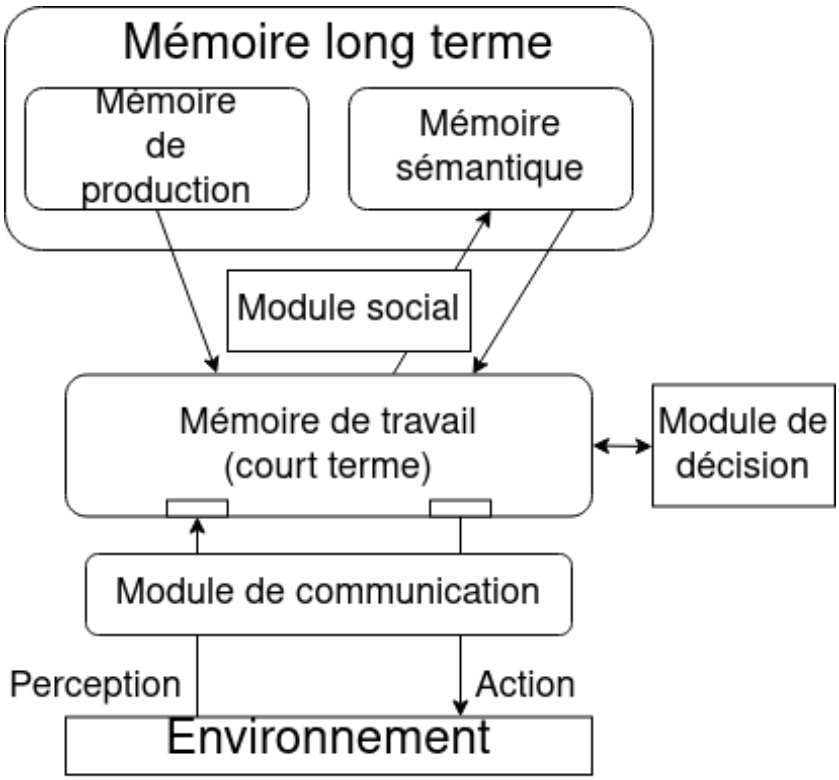


Figure 5.1 – Architecture ABSG

5.3.1 Mémoires

Le système d'aide à la décision fonctionne dans un contexte très variable où les agents, et leur nombre, peuvent être modifiés à tout moment. Notre architecture doit être capable de prendre en compte les modifications concernant l'agent lui-même mais aussi celles de ses accointances. Cette flexibilité de l'architecture permet aux agents d'être suffisamment dynamiques pour former des coalitions avec de nouveaux entrant ou quitter une coalition en fonction des caractéristiques modifiées ou non des autres membres. Afin de prendre en compte leur propre état et celui de leurs accointances, les agents ABSG doivent pouvoir construire une représentation d'eux-mêmes et de leur environnement. Dès lors, l'architecture ABSG doit avoir des capacités de représentation des désirs et des caractéristiques de l'agent qui l'implémente ainsi que ceux et celles de ses accointances (*e.g.* comment les contacter, quelles sont leurs caractéristiques, *etc.*). Il doit aussi être capable d'utiliser ces informations lorsqu'il effectue une action, par exemple envoyer un message à une accointance. Ces connaissances peuvent être stockées dans trois types de mémoires :

— une mémoire de production : c'est une mémoire à long terme stockant les règles de prise

de décision vues dans le chapitre précédent. Ces règles sont conçues pour être spécifiques aux agents ABSG et leur permettre d'interagir avec leurs accointances pour former des coalitions ;

— une mémoire sémantique : c'est une mémoire à long terme stockant des faits sur l'environnement de l'agent. Elle sert par exemple aux agents ABSG à mémoriser les caractéristiques de leurs accointances ainsi que l'avis qu'ils en ont. La mémoire sémantique se met à jour via la mémoire de travail. Par exemple, un message reçu d'une accointance est d'abord chargé dans la mémoire de travail qui traite le message en fonction de son contenu. Si celui-ci contient une information à propos d'une accointance, la mémoire sémantique met à jour ses connaissances sur cet agent. La mémoire sémantique est mise à jour à chaque réception d'un nouveau message. Cette mise à jour régulière est importante pour permettre à l'agent de mettre à jour son attirance pour ses accointances et prendre des décisions éclairées. La mémoire sémantique est celle qui contient les caractéristiques et les désirs de chaque agent ;

— une mémoire de travail : c'est une mémoire à court terme stockant les variables temporaires utilisées par le module de prise de décision. Pour un agent ABSG, la mémoire de travail est utilisée pour charger et interpréter les messages envoyés par les autres agents. Elle permet aussi aux agents de répondre à leur interlocuteur. Les connaissances de la mémoire sémantique sont utiles à la prise de décision ou à l'exécution d'une tâche et sont chargées dans la mémoire de travail pour être traitées. Par exemple, retrouver l'avis que l'agent a sur une accointance en vue de lui demander de rejoindre une coalition.

Nos agents ont besoin de se représenter leurs caractéristiques et leurs désirs. Nous proposons de les décrire sous forme de décimaux. Chaque décimal représente une caractéristique (*e.g.* poids, volume, prix, *etc.*) ou un désir (*e.g.* volume recherché, prix recherché, *etc.*) et évolue dans un intervalle $[0,1]$. Les décimaux ont l'avantage de permettre une bonne expressivité, autant pour des valeurs qualitatives que quantitatives. Par exemple, si un composant doit être décrit par une couleur l'intervalle $[0,1]$ peut représenter l'ensemble des couleurs disponibles (*rouge*, *bleu* ou *vert*) à travers la discrétisation $rouge = 0$, $bleu = 0.5$ et $vert = 1$. Le décimal d'un agent correspondant à sa couleur pourrait donc prendre les valeurs 0, 0.5 ou 1. Nous reviendrons sur la manière de représenter des connaissances sous la forme de décimaux dans la section *Opérationnalisation de l'architecture d'agent* du chapitre 7.

Ces trois types de mémoires permettent aux agents de stocker leur représentation de leur environnement et de la prendre en compte pour évaluer et générer des coalitions avec les meilleures accointances. La section suivante définie la métrique que les agents ABSG utilisent pour évaluer une accointance.

5.3.2 Module social

Évaluer les accointances

En plus de la représentation en mémoire de connaissances, les agents ABSG doivent interagir avec leurs pairs et déterminer quels agents sont les plus appropriés pour former un groupe. Le rôle du module social est de déterminer l'*attraction interpersonnelle* que les agents ont pour leurs accointances. Comme expliqué dans le chapitre 3, l'attraction interpersonnelle est un concept basé

sur de nombreux facteurs. Le module social s'en inspire pour permettre aux agents d'évaluer leurs pairs. Le module utilise les connaissances de la mémoire sémantique sur les accointances d'un agent pour calculer et mettre à jour une valeur d'attraction associée à chacune d'entre elles. L'attraction utilisée par nos agents est issue d'une métrique (*cf.* la section suivante) quantifiant à quel point un agent est attiré par une de ses accointances. Elle peut être vue comme une distance entre deux agents où plus la distance est faible, plus les agents sont attirés l'un par l'autre. Cependant cette distance n'est pas nécessairement symétrique, un agent peut être fortement attiré par une de ses accointances tandis que l'autre ne l'est pas du tout. Par exemple dans l'exemple du lave-linge, un agent *tambour* peut apprécier que son accointance *cuve* possède un diamètre plus large que lui alors que la *cuve* peut trouver que le diamètre du *tambour* est trop élevé. Dans une certaine mesure, cette métrique de l'attraction est une opérationnalisation de l'attraction interpersonnelle des sciences sociales pour notre problématique. Notons tout de même que l'attraction interpersonnelle est une source d'inspiration et que notre opérationnalisation ne reflète pas la réalité des sciences sociales. Le module s'inspire aussi de la définition de la cohésion de groupe de Lott & Lott (*cf.* chapitre 3) pour adapter le comportement des agents au sentiment d'unité qu'ils ont pour leurs groupes. Cette définition a l'avantage de réutiliser le concept d'attraction interpersonnelle pour définir la cohésion. D'après elle, la cohésion est un agrégat des attractions interpersonnelles des membres du groupe. Pour respecter cette définition, le module social voit la cohésion comme étant la moyenne des valeurs d'attractions que tous les membres ont les uns pour les autres.

Intégrer les principes de l'attraction interpersonnelle

Le module social intègre les principes de l'attraction abordés dans le chapitre 2 dans la variable v_a permettant alors de modifier la fonction d'attraction/répulsion des agents en fonction des accointances avec lesquelles ils interagissent.

Nos agents doivent former des groupes en fonction de leurs propres objectifs. Nous reprenons donc la majorité des principes à l'exception de la réciprocité qui les pousse à adopter un comportement similaire l'un envers l'autre en moyennant l'attraction qu'ils ont l'un pour l'autre. Ce principe diminue leur capacité à prendre en compte leurs propres désirs.

Nous appelons dans la suite de cette section C et D les tableaux de décimaux représentant les caractéristiques et les désirs de nos agents.

— principe de similarité (s) : la similarité prend normalement en compte les similitudes entre les attitudes des individus. Dans notre architecture nous l'interprétons comme la similarité des désirs des agents leur permettant de se regrouper par objectif. Par exemple, dans le cas d'un vélo, faciliter le regroupement des agents *guidon* et *roues* si les deux agents recherchent la même couleur de cadre de vélo pour remplir leurs deux désirs à l'aide d'un unique agent *cadre de vélo*. Nous représentons donc la similarité comme une distance entre les désirs de deux agents. D'un point de vue plus technique, ce principe représente la distance euclidienne entre le vecteur de décimaux D de désirs d'un agent a_i et celui d'un agent a_j tel que :

$$d = D_i - D_j \qquad\qquad s = \sqrt{d^T d} \tag{5.1}$$

avec d^T la transposée du vecteur d. Plus la distance est faible, plus les agents possèdent des désirs similaires et plus il est intéressant de les regrouper afin de former une coalition d'agents ayant des objectifs communs ;

— principe de complémentarité (c) : la complémentarité a tendance à faire s'apprécier les individus qui ont des capacités complémentaires. Nous l'interprétons dans notre architecture comme la complémentarité des objectifs d'appariement des agents en arguant que ce mécanisme peut permettre de faciliter l'insertion d'agents peu prisés par leurs pairs dans une coalition. Par exemple dans le cas d'un appareil complexe tel qu'un lave-linge, un agent *cuve* et un agent *moteur* ont peu d'objectifs d'appariement en commun. Autrement dit, il n'existe aucun type d'agent avec lequel les deux souhaitent s'assembler. Il semble donc intéressant de favoriser cette relation car l'agent *moteur* possède des désirs d'appariement avec des types d'agents que la *cuve* ne connaît pas. Favoriser cette relation permet donc de faciliter la complétion des groupes. Ce principe est représenté dans l'architecture comme la complémentarité des désirs des types d'agents.

$$c = 1 - \frac{d_a}{|D|} \tag{5.2}$$

avec d_a le nombre de désirs d'association différents entre deux agents et $|D|$ le nombre de désirs d'association maximum que l'un des deux agents possède ;

— principe d'attractivité physique (a) : l'attractivité physique a tendance à rendre appréciables les individus considérés comme beaux. Nous la considérons dans l'architecture comme la distance entre les caractéristiques d'un individu et les désirs d'un autre. Techniquement parlant, l'attractivité physique est la distance entre les désirs D d'un agent et les caractéristiques C d'une de ses accointances. Plus la distance est faible et plus l'accointance est attirante.

$$d = D_i - C_j \qquad\qquad a = \sqrt{d^T d} \tag{5.3}$$

Ce principe a comme intérêt de rendre attirant les agents qui possèdent des caractéristiques correspondants aux désirs des autres. Il joue donc un rôle majeur dans le comportement de nos agents.

— principe du minimax (m) : enfin le minimax représente dans les *Dynamiques de Groupes* un ratio entre le bénéfice et le coût d'une relation. Nous l'interprétons dans l'architecture ABSG comme le coût à maintenir une relation avec une accointance. Autrement dit, le bénéfice est considéré comme constant tandis que seul évolue le coût de la relation à travers les refus d'un agent à répondre aux propositions de ses accointances. Par exemple, si un agent A demande à un agent B de former un groupe et que l'agent B refuse la proposition alors l'agent A dégrade l'attraction pour cet agent de façon à ne pas lui proposer à nouveau de former un groupe.

$$m = nbR * \varepsilon \tag{5.4}$$

avec *nbR* le nombre de rejets de demande de formation ou d'affiliation à une coalition qu'une accointance a émis vers un agent et ε le taux de dégradation de l'attraction pour chaque refus. Le chapitre 7 explique comment ce taux est choisi dans le cadre de notre expérimentation.

Les principes de l'attraction utilisés étant sélectionnés et détaillés, nous les intégrons dans notre architecture à travers une équation que nous appelons la *fonction d'attraction*.

Calcul de l'attraction interpersonnelle

La fonction d'attraction appelée ici v_a est la fonction permettant aux agents d'évaluer leurs accointances pour prendre des décisions. Nous la définissons grâce aux principes de l'attraction de la section précédente : la similarité **(s)**, la complémentarité **(c)**, l'attractivité physique **(a)** et le minimax **(m)**. $v_a(A)$ représente la valeur d'attraction qu'à un agent pour son accointance A.

$$v_a(A) = \frac{\lambda_1 c + \lambda_2 s + \lambda_3 a}{\lambda_1 + \lambda_2 + \lambda_3} + m \tag{5.5}$$

Cette équation est une moyenne pondérée de la complémentarité, similarité et attractivité physique additionnée au minimax. L'attractivité physique étant primordiale à la conception de produits répondant aux désirs de l'utilisateur, nous proposons de surpondérer ce principe dans l'équation. La complémentarité et la similarité étant au contraire des principes de supports, c'est-à-dire facilitant la formation de coalitions, leur pondération est moins important que celle de l'attractivité physique. Le chapitre 7 revient sur les diverses configurations de la fonction d'attraction, évalue leur efficacité dans la formation de coalitions et propose des pondérations en fonction des cas applicatifs.

L'attraction pour une coalition

La cohésion de groupe est définie de nombreuses manières dans la littérature (*cf.* chapitre 3). La définition de Lott & Lott est particulièrement intéressante dans notre cas puisqu'elle la défini comme une agrégation de l'attraction qu'un membre du groupe à pour les autres membres, attraction que nous avons opérationnaliser dans la section précédente. Nous nous inspirons donc de cette définition pour calculer l'attraction qu'un agent a pour sa coalition. Nous définissons l'attraction qu'un agent a pour sa coalition comme la moyenne de l'attraction qu'il a pour tous les autres membres avec qui il désire être en groupe. $V_{ac}(C)$ représente l'attraction qu'un agent a pour sa coalition C.

$$v_{ac}(C) = \frac{\sum_A^{Acc} v_a(A)}{|Acc|} * (1.1 - \frac{|Acc|}{|D|}) \tag{5.6}$$

avec Acc l'ensemble des accointances de l'agent dans la coalition concernée et A l'une de ces accointances. $|D|$ représente le nombre total d'accointances avec lesquelles un agent désire former un groupe. La partie droite de la fonction (après la multiplication) est un coefficient représentant à quel point une coalition est complète du point de vue de l'agent. Plus $|Acc|$ est proche de $|D|$, et plus l'agent a d'accointances avec qui il désire être en groupe dans sa coalition. Ce coefficient permet la valorisation des coalitions les plus remplies pour inciter les agents à y entrer et ainsi

obtenir des coalitions les plus complètes possible. De la même manière que pour l'attraction interpersonnelle, plus la valeur de l'attraction est faible, meilleure elle est pour les agents. Notons que l'agent ne prend en compte que l'attraction qu'il a pour ses accointances puisqu'il lui est impossible d'évaluer les agents pour lesquels il n'a pas de désirs. Par exemple, si l'objectif était de concevoir un vélo, un agent *roue* ne pourrait évaluer un agent *guidon* car les deux n'étant pas liés dans le produit, ils ne se connaissent pas.

Une fois l'attraction calculée et stockée dans la mémoire sémantique d'un agent ABSG, ce dernier a besoin de l'utiliser pour savoir à quelles accointances demander de former une coalition. La section suivante présente la façon dont l'attraction est utilisée dans la prise de décision et détaille le cycle de décision de l'architecture ABSG.

5.3.3 Module de décision

Nous présentons dans cette section la manière dont un agent ABSG prend la décision de s'apparier avec d'autres agents. Plus précisément, nous présentons les diverses règles de comportement stockées dans la mémoire de production des agents ainsi que leur cycle de décision général et définissons une limite permettant aux agents de savoir quelles accointances sont bonnes ou non pour former une coalition.

Prise de décision

La prise de décision du modèle ABSG est basée sur deux facteurs : (1) les règles de comportement issues de la mémoire de production, (2) les valeurs d'attraction affectées aux accointances ou aux groupes. Les comportements (présentés dans la table 5.1) sont un ensemble de règles qui indiquent comment un agent doit agir dans des situations spécifiques. Ils sont similaires à une déclaration "si/alors". Nous présentons table 5.1 ces huit règles.

Il est important que les agents priorisent l'échange de leur caractéristiques pour adapter leur prise de décision. Par conséquence les règles les plus prioritaires sont les 1, 2 et 3 qui permettent cet échange. Ensuite la priorité est à la formation de groupe. Les agents ne créent pas de coalitions avec les accointances qui sont déjà dans un de leur groupe afin d'éviter de faire des groupes doublons dans lesquels les membres sont exactement les mêmes. Donner la priorité à la règle de formation avant celle de l'affiliation permet donc aux agents de former des groupes avant que les membres ne les remplisse avec de nouveaux membres. Les agents ABSG peuvent faire partie de plusieurs coalitions, ce qui permet au système d'aide à la décision d'essayer plus de solutions. Bien que le chevauchement des coalitions paraisse pratique, il serait inutile de former des coalitions possédant exactement les mêmes membres. C'est pourquoi la règle 4 spécifie de ne pas former de groupes avec une accointance déjà présente dans une coalition. En revanche, un agent est libre d'ajouter n'importe quel membre à une de ses coalitions s'il estime qu'il répond à ses besoins. Enfin, la règle 8 spécifie qu'un agent a la possibilité de quitter un groupe si son attraction pour lui n'est pas suffisamment bonne.

Ces règles font partie du modèle ABSG et sont utilisées par les agents pour savoir quelle action exécuter en fonction de leur état actuel. Mais un agent doit choisir la bonne règle à appliquer. Pour cela, ils suivent un cycle de décision qui leur permet de sélectionner leur prochaine action.

Règles	Description
Règle 1	Si l'agent ne connaît pas les caractéristiques d'une de ses accointances, alors il les lui demande.
Règle 2	Si l'agent reçoit une demande de caractéristiques, alors il les envoie à l'accointance qui les demande.
Règle 3	Si un agent change de caractéristiques, alors il renvoie les nouvelles à toutes ses accointances.
Règle 4	Si l'agent a une valeur d'attraction inférieure à un seuil α pour une de ses accointances et qu'il n'est pas déjà dans un groupe avec elle, alors il lui envoie une requête de formation de groupe.
Règle 5	Si l'agent reçoit une demande de formation de groupe et que la valeur de l'attraction pour l'émetteur de la demande est inférieure à un seuil α, alors il accepte la requête, sinon il la refuse.
Règle 6	Si l'accointance possédant la plus faible valeur d'attraction n'est pas dans le groupe qui possède la plus faible valeur d'attraction de l'agent, alors il lui envoie une requête à son accointance pour lui demander de rejoindre son groupe.
Règle 7	Si l'agent reçoit une demande pour rejoindre un groupe et que son attraction pour l'accointance émettrice est inférieure à un seuil β, alors il accepte, sinon il refuse.
Règle 8	Si l'attraction de l'agent pour un groupe est supérieure à un seuil γ, alors il le quitte en notifiant tous les autres membres de son départ.

Tableau 5.1 – Règles de comportement instanciées dans l'architecture ABSG

Seuils de sélectivité

Dans cette section, nous définissons les seuils α, β et γ utilisés dans les règles de la table 5.1. Plus les seuils sont élevés et plus ils laissent aux agents l'opportunité de créer de coalitions. Au contraire, plus ils sont bas, moins les agents sont enclins à former des groupes. Nous proposons une fonction de seuil qui correspond à une fonction de normalisation prenant en compte le nombre d'accointances des agents pour calculer α, β et γ. En effet, nous avançons que plus un agent a d'accointances, plus il a de chance d'en trouver de très intéressantes pour former une coalition. Il peut donc se permettre d'être plus sélectif. Au contraire, moins il a d'accointances, plus il y a de risque qu'une forte sélectivité sur la qualité de l'attraction pour une accointance mène l'agent à s'isoler sans former de groupes. Comme nous le verrons dans le chapitre 7, l'attraction moyenne des agents pour leurs accointances est d'environ 0.35, nous affectons donc le seuil minimum *minout* à 0.35 et *maxout* à 0.45 de façon à réduire légèrement la sélectivité dans les systèmes composés de peu d'agents. max_{acc} et min_{acc} respectivement le nombre maximum et minimum d'accointances d'un système sont initialisés différemment selon les expériences dans le chapitre présentant les évaluations. Enfin x est le nombre d'accointances des agents au cours des expériences. Nous reviendrons sur l'initialisation de ces paramètres dans le chapitre abordant les évaluations.

$$s = \frac{(max_{acc} - x - minout) * (maxout - minout)}{max_{acc} - min_{acc}} + minout \tag{5.7}$$

Bien que les trois seuils portent des noms différents et ne sont pas utilisés dans les mêmes conditions, nous proposons leur initialisation avec la même valeur. Rendre identique le seuil d'affiliation à un groupe et celui du départ de la coalition si cette dernière est de trop mauvaise qualité permet de ne pas tolérer de dégradation de la qualité des coalitions dans le cas de la modification d'un de leurs membres. Cette méthode montrera plus particulièrement son intérêt dans la partie évaluation.

Ces seuils permettent aux agents de différencier une accointance jugée attirante d'une accointance avec qui il n'est pas intéressant de former une coalition. Ils influent sur le comportement des agents à travers les règles explicitées dans la section précédente. La section suivante présente le cycle de décision de l'architecture ABSG.

Cycle de décision

Le modèle ABSG étant basé sur Soar, leurs cycles de décision sont très similaires. Les agents ABSG doivent former des groupes dans un système dynamique où leurs accointances peuvent changer de caractéristiques à n'importe quel moment. La valeur de l'attraction doit donc être mise à jour régulièrement pour chaque accointance. Ainsi, un agent ABSG possède dans son cycle de décision une étape supplémentaire par rapport à un agent Soar lui permettant de charger les messages reçus dans la mémoire sémantique et de mettre à jour l'attraction pour ses groupes et ses pairs. Cette phase supplémentaire est appelée "phase de socialisation". Le cycle de décision est divisé en quatre phases que nous présentons ci-dessous :

— la phase d'élaboration : l'agent charge dans la mémoire de travail toutes les règles de production qui peuvent être déclenchées ;

— la phase de décision : l'agent choisit une règle à lancer. Si plusieurs règles peuvent être lancées, il choisit la règle de plus haute priorité qui est l'ordre dans lequel elles sont présentées dans la partie précédente. Toutes les règles chargées au moment de la phase de décision seront exécutées dans cet ordre ;

— la phase d'application : les actions des règles sélectionnées sont exécutées. La mémoire de travail et la mémoire sémantique sont modifiées ;

— la phase de socialisation : les messages reçus par l'agent depuis le dernier cycle sont traités et les valeurs d'attraction des connaissances et des coalitions sont mises à jour en fonction du nouvel état de la mémoire sémantique.

Synthèse

Le comportement des agents ABSG est régi par le cycle de décision. Celui-ci sélectionne et exécute les règles dont les agents sont intrinsèquement dotés. En outre, il permet la mise à jour de l'attraction qu'un agent a pour ses accointances à travers une fonction d'attraction socialement inspirée. Les règles utilisées par les agents sont principalement basées sur cette métrique de l'attraction qui leur permet de créer, intégrer ou quitter une coalition. Plus la valeur d'attraction entre deux agents est faible, plus les agents sont considérés comme proche et intéressants l'un pour l'autre.

Ce fonctionnement individuel dote les agents ABSG d'un comportement adapté à la formation de coalitions. Mais pour réellement les former, ceux-ci nécessitent d'interagir les uns avec les autres à travers un langage commun et un protocole de communication. La partie suivante les présente.

5.3.4 Interactions

La structure des messages

Les interactions sont essentielles à la formation des coalitions. Les messages permettent aux agents d'échanger leurs caractéristiques et leurs désirs, d'envoyer des requêtes afin de former des groupes, *etc*. Ils utilisent une réduction du langage de communication FIPA-ACL pour communiquer que nous présentons ci-dessous. Le langage FIPA-ACL est originellement très complet et les agents ABSG n'ont pas besoin d'une structure aussi détaillée pour interagir. Un message généré par le module de communication d'ABSG est composé des paramètres suivants :

— *sender* : l'identifiant de l'agent ayant envoyé le message ;

— *receiver* : l'identifiant de son destinataire ;

— *performative* : caractérise le type de communication. Par exemple une demande de formation de groupes, ou une réponse à une demande d'affiliation. Les performatifs utilisés sont détaillés dans la suite de cette partie ;

— *ontology* : sert de contexte au message donnant un sens à son contenu. Par exemple, les agents ABSG peuvent utiliser l'ontologie *coalition* pour spécifier que le contenu du message est lié à un groupe ;

— *content* : le corps du message, contient toutes les données que l'agent veut transmettre. Par exemple dans le cas d'une demande d'affiliation à une coalition, l'identifiant du groupe concerné, les identifiants de ses membres, *etc*.

Les paramètres *reply-to*, *language*, *encoding*, *conversation-id*, *reply-with*, *in-reply-to* et *reply-by* ne sont pas ajoutés car inutiles dans notre cas. Dans le cadre de leurs échanges, les agents ne peuvent communiquer que de un à un et ils n'ont pas la nécessité de transmettre un même messages à plusieurs accointances. Le paramètre *reply-to* est donc implicitement équivalent au *sender* que les agents utilisent. Nos agents utilisant toujours le même langage, le paramètre *language* serait constant s'il était utilisé et n'apporterait pas d'information. Nous le mettons donc de coté. Le paramètre *encoding* est lui aussi inutilisé pour la même raison que le *language*. Comme nous le verrons dans les sections suivantes, les conversations des agents ne contiennent pas de nombreux messages et chacune d'entre elles concerne une coalition spécifique. Dés lors il leur est facile de suivre le fil d'une conversation sans avoir besoin du paramètre *conversation-id*. De la même manière, le paramètre *reply-with* a pour rôle de faciliter le suivi d'une conversation lorsque plusieurs d'entre elles ont lieu en même temps. Les conversations entre les agents ABSG étant courtes et basées sur un ensemble de requêtes / réponses peu ambiguës, ce paramètre n'est pas utile à nos agents. Le paramètre *in-reply-to* est lui aussi peut utile dans notre cas puisque les agents n'ont pas la nécessité de communiquer à propos d'une action. Enfin le paramètre *reply-by* aurait pu être utile pour désigner un délai maximum de réponse avant que la requête soit annulée, mais nous avons fait le choix de forcer les agents à répondre aux requêtes reçues – même dans le cas d'un rejet de la proposition – de façon à ce qu'en cas de congestion de leur pile de messages, les agents puissent prendre le temps qui leur est nécessaire pour répondre à tout le monde.

Description des paramètres

Deux des paramètres de la structure d'un message utilisé par les agents de notre système peuvent contenir une multitude de valeurs. Nous les présentons et justifions ci-dessous.

Ontologies. L'ontologie donne un contexte permettant de comprendre le contenu du message. Dans le cas des messages générés par notre architecture, l'ontologie peut prendre 3 valeurs :

— *agent* : désigne les interactions dont le sujet est l'agent. Toutes les interactions permettant de demander ou transmettre les caractéristiques et les désirs des agents utilisent cette ontologie ;

— *coalition* : désigne les interactions qui ont pour sujet la formation de coalition. Par exemple, une demande de formation de coalition utilise cette ontologie ;

— *scheduling* : désigne toutes les interactions permettant aux agents de se coordonner. Par exemple lorsqu'un agent souhaite ajouter un membre dans sa coalition.

Performatifs. Le performatif exprime l'action du message. Les performatifs manipulés par l'architecture ABSG sont les suivants :

— *proposes* : est utilisé lorsqu'un agent propose à son accointance de former une nouvelle coalition ou d'en rejoindre une existante ;

— *requests* : la requête est utilisée pour les interactions de coordination, par exemple quand un agent demande l'autorisation à ceux de sa coalition d'ajouter un nouveau membre. Les processus de coordination au sein d'une coalition sont détaillés dans la section suivante traitant

des protocoles de communication. Ce performatif est également utilisé lorsqu'un utilisateur extérieur au système comme son concepteur souhaite connaître des données sur l'agent. Ce performatif est par exemple utilisé pour la génération des graphiques d'évaluation dans le chapitre 7 ;

— *queries* : est utilisé lorsqu'un agent souhaite connaître les caractéristiques et désirs d'une de ses accointances ;

— *informs* : est utilisé en réponse au performatif précédent pour renseigner les informations demandées à l'agent concerné ;

— *accept_proposals* : ce performatif est l'une des deux réponses possibles au performatif *proposes* lorsqu'un agent en invite un autre dans une coalition ;

— *reject_proposals* : le refus est la seconde réponse envisageable au performatif *proposes* en cas de demande de formation ou d'affiliation à une coalition.

Afin de mieux visualiser les liens entre ces performatifs et leur enchaînement dans une conversation entre deux agents, nous pouvons présenter le protocole de performatif de la manière suivante :

— $réception(proposes) = \{accept\_proposal, reject\_proposal\}$

— $reception(requests) = \{informs\}$

— $reception(queries) = \{informs\}$

— $reception(informs) = \emptyset$

— $reception(accept\_proposal) = \emptyset$

— $reception(reject\_proposal) = \emptyset$

Nous pouvons lire chaque élément de cette liste comme : *à la réception d'un message contenant un performatif P, l'agent peut répondre par l'un des performatifs de l'ensemble suivant* $\{P1, P2, ...\}$. Ensemble où $\emptyset$ désigne un ensemble vide et la fin de la conversation.

Après avoir présenté la structure des messages utilisés par les agents ABSG et détaillé leurs paramètres, nous présentons dans la section suivante le protocole de communication suivi par chaque agent. Ce protocole peut être vu comme la structure de la conversation que les agents doivent suivre pour travailler ensemble.

Protocole de communication

Les figures 5.2, 5.3, 5.4 et 5.5 illustrent des instances de protocoles internes d'un agent dans des scénarios de formation de groupe, de demande d'informations, de demande d'affiliation à un groupe et d'abandon de groupes. L'état initial est l'état numéro 1 et l'état final est indiqué par un double cercle. Les marqueurs *!* sont les messages émis, *?* les messages attendus et x un paramètre échangé entre les protagonistes. Dans le cas des figures 5.2, 5.4, 5.5 et 5.6, le paramètre x représente le numéro du groupe formé, rejoint ou quitté. Dans le cas de la figure 5.3 il représente les caractéristiques physiques et les désirs de l'agent. Ce protocole est très proche du protocole *FIPA Contract Net* [Qasim2021], la seule différence étant que nos agents n'ont pas la nécessité de faire un appel d'offre puisque qu'ils contactent directement l'accointance avec laquelle ils veulent traiter.

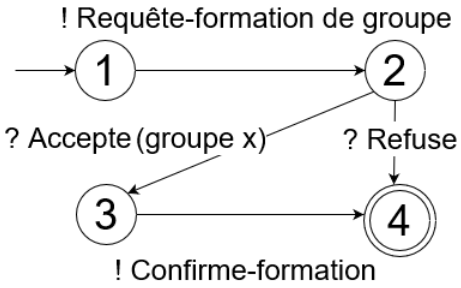


Figure 5.2 – Instance du protocole de proposition de formation de groupe

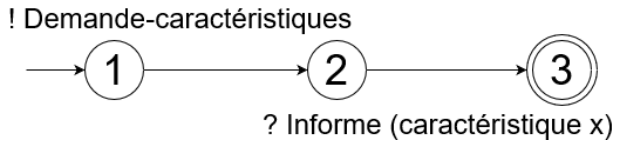


Figure 5.3 – Instance du protocole de demande d'information

Les coalitions formées grâce à ce protocole d'interactions sont distribuées dans la mémoire de chaque agent. Puisqu'il n'existe pas de représentation centralisée des coalitions et que chaque membre a la charge de sa propre représentation de la coalition, il est nécessaire aux agents de se synchroniser pour modifier la composition d'un groupe. Il faut par exemple éviter qu'un membre ajoute un nouvel agent au groupe tandis qu'un autre membre le quitte. Par défaut, le nouvel entrant suppose que personne n'a quitté le groupe entre le moment où il a reçu l'invitation et le moment où il l'a acceptée. Par conséquent, modifier la coalition pendant ce laps de temps provoquerait un manque de cohérence parmi les représentations de la coalition des différents membres. Les agents ont donc besoin de se coordonner pour maintenir une même représentation de leur groupe. Pour cela, les coalitions implémentent un système de *leader* capable de coordonner les membres d'une coalition lorsqu'ils souhaitent la modifier. L'agent à l'origine de la formation du groupe est automatiquement considéré comme son *leader*. Le *leader* initialise un sémaphore sur lequel il a le contrôle. Son rôle est d'indiquer aux autres membres quand ils peuvent ou non modifier la coalition. Il accorde son autorisation à un membre du groupe par l'envoi d'un jeton. Un agent souhaitant ajouter un nouveau membre au groupe doit utiliser le protocole de demande de jeton illustré dans la figure 5.6. Notons que le transfert du jeton ne peut être refusé ce qui évite aux agents d'une coalition les situations de blocage. Lorsque les agents ont fini de modifier le groupe, le jeton est renvoyé au *leader* qui peut alors le transmettre à d'autres membres. Le rôle de *leader* et le sémaphore qui lui est rattaché sont associés à un groupe. Un agent peut donc être le *leader* de plusieurs coalitions qui possèdent toutes leur propre sémaphore. Lorsque le *leader* d'un groupe souhaite le quitter, il nomme son successeur et transmet son identifiant à tous les membres du groupe.

Dans les protocoles permettant la création ou la modification d'une coalition, les agents sont soumis à un message de confirmation permettant à l'initiateur de la conversation de savoir que le message a bien été traité par son destinataire. Le mécanisme de confirmation est particulièrement important puisque l'expéditeur du message n'exécute son action qu'une fois certain que son message a bien été traité. Par exemple dans le cas d'un abandon de coalition, il notifie tous les membres du groupe et ne le quitte réellement que lorsque tous les destinataires ont renvoyé la

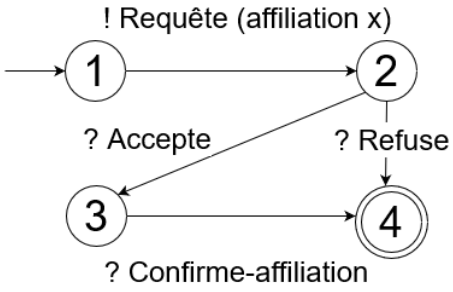


Figure 5.4 – Instance du protocole de demande d'affiliation à un groupe

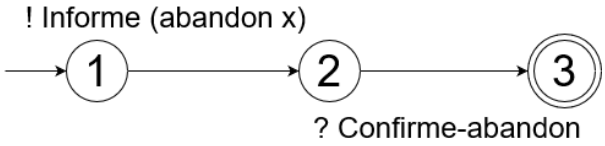


Figure 5.5 – Instance du protocole d'abandon de groupe

confirmation. Ce mécanisme de confirmation a pour intérêt de permettre au leader d'être certain que tous les membres de sa coalition ont pris en compte la modification du groupe au moment où il récupère le jeton de l'accointance initiatrice du changement.

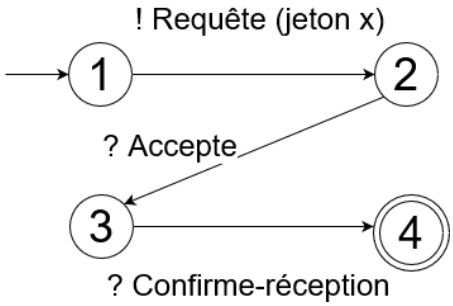


Figure 5.6 – Instance du protocole de demande de jeton

5.3.5 Synthèse

Cette partie a présenté divers mécanismes permettant de répondre à deux de nos besoins : l'évaluation sociale et les interactions inter-agents. L'architecture ABSG confère aux agents un comportement dynamique en fonction des connaissances qu'ils ont de leurs pairs et de la valeur d'attraction qu'ils leur attribuent. Son cycle de décision permet la mise à jour des accointances et les règles de prises de décisions utilisent fortement le concept d'attraction pour différencier une accointance attractive, avec qui les agents veulent former une coalition, de celles qui présentent moins d'intérêts. La fonction d'attraction permettant à un agent d'évaluer ses pairs s'inspire des principes des *Dynamiques de Groupes* et nous avançons que certains de ces principes permettent d'améliorer la qualité des coalitions formées par rapport à un processus plus classique dans lequel les agents recherchent uniquement l'adéquation de leurs désirs aux caractéristiques de leurs accointances. Cette supposition fera l'objet d'une évaluation dans le chapitre 7. Enfin, les mécanismes d'interaction intégrés à l'architecture ABSG permettent aux agents de s'échanger des

connaissances et de se synchroniser grâce à la mise en place d'un langage commun et à la conception d'un protocole de communication.

Dans la partie suivante, nous répondrons au besoin de l'adaptation de l'architecture à notre contexte applicatif : le système d'aide à la décision pour le conception de produits remanufacturés.

5.4 Mécanismes associés au cas applicatif

L'architecture ABSG est basée sur des capacités cognitives permettant aux agents de manipuler des connaissances. L'attraction est une métrique importée des dynamiques de groupes dessinant les comportements de nos agents. Cette mesure se veut très générique et adaptable selon le contexte d'utilisation. L'architecture et notre métrique d'attraction possèdent en effet le même fonctionnement indépendamment du contexte applicatif du système. L'architecture telle que présentée dans la section précédente permet aux agents de se regrouper en coalitions en prenant en compte des critères d'appariement peu précis comme l'adéquation des désirs d'un agent par rapport aux caractéristiques de ses accointances. Cependant, l'adéquation des désirs d'un agent *A* pour les caractéristiques d'un agent *B* peut être très mauvaise et quand même être la meilleure disponible dans le système, même si pour cela les contraintes physiques des composants ne sont pas respectées. Par exemple dans le cas d'un lave-linge, un agent *cuve* peut vouloir trouver un agent *tambour* du plus petit volume possible afin de rendre possible l'assemblage. Mais l'agent *tambour* possédant le plus petit volume peut être plus volumineux que l'agent *cuve*. L'adéquation est mauvaise, pour autant elle est la meilleure possible. Mais il ne faut pas que ces deux agents forment une coalition ensemble car ils sont physiquement incompatibles. Les mécanismes présentés jusqu'à présent ne permettent pas à nos agents de décrire une contrainte empêchant les agents de former des coalitions si certaines contraintes physiques ne sont pas respectées. L'architecture ABSG requiert des mécanismes de description de contraintes plus précises que les désirs des agents. Nous présentons dans cette partie des mécanismes permettant l'ajout de capacités descriptives aux agents ABSG. Nous présentons également des mécanismes plus spécifiques à notre contexte applicatif afin d'adapter l'architecture ABSG aux applications de conception de produits pour le génie industriel.

5.4.1 Contraintes

Conférer des désirs aux agents ABSG les incite à rechercher la présence de certains agents spécifiques dans leur coalition. Mais lorsque leurs désirs ne peuvent pas être satisfaits, les agents cherchent à les combler grâce à la métrique de l'attraction par la présence dans leur coalition de ceux qui possèdent les caractéristiques les plus proches de leurs désirs. Reprenons l'exemple du tambour et de la cuve, si la plus grande cuve que le tambour trouve dans le système était exactement du même volume que lui, les agents ne devraient pas s'apparier puisque les composants ne pourraient être assemblés. Il est donc nécessaire de pouvoir spécifier des contraintes aux agents pour leur éviter de concevoir des produits irréalistes.

De la même manière que les agents ABSG associent des désirs à chacune des caractéristiques de leurs accointances, les contraintes s'associent à un désir et une caractéristique permettant de définir d'une part sa valeur de référence (le désir) et d'autre part la caractéristique concernée. Par exemple, l'agent *tambour* souhaite contraindre son choix de cuve sur la caractéristique *volume* en recherchant une cuve d'un volume égal au désir renseigné. Ainsi, l'expression des désirs des agents peuvent être vus comme des tuples $\langle contraintes, désir, caractristique \rangle$ pouvant s'interpréter comme : *"Je recherche un agent dont la caractéristique* C^a *respecte la ou les contraintes* C^o *par rapport au désir d"*. Notons qu'afin de décrire au mieux les contraintes entre les composants, l'une des caractéristiques d'une accointance peut être la cible de plusieurs désirs et plusieurs contraintes. Cela peut par exemple être utile dans le cas où le volume de la cuve doit être supérieur au volume de tambour tout en étant inférieur à un volume maximum.

Après avoir défini le lien entre contraintes, désirs et caractéristiques, nous présentons les différents types de contraintes que peuvent manipuler les agents ABSG. De la même manière que dans l'exemple du tambour et de la cuve, il est nécessaire que les agents puissent exprimer le besoin de s'apparier uniquement avec des agents possédant une caractéristique strictement inférieure ou supérieure à un de leurs désirs. Il est aussi possible d'imaginer que deux composants puissent s'assembler dans le cas où l'une de leurs caractéristiques est identique. Nous proposons donc les contraintes : *LSS* (strictement inférieur), *GRT* (strictement supérieur), *LEQ* (inférieur ou égal), *GEQ* (supérieur ou égal) et *EQ* (égal). Certain cas exigent également une plus grand indifférence dans la conception du produit, par exemple sur des caractéristiques de moindre importance, comme la couleur d'un produit dont l'utilisateur se soucie peu. Celui-ci devrait alors soit pouvoir renseigner son indifférence, soit exprimer un unique veto. Nous ajoutons donc les contraintes : *ALL* (tout) et *NEQ* (différent de). Enfin, la dernière contrainte qui est le fonctionnement par défaut de nos agents est la contrainte *ANY* (tout, au plus proche) qui permet aux agents de rechercher les caractéristiques les plus proches de leurs désirs sans autre restriction. Notons que puisqu'il est possible d'associer plusieurs contraintes à une même caractéristique, il est tout à fait possible à des agents de rechercher des accointances possédant des caractéristiques dans un intervalle de valeur en utilisant les contraintes *GRT* et *LSS*. Par exemple, un tambour peut souhaiter une cuve plus volumineuse que lui mais tout de même moins volumineuse qu'un châssis spécifique. Il est aussi possible de rechercher des composants dans une gamme de prix tout en le minimisant en appliquant la contrainte *GRT* et *ANY* dans le cas où l'on recherche un composant au prix s'approchant le plus possible de 20€ sans y être inférieur.

Pour résumer, les contraintes manipulables par les agents ABSG sont les suivantes :

— *ANY* : rechercher une caractéristique au plus proche du désir fourni ;

— *LSS* : rechercher une caractéristique inférieure au désir fourni ;

— *LEQ* : rechercher une caractéristique inférieure ou égale au désir fourni ;

— *GRT* : rechercher une caractéristique supérieure au désir fourni ;

— *GEQ* : rechercher une caractéristique supérieure ou égale au désir fourni ;

— *EQ* : rechercher une caractéristique égale au désir fourni ;

— *NEQ* : rechercher une caractéristique différente du désir fourni ;

— *ALL* : ne pas chercher de caractéristique spécifique.

Ce mécanisme de contrainte permet aux agents ne de pas s'associer à ceux qui ne respectent pas les contraintes physiques des composants du système d'aide à la décision. Cependant, les agents agissent de manière égoïste dans le système multi-agent et il est possible qu'un des membres d'une coalition ajoute un agent au groupe qui ne respecte pas les contraintes d'un autre agent. C'est par exemple le cas si dans le contexte d'un vélo et de la coalition *guidon, roues* les roues décident d'ajouter un agent *cadre de vélo* possédant la couleur rouge tandis que le guidon souhaite un cadre de couleur bleue. Le non respect d'une contrainte d'un agent entraîne une forte dégradation de l'attraction qu'il a pour sa coalition et implique son abandon. En outre, les agents ont la capacité de mémoriser les caractéristiques des agents qu'ils ont rencontrés dans une coalition ce qui leur permet d'éviter d'y retourner si un agent qu'ils apprécient leur demande de revenir.

La section suivante présente un autre mécanisme spécifique à notre contexte applicatif adaptant un peu plus le comportement des agents ABSG pour la conception de produits.

5.4.2 Transformations

Une des spécificités du remanufacturing est de laisser la capacité aux industriels récupérant des composants de leur appliquer une transformation afin de les adapter au mieux à leur nouvelle application. Comme nous le verrons dans le chapitre suivant, la transformation de composants est encore une pratique rendue très compliquée –voire interdite– par la législation. Les transformations pratiquées de nos jours sont plus de l'ordre de la réparation que d'une réelle modification des composants. Elles consistent par exemple en la reprogrammation des cartes électroniques des lave-linge pour restaurer leurs fonctionnalités initiales ou encore à remplacer un connecteur endommagé par un de même type sur un appareil électronique. L'économie circulaire et le remanufacturing se développant de plus en plus, la capacité à transformer des composants est une méthode intéressante permettant de minimiser notre émission de déchets, nous présentons donc tout de même la capacité de transformation des agents ABSG pour s'adapter à leur nouvelle application.

Dans l'architecture ABSG, une transformation est une opération consistant à modifier l'une des caractéristiques d'un agent. Les caractéristiques peuvent se voir associer plusieurs transformations de façon à pouvoir décrire diverses manières de modifier un composant physique. Ainsi, si l'on veut décrire la modification d'un contrepoids de lave-linge, on peut lui associer une opération pour augmenter son nombre de fixations (ce qui correspond par exemple, physiquement, à lui en percer une nouvelle), mais aussi une opération pour le réduire (ce qui correspond à ne pas toutes les utiliser au moment de l'assemblage avec une cuve).

Lorsqu'un agent calcule l'attraction qu'il a pour ses accointances, il a le choix d'utiliser ou non leurs transformations en fonction des cas. Dans le cas où l'agent souhaite former une nouvelle coalition, celui-ci peut sélectionner les transformations de l'accointance qui l'intéresse. Si l'attraction pour l'accointance transformée est bonne, alors l'agent lui envoie une demande de formation de groupe en renseignant les transformations nécessaires. De la même manière, si un agent veut envoyer une demande d'affiliation, alors il pourra demander au nouveau venu de se transformer. Au contraire, dans le cas d'une réception d'une demande d'affiliation, l'agent ne pourra demander au membre de la coalition de se transformer puisque ce dernier est déjà intégré au groupe et modifier ses propres caractéristiques pourrait détériorer la qualité de la coalition.

5.4.3 Synthèse

Nous avons présenté dans cette partie des mécanismes spécifiques au cas applicatif qu'est le remanufacturing. La spécification de contraintes physiques apporte à nos agents la capacité de manipuler des connaissances plus détaillées et leur offre la possibilité de concevoir des produits plus complexes. La possibilité de définir des contraintes physiques et des transformations applicables aux composants rend notre architecture plus apte à s'attaquer à des problèmes réalistes liés à l'économie circulaire. Dans la partie suivante, nous présenterons la façon dont nos agents communiquent et interagissent pour se synchroniser et rendre la formation de coalitions possible.

5.5 Environnement

L'environnement des agents est composé de deux parties avec lesquelles ils peuvent interagir. La première partie est représentée par les autres agents du système avec qui il leur est possible de communiquer selon les protocoles présentés dans la partie précédente. La seconde composante de l'environnement tient moins du système multi-agent que du système d'aide à la décision. La figure 5.7 présente cette seconde partie qui est un système d'information auquel peuvent se connecter les agents du système. Celui-ci contient toutes les connaissances renseignées par les divers utilisateurs du système d'aide à la décision [Bettinelli2020c]. Nous les détaillons ci-dessous :

— l'objectif requêté par l'utilisateur : le produit est décrit par l'utilisateur. Les caractéristiques du produit décrit représentent les désirs des agents. Par exemple si un vélo rouge est demandé, les agents auront comme désir la formation d'un groupe avec d'autres agents possédant la caractéristique *couleur rouge* ;

— la description des composants : cette description représente toutes les caractéristiques associées aux composants utilisables dans le système d'aide à la décision. Elles sont renseignées par un ou des fournisseurs. Ce sont ces caractéristiques qui sont transférées aux agents ;

— la description des transformations : les fournisseurs peuvent indiquer leurs capacités à modifier un de leur composant afin d'aider son intégration à un nouveau produit. Il peut par exemple s'agir de la possibilité de modifier la taille d'une pièce ou de modifier sa couleur. Les transformations sont entrées dans le système d'information par le fournisseur. Le mécanisme de transformation peut permettre aux agents de rentrer dans des coalitions dans lesquelles leur présence ne serait pas souhaitable autrement. Par exemple un agent *cadre de vélo* de couleur bleue ne serait pas accepté dans une coalition nécessitant des composants rouges. Cependant, le fournisseur du composant peut indiquer au système sa capacité à repeindre le composant. Ainsi, un *cadre de vélo* de couleur bleue pourrait changer sa couleur pour s'intégrer à la coalition ;

— les connaissances du domaine d'expertise : ces connaissances sont une description du domaine d'expertise du fournisseur afin d'indiquer aux agents avec qui ils peuvent s'associer. Par exemple, indiquer qu'un *cadre de vélo* peut s'associer avec un *guidon*. Comme pour les descriptions des composants et les transformations, le domaine d'expertise est rensei-

gné par un fournisseur. Ces connaissances sont un complément d'information à l'objectif de l'utilisateur et permettent aux agents s'associer avec les bons types d'accointances.

Les désirs des agents sont représentés par la description du domaine d'expertise et le produit demandé par l'utilisateur. La première indique aux agents quels sont les composants avec lesquels former une coalition. Par exemple dans le cas d'un lave-linge une *cuve* s'assemble avec un *contrepoids* pour éviter à l'appareil de se déplacer pendant les phases de rotation rapide. Le second renseigne les caractéristiques détaillées des composants qu'ils doivent rechercher. Par exemple, la *cuve* doit s'associer avec un *tambour* de 10L pour accueillir le linge de de ses utilisateurs. Lorsque la demande utilisateur est modifiée ou lorsque le fournisseur change la description du domaine d'expertise, le système d'information envoie les nouvelles connaissances aux agents concernés afin de leur permettre de se réorganiser. Les caractéristiques des agents sont représentées par la description des composants. Lorsqu'un composant est modifié, le système d'information envoie la mise à jour à l'agent concerné.

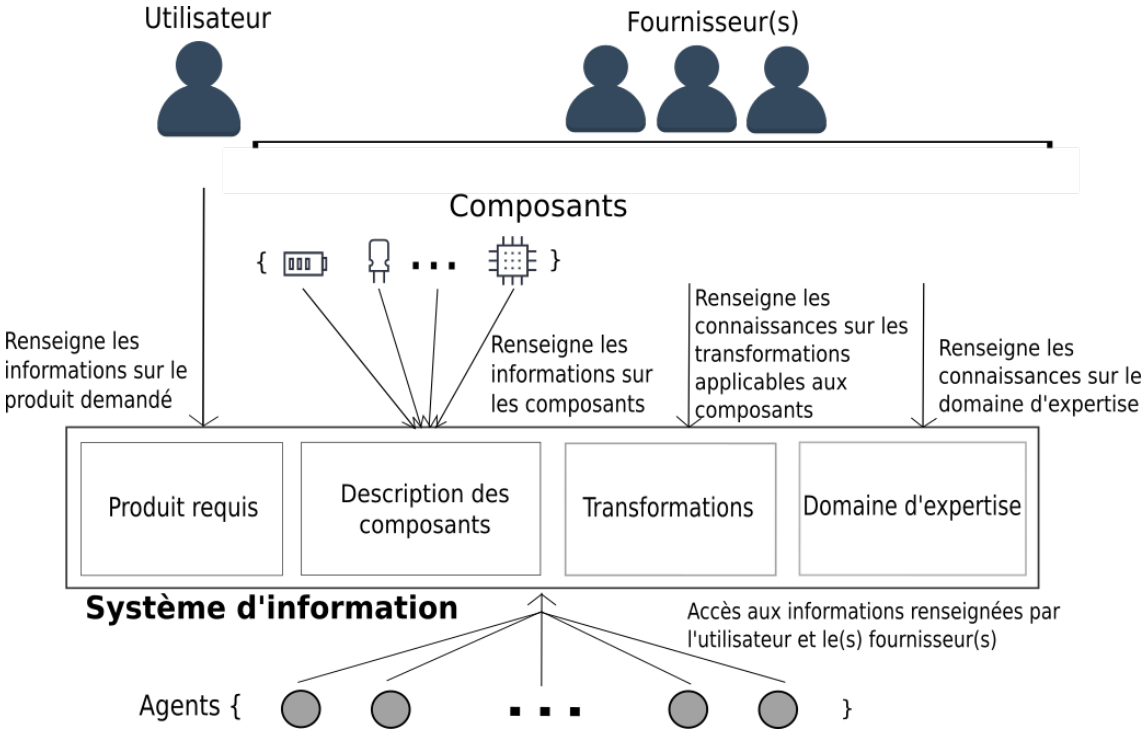


Figure 5.7 – Système d'information

5.6 Exemple de fonctionnement

Pour illustrer le fonctionnement du modèle, nous imaginons un cas d'étude très simple (figure 5.8) dans lequel l'objectif du système est de construire une batterie de véhicule électrique. Pour cela, il a à sa disposition les composants suivants : des cellules électriques, des modules de cellules permettant de monter les cellules en série, des systèmes de refroidissement, des boîtiers et

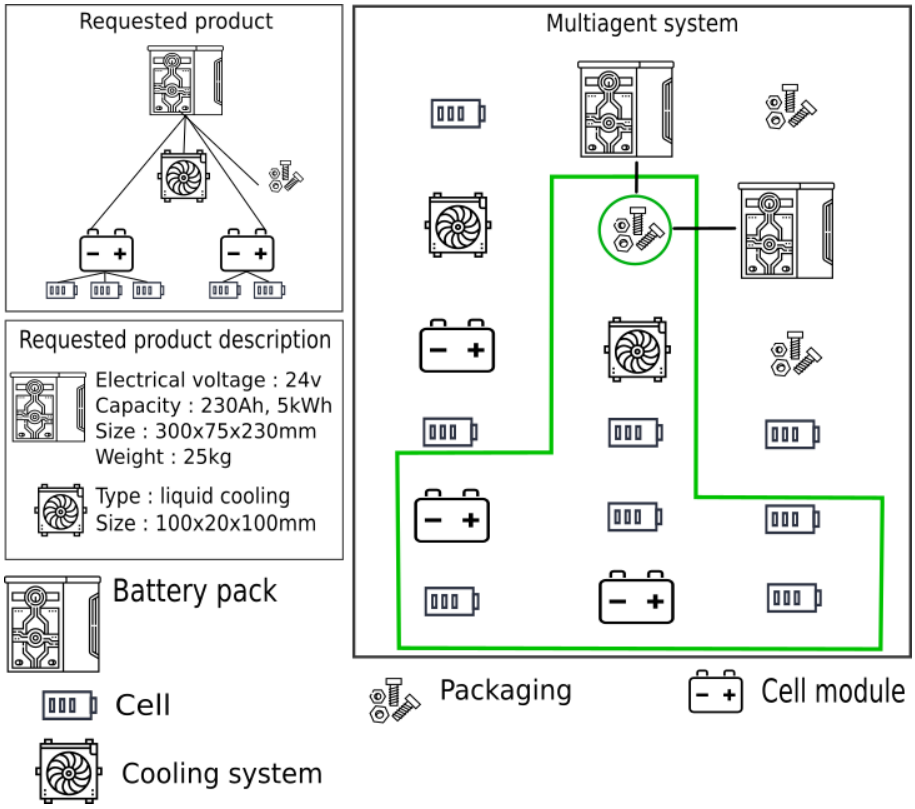


Figure 5.8 – État du système d'aide à la décision pour un cas d'étude sur les batteries de véhicule électrique

des ensembles de vis et écrous pour assembler les composants. Le plan de montage d'une batterie électrique est présenté en haut à gauche de la figure 5.8. Le système multi-agent qui contient les agents est présenté sur la droite de la figure. Une coalition est déjà présente au sein du système, elle est détourée par une ligne pleine de couleur verte et contient tous les composants nécessaires excepté le boîtier de la batterie. En observant le plan de montage d'une batterie, nous pouvons observer que plusieurs composants ont le désir de se connecter au boîtier. Il se trouve que dans notre cas pour les besoins de l'explication, l'agent *vis* est celui qui a trouvé les deux agents *boîtiers* et qui se demande lequel des deux ajouter. Mais avant d'ajouter un membre à la coalition, l'agent *vis* vérifie qu'une règle prioritaire sur l'affiliation soit exécutable. Supposons que ce n'est pas le cas et que l'agent n'a reçu aucun message. Il lance la procédure d'ajout d'un membre à sa coalition. Pour cela, il compare les valeurs d'attraction qu'il a pour les deux agents *boîtiers* et sélectionne celui qui a la plus faible. Il vérifie ensuite que cet agent n'est pas déjà dans la coalition et qu'il détient bien l'autorisation du leader de la coalition d'ajouter un membre. Dans cette coalition le leader est un des agents *cellules*. Notre agent *vis* utilise donc le protocole de demande de jeton et envoie sa requête au leader. Le leader lui retourne rapidement le jeton pour lui signifier l'autorisation de modification du groupe. L'agent *vis* envoie une demande d'affiliation à l'agent *boîtier* (figure 5.9). Dans cet exemple d'interaction les agents exécutent le protocole décrit dans les parties précédentes. L'agent *vis* termine son cycle de décision en traitant les messages reçus et note la présence d'un message en attente. Il s'agit de la notification de refus de l'agent *boîtier* pour la demande d'affiliation. Dans ce cas, cela signifie que ce dernier possède une mauvaise valeur attraction pour

les membres présents dans cette coalition. L'agent *vis* décide donc de noter le rejet de sa demande et de mettre à jour les valeurs d'attraction qu'il a pour ses accointances et ses coalitions. Le rejet du *boîtier* induit une augmentation du principe *minimax* et détériore l'attraction qu'il a pour son accointance. Lors du cycle de décision suivant, la règle d'affiliation est de nouveau la seule applicable. L'agent *vis* compare à nouveau les *boîtiers* disponibles et voit que, cette fois, l'autre *boîtier* possède une meilleure attraction que celui qui vient de rejeter la demande. La demande d'affiliation lui est envoyée et le destinataire accepte l'offre rendant la coalition complète.

Pour cela, les agents interagissent grâce au protocole défini dans la partie précédente. La figure 5.9 illustre l'échange de messages entre les agents A et B du système pour la formation d'un groupe.

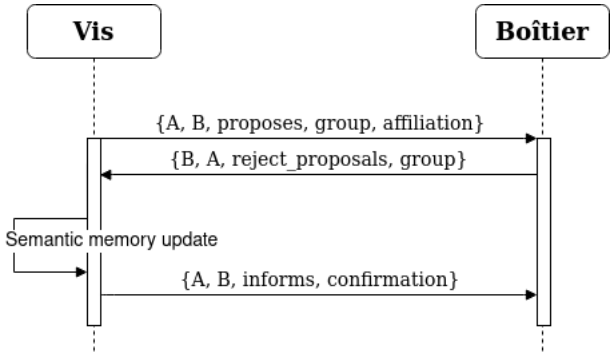


Figure 5.9 – Exemple de deux agents formant un groupe

5.7 Synthèse et discussion

L'architecture ABSG répond à nos besoins initiaux qui étaient ceux de faire interagir les agents, de les rendre capable d'évaluer leurs accointances et de s'adapter à certaines spécificités du remanufacturing. Les agents ABSG possèdent des capacités cognitives à travers leurs mémoires et leurs mécanismes de décision leur permettant d'évaluer leurs pairs selon des principes inspirés des sciences sociales. Ces capacités et mécanismes leur confèrent également la capacité d'agir dynamiquement dans un environnement variable auquel ils doivent s'adapter pour former des coalitions. Mais les coalitions ne doivent pas seulement être théoriquement viables, il est nécessaire que les conceptions produites par le système multi-agent soient cohérentes avec les spécificités physiques des composants et les capacités de transformation que leurs fournisseurs possèdent. Les mécanismes de contraintes et de transformation intégrés à notre architecture offrent la possibilité aux agents ABSG de s'adapter à ce contexte applicatif.

La possibilité de spécifier des contraintes sur les désirs des agents améliore grandement la représentation de l'objectif des divers acteurs du système. Celles-ci pouvant se cumuler sur un même désir, l'objectif peut être décrit avec précision en utilisant par exemple les contraintes *LSS* et *GRT* pour indiquer un intervalle de valeurs acceptables pour les valeurs quantitatives, ou une succession de *EQ* pour désigner les valeurs qualitatives souhaitées. Comme nous l'avons vu, ces

connaissances sont apportées par deux types d'acteurs ; d'un coté les utilisateurs qui renseignent leur objectif, de l'autre les fournisseurs qui intègrent dans le système les informations sur les composants et les connaissances liées au domaine d'expertise (notamment comment assembler les types de produits dont ils sont experts). La dualité de la provenance des connaissances peut permettre à un large panel d'utilisateurs d'user du système. En effet, si toutes les connaissances expertes liées au domaine sont renseignées par les fournisseurs, il suffirait à un utilisateur novice de demander au système un lave-linge d'une certaine capacité sans avoir à décrire les caractéristiques précises des composants nécessaires à sa conception. Un tel système pourrait donc être aussi bien utilisé par un professionnel du remanufacturing qui sait exactement ce dont il a besoin que par un client derrière l'interface web de son entreprise préférée d'appareils électroménagers.

Comme nous l'avons vu, cette architecture d'agent possède les traits des architectures cognitives qui sont en général moins utilisées dans les applications nécessitant un large passage à l'échelle. Or, le domaine applicatif de ce système multi-agent est la conception de produits remanufacturés. Nous pouvons supposer que ce type d'application devrait nécessiter de nombreux agents pour fonctionner de manière réaliste. Dans la dernière partie de ce chapitre où nous avons présenté un exemple de fonctionnement d'une instance du système, nous sommes partis du principe qu'un agent représentait un unique composant physique. Mais le passage à l'échelle sur des milliers de composants avec une architecture cognitive serait extrêmement complexe. En réponse à cet inconvénient, nous pouvons également voir chaque agent comme un lot de composants possédant des caractéristiques similaires. Bien qu'il soit peut aisé pour deux composants d'être physiquement identiques, nous pouvons supposer que devant le grand nombre de composants manipulés par les sociétés de remanufacturing il serait possible de les regrouper par similitudes afin de les réunir en un seul agent. Finalement, si chaque agent du système représente des lots de composants de même taille, le passage à l'échelle en terme de nombre d'agents peut être évité et le passage à l'échelle en terme de nombre de composants n'aurait pas d'impact sur le fonctionnement du système multi-agent tel qu'il a été présenté dans ce chapitre.

La question du passage à l'échelle soulève des interrogations liées au domaine du remanufacturing. Par exemple, sur le nombre de composants pouvant être manipulés par les acteurs de ce domaine mais aussi sur la façon dont se font les transformations du point de vue industriel ? Le chapitre suivant présente un cas d'étude construit à partir d'une collaboration avec des entreprises remanufacturant des appareils électroménagers. Celui-ci détaille la façon dont s'assemblent et fonctionnent les lave-linge ainsi que les transformations applicables à leurs composants. Nous évaluerons l'architecture ABSG à travers ce cas d'étude dans les chapitres suivants afin de la tester dans un cadre réaliste.

6

Vers l'application au système d'aide à la décision

Sommaire

Après avoir présenté la conception de notre modèle d'agent et son fonctionnement, ce chapitre introduit un cas d'étude tiré des industriels SOFIE et ENVIE sur lequel nous l'évaluerons. Comme nous le verrons, ce cas d'étude se concentre sur de l'équipement électroménager, spécialité de ces sociétés.

6.1 SOFIE et ENVIE

SOFIE est basée à Seraing en Belgique. C'est une société coopérative à finalité sociale qui œuvre dans le domaine de la collecte, du tri, du démantèlement et de la réutilisation de déchets d'équipements électriques et électroniques (DEEE) provenant d'appareils électroménagers.

ENVIE est un réseau d'entreprises d'insertion professionnelle œuvrant pour l'économie circulaire en France. Leur activité est très similaire à celle de SOFIE : ils collectent, nettoient et trient des DEEE dans l'objectif de les réutiliser pour la réparation de divers types d'appareils. Leur réseau étant composé de plusieurs entreprises, ils travaillent sur une gamme plus large de produits. Dans le cadre de ce travail, nous avons contacté ces deux entreprises et visité les locaux d'ENVIE sur le site de Villeurbanne dans lequel ils démontent, nettoient et réparent des appareils électroménagers.

6.1.1 Activités de remanufacturing

ENVIE et SOFIE remanufacturent une large gamme de produits électroménagers. Ceux-ci sont en général complexes et fabriqués à partir de nombreux composants. Les pièces possèdent une forte diversité ce qui les rend très peu interchangeables. En effet, chaque constructeur produit ses machines avec ses propres spécificités pour répondre à ses propres besoins. Il peut par exemple s'agir des fixations d'un composant de lave-linge qui ne sont pas placées aux mêmes endroits pour toutes les machines empêchant d'échanger une pièce entre deux modèles, même si les dimensions correspondent. De la même manière, tous les composants électroniques ne nécessitent pas la même puissance électrique pour fonctionner ce qui contraint fortement le choix des composants au moment de leur remplacement. D'un point de vue technique, ce manque de normes entre les composants de constructeurs différents – et parfois entre les machines d'un même constructeur – complexifie fortement le processus de remanufacturing. Pour cette raison, les pièces sont le plus souvent remplacées par leurs homologues d'une autre machine du même modèle. Par conséquent, dans le cas où une pièce ne peut pas être trouvée dans une autre machine il est nécessaire de l'acheter neuve chez le fabricant d'origine.

6.1.2 Défis futurs

En imaginant que le remanufacturing se développe dans les prochaines années, le nombre de produits et de composants réutilisables deviendra rapidement très important. Une grande partie des sociétés qui pratiquent le remanufacturing sont aujourd'hui des PME qui réutilisent les composants de grands groupes qui conçoivent les produits. Par exemple, lorsque les employés de

SOFIE ne peuvent pas réparer une machine par manque de pièce de rechange, ils rachètent les composants nécessaires chez les grands groupes tels que Bosch. Les ateliers de réparation de ces PME comptent sur l'expertise de leurs employés pour savoir quels composants réutiliser selon les machines à réparer. Cependant, avec le développement du remanufacturing, l'expertise humaine dont ils bénéficient actuellement pourrait être dépassée devant l'explosion du nombre de composants réutilisables et de leur diversité. Un système d'aide à la décision pour la conception de produits présente l'avantage de répondre à ce problème en donnant la possibilité à une entreprise ou un consommateur de concevoir des produits sans que son utilisateur n'ait à connaître lui-même le stock de composants disponibles, ni n'ait même nécessairement des connaissances expertes sur la constitution du produit souhaité.

6.2 Cas d'étude

Les produits électroménagers comportent souvent un grand nombre de composants. Par exemple, une machine à laver est composée d'environ une dizaine de pièces importantes. On peut y inclure le moteur, la cuve, le tambour, le programmateur, la pompe de vidange, *etc*. Mais il en existe un bien plus grand nombre si nous prenons en compte les joints, les condensateurs, les câbles et les vis. Le grand nombre de pièces à manipuler est un intérêt pour la construction de notre cas d'étude puisqu'il permet de montrer comment notre système d'aide à la décision réagit avec une machine complexe et une étude de cas réaliste. En plus de leur expertise dans le remanufacturing, c'est pour cette raison que nous nous sommes tournés vers ces entreprises pour réaliser un cas d'étude.

Outre fournir une étude de cas réaliste, les lave-linge sont des appareils suffisamment riches en composants pour que nous puissions l'adapter à ce que nous souhaitons évaluer dans le chapitre suivant. Les lave-linge possèdent par exemple suffisamment de composants pour que nous n'ayons pas besoin de prendre en compte les petites pièces comme les joints ou les vis pour l'enrichir.

6.2.1 Sélection des composants

Une machine à laver contient de nombreuses petites pièces comme des joints, des vis et divers connecteurs pour assembler des composants électroniques. Dans ce chapitre, nous appellerons un composant une pièce de la machine ayant un rôle déterminant dans son fonctionnement comme un moteur, une vanne ou un programmateur. Pour des raisons pratiques, nous ne considérons que ce type de pièces pour le cas d'étude sans quoi il contiendrait probablement plus d'une centaine de pièces ce qui le rendrait inutilement détaillé. Les lave-linge démontés et remanufacturés que nous avons vus lors de la visite de l'entrepôt d'ENVIE ne comportaient pas la fonctionnalité de sèche-linge. Nous construisons donc notre cas d'étude autour de la conception de simples lave-linge.

Il existe deux types de lave-linge, ceux dont l'accès au tambour se fait par le haut (*top*) et ceux par lequel il se fait par l'un des côtés (*front*). Dans les deux cas le fonctionnement et les pièces sont les mêmes. La position des composants va par contre être très différente. Par exemple, dans un lave-linge *front*, un contrepoids est posé au dessus du tambour pour éviter que la machine ne

bouge lorsque celui-ci tourne trop vite. Sur un lave-linge *top*, le contrepoids ne pouvant pas être placé au-dessus du tambour, celui-ci est positionné sur l'un des côtés de la cuve. Cependant le fonctionnement des machines reste le même. Bien que cet appareil électroménager soit populaire, son fonctionnement n'est pas nécessairement bien connu. Afin de présenter les différents éléments du lave-linge, nous allons détailler son fonctionnement sur un cycle de lavage complet. Celui-ci est constitué de deux cycles, le prélavage et le lavage.

Le prélavage. Tout d'abord, l'utilisateur doit choisir son programme avec le bandeau de commande. Grâce à celui-ci, il agit sur le programmateur qui a pour rôle de synchroniser le fonctionnement de tous les composants électroniques de l'appareil. Par exemple, lorsqu'un lavage de 40°C est lancé, le programmateur verrouille la porte du lave-linge et actionne l'électrovanne permettant de remplir la cuve pour une phase de prélavage. Un capteur de niveau d'eau surveille le remplissage et s'assure d'un niveau suffisant pour commencer à brasser le linge. Une fois le niveau atteint le tambour qui se trouve dans la cuve est entraîné par un moteur placé sous cette dernière et commence à tourner lentement. Le moteur est relié au tambour par une courroie, une poulie et un roulement. Le roulement est inséré dans la cuve et permet la rotation de l'axe du tambour sans frottement. La poulie est fixée à l'axe du tambour et la courroie relie le moteur à la poulie ce qui permet la transmission de son mouvement de rotation au tambour. Après quelques minutes de brassage, la cuve est vidée à l'aide d'une pompe de vidange. Enfin, le prélavage se termine sur un essorage. La vitesse de rotation du tambour augmente alors à plusieurs centaines de tours par minute. Le linge ayant absorbé de l'eau, il est beaucoup plus lourd qu'au moment de son introduction dans la machine. Un contrepoids de 10 à 15 kilogrammes rattaché à la cuve permet d'amortir les vibrations induites par la grande vitesse de rotation du tambour.

Le lavage. Lorsque l'essorage prend fin, le lave-linge démarre un cycle de lavage presque identique à celui du prélavage. L'eau est d'abord introduite dans la cuve puis chauffée avant le brassage grâce à une résistance placée au fond de la cuve. La température de l'eau est surveillée à l'aide d'une sonde de température. Une fois la température voulue atteinte, le tambour commence une rotation lente pour brasser le linge. Le cycle de lavage continue ensuite comme le cycle de prélavage. Après un certain temps de brassage, l'eau est vidangée puis le linge essoré. Cet exemple permet de se rendre compte du grand nombre de composants présents dans la machine.

Pour résumer, les composants essentiels au bon fonctionnement du lave-linge sont les suivants :

— un bandeau de commande : qui fait office d'interface entre l'utilisateur et le programmateur de l'appareil. Ce bandeau permet à l'utilisateur de choisir son programme de lavage. Le bandeau de commande est directement connecté au programmateur ;

— un programmateur : permettant la coordination du fonctionnement de tous les éléments électroniques de l'appareil après la sélection du programme par l'utilisateur ;

— un bac à lessive : pour y déposer la lessive avant de lancer le lavage ;

— un moteur électrique : permettant de faire bouger le tambour et de brasser le linge ;

— un tambour : qui contient le linge ;

— une cuve : servant de récipient à l'eau et au tambour. Elle est fermée par le hublot.

— une résistance : pour chauffer l'eau de la cuve lors des lavages ;

— une sonde de température : pour s'assurer d'atteindre la bonne température en fonction du type de lavage sélectionné ;

— un roulement : permettant la rotation du tambour en évitant les frottements ;

— une poulie : rattachée à l'axe de rotation du tambour ;

— une courroie : liant le moteur et la poulie et permettant le transfert de la rotation de l'un à l'autre ;

— un contrepoids : pour amortir les vibrations de la machine ;

— une électrovanne : pour contrôler l'arrivée d'eau dans la cuve ;

— une pompe de vidange : pour retirer l'eau usée de la cuve à la fin du lavage ;

— un capteur de niveau d'eau : pour surveiller la quantité d'eau lors des différentes étapes de lavage.

Étant donné le nombre de pièces présentes dans cet appareil, cette liste n'a pas vocation a être exhaustive. Elle permet cependant d'avoir un aperçu des composants principaux d'un lave-linge. À cette liste, nous ajoutons le chassis qui n'a pas réellement de rôle dans le lavage mais est nécessaire à l'intégration de l'appareil dans un logement ou un local.

L'objectif de ce cas d'étude est d'évaluer notre système d'aide à la décision dans une application réaliste. Pour cela, il n'est pas nécessaire que les appareils à concevoir soient composés d'un grand nombre de composants. Une dizaine de composants suffit à illustrer et évaluer son fonctionnement. Pour cette raison, et puisque les trois composants suivants sont normalement assemblés les uns avec les autres, nous regroupons la sonde de température, le capteur de niveau d'eau et la résistance dans le composant *cuve*. De la même manière, la poulie et le roulement sont regroupés en un seul composant sous le terme *système de roulement*. Nous regroupons également le bandeau de commande et le programmateur qui ont un couplage très important. En effet, la forme des bandeaux est conçue pour s'adapter avec un programmateur ce qui rend difficile l'intéropérabilités de ces pièces avec d'autres modèles. Enfin, nous retirons le bac à lessive qui, en plus de ne pas avoir un rôle prépondérant dans le processus de lavage, ne présente pas une grande variété de caractéristiques ce qui le rend peu intéressant pour notre cas d'étude.

Pour résumer, nous sélectionnons les composants suivants :

— le programmateur ;

— le châssis ;

— le contrepoids ;

— le moteur électrique ;

— le tambour ;

— la cuve ;

— le système de roulements ;

— la pompe de vidange ;

— l'électrovanne.

6.2.2 Les composants et leurs caractéristiques

Les composants d'un produit possèdent leurs propres caractéristiques leur permettant de s'assembler ou non les uns avec les autres. Dans le cas du lave-linge la rotation du moteur entraîne la courroie ainsi que le système de roulement qui fait lui-même tourner le tambour. Pour que cela soit possible, il faut entre autre que le roulement possède le bon diamètre pour se fixer à l'axe de rotation du tambour et que la courroie ait la bonne longueur pour relier le moteur et la poulie. On voit donc qu'il n'est pas suffisant de prendre deux composants assemblables pour les faire fonctionner ensemble. En outre, les composants peuvent disposer de caractéristiques logistiques pouvant influencer leur assemblage. Nous pouvons imaginer le cas de conception d'un lave-linge dans une entreprise répartie sur plusieurs sites en France. Si un type spécifique de moteur devait se trouver uniquement dans un entrepôt nantais tandis que tout le reste des composants se trouve dans un entrepôt lyonnais, le coût de transport des pièces pourrait être suffisamment élevé pour vouloir changer la conception de l'appareil et y intégrer un autre type de moteur disponible avec le reste des pièces. À partir des composants sélectionnés dans la partie précédente et des discussions que nous avons eu avec les employés de chez ENVIE et SOFIE, nous proposons les caractéristiques de composants suivantes (table 6.1) avec lesquelles notre système d'aide à la décision devra travailler pour concevoir des produits.

Les entreprises pratiquant le remanufacturing sont soumises à deux problématiques : 1. la législation qui leur interdit de remplacer les composants d'un appareil par des composants transformés et 2. la complexité de réutiliser et assembler des composants qui n'ont pas été prévus pour fonctionner ensemble. Les PME comme SOFIE et ENVIE ne sont pas des constructeurs d'appareils électroménagers, ils ne produisent pas leurs propres produits et ne sont donc pas soumis aux mêmes règles lors de la conception des machines. Ils ne possèdent pas la même liberté d'assemblage des composants ni les mêmes obligations de test de leurs appareils que les constructeurs. En outre, les composants originaux sont conçus pour fonctionner dans un modèle spécifique de machine. Récupérer le composant d'une machine et le réparer ou le modifier pour l'intégrer à un nouvel appareil soulève des questions de sécurité et de responsabilité. Les PME pratiquant le remanufacturing sont donc limitées dans leurs possibilités de conception de produits. En plus de la legislation, la forme des composants et leur fonctionnement est peu propice à l'intéropérabilité. Les caractéristiques des composants étant nombreuses, leurs capacités de réutilisation sont très limitées entre les différents appareils. Pour cette raison, les caractéristiques sélectionnées (table 6.1) pour notre cas d'étude représentent uniquement un sous-ensemble de ce que l'on peut trouver sur des produits réels de façon à rendre plus évidente la réutilisation de composants. Comme expliqué dans la partie précédente, ce choix n'est pas naïf puisque le fort développement du remanufacturing pourrait amener les différents constructeurs d'appareils électroménager à rendre leurs composants plus interopérables. Un exemple est celui du châssis Malice (les lave-lings *top*) qui a initialement été crée par l'entreprise Fagor et qui est réutilisé depuis par de nombreux autres constructeurs.

Composants	Caractéristiques	Unité	Valeurs possibles	Descriptif
Programmateur	Type	∅	Electromagnétique / électronique	Correspond au type de fonctionnement du programmateur.
	Nombre de fils	∅	6 / 7	Dénomination faisant référence au nombre de fils du moteur de l'appareil. Les deux doivent être identiques pour pouvoir fonctionner ensemble.
Châssis	Type	∅	Top / front	Les machines tops ont une trappe d'accès au tambour au dessus de l'appareil. La trappe d'accès se trouve sur le coté des machines de type front.
	Diamètre de la trappe de vidange	cm	[15, 16]	La taille de la trappe du châssis doit être similaire au diamètre du tuyau de vidange de la pompe de vidange.
	Volume	L	[70, 120]	Le volume du châssis doit être adapté à ce qu'il contient .
contrepoids	Poids	kg	[10, 40]	Le poids du contrepoids doit être adapté à la capacité du tambour.
	Nombre de fixations	∅	[2, 4]	Le nombre de fixation du contrepoids doit être le même que celui de la cuve sur laquelle il va se fixer.
Moteur électrique	Type	∅	Induction / contact électrique par charbon	Un moteur à induction n'a pas besoin de système de roulements, il est directement fixé sur la cuve.

Composants	Caractéristiques	Unité	Valeurs possibles	Descriptif
	Nombre de fils	∅	6 / 7	Le nombre de fils doit être identique à celui du programmateur qui le contrôle.
	Type de courroie	∅	[1, 2]	L'élasticité de la courroie qui fonctionnera avec le système de roulements. Ne s'applique pas dans le cas d'un moteur à induction.
Tambour	Poids supporté	kg	[3-18] pour les fronts [5-8] pour les tops	La capacité de linge supportée en fonction du type de l'appareil.
	Volume	L	[20, 50]	Les dimensions du tambour doivent être inférieures à celles de la cuve.
	Diamètre de l'axe	cm	[1, 2]	L'axe permet au tambour de se visser sur le système de roulement.
Cuve	Volume	L	[40, 80]	La cuve doit être suffisamment petite pour rentrer dans le châssis
	Nombre de fixations	∅	[2, 4]	Ces fixations servent à la fixation du contrepoids.
	Puissance de la résistance	W	[1300, 2500]	La résistance chauffe l'eau de la cuve, plus la cuve est grande, plus la résistance doit être puissante.
	Diamètre du tuyau de sortie d'eau	cm	[15, 16]	Ce tuyau amène sur le tuyau de durite de la pompe de vidange qui expulse l'eau usée de la machine.

Composants	Caractéristiques	Unité	Valeurs possibles	Descriptif
Système de roulements	Diamètre intérieur de la poulie	cm	[1, 2]	Le diamètre intérieur de la poulie doit être le même que le diamètre de roulement du tambour afin d'assembler les deux composants.
	Type de courroie	∅	[1, 2]	La courroie doit être la même que celle supportée par le moteur. Selon leur type, les courroies ont différents niveaux d'élasticité.
Pompe de vidange	Diamètre du tuyau de durite	cm	[15, 16]	Afin de pomper les eaux usées, le tuyau de durite doit avoir le même diamètre que le tuyau de sortie d'eau de la cuve.
	Diamètre du tuyau de vidange	cm	[15, 16]	Ce tuyau mène à une trappe de vidange du châssis. Le tuyau de vidange de la pompe ne doit pas être plus large que le diamètre de la trappe du châssis.
Electrovanne	Type	∅	1 voie, 1 voie à sortie angle droit, 2 voies ou 3 voies	Ces différents types d'électrovanne ont un fonctionnement spécifique leur permettant de contrôler l'arrivée d'eau.

À ces caractéristiques sont ajoutés un prix et un lieu de stockage représenté comme une coordonnée *x* et une coordonnée *y* pour chaque type de composant afin à montrer la capacité du système de prendre en compte un large éventail de caractéristiques, dont certaines contraintes logistiques.

Afin d'améliorer leur intéropérabilité, nous proposons la possibilité de transformer les composants afin de les adapter à leur nouvelle application.

6.2.3 Les transformations

Nous définissons une transformation comme un procédé industriel exécutable par le fournisseur d'un composant capable de modifier les caractéristiques d'un composant dans le but d'aider à son intégration dans une nouvelle application. Une transformation peut-être effectuée dans le cas de deux stratégies de réutilisation : 1. le remanufacturing : pour adapter le composant d'un appareil et l'intégrer dans un autre produit du même type. Par exemple récupérer le programmateur d'un lave-linge et changer sa programmation pour l'adapter aux capacités de lavage d'un autre appareil. 2. le repurposing : pour intégrer un composant d'un appareil dans autre type d'appareil. Par exemple, récupérer des batteries de véhicules électriques pour les réutiliser dans une application de stockage d'énergies renouvelables irrégulières (éoliennes ou panneaux solaires).

Dans le cadre de notre système d'aide à la décision et de notre cas d'étude, nous allons nous concentrer sur les les transformations de composants pour le remanufacturing. Rappelons qu'il est actuellement difficile de transformer des composants pour des raisons techniques et juridiques. Puisque les entreprises actuelles n'ont pas le droit ni parfois les moyens de transformer des composants, nous proposons pour notre cas d'étude des transformations fictives :

— modification du nombre de fixations nécessaires à l'assemblage du contrepoids sur la cuve. Le contrepoids peut par exemple utiliser une fixation de moins ou de plus que ce qu'il possède pour s'assembler à une cuve. Pour l'ajout d'une fixation, cette transformation consisterait à en percer une supplémentaire dans le contrepoids pour l'adapter à celles de la cuve ciblée. Dans le cas où l'on souhaite fixer le contrepoids sur une cuve qui nécessite une fixation de moins, le procédé reviendrait à ignorer l'une des fixations du contrepoids. Pour des raisons de sécurité, nous supposons qu'il n'est pas envisageable de fixer un contrepoids sur une cuve en ignorant plus d'un de ses points d'ancrage et qu'il serait également dangereux de percer plus d'un point de fixation supplémentaire au risque de fragiliser la pièce ;

— changement de la résistance de la cuve en utilisant un composant plus ou moins puissant selon les besoins. Cette transformation ne nécessite pas le réusinage d'une pièce puisque les résistances sont des pièces facilement détachables de la cuve. Cependant, puisque nous considérons dans le cas d'étude la résistance comme faisant partie de la cuve, nous considérons également que la changer revient à transformer la cuve ;

— suppression de matière pour diminuer le diamètre de l'axe du tambour de façon à l'adapter au système de roulement. Le réusinage d'une pièce métallique pour adapter la largeur d'une pièce et refaire son filetage est un procédé maîtrisé par certains industriels. Nous ne rentrons pas dans le détail de l'intérêt du réusinage pour un axe de tambour et partons du principe que l'opération est peu coûteuse et rapide. Lorsque l'axe du tambour est trop large pour s'adapter au système de roulement, nous supposons que le fournisseur du composant peut

être capable de réduire son diamètre et refaire son filetage afin de le fixer au système de roulement voulu.

6.2.4 Synthèse

La figure 6.1 synthétise les composants et leurs caractéristiques introduits dans les parties précédentes. Les liens entre les composants indiquent ceux qui sont assemblables. Afin de rendre plus visibles les interactions inter-composants, les caractéristiques permettant d'assembler deux composants sont affichées sur ces liens. Par exemple, un programmateur et un moteur peuvent s'assembler à condition d'avoir le même nombre de fils. Plus précisément, qu'il y ait le même nombre de fils sortants du programmateur que de connectiques pour les brancher sur le moteur. De la même manière, une cuve ne peut s'assembler avec une pompe de vidange que si le *diamètre du tuyau de la sortie d'eau* est équivalent à celui du *diamètre du tuyau de durite* de la pompe. Le *diamètre du tuyau de durite* est une caractéristique appartenant à la pompe de vidange mais elle se trouve sur le lien du côté de la cuve pour indiquer qu'elle est une contrainte pour l'assemblage avec la cuve. De la même manière le *diamètre du tuyau de sortie de l'eau usée* est située vers la pompe de vidange puisqu'elle est une contrainte pour l'association avec cette dernière. Les caractéristiques présentes sur les liens représentent donc qu'une contrainte doit être respectée sur l'une des caractéristiques des deux composants. Cependant, cette contrainte n'est pas nécessairement une égalité de valeur. La cuve doit par exemple avoir un volume inférieur à celui du châssis pour pouvoir être montée dedans. Certain liens n'ont pas de contraintes comme celui présent entre le programmateur et le châssis. Cela indique que le programmateur se monte dans le châssis mais qu'il n'y a pas de caractéristique spécifique pouvant restreindre cet assemblage.

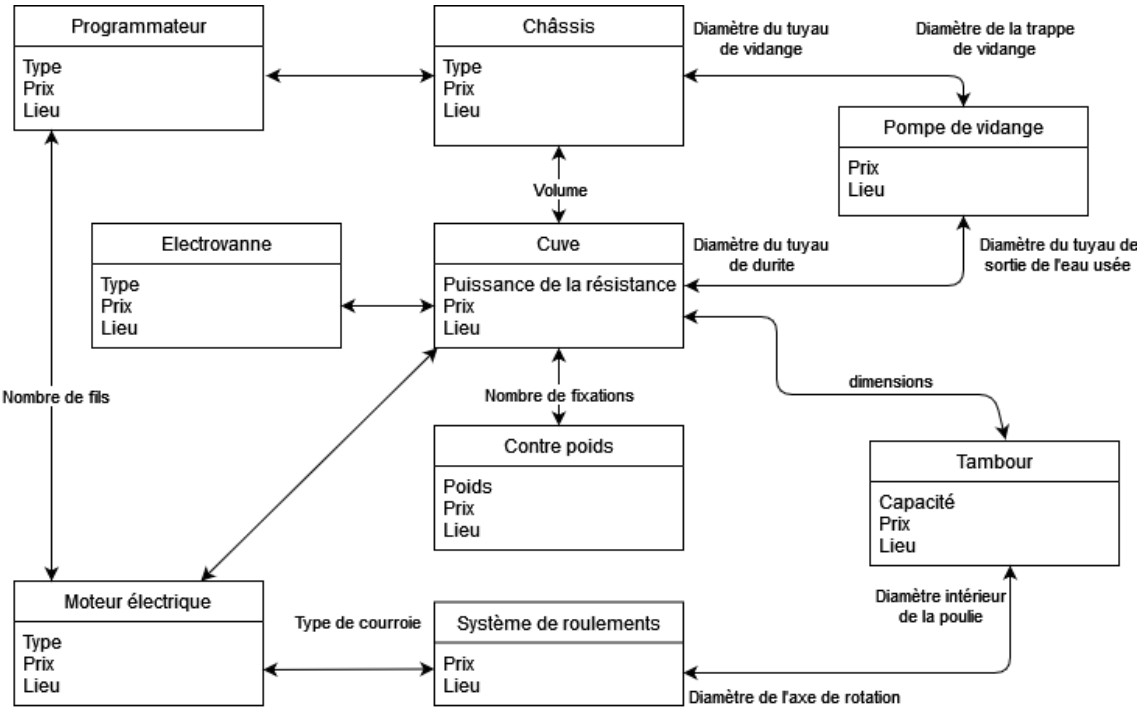


Figure 6.1 – Diagramme synthétisant les composants et leurs caractéristiques.

Toutes les données présentées dans ces dernières parties sont destinées à être saisies dans un

système d'information faisant partie du système d'aide à la décision afin que les agents du SMA puissent y avoir accès et mettre à jour leurs connaissances. La partie suivante présente la structure du système d'aide à la décision.

6.3 Architecture du système d'aide à la décision

Notre cas d'étude a partiellement été construit grâce aux connaissances transmises par ENVIE. Les entreprises du réseau ENVIE fournissent des appareils pour les revendre aux consommateurs à l'aide de leur stock de composants issus de DEEE et de leur expertise dans l'assemblage d'appareils électroménagers. Puisque ces entreprises réparent et fournissent des appareils aux consommateurs, nous les appelons des *fournisseurs* dans la suite de ce chapitre. ENVIE étant un réseau d'entreprises travaillant sur de nombreux types d'appareils, les divers acteurs du réseau peuvent travailler sur les mêmes types de machines. Notre système d'aide à la décision peut alors considérer la collaboration de plusieurs fournisseurs dans la conception des produits. L'intérêt de cette collaboration est par exemple le partage des composants réutilisés et le partage de l'expertise pour la conception d'appareils complexes. Cette collaboration peut se faire à travers un système d'information enregistrant les données des composants disponibles dans les entrepôts des différents fournisseurs, mais aussi en partageant la façon dont les produits qu'ils souhaitent remanufacturer sont assemblés. Ainsi l'expertise d'une des entreprises peut être réutilisée par les autres, voire augmentée s'ils possèdent aussi des connaissances dans le domaine. Pour illustrer, nous pouvons imaginer que les entreprises du réseau ENVIE remanufacturant les mêmes types d'appareils électroménagers mettent en commun leurs stocks de composants pour maximiser le nombre de pièces réutilisées. En outre, la mise en commun de leurs connaissances sur la conception d'un type d'appareil peut leur être bénéfique en apportant à chacun de ces acteurs une expertise différente et potentiellement complémentaire à la leur.

Toutes les données entrées dans le système d'information sont rendues disponibles au système multi-agent qui a pour objectif de concevoir les produits en fonction des connaissances fournies. En plus des informations données par les fournisseurs, le système a besoin de connaître le besoin du client pour lui proposer des produits. Ce dernier doit exprimer son besoin en décrivant de manière plus ou moins détaillée le produit qu'il souhaite construire. Notons que le client peut aussi bien être un consommateur manipulant le système d'aide à la décision à travers une interface web que l'un des fournisseurs qui souhaite produire des machines pour les vendre dans son magasin. La requête d'un client provoque la génération des agents du SMA via un processus synchronisant les composants du système d'information au SMA. À chaque nouveau composant, le processus ajoute un agent au SMA. Les composants étant modifiables, il est nécessaire que les agents –les avatars virtuels représentant un composant physique [Mrissa2015]– du SMA soient mis à jour en même temps que les informations sur leur avatar physique. Pour cela, les agents vérifient régulièrement les informations et les compare à leurs connaissances pour les mettre à jour au besoin. Tout au long du processus de conception des produits, un processus d'observation du SMA lit son état. Lorsque celui-ci semble avoir convergé vers un état stable, le processus propose les solutions au

client. Le client peut alors décider d'en sélectionner une ou de soumettre une nouvelle demande au système.

La figure 6.2 illustre l'architecture du système d'aide à la décision ainsi que son écosystème et certains de ses mécanismes. Les principaux éléments du système d'aide à la décision sont le système d'information (SI) et la couche logicielle sur laquelle les agents évoluent. Le SI contient toutes les informations dont a besoin le système d'aide à la décision pour fonctionner : l'objectif du client, la liste détaillée des composants disponibles, les possibilités de transformation des composants et les connaissances du domaine d'expertise des fournisseurs. La couche logicielle peut interagir avec le SI à travers l'utilisation d'une API. L'écosystème du système d'aide à la décision est composé de deux logiciels pour deux types d'acteurs apportant des informations complémentaires. Les fournisseurs renseignent leur expertise au système et les clients leurs objectifs. Les fournisseurs connectent leur stock de composants au système d'aide à la décision. Ce stock représente l'ensemble des composants usagés disponibles. Chaque module interagit avec le SI via une API lui permettant d'insérer ou modifier des données. En plus de pouvoir envoyer des données au SI, le logiciel Client peut recevoir des données du système d'aide à la décision grâce au processus d'observation qui permet au système de lui proposer des conceptions de produits.

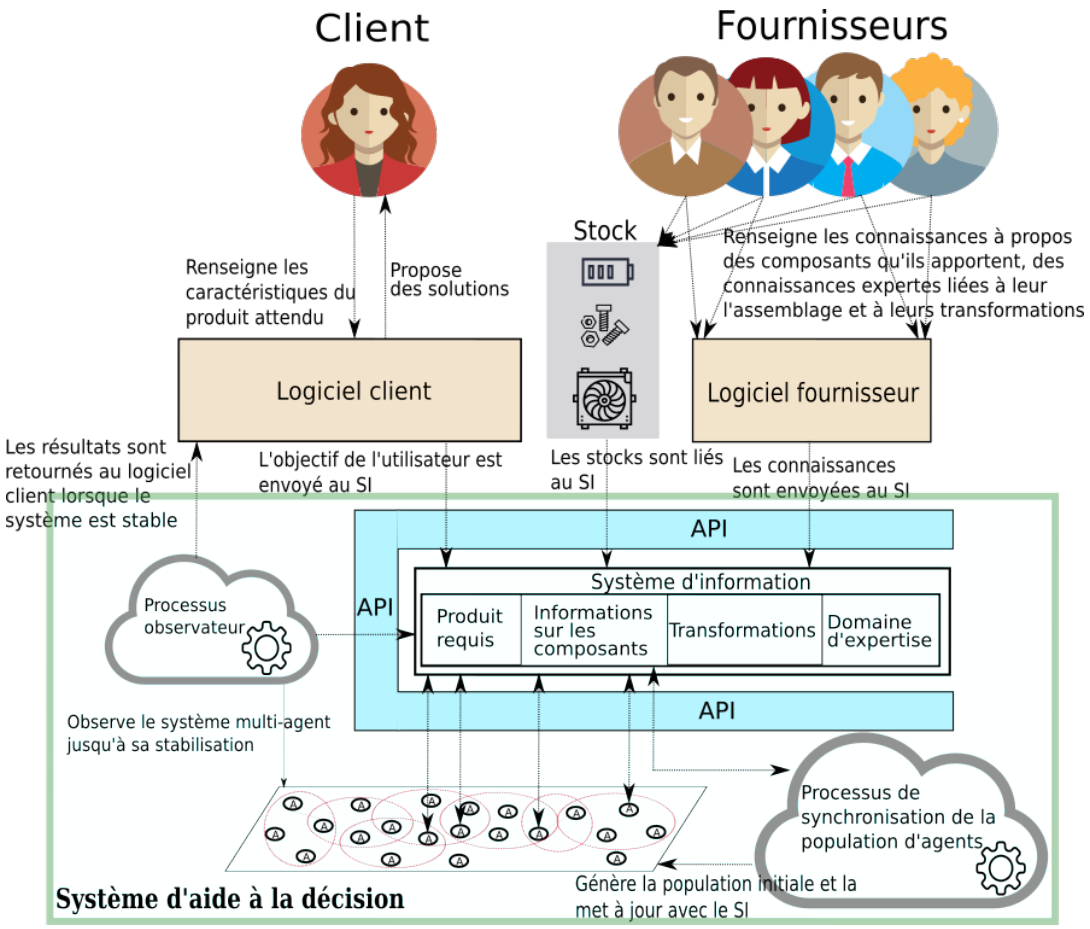


Figure 6.2 – Architecture du système d'aide à la décision

Une fois que les agents se sont organisés pour former une solution, le processus observateur détecte leur stabilisation et demande à chacun les coalitions dont il est membre. En agrégeant les retours des agents, il est à même de reconstituer une vue d'ensemble des coalitions formées par

le système, de les évaluer en fonction des désirs du client et de leur niveau de complétude et de les lui proposer. Le client peut alors lire les solutions et les trier sur différents critères tels que la complétude des groupes, leur qualité, ceux qui nécessitent le moins de transformations, *etc*. En cas d'insatisfaction l'utilisateur peut relancer le système en modifiant sa requête initiale. Au contraire, s'il trouve une conception qui lui convient, il peut s'approprier les composants en les achetant ce qui les fera disparaître des composants disponibles pour les autres utilisateurs.

6.4 Discussion et synthèse

Le cas d'étude décrit dans ce chapitre prend en compte une grande partie des composants clefs d'un lave-linge. Bien que les caractéristiques nécessaires à la description de ces composants soient en réalité plus nombreuses que celles utilisées, leur nombre suffit à rendre le cas d'étude suffisamment réaliste pour évaluer le fonctionnement du cœur du système d'aide à la décision : son système multi-agent. Un tel système d'aide à la décision a l'avantage de permettre à de multiples fournisseurs de mutualiser leur expertise et leur inventaire. La mise en commun de leurs connaissances en un seul système peut faciliter la conception des produits. L'outil semble donc intéressant pour de grandes entreprises possédant plusieurs entrepôts chacun spécialisé dans la conception d'un type d'appareil. Les composants sont aujourd'hui conçus pour ne s'adapter qu'à un seul type de machine. Comme nous l'avons vu dans les parties précédentes, les composants d'un lave-linge sont par exemple conçus de manière à s'adapter au besoin très spécifique d'un modèle de machine. En plus d'une législation peu favorable au remanufacturing, ce manque d'interopérabilité dans la conception des appareils complexifie son usage. Mais le développement de la conception des produits pour leur réutilisation pourrait permettre à des entreprises d'utiliser des composants dans des appareils pour lesquels ils n'étaient pas initialement fabriqués. En imaginant que cette pratique se développe, ce système d'aide à la décision pourrait intégrer le repurposing dans son fonctionnement et prendre en compte plus de composants dans la conception de produits afin de diminuer notre production de déchets. Le chapitre suivant évalue le système multi-agent et détaille les conditions expérimentales ayant conduit à ces résultats.

7

Expérimentation et évaluations

Sommaire

Les chapitres précédents ont présenté notre architecture d'agent et ont permis d'établir un cas d'étude tiré d'un contexte industriel. Dans ce chapitre, nous présentons plus en détail l'opérationnalisation de l'architecture ABSG permettant son fonctionnement dans un système multi-agent. Nous évaluons ensuite notre approche afin de mieux estimer ses avantages et inconvénients par rapport à la littérature. Enfin, nous évaluons ABSG dans notre cas d'étude pour déterminer son utilisabilité dans un problème de remanufacturing réaliste.

7.1 Opérationnalisation de l'architecture d'agent

7.1.1 Représentation des connaissances

L'architecture a été présentée de manière théorique dans le chapitre 5 mais nécessite quelques précisions pour son utilisation pratique. Pour compléter notre présentation, il est nécessaire de décrire comment les connaissances sont représentées dans la mémoire des agents. Comme décrit dans le chapitre 5, les agents se représentent leurs caractéristiques et désirs à l'aide de deux tableaux de décimaux, le tableau contenant les caractéristiques et celui contenant les désirs. Le décimal d'un agent peut par exemple correspondre à sa couleur. Dans le cas où l'ensemble des couleurs que l'agent peut avoir est *{bleu*, *rouge*, *vert}* alors la caractéristique couleur pourrait prendre les valeurs 0, 0.5 ou 1 où chaque décimal correspond à une unique couleur. Pour les caractéristiques quantitatives, comme le poids d'un contrepoids de lave-linge, la valeur peut être normalisée grâce à l'équation :

$$n = \frac{x}{maxin - minin} \tag{7.1}$$

Avec *maxin* et *minin* les intervalles de valeurs possibles que peut prendre la valeur à normaliser x. En outre, cette normalisation a l'avantage de limiter le biais du calcul de l'attraction en comparant des caractéristiques qui évoluent sur des intervalles et des échelles de valeur différentes. Par exemple dans un lave-linge un tambour doit choisir une cuve en fonction de son volume et de la puissance de sa résistance. Le volume d'une cuve évolue sur un intervalle de valeur d'environ $[80, 120]$ litres et la puissance de sa résistance sur $[1300, 2500]$ Watts. Nous voyons que quelle que soit la cuve choisie, il y aura au maximum une différence de 40 entre le désir d'un tambour et le volume de la cuve alors que cette différence s'élève à 1200 dans le cas de la puissance de la résistance. Sans normalisation, la différence d'échelle peut laisser l'agent *tambour* penser qu'il sera très souvent plus proche de trouver la cuve au bon volume que la cuve à la bonne puissance de résistance, même si la différence avec le volume souhaité est de 40L alors que la différence avec la puissance souhaitée n'est que de 100W. D'une part, l'impact de la puissance sur l'attraction pourrait devenir trop important par rapport à d'autres caractéristiques d'un même agent, d'autre part elle influerait aussi sur les relations avec les autres agents. En effet, le tambour se sentirait souvent plus proche des agents *système de roulement* qui évoluent sur un intervalle de prix de 50 à 80 euros, que des cuves qui possèdent des résistance de quelques centaines de watts. Par conséquent, dans ces expériences, nous instancions les désirs et les caractéristiques de nos agents sous

la forme de tableaux de décimaux normalisés entre 0 et 1 permettant de limiter cet effet.

Pour rappel, chaque agent est caractérisé par deux types de données qu'il représente en mémoire comme des tableaux :

— ses caractéristiques : ce sont les caractéristiques de l'agent (*e.g.* poids, volume, prix, *etc.*) ;

— ses désirs : ce sont les caractéristiques que l'agent aimerait voir chez les membres des groupes auxquels il appartient (*e.g.* volume recherché, prix recherché, *etc.*).

La section suivante illustre l'implémentation de l'architecture à l'aide d'un diagramme de classes.

7.1.2 Implémentation de l'architecture

La figure 7.1 présente le diagramme de classe de l'architecture ABSG. Elle représente une vue partielle de l'architecture en y excluant les classes de l'architecture propres à Soar qui n'ont pas été modifiées. Pour faciliter la lisibilité, les classes du diagramme n'ont pas été remplies de leurs attributs et méthodes et leur fonction est décrite dans le corps du texte ci-dessous.

Les classes sont réparties en quatre catégories, celles qui permettent aux agents d'adopter un comportement social, celles servant à l'envoi et la réception des messages, celles qui permettent l'interprétation des messages reçus et celles qui sont liées au cycle de décision et au comportement des agents ABSG. Nous les présentons dans les sections suivantes.

La communication

Les agents ABSG peuvent communiquer à travers internet via le protocole TCP/IP. Pour cela, chaque agent est à la fois serveur et client. Le rôle de serveur leur permet d'écouter les messages de leurs accointances tandis que le rôle de client leur permet d'en envoyer lorsqu'ils souhaitent contacter d'autres agents. Les classes *TCP_server* et *TCP_client* servent respectivement à lancer le serveur d'écoute des messages et à créer une connexion vers un autre agent. La classe *Client* quant à elle permet aux agents d'associer un numéro d'agent à une adresse IP et un port afin de contacter l'agent. La classe *GestionnaireClients* est une liste de clients et sert d'interface entre eux et les classes s'occupant du comportement de l'agent. Lors de leur exécution, les agents ABSG ne connaissent pas leurs accointances mais savent à quelle adresse les contacter (*127.0.0.1* puisque les expériences sont lancées en local) et sur quel intervalle de ports elles sont présentes. À leur lancement, les agents diffusent un message de présentation pour déclarer leur présence et se présenter aux autres agents du système.

La seconde partie du module de communication sert à l'interprétation des messages reçus par l'agent. La classe *Message* définit la structure d'un message selon les modifications apportées à la norme *FIPA-ACL* que nous avons décrite dans le chapitre 5. La classe *QueueMessages* représente une boîte de réception où les messages sont stockés avant d'être traités. Lorsqu'un message est traité, cette même classe l'analyse et met à jour la mémoire de l'agent en fonction de son contenu. *QueueMessages* contribue par ce mécanisme de mise à jour de la mémoire au choix de la prochaine action de l'agent. Les classes *Requête* et *GestionnaireRequêtes* tiennent à jour le déroulement des conversations entre les agents. Par exemple, lorsqu'un agent veut quitter l'une de ses coalitions, ces classes lui permettent de mémoriser à qui il a déjà notifié son départ et qui lui a répondu. Dans

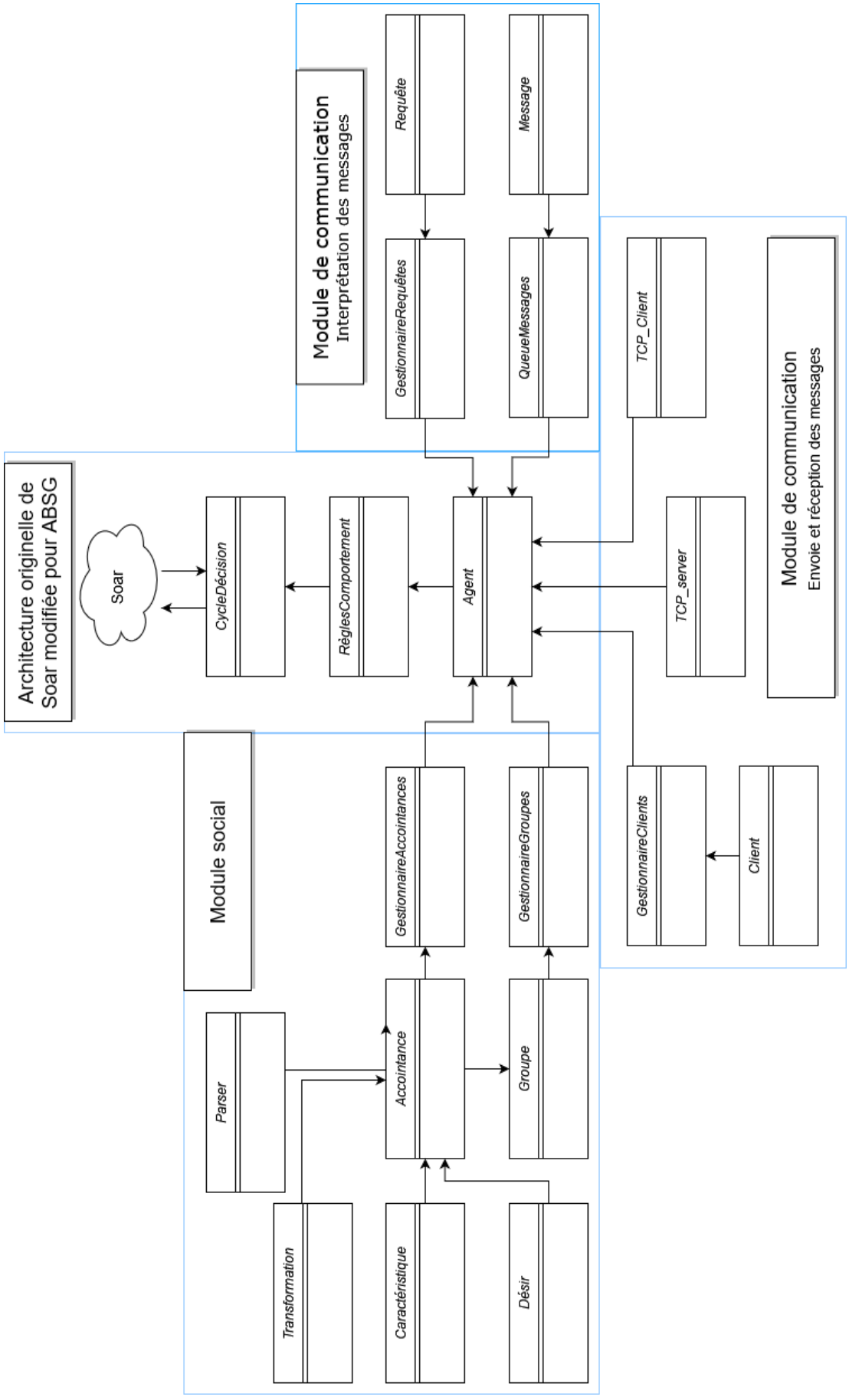


Figure 7.1 – Diagramme de classe de l'architecture ABSG

ce cas, c'est uniquement lorsque tous les agents ont envoyé un accusé de réception que l'agent quitte réellement la coalition.

Le social

Les classes *Transformation*, *Caractéristique* et *Désir* stockent et manipulent les données des agents. Chaque instance de ces classes est propre à chaque instance de la classe *Accointance*. Toutes les accointances sont stockées dans une liste appartenant à la classe *GestionnaireAccointances* servant d'interface entre la logique du comportement de l'agent et ses accointances. Ces mêmes accointances sont utilisées dans les groupes qui sont manipulés dans la classe *GestionnaireGroupes*. Les classes *Groupe* et *Accointance* représentent respectivement les groupes auxquels appartiennent les agents et leurs accointances. Ces classes sont celles où se déroule le traitement des valeurs d'attraction. Les désirs, caractéristiques et transformations des agents sont déterminés avant leur instanciation grâce à un script Python qui les génère aléatoirement et les écrit dans un fichier texte. La classe *Parser* permet à l'agent de lire et récupérer ces informations pour les stocker dans sa mémoire. En plus de posséder des caractéristiques, une instance de la classe *Accointance* possède un numéro d'agent qui peut lui permettre de s'associer à une instance de la classe *Client*. Ainsi, lorsqu'un agent souhaite envoyer un message à une instance d'*Accointance*, il recherche son numéro d'agent dans les listes de clients afin de retrouver son adresse IP et son port.

Le comportement

Le comportement de l'agent se déroule dans des classes originellement présentes dans la structure de l'architecture Soar. L'architecture ABSG les reprend et les modifie de façon à adapter le comportement aux besoins des agents. L'architecture Soar est vaste et ce diagramme ne la reprend pas entièrement. Nous la représentons dans le diagramme de classe comme un nuage avec lequel interagit la classe *CycleDécision* que nous avons modifiée pour prendre en compte la phase de socialisation ajoutée au cycle des agents ABSG. Les classes les plus modifiées sont les classes *Agent* et *RèglesComportement* qui, respectivement, servent à décrire l'agent ABSG, à définir les modules qu'il peut utiliser et à définir la logique du comportement de l'agent telle qu'elle a été présentée dans le chapitre 5.

Cette architecture a été développée à l'aide des langages *C* et *C++* à partir du code source de l'architecture Soar téléchargé sur son dépôt *Github*. Dans la partie suivante, nous présenterons les métriques utilisées pour les évaluations ainsi que les conditions expérimentales dans lesquelles notre système a été évalué.

7.2 Expérimentation

Nous définissons dans les sections suivantes un certain nombre de métriques servant à évaluer l'approche multi-agent ainsi que l'architecture ABSG.

7.2.1 Métriques d'évaluation

Afin d'évaluer notre approche, il nous est nécessaire de définir des critères d'évaluation nous permettant de nous comparer à la littérature et de valider ou non les propriétés de notre architecture. En plus d'évaluer la capacité de notre approche à satisfaire une demande de conception de produit en un temps raisonnable, nous souhaitons voir si elle est robuste face à un environnement ouvert et variable. Enfin, nous souhaitons confirmer la capacité de notre approche a être explicable pour un utilisateur non expert développeur. Pour cela, nous avons besoin de trois métriques :

— la mesure de la qualité des coalitions formées : elle permet de comparer les coalitions entre elles et de déterminer les paramètres du système pouvant les influencer ;

— la mesure du temps d'exécution : elle permet de comparer le temps d'exécution avec d'autres travaux, mais aussi, tout comme la mesure de la qualité, de déterminer les facteurs qui l'impactent ;

— la mesure du nombre de cycles exécutés par chaque agent : elle est corrélée avec le temps d'exécution. Cette métrique ajoute une autre dimension à la mesure du temps car elle est indépendante de la machine sur laquelle les expériences ont lieu ;

— la mesure de la robustesse face au nombre de communications : cette métrique a pour objectif d'évaluer le nombre de messages émis par le système et de comprendre les facteurs capables de l'influencer.

La qualité

La qualité d'une solution peut être définie comme l'adéquation de chaque agent d'une coalition avec ses accointances. Ainsi, au mieux deux composants s'assemblent ensemble, au plus leur présence dans la même coalition a de sens. De manière plus large, plus les agents d'une même coalition voient leurs désirs satisfaits par les caractéristiques de leurs accointances, plus la conception du produit est bonne. Pour nuancer ce propos, notons qu'une conception sera jugée de bonne qualité grâce aux données renseignées par un fournisseur ou un quelconque expert technique dans la fabrication du produit en question. Les agents n'ont aucune connaissance experte et ne jugent leurs coalitions qu'à travers les informations qu'ont leur a renseigné. Une conception de produit peut donc être de très bonne qualité, voire la meilleure que le système puisse trouver, tout en étant totalement dénuée de sens si les connaissances expertes apportées aux agents ne sont pas correctes. Par conséquent, notre métrique de la qualité ne prend pas en compte la qualité objective d'une coalition mais plutôt la manière dont les composants d'un groupe voient leurs désirs et contraintes satisfaits.

Dans notre évaluation, la qualité d'une coalition correspond à la moyenne de la satisfaction moyenne des désirs de chaque agent du groupe. La satisfaction des désirs d'un agent pour les caractéristiques de l'une de ses accointances correspond à la normalisation de la distance entre ses désirs et les caractéristiques de l'accointance :

$$d = \sqrt{s^T s} \qquad s = a_d - a_c \tag{7.2}$$

$$satisfaction(a_d, a_c) = \frac{d}{\sqrt{|a_c|}} \tag{7.3}$$

avec a_d les désirs de l'agent et a_c les caractéristiques de son accointance. La normalisation a pour objectif de ne pas faire augmenter la distance artificiellement en augmentant le nombre de caractéristiques des agents. En effet, la distance entre un désir et une caractéristique est comprise entre 0 et 1, alors qu'une distance entre dix désirs et dix caractéristiques est comprise entre 0 et $\sqrt{10}$. La normalisation évite alors de biaiser la mesure de la qualité pour les agents étant décrits par un grand nombre de caractéristiques. $S(a)$, la satisfaction moyenne d'un agent a pour sa coalition C peut être définie telle que :

$$S_c(a) = \frac{\sum_{m=0}^{|Acc_a|}(1 - satisfaction(a_m))}{|Acc_a|} \tag{7.4}$$

avec Acc_a les accointances de a présentes dans la coalition c, $satisfaction(a_m)$ la satisfaction des désirs de l'agent a pour son accointance m et $|Acc_a|$ le nombre d'accointances de a dans la coalition c.

La satisfaction moyenne d'une coalition C est ainsi définie comme :

$$V(C) = \sum_{n=0}^{|C|} S_c(n) \tag{7.5}$$

avec $S_c(n)$ la satisfaction de l'agent n pour la coalition c et $|C|$ la cardinalité de la coalition C. L'objectif de la somme est à la fois de pouvoir tenir compte de la satisfaction de chaque agent pour ses accointances dans la coalition et à la fois du nombre d'agents dans le groupe, ce qui permet de faire la différence entre une coalition complète et incomplète.

Enfin, une coalition ne respectant pas les contraintes du domaine d'expertise, imposant par exemple l'assemblage de deux pièces électroniques uniquement si les connecteurs sont du même type, ne représente pas une solution et possède par défaut une qualité négative.

Le temps d'exécution

Un grand nombre de travaux de la littérature présentent le temps d'exécution de leurs méthodes en secondes ou millisecondes. Nous choisissons cette même mesure pour nous comparer à ces approches. Cependant, le temps étant fortement impacté par la qualité du matériel, du système d'exploitation et des logiciels exécutés par l'ordinateur, nous introduisons également le concept de cycle de décision. Chaque cycle de décision entraîne la sélection puis l'exécution d'une action de la part de l'agent. Les cycles représentent donc le nombre d'actions qu'a effectué un agent. Ces derniers sont exécutés en parallèle, ils ne subissent aucun ordonnancement autre que celui du système d'exploitation. Leurs cycles de décision ne sont pas synchronisés et des agents peuvent exécuter plus de cycles que d'autres au cours de l'expérience. C'est pour cette raison que nous mesurons le nombre de cycles moyen par agent lors d'une exécution. Cette métrique nous permet donc de mesurer l'évolution du temps d'exécution des agents tout en nous affranchissant du support matériel et logiciel de notre machine.

La robustesse face au nombre de communications

Cette métrique a pour objectif d'évaluer le nombre de communications dans le système et de voir quels facteurs ou évènements peuvent l'influencer. Dans les expériences suivantes, nous utiliserons le nombre total de messages émis par unité de temps pour l'ensemble du système comme mesure des interactions inter-agents.

Synthèse

Nos évaluations se basent sur trois métriques permettant d'évaluer et comparer les performances de notre approche. La valeur $V(C)$ des coalitions mesure leur qualité et permet d'observer leur évolution dans le système à mesure de son exécution. Le temps et le nombre de cycles permettent d'évaluer la durée d'exécution du système. Alors que la première métrique permet de se faire une idée du temps nécessaire au système pour proposer des solutions à l'utilisateur, la seconde s'affranchit de contraintes matérielles liées à l'environnement d'exécution des agents en ne prenant pas en compte leur charge computationnelle.

La section suivante présente les conditions d'expérimentations utilisées lors de l'exécution de notre système.

7.2.2 Conditions d'expérimentations

Observateur

L'évaluation de l'ouverture et de la variabilité nécessite une mesure de la qualité des coalitions formées au fil des expériences afin d'observer les effets de l'ajout, de la suppression ou de la modification des caractéristiques des agents sur le système. Pour cela, un processus observateur est lancé en même temps que les agents du système. Celui-ci a pour objectif l'observation du système multi-agent et la sélection des meilleures coalitions satisfaisant une demande de conception de produit. Tous les agents savent intrinsèquement comment communiquer avec l'observateur. Ils le notifient de leur arrivée dans le système au moment de leur exécution et lui envoie une copie de chaque message transmis à leurs accointances. Notons que ces copies ne sont pas comptées dans la mesure de la robustesse au nombre de communications dans les expériences de la section suivante. Les copies des messages transmises à l'observateur ont comme seul intérêt la génération de données pour la génération des graphiques de la partie évaluation de ce chapitre. En plus de ces copies, l'observateur demande régulièrement aux agents les coalitions dans lesquelles ils se trouvent. Bien que ce mécanisme alourdisse la charge en communication du système pour nos évaluations, il nous permet de visualiser – à travers les données récoltées par l'observateur – la qualité des coalitions formées par le système au cours des expériences. Ces visualisations sont utilisées dans les évaluations de l'ouverture et de la variabilité du système. En outre, ce mécanisme de demande d'information est utile du point de vue du système d'aide à la décision puisqu'il permet à celui-ci de proposer des solutions à son utilisateur à n'importe quel moment.

Paramétrage des agents

Dans les expériences suivantes les caractéristiques, les désirs, les contraintes et les transformations des agents sont générés aléatoirement. Les seuils de formation, d'affiliation et de sortie d'une coalition sont calculés en fonction du nombre d'agents dans le système tel que :

$$f_s(x) = \frac{(maxin - x - minout) * (maxout - minout)}{maxin - minin} + minout \tag{7.6}$$

avec $maxin = 80$, $minin = 0$, $maxout = 0.45$ et $minout = 0.35$. Cette équation permet la normalisation des seuils en fonction du nombre d'accointances d'un agent. Plus un agent a d'accointances, meilleure l'attraction doit être pour que celui-ci accepte une requête de formation de groupe ou d'affiliation. Cette méthode permet de rendre les agents moins exigeants lorsqu'ils ont peu de choix de formation de groupe ou d'affiliation, par exemple dans les systèmes à petite échelle. Au contraire, dans des systèmes à plus grande échelle, les agents ont de multiples possibilités de former ou rejoindre des groupes, diminuer la valeur des seuils permet alors de se concentrer sur les meilleures d'entre elles. En plus de former des coalitions de meilleure qualité, ce mécanisme réduit le temps d'exécution des systèmes les plus larges. Dans ce cas précis, nous affectons *maxin* comme étant le nombre d'accointances maximum que peut avoir un agent puisque, comme nous le verrons dans la partie suivante, la majorité des évaluations sont exécutées avec un maximum de 80 agents. *minin* est initialisé à 0 dans le cas où un agent n'a aucune accointance et *minout* et *maxout* sont respectivement associés aux valeurs 0.35 et 0.45. L'attraction étant normalisée entre 0 et 1 (sans prendre en compte le principe du minimax qui peut la faire dépasser 1), nous estimons qu'une accointance intéressante est une accointance pour laquelle l'attraction est d'un tiers de la pire valeur possible. Ce seuil a été fixé empiriquement de façon à faire un compromis d'une part entre une trop faible sélectivité des accointances menant a la formation de multiples coalitions de faible qualité et d'autre part une trop forte sélectivité empêchant toute coalition d'émerger. *maxout* représente le seuil maximum de sélectivité et permet aux agents ayant peu d'accointances de réduire leur sélectivité afin de tout de même pouvoir former ou s'intégrer à des coalitions.

Les seuils de sélectivités étant compris entre 0.35 et 0.45, nous fixons ε à 0.2 afin de rapidement dépasser *minout* et d'empêcher les agents de trop insister à intégrer une accointance dans leurs coalitions si le système comprend un grand nombre d'agents. L'objectif étant de leur éviter de s'entêter à s'affilier avec un agent réticent si beaucoup d'autres sont disponibles. Au contraire dans le cas où un agent a peu d'accointances et le seuil de sélectivité est plus élevé, $\varepsilon = 0.2$ lui permettrait d'insister auprès des accointances pour lesquelles il a les meilleures valeurs d'attraction pour les intégrer à plusieurs coalitions. Nous ne clamons pas avoir proposé la valeur optimale de ε en l'initialisant de cette manière, cependant nous suggérons que celle-ci est suffisamment convenable pour permettre le bon fonctionnement de nos agents et permettre au système de proposer des solutions de bonne qualité. Les parties suivantes investigueront plus en détail de l'effet de ε et l'influence que ce paramètre a sur le comportement des agents et sur leur capacité à former des coalitions.

Les parties suivantes évaluerons l'approche multi-agent et les capacités de notre système à faire émerger des coalitions dans un cadre théorique où les agents ne représentent pas de composant spécifique et où leurs caractéristiques, leurs désirs et leurs transformations sont générées aléa-

toirement. Elles évalueront également notre architecture sur le cas d'étude présenté dans le chapitre précédent dans lequel les caractéristiques et les désirs des agents sont pseudo-aléatoire car représentant des grandeurs physiques réalistes de vrais produits. Dans ce cas d'étude les contraintes utilisées par nos agents sont connues à l'avance et dictées par les contraintes physiques réelles des produits à concevoir. Au contraire, dans l'évaluation de notre approche dans un cadre théorique, les contraintes utilisées sont celles qui ont été présentées dans le chapitre 5 : ANY, ALL, LSS, GRT, LEQ, GEQ, EQ, ALL. Exception faite des systèmes à petites échelles dans lesquels des contraintes trop fortes empêcheraient toute émergence de coalitions, toutes les contraintes sont utilisées durant ces expériences. Dans l'évaluation de l'architecture dans le cadre théorique, les transformations sont générées de manière aléatoire dans un intervalle de valeur $[-1,1]$. Cependant, celles-ci sont générées de manière à ne pas faire sortir les caractéristiques des agents de leur intervalle $[0,1]$. Enfin, dans une application réaliste, seule une minorité de composants sont transformables. Les parties suivantes tenteront d'évaluer l'impact des transformations sur le comportement du système. Enfin, les composants de l'évaluation théorique possèdent par défaut 5 caractéristiques et désirs. Le comportement des agents étant indépendant du nombre de caractéristiques et de désirs qu'ils manipulent, nous avons choisi de leur en affecter 5 pour l'évaluation théorique. Les effets de ce paramètre sur le temps d'exécution du système seront évalués dans les sections suivantes. Quant à l'évaluation sur le cas d'étude, le nombre de caractéristiques et de désirs est celui indiqué pour chaque type de composants dans le chapitre précédent.

Environnement matériel et logiciel

Les agents sont exécutés en parallèle sur la même machine de manière asynchrone et communiquent avec le protocole TCP/IP. Les expériences sont effectuées sur un PC multi-cœur Intel i7-6600U CPU (2.60GHz), 16 Go, 64-bit, Manjaro Linux 20.1.

Synthèse

Un processus observateur nous permet de récupérer des données sur les coalitions formées par les agents au cours de l'exécution du système. Nos agents sont paramétrés de la manière suivante pour toutes les expériences, exceptées celles qui les redéfinisse explicitement afin d'évaluer leur impact sur le système :

- seuil minimum de sélectivité : 0.35
- seuil maximum de sélectivité : 0.45
- ε : 0.2
- contraintes : ANY, ALL, LSS, GRT, LEQ, GEQ, EQ, ALL
- transformations : chaque caractéristique est additionnée à une valeur aléatoire (la transformation) générée dans un intervalle de $[-1,1]$. Cependant leur génération est telle qu'elle ne permet pas aux caractéristiques de sortir de leur intervalle $[0,1]$;
- nombre de caractéristiques et désirs par agent : 5

Dans la partie suivante, nous évaluerons notre approche multi-agent en exécutant le système dans les conditions expérimentales précédemment décrites. Bien que notre système puisse retourner des solutions à n'importe quel moment, tous les résultats sont obtenus après stabilisation du

système, c'est-à-dire au moment où les agents sont tous satisfaits de leurs coalitions et que plus aucun message n'est émis pendant plus de dix secondes. En outre, tous ont été obtenus en moyennant les données récoltées sur 5 expériences.

Le nombre d'agents variera selon les expériences. Une entreprises comme Envie peut stocker et réutiliser des centaines de composants pour un même type d'appareil électroménager. Nous limiterons cependant la plupart de nos évaluations à quelques dizaines d'agents pour mieux appréhender les résultats et obtenir des temps d'exécution raisonnables. En effet, obtenir la qualité des solutions optimales nécessite un algorithme brute force demandant un long temps d'exécution allant de quelques dizaines de secondes pour 50 agents, à plusieurs heures lorsque l'on dépasse la centaine d'agents. Limiter la taille du système nous permet donc de nous comparer à un optimal et mesurer l'efficacité de notre architecture pour la formation de groupes. Nous évaluerons tout de même de manière moins exhaustive le fonctionnement de notre système dans une application à plus grande échelle.

7.3 Évaluation des aspects multi-agents

Les évaluations de cette partie ont deux principaux objectifs ; premièrement, d'évaluer l'intérêt de notre approche pour notre problématique de formation de coalitions en la comparant aux approches de la littérature, secondement, d'analyser l'impact des paramètres de notre architecture dans la capacité du système multi-agent à résoudre son problème. Ces paramètres incluent notre fonction d'attraction définie dans le chapitre 5, les contraintes et les transformations renseignées aux agents par l'expert, et le nombre de caractéristiques et désirs manipulés par les agents ABSG. Enfin, nous finirons cette partie par l'évaluation de la robustesse de notre système face à la charge en communication et par l'évaluation des capacités d'adaptation de nos agents face à un système ouvert et très variable.

7.3.1 Comparaison à la littérature

Dans ces expériences, nous comparons la qualité des coalitions de notre approche à celles de l'art. Les courbes représentant les valeurs de structure de coalitions optimales sont obtenues par un algorithme brute-force testant toutes les solutions pour ne sélectionner que la meilleure. Les conditions expérimentales sont celles renseignées dans la partie précédente à l'exception de l'ensemble des contraintes qui a été réduit à la contrainte *ANY* pour éviter qu'un excès de contraintes rende impossible la formation de coalitions. Dans ces expériences les agents ne possèdent pas d'objectif commun. Nous générons leurs désirs de manière aléatoire et observons le temps dont le système a besoin pour faire émerger des coalitions.

La figure 7.2 présente les résultats des approches SACF [Ramos2013] et du framework de Janovsky et Deloach [Janovsky2016].

Dans leur travail, Janovsky et Deloach présentent un framework pour la formation de coalitions à grand échelle. Leur méthode s'inspire d'une approche de clustering hiérarchique afin de construire des structures de coalitions de manière dynamique selon un processus itératif. Contrai-

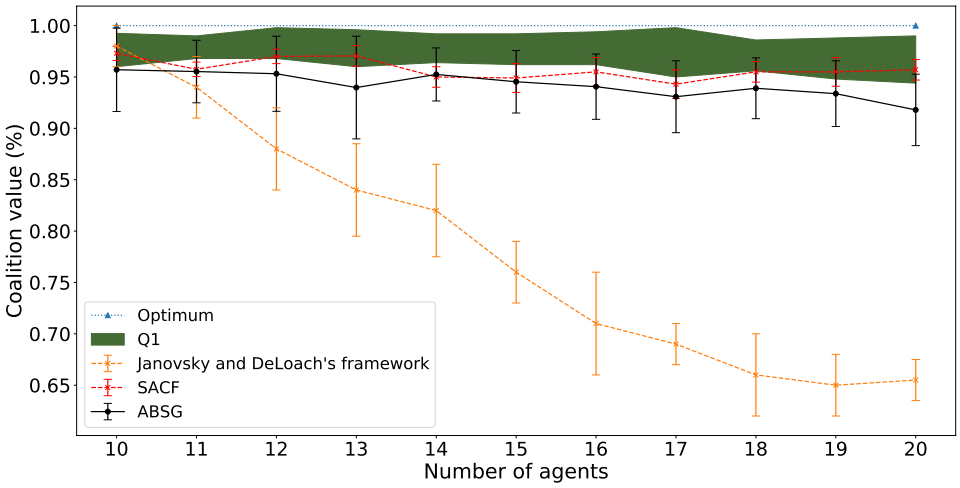


Figure 7.2 – Comparaison avec la littérature de la qualité des structures de coalitions formées en fonction de l'optimum pour des expériences de 10 à 20 agents.

rement à nous et à la méthode SACF, leur approche n'est pas distribuée et fonctionne exclusivement en monde fermé. Les coalitions sont formées itérativement de manière à maximiser leur gain. Comme le montre la figure 7.2, leur méthode ne fonctionne pas aussi bien que les autres sur des systèmes à petite échelle. Ce framework étant basé sur une approche de clustering, il n'est pas adapté à fonctionner dans un monde ouvert et variable. Cependant, il se trouve être plus adapté à des systèmes à très large échelle (jusqu'à 10 000 agents) dans lesquels les agents sont des entités invariables.

SACF est une méthode distribuée fonctionnant en monde ouvert dans un environnement situé. Leur approche présente des similitudes à la nôtre puisque leurs agents ont la capacité de créer ou rejoindre des coalitions en fonction de la valeur qu'ils associent à leurs accointances. L'objectif de chaque agent de leur système est de maximiser la valeur de sa coalition en s'appariant avec les autres. L'une des différence entre les agents ABSG et les agents de la méthode SACF est leur capacité à se réorganiser en quittant leurs coalitions, ce que les agents SACF ne sont pas capables de faire. Bien que les agents SACF soient conçus pour être flexibles, les coalitions générées ne sont pas remises en question en cas de changement dans le système.

La courbe noire représente la qualité moyenne des coalitions formées par les agents ABSG. On peut la considérer comme la valeur de notre structure de coalitions de manière à mieux nous comparer à la courbe de SACF et du framework de Janovsky et Deloach. Afin de mieux comparer les résultats entre eux, tous sont comparés à la valeur optimale que les systèmes auraient pu trouver dans leur contexte d'exécution. Dans le cas de notre système, puisque nous recherchons les meilleures coalitions possibles pour de l'aide à la décision, l'optimal représente la qualité de la meilleure coalition concevable. Notre approche génère des ensembles de coalitions dont la qualité moyenne approche environ 95% de l'optimal sur des systèmes à petite échelle. La méthode SACF présente des résultats légèrement supérieurs avec une proximité d'environ 96% de son optimal. Cette différence peut s'expliquer par le fait que notre contexte applicatif nécessite que notre système propose des conceptions de produits. Or un produit est composé d'un certain nombre de

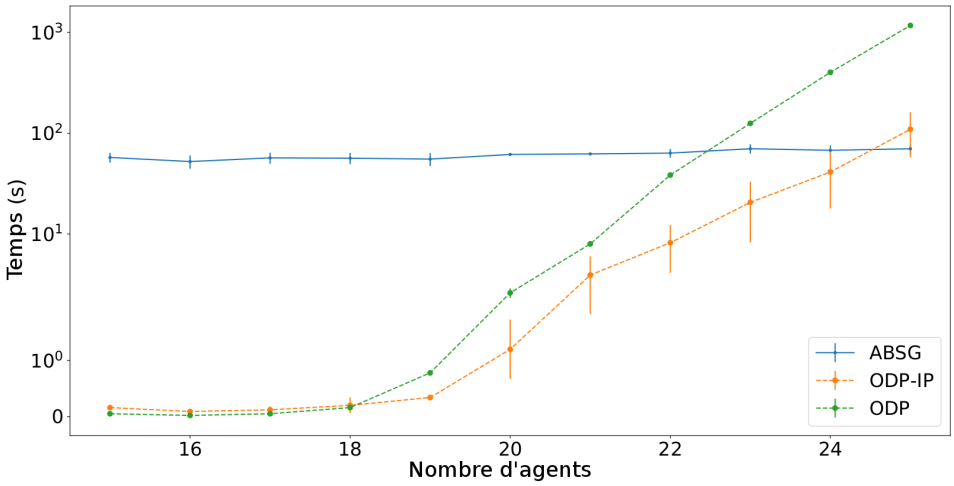


Figure 7.3 – Comparaison du temps d'exécution du système avec la littérature

composants fini et nous prenons en compte la quantité de composants dans la mesure de la qualité des coalitions formées (équation 7.5). Par conséquent, certaines coalitions incomplètes dans lesquelles les désirs individuels des agents sont satisfaits par les caractéristiques de leurs accointances voient leur qualité détériorée car elles ne possédent par tous les composants requis. Ce mécanisme n'existant pas dans la méthode SACF puisque les coalitions n'ont pas d'impératif de nombre d'agents, il est normal que la qualité de nos coalitions soit légèrement inférieure à celle de cette méthode. Autre facteur dégradant la qualité de nos structures ; les agents de la méthode SACF ne peuvent appartenir à plusieurs coalitions et se contentent donc de maximiser leur valeur de coalition. Au contraire notre approche est plus exploratoire, les agents peuvent faire partie de plusieurs coalitions et essaient de former des groupes avec tous les agents ayant une attraction meilleure qu'un seuil prédéfini. Le seuil a donc un impact très important sur la qualité moyenne des coalitions formées. Sur la figure, l'aire Q1 représente la qualité des 25% meilleures coalitions que les agents ont formées. Comme nous pouvons le constater, ces coalitions sont pour la plupart plus proches de l'optimal que les structures générées par SACF. Un seuil de sélectivité mieux ajusté permettrait donc d'augmenter la qualité moyenne de nos coalitions à un niveau plus proche de l'optimal.

Pour notre seconde expérience tous les agents sont lancés en même temps, le décompte du temps commence au moment où l'observateur reçoit le premier message d'un des agents du système. L'observateur estime que le système a convergé vers des solutions et arrête le décompte lorsqu'il ne reçoit plus de nouveaux messages pendant dix secondes. La figure 7.3 compare les temps nécessaires à notre approche pour générer des solutions face à des algorithmes exacts. Ces algorithmes sont très efficaces pour de très petites quantités d'agents. Cependant, leur temps d'exécution dépasse rapidement celui de notre approche à partir de 22 et 25 agents. Plus important, leur complexité en temps est exponentielle et leur consommation de mémoire étant elle aussi exponentielle, il leur est difficile de s'exécuter avec plus de 25 agents. L'échelle rend difficile la mesure de l'évolution du temps d'exécution de notre approche sur cette figure. Nous verrons dans la section suivante que sa complexité en temps est polynomiale (figure 7.5). Du point de vue du temps

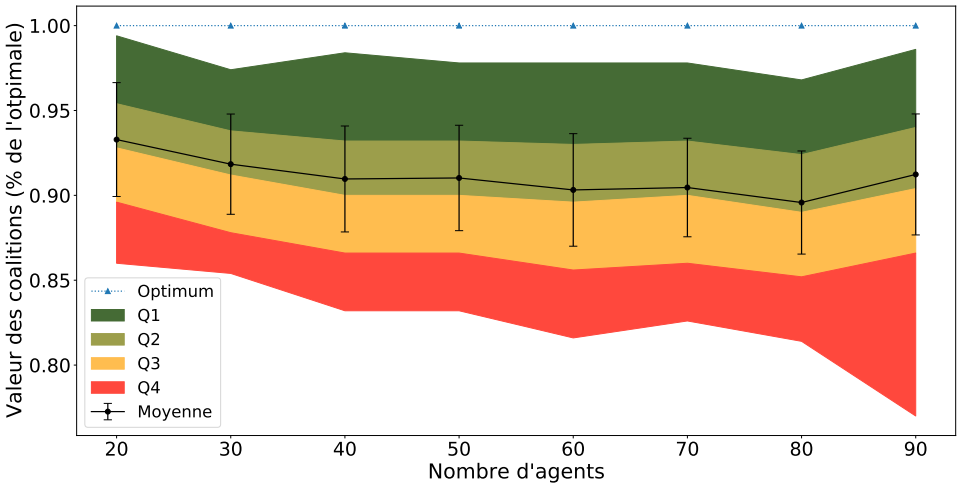


Figure 7.4 – Qualité des coalitions formées en fonction de l'optimum pour des expériences de 20 à 90 agents

d'exécution, notre approche semble donc plus pertinente que celles des méthodes exactes de programmation dynamique.

D'autres approches comme celles du clustering [Bistaffa2017], ou des jeux hédoniques [Morge2017] obtiennent des temps d'exécution plus faibles que les approches présentées dans la figure 7.3. Cependant ces algorithmes sont pour la majorité conçus pour résoudre une instance non variable d'un problème de formation de coalitions de manière non exacte (contrairement à l'approche de programmation dynamique). Contrairement à l'approche multi-agent, celles-ci sont soit centralisées soit assument que les agents connaissent tout de leurs accointances au lancement du système ce qui réduit voire enlève totalement les temps de communication inter-agent. Cependant, comme nous le verrons dans les sections suivantes, occulter l'individualité des agents en les traitant comme des données et leur retirer leur capacité de décision peut également réduire la capacité des systèmes à s'adapter à un environnement variable.

Dans les sections suivantes, nous évaluerons le système multi-agent sur une plus grande échelle et évaluerons les effets des paramètres utilisés par notre architecture sur ses performances. Nous apprécierons également les capacités d'adaptation de notre système face à un environnement ouvert et variable.

7.3.2 De l'effet du nombre d'agents

Dans une première expérience (figure 7.4), nous souhaitons évaluer la capacité de notre système à maintenir la qualité des coalitions à un niveau proche de l'optimal. Nous limitons le nombre d'agents à 90 pour éviter que le temps de recherche de l'optimal par l'algorithme brute-force devienne déraisonnablement long. Contrairement à l'évaluation précédente, toutes les contraintes sont utilisées lors de la génération des désirs des agents. Pour une meilleure visualisation, nous répartissons nos coalitions en quartiles en fonction de leur qualité. Chaque quartile représente une tranche de 25% des coalitions formées. Ainsi, Q1 représente les 25% meilleures coalitions, Q2 les

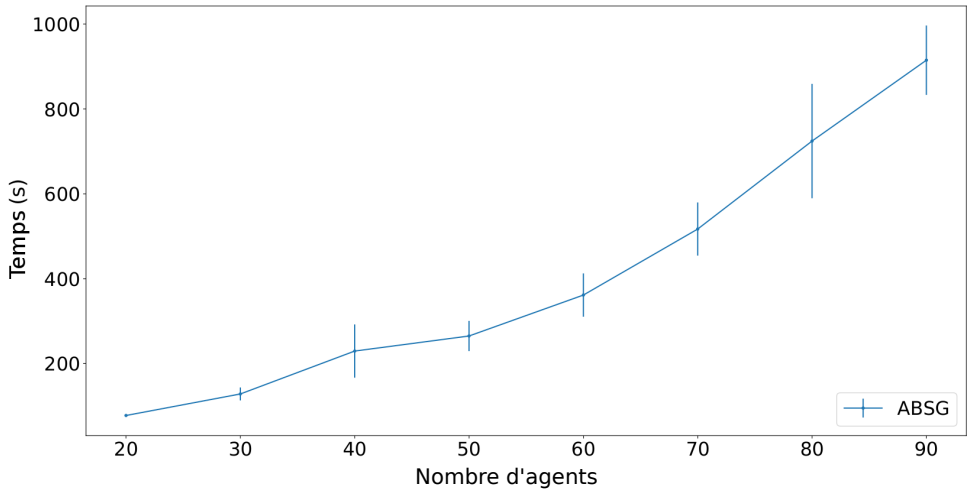


Figure 7.5 – Évolution du temps d'exécution du système en fonction du nombre d'agents

25 à 50% meilleures, Q3 les 50 à 75% meilleures et Q4 les 25% plus mauvaises. Finalement, la courbe noire représente la moyenne des valeurs de chaque coalition. Contrairement à l'expérience précédente où le nombre d'agent est très limité, nous pouvons voir que la qualité de la meilleure coalition formée ne dépasse jamais 99% de l'optimal (sauf exception sur l'expérience à 20 agents qui l'atteint). La qualité moyenne des coalitions est plus faible que dans les très petites instances du système (environ 93% de moyenne contre 95%). Il est cependant intéressant de noter que les coalitions du premier quartile sont de qualité presque équivalente aux expériences de petites échelles (>=94%), même sur des instances de 90 agents. Exception faite des expériences lancées avec 90 agents, les coalitions maintiennent de manière générale une bonne qualité face à l'augmentation du nombre d'agents. Malgré cela, ce paramètre semble tendre à légèrement faire diminuer leur qualité moyenne. De manière générale la répartition des coalitions dans les quartiles reste équivalente quelle que soit la quantité d'agents. L'agrandissement du quartile 4 sur les systèmes de 90 agents pourrait avoir lieu dans le cas où de nombreuses coalitions du Q4 perdraient subitement en qualité. Mais le fait que la qualité moyenne soit en légère augmentation malgré la chute du Q4 laisse plutôt penser qu'une seule des coalitions a été suffisamment mauvaise pour faire baisser la moyenne des qualité des autres coalitions de son quartile. Ce phénomène est donc probablement dû à une erreur statistique que le calcul des moyennes n'a pas suffi à estomper.

Dans la seconde évaluation (figure 7.5), nous observons l'évolution du temps d'exécution en fonction du nombre d'agents. L'axe y nous permet de d'estimer la complexité en temps comme étant polynomiale. Cette évolution semble intuitive puisque plus les agents ont d'accointances, plus ils ont de possibilité de former des coalitions, la formation de coalitions menant ensuite à la recherche de nouveaux membres pour concevoir un groupe complet. Le temps d'exécution étant très lié à la capacité computationelle d'une machine, la figure 7.6 apporte des informations complémentaires. Ce troisième graphique montre l'évolution du nombre de cycles moyen des agents ABSG en fonction de leur nombre. Un cycle représente une prise de décision. Lorsque les agents du système ont exécuté N cycles, ils ont alors exécuté N actions. Les actions correspondent à la lecture d'un message et à l'émission de l'un d'eux à la suite d'une prise de décision, par exemple

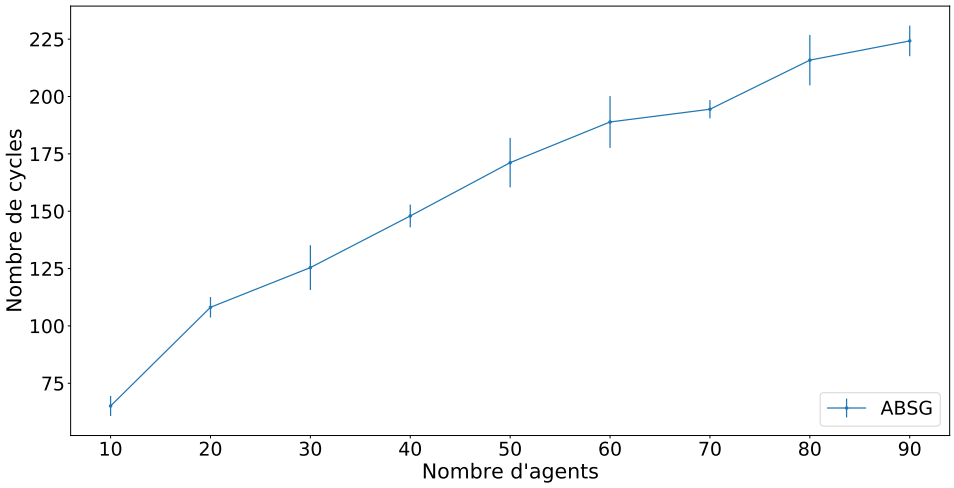


Figure 7.6 – Évolution du nombre de cycles moyens par agent en fonction de leur nombre

lorsqu'un agent veut rejoindre une coalition ou en former une nouvelle avec l'une de ses accointances. L'évolution du nombre de cycles moyen est plus faible que l'évolution du temps d'exécution moyen du système et semble être linéaire. Cela peut s'expliquer par le coût computationnel des agents inactifs satisfaits par leur présence (ou non) dans une coalition. Ces agents n'incrémentent plus leur nombre de cycles puisqu'ils ne prennent plus aucune décision, mais ils sont tout de même prêts à écouter les messages de leurs accointances pour se réorganiser au besoin. Cette charge computationnelle passive pèse sur la machine et peut ralentir les agents encore actifs du système. La figure 7.6 laisse donc penser que le temps d'exécution des agents est biaisée par la capacité computationnelle d'une machine. Cependant, le nombre de cycles moyen ne capture pas le temps nécessaire aux agents pour se synchroniser au sein d'un groupe. Par exemple, supposons l'existence d'une coalition $\{A,B,C\}$ dont A et le *leader*. Si les agents B et C souhaitent modifier la coalition en même temps pour y ajouter les agents D et E, B et C devront tout deux demander le jeton de la coalition à A. Ce jeton leur sera accordé à tour de rôle pour que chacun exécute ses actions sans gêner l'autre. Dans ce cas, C ne peut demander à E de les rejoindre que lorsque B aura fait sa demande à D. Or, le nombre de cycles serait le même si les deux demandes étaient faites en parallèle alors que le temps d'exécution serait divisé par deux. Aucune de ces deux métriques ne capturent avec précision le temps d'exécution indépendamment du support matériel mais elles permettent tout de même de définir la complexité en temps du système dans le meilleur et le pire cas. Le meilleur étant celui où le système multi-agent peut être distribué sur plusieurs machines ce qui lui permettrait d'être réellement exécuté en parallèle sans se soucier de l'ordonnancement des agents sur un unique processeur et le pire étant celui où tous les agents sont exécutés sur la même machine les obligeant à s'exécuter à tour de rôle. Dans le meilleur cas, la complexité en temps se rapprocherait un peu plus de celle du nombre de cycle moyen par agent (figure 7.6), dans le pire des cas, la complexité ressemblerait à celle de notre mesure du temps d'exécution (figure 7.5).

L'objectif de cette évaluation est d'analyser l'évolution du nombre de messages en fonction du nombre d'agents présents dans le système. La figure 7.7 montre les résultats de cette expérience avec des systèmes allant de 10 à 70 agents où l'axe y est représenté par une échelle logarithmique.

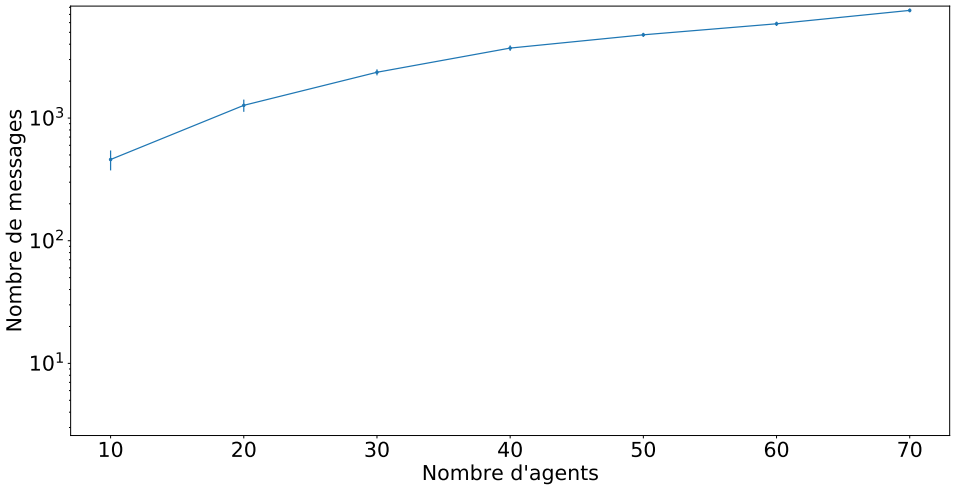


Figure 7.7 – Évolution du nombre de messages en fonction du nombre d'agents

Le nombre de messages semble évoluer de manière logarithmique sur cette échelle ce qui se rapproche d'une fonction affine sans la modification de l'échelle. Ce résultat suggère que chaque agent émet un nombre de messages relativement fixe au cours de l'exécution du système. Pour rappel, lorsqu'un agent entre dans le système, il diffuse un message d'introduction à toutes ses accointances afin de se présenter aux autres et de les notifier de sa présence. Ce mécanisme ne semble pas impacter particulièrement le nombre de messages émis par les agents du système sur l'intervalle d'agents testé, mais il possible que cette tendance ne soit pas maintenue avec un plus grand nombre d'agents. En outre comme pour le temps d'exécution, le nombre de messages émis peut être fortement impacté par les paramètres du système et des agents ABSG.

Après avoir évalué notre approche sur une plus grande échelle, nous souhaitons vérifier la pertinence de notre interprétation des principes de l'attraction dans le traitement de la valeur d'attraction. La section suivante tente de mieux apprécier l'impact de chaque principe sur la qualité et le temps d'exécution de notre système multi-agent.

7.3.3 De l'effet des principes de l'attraction

Définition des mesures. Le chapitre 5 justifie l'utilisation et l'interprétation de chaque principe de notre fonction d'attraction. Il n'est toutefois pas évident d'estimer leur impact sur le travail de formation des coalitions du système. Nous évaluons dans cette partie cet impact en exécutant les agents ABSG avec des versions modifiées de la fonction d'attraction et comparons les résultats. Pour cette évaluation, nous configurons la fonction d'attraction de cinq manières différentes de manière à isoler les principes que nous souhaitons évaluer :

1. Nous retirons la similarité de la fonction pour en déterminer l'impact sur la qualité moyenne et maximum des coalitions formées. Nous évaluons donc la fonction sur la base des principes d'attractivité physique, de complémentarité et sur le minimax ;
2. Pour les mêmes raisons nous retirons uniquement la complémentarité. Restent la similarité, l'attractivité physique et le minimax ;

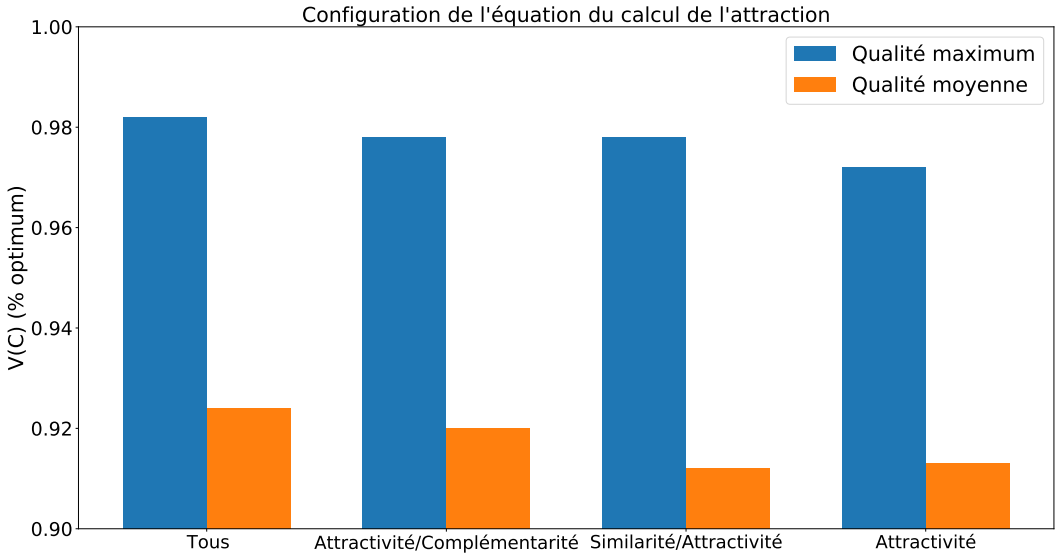


Figure 7.8 – Comparaison de la qualité des coalitions produites par différentes configurations de notre fonction d'attraction

3. Nous retirons l'attractivité physique tout en gardant la similarité, la complémentarité et le minimax ;
4. Seule l'attractivité physique et le minimax sont préservés.
5. Cette dernière configuration représente la fonction dans son ensemble sans modification de façon à comparer les résultats des autres configurations à celle-ci.

Conditions expérimentales. Toutes ces configurations gardent le principe du minimax qui est essentiel au bon fonctionnement des agents. Sans lui ces derniers enverraient indéfiniment des demandes de formation de groupe à leurs accointances préférées sans se soucier de leurs refus. Enfin, nous exécutons et moyennons les résultats sur 30 expériences en générant des systèmes composés de 30 agents ce qui nous permet d'obtenir des temps d'exécution raisonnables. Pour chaque expérience, nous avons recueilli la qualité moyenne des coalitions après stabilisation du système et en avons fait un jeu de données pour comparer nos différentes configurations. Pour cela, nous avons utilisé le test statistique de Shapiro-Wilk permettant de montrer la distribution normale de nos données parmi les jeux. Chacun des jeux de données a présenté une distribution normale avec une *p-valeur* supérieure à 0.05. Les variances étant inégales pour les jeux de données, nous avons vérifié l'égalité de leur moyenne à l'aide d'un test t de Welch.

	Tous	Attractivité/Complementarité	Attractivité/Similarité	Attractivité	Complémentarité/Similarité
Attraction moyenne	0.369	0.381	0.305	0.318	0.413

Tableau 7.1 – Moyenne de l'attraction inter-agent pour nos cinq configurations

Présentation des figures. Deux facteurs peuvent influencer la qualité moyenne des coalitions à travers la fonction d'attraction, d'une part la capacité des principes à modifier l'attraction moyenne inter-agent dans tout le système, d'autre part leur capacité à modifier l'attraction localement pour des agents spécifiques. Dans le premier cas, un principe peut modifier la qualité moyenne des coa-

litions en augmentant l'attraction globale que tous les agents ont pour leurs accointances. Puisque les seuils de formation et d'affiliation aux coalitions ne changent pas, ce mécanisme revient à demander aux agents d'être plus sélectifs au moment de choisir leurs accointances. Autrement dit, au lieu de diminuer les seuil d'attraction permettant de former des coalitions, tous les agents augmentent leur attraction pour leurs accointances ce qui a un effet similaire sur leur comportement. Au contraire, dans le second cas, la qualité des coalitions peut être augmentée si les agents choisissent avec plus de soin au cas par cas avec quelles accointances se regrouper. C'est ce qui a été appelé précédemment la capacité à modifier l'attraction localement. Afin de prendre en compte ces effets, la table 7.1 renseigne la moyenne de l'attraction inter-agent dans les cinq configurations testées. Notons qu'en usant de la fonction de seuil de formation de groupe définie dans le chapitre 5, nous obtenons une valeur de seuil à **0.393** dans ces expériences. Pour rappel, les agents ne souhaitent plus former ou rejoindre de coalitions si l'attraction pour leurs accointances dépasse ce seuil.

La figure 7.8 présente la qualité moyenne des coalitions obtenues avec quatre des cinq configurations. Seule manque la configuration *Complémentarité/Similarité* avec laquelle aucune coalition ne s'est formée.

Analyse des résultats. L'attractivité physique étant le principal facteur de notre fonction dans ce cas d'étude, il permet aux agents de juger les autres par rapport à la satisfaction de leurs désirs. En le retirant, les agents n'ont pu former aucune coalition ce qui explique l'absence de résultat sur le graphique. Ce résultat s'explique en regardant la table 7.1. Celle-ci montre que l'attraction moyenne inter-agent dans cette configuration est de 0.413, elles est donc supérieure au seuil permettant aux agents de former des coalitions.

Les qualités moyenne et maximum sont les plus faibles dans la quatrième configuration (*Attractivité*). Ce principe étant le plus important de la fonction, il emporte tout de même la qualité moyenne des coalitions à environ 91%. Contre nos attentes initiales, le fait d'y ajouter la similarité (configuration *Similarité/Attractivité*) ne permet pas d'augmenter la qualité moyenne et semble même la diminuer légèrement. Le test statistique t de Welch ne permet pas de dire que les deux jeux de données de ces configurations ont une moyenne différente. Par conséquent, il semble que dans ce cas d'étude le principe de similarité ne joue pas de rôle dans le processus de formation de coalition.

La configuration suivante est la combinaison de l'attractivité et de la complémentarité. Elle présente une qualité moyenne plus élevée que la combinaison *Similarité / Attractivité* et que l'*Attractivité* seule. La complémentarité a originellement pour rôle l'incitation des agents à se regrouper avec ceux qui ont des objectifs d'appariement différents. Or, dans cette expérience les agents possèdent tous des objectifs d'appariement similaires. En pratiquant un test t de Welch nous pouvons dire que les moyennes de ces deux groupes sont différentes (*p-valeur* de 0.045). Mais en observant la table 7.1, nous voyons que la moyenne de l'attraction dans ce cas est beaucoup plus élevée que dans les configurations précédentes. Cette hausse globale de l'attraction permet une meilleure sélectivité des accointances pour la formation de coalitions. Ce phénomène pourrait expliquer l'augmentation de la qualité des coalitions observée sur le graphique sans que le principe de complémentarité n'ait un effet réellement stratégique sur le choix des accointances des agents.

Enfin, le regroupement de tous ces principes dans la configuration *Tous* montre les qualités moyenne et maximum les plus élevées avec une attraction moyenne de 0.369. Le test t de Welch

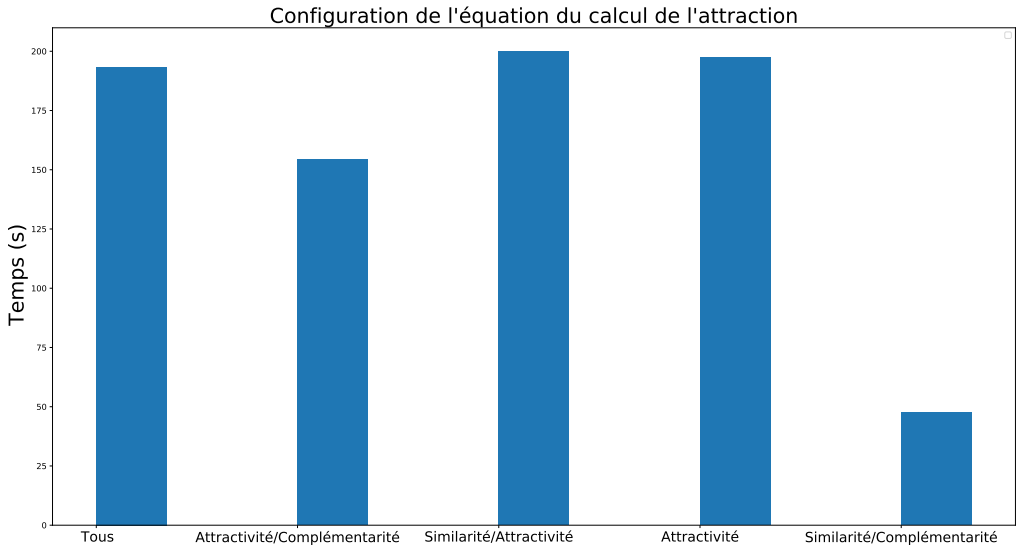


Figure 7.9 – Comparaison du temps d'exécution de systèmes en fonction de diverses configurations de notre fonction d'attraction

montre une *p-valeur* de *0.38* lorsque l'on compare cette moyenne avec la configuration *Complémentarité/Attractivité* et une *p-valeur* de *0.017* lorsqu'on la compare avec la configuration *Attractivité* ce qui montre une vraie différence dans la moyenne de ces jeux de données. Cette différence est également explicable par une moyenne de l'attraction plus importante dans ce cas que dans la configuration *Attractivité*.

La figure 7.9 consiste à mesurer l'impact de ces mêmes principes sur le temps d'exécution du système. Pour cela, nous réutilisons les mêmes configurations que dans l'évaluation précédente. Les expériences les plus courtes ont été réalisées avec la configuration *Similarité/Complémentarité* qui ne faisait émerger aucune coalition. Comme expliqué précédemment, le manque de coalitions est dû à une trop haute attraction inter-agent qui se situait en moyenne au dessus du seuil de formation de groupes (table 7.1). Le même effet est retrouvé de manière atténuée sur la configuration *Attractivité/Complémentarité*. L'attraction moyenne y étant élevée mais tout de même inférieure au seuil de formation de groupes, peu de coalitions se sont formées et le temps d'exécution moyen s'est vu diminué. Les autres configurations montrent des temps d'exécution très proches. Dans chacune de ces expériences, les agents du système ont eu suffisamment d'accointances jugées intéressantes pour prendre le temps de former des coalitions ce qui explique les temps un peu plus longs.

Synthèse. Du point de vue de la qualité des coalitions, ces résultats infirment l'intérêt du principe de similarité et de complémentarité dans notre fonction d'attraction dans ce cas d'étude. Toutefois, la *complémentarité* n'ayant pas été conçue pour fonctionner dans ce type de contexte, ce résultat n'est pas surprenant. Le principe de *similarité* en revanche avait été conçu de manière à améliorer la qualité des coalitions dans ce type d'application en permettant aux agents de se regrouper par similarité des désirs. Il semble que ses effets soient finalement inexistants et que regrouper des individus par similarité de leurs désirs ne soit pas une stratégie pertinentes pour former de bonnes coalitions. Le principe de *l'attractivité* montre une très grande efficacité pour la formation de coalitions et se révèle dans ces expériences être le facteur principal dans ce pro-

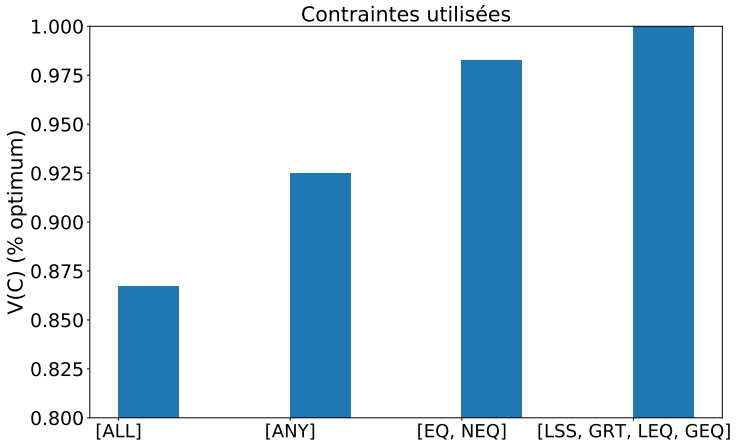


Figure 7.10 – Qualité moyenne des coalitions sur 10 expériences de 30 agents.

cessus. L'attraction moyenne inter-agent est également un élément prépondérant influant sur la qualité des résultats. Les seuils de formation et d'affiliation aux coalitions peuvent être adaptés en conséquence pour permettre une plus ou moins forte sélectivité des agents lors du processus de formation de groupes et ainsi jouer sur la qualité des coalitions.

Le principe de *complémentarité* ne s'est pas révélé efficace dans ces expériences. Pour nous permettre de mieux l'évaluer, nous ré-aborderons une évaluation similaire à celle-ci en utilisant le cas d'étude sur les appareils électroménagers présenté dans le chapitre 6.

7.3.4 De l'effet des contraintes

L'objectif de cette expérience est de déterminer l'influence des contraintes sur la qualité des coalitions formées. La figure 7.10 présente la qualité moyenne des coalitions formées sur 10 expériences en fonction des contraintes utilisées. Les contraintes ont été triées selon leur force, *ALL* n'en présentant aucune puisqu'elle autorise les agents à s'associer avec toutes leurs accointances. *ANY* ne pose pas réellement de contraintes mais permet aux agents d'évaluer leurs accointances. Elle ne les autorise à former ou s'affilier à des coalitions que si celles-ci ont une attraction suffisamment bonne. *EQ* force les agents à s'affilier uniquement si leur désir correspond exactement aux caractéristiques de leurs accointances et *NEQ* uniquement si elles leur sont différentes. *LSS*, *GRT*, *LEQ* et *GEQ* posent des conditions moins strictes que *EQ* en demandant aux caractéristiques des accointances d'être supérieures ou inférieures à leurs désirs. De manière générale plus les contraintes sont strictes et meilleure est la qualité moyenne des coalitions. Pour cause, *ALL* permet aux agents de créer n'importe quelles coalitions ce qui leur permet de produire des coalitions complètes facilement mais d'une qualité moyenne plus faible puisque les agents ne s'évaluent pas. La contrainte *ANY* permet aux agents de s'évaluer et par conséquent de former des coalitions de meilleure qualité tout en ne limitant que peu les agents dans la formation des groupes. Au contraire les autres contraintes sont beaucoup plus fortes et empêchent les agents de former des coalitions. La figure 7.11 présente le nombre d'expériences où il était possible de former des coalitions en

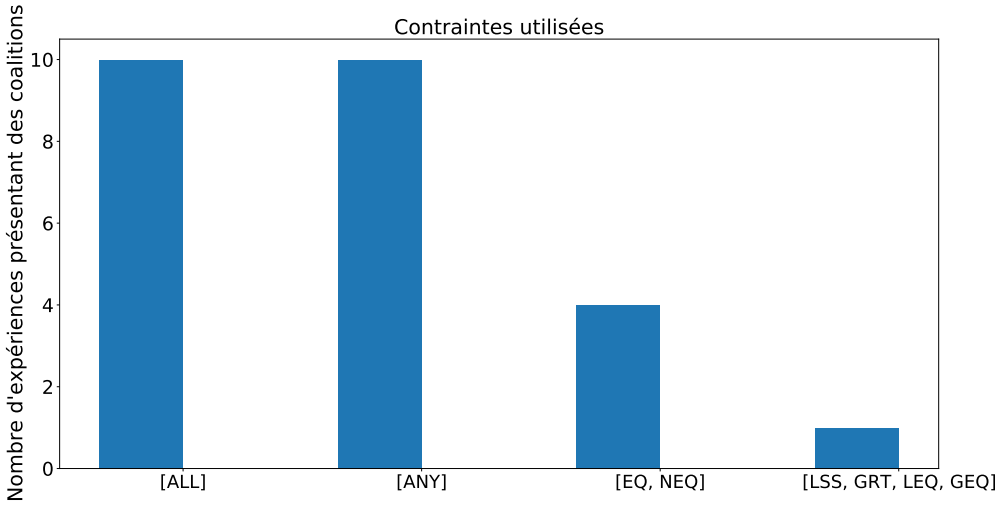


Figure 7.11 – Nombre d'expériences où il est possible de générer au moins une coalition en fonction des contraintes utilisées

fonction des contraintes utilisées. L'utilisation de contraintes plus strictes réduit de beaucoup l'espace des solutions, il en résulte des temps d'exécution plus faibles et une réduction du nombre d'expériences où le système trouve des coalitions. C'est pour cette raison que les expériences exécutées avec des contraintes strictes présentent une qualité moyenne supérieure aux autres, et il y a si peu de solutions que lorsque le système en trouve une elle est le plus souvent très proche de l'optimal.

7.3.5 De l'effet des transformations

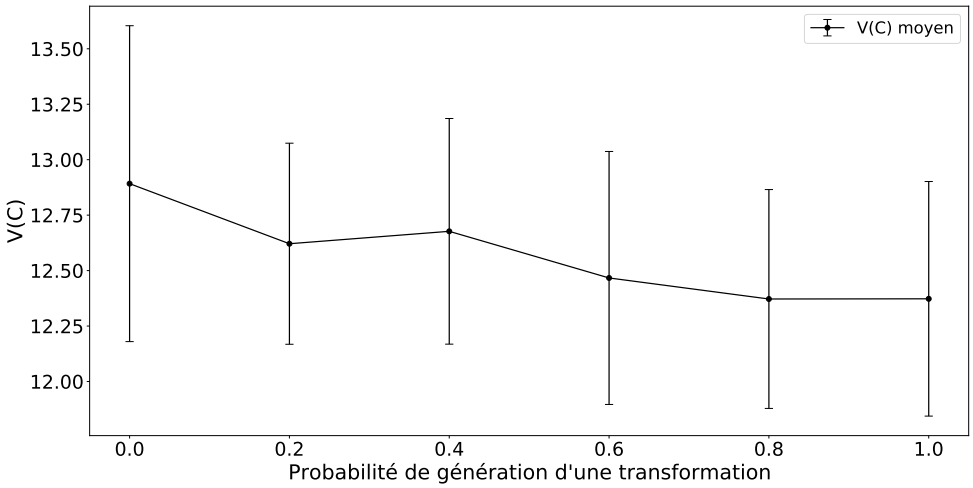


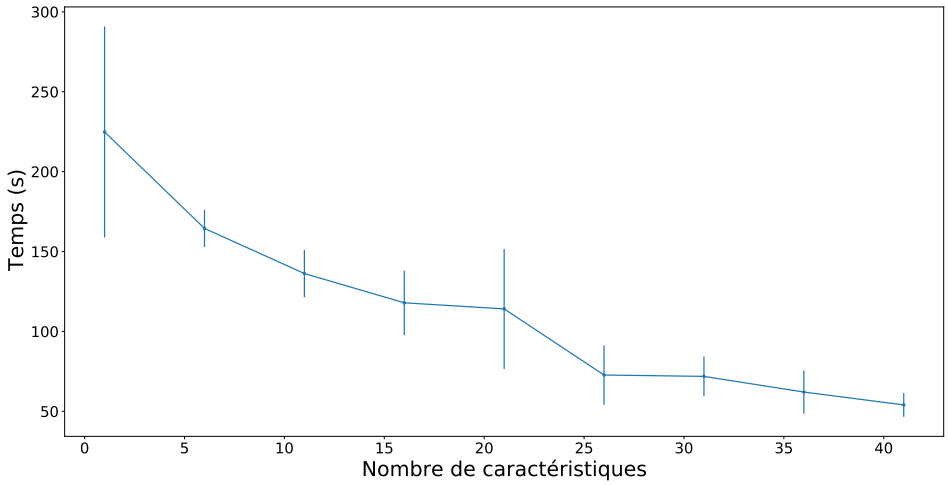
Figure 7.12 – Qualité des coalitions formées en fonction de la probabilité de génération d'une transformation dans un système de 30 agents

Dans cette expérience, nous générons des systèmes de 30 agents auxquels nous attribuons les paramètres par défaut définis dans la partie précédente. Seule exception, nous n'utilisons que la contrainte par défaut *ANY* afin de supprimer l'impact d'une génération aléatoire des contraintes et limiter le bruit sur notre graphique.

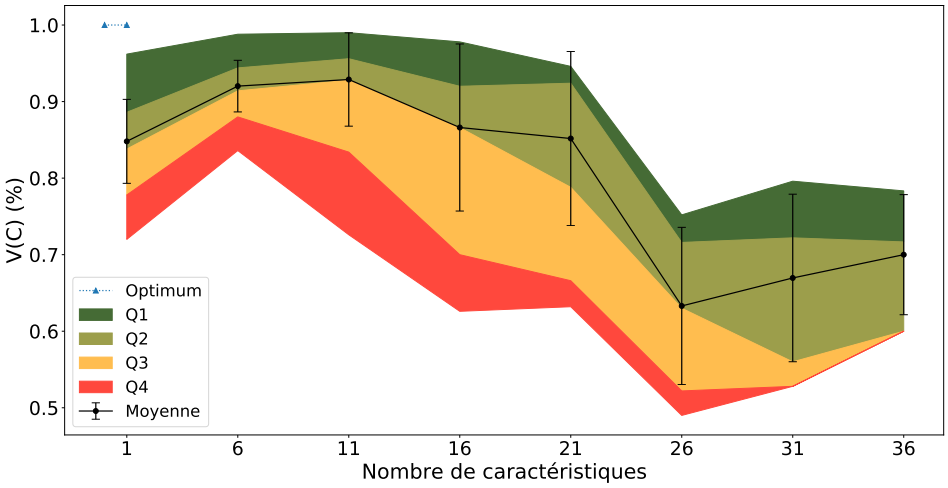
La figure 7.12 présente l'influence des transformations sur la qualité moyenne des coalitions. L'abscisse représente les probabilités que chaque caractéristique se voit affectée une transformation aléatoire dans un intervalle de $[-1,1]$. l'abscisse $x = 0$ représente donc les expériences où aucune caractéristique ne se voit affectée de transformation, l'abscisse $x = 0.1$ celle où chaque composant à 10% de chance d'obtenir une transformation et ainsi de suite jusqu'à ce que tous les composants en aient une dans le cas $x = 1.0$. Étant donné la variance des résultats et la similarité des moyennes qui varient entre 12.5 et 13, la figure ne montre pas un réel impact du nombre de transformations sur la qualité. Bien que la capacité des composants à modifier leurs caractéristiques puisse faciliter la formation de coalitions, il est probable qu'elle n'aide pas à former des coalitions de meilleure qualité au moment de l'ajout d'un membre. En effet le nouveau membre aura plus de chance de correspondre aux désirs de l'agent qui l'intègre à la coalition, mais ces caractéristiques ne seront pas forcément mieux adaptées aux désirs des autres membres qui n'ont pas leur mot à dire dans le processus d'affiliation.

7.3.6 De l'effet du nombre de caractéristiques

Nous avons souhaité voir si l'ajout de caractéristiques peut impacter le temps d'exécution du système. Nous continuons d'utiliser les paramètres renseignés dans la partie présentant nos conditions expérimentales. Nous nous limitons cependant à la contrainte *ANY* pour éviter le bruit infligé par un paramétrage aléatoire des contraintes sur la mesure du temps. Nous lançons l'expérience (figure7.13 *(a)*) avec 40 agents possédant de 1 à 41 caractéristiques. La tendance monotone décroissante de la courbe étant établie avec un nombre limité de caractéristiques, nous nous restreignons ici à en générer 41. La décroissance du temps d'exécution est contre intuitive et est liée avec l'augmentation du nombre de caractéristiques. L'augmentation de la taille des vecteurs de caractéristiques, de désirs, de transformations et de contraintes manipulées par les agents devrait au mieux montrer une courbe constante, au pire croissante. En réalité, augmenter la taille de ces vecteurs a tendance à augmenter la distance entre les désirs des agents et les caractéristiques des accointances. Supposons pour simplifier qu'un agent juge son accointance intéressante lorsque la distance entre ses désirs et les caractéristique de cette dernière est inférieure ou égale à 0.25. Supposons maintenant dans un premier temps que nos agents ne possèdent qu'une caractéristique et qu'un désir. Dans ce cas, lors de la génération des agents, chacun d'entre eux a, au maximum, une chance sur deux d'être jugé attirant (jusqu'à 0.25 de moins que le désir de l'agent et 0.25 de plus). Supposons maintenant que les agents possèdent 2 caractéristiques et 2 désirs. Nous pouvons représenter la zone des accointances intéressantes (noté *zone d'attrait*) sur un plan 2D comme un cercle (figure 7.14). La probabilité maximum qu'une accointance soit générée avec les caractéristiques entrant dans cette zone correspond à l'aire du cercle divisé par l'aire totale de l'espace, soit environ 0.19. Nous parlons ici de probabilité maximum puisque le désir de l'agent peut aussi bien se situer au centre du plan comme dans l'exemple que sur son origine *(0, 0)*. Dans le second cas, puisque les agents ne peuvent générer de caractéristiques négatives, la probabilité de voir une



(a) Temps des expériences pour 40 agents en fonction de leur nombre de caractéristiques



(b) Qualité des coalitions en fonction de nombre de caractéristiques

Figure 7.13 – Évaluation de l'effet du nombre de caractéristiques sur le temps d'exécution et la qualité des coalitions

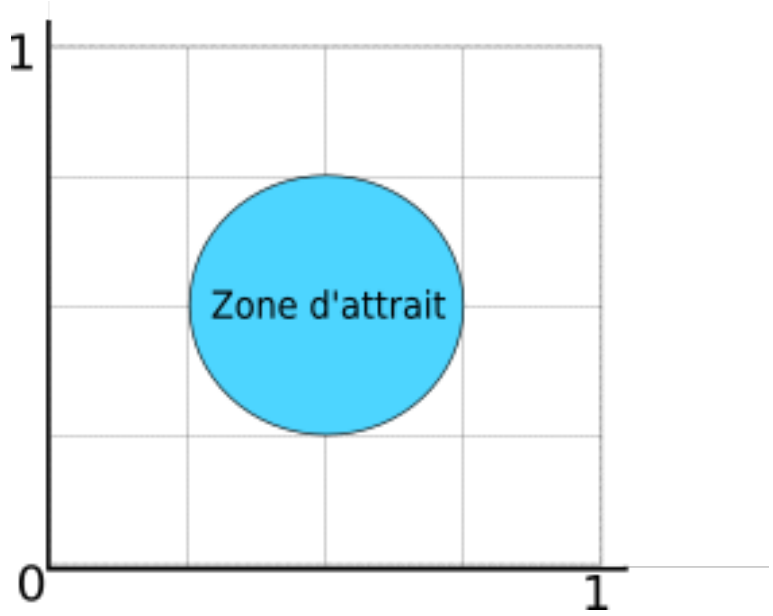


Figure 7.14 – Exemple de la zone d'attrait pour un plan en 2 dimensions

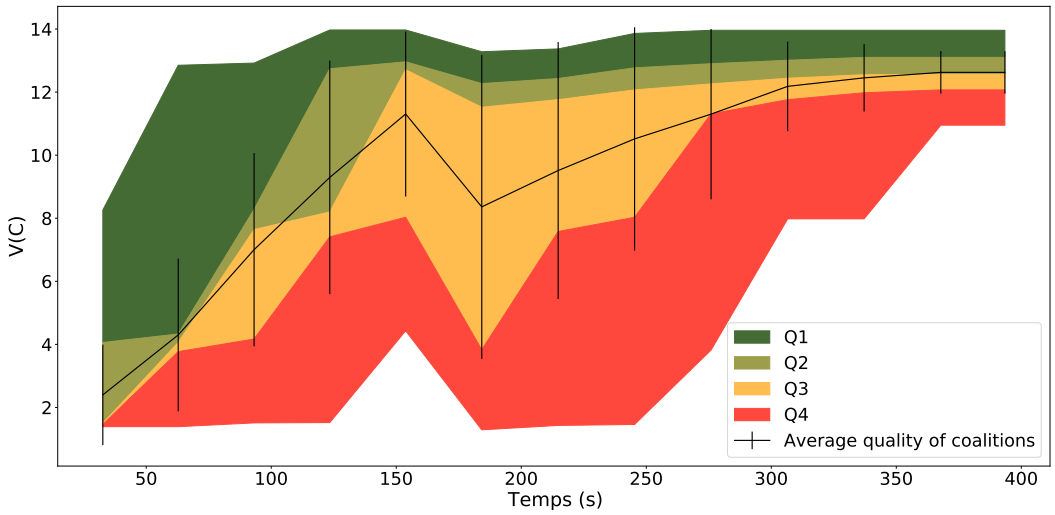
accointance dans cette zone serait quatre fois moins importante. Continuons avec un plan en 3 dimensions, avec le même raisonnement, la plus haute probabilité qu'un agent soit dans la zone jugée attrayante des désirs en 3 dimensions d'un agent est le ratio entre le volume de la sphère et le volume du plan, soit environ 0.065. Nous pouvons observer le rétrécissement progressif et rapide des probabilités d'apparition d'une accointance désirable au fur et à mesure que l'on augmente le nombre de caractéristiques. Il résulte de ce mécanisme une diminution progressive du nombre d'accointances jugées attrayantes par les agents du système et une augmentation de l'attraction moyenne que les agents ont les uns pour les autres. Or, plus l'attraction est élevée et moins les agents sont enclins à former des coalitions et moins le système peut produire de solutions, d'où l'augmentation de la vitesse d'exécution à mesure que l'on ajoute des caractéristiques. Cet effet indique qu'il serait intéressant d'adapter les seuils permettant les formations et les affiliations aux coalitions en fonction du nombre de caractéristiques manipulées par les agents, notamment s'ils en manipulent des quantités très différentes. Notons que cet effet n'a pas impacté les résultats de cette partie puisque toutes les expériences ont été lancées avec des agents possédant le même nombre de caractéristiques et désirs.

La figure 7.13 *(b)* présente la qualité des coalitions formées et la moyenne de leur qualité pour un nombre de caractéristiques et désirs variables. La figure précédente (7.13 *(a)*) montrait une réduction du temps d'exécution avec l'augmentation du nombre de caractéristiques et désirs des agents. La qualité des coalitions subit le même effet et se voit fortement impactée à partir de 26 caractéristiques. Comme expliqué, plus les agents manipulent un nombre de caractéristiques important, moins ils trouvent facilement d'accointances intéressantes. Sur notre figure, la 26^{e} caractéristique représente le point de rupture à partir duquel les agents n'ont plus assez d'accointances pour former des groupes de bonne qualité. Nous pouvons également observer la diminution de la taille des quartiles, par exemple à la 36^{e} caractéristique où les quartiles 3 et 4 disparaissent. Leur disparition est due au trop faible nombre de coalitions formées par les agents (2 dans ce cas là),

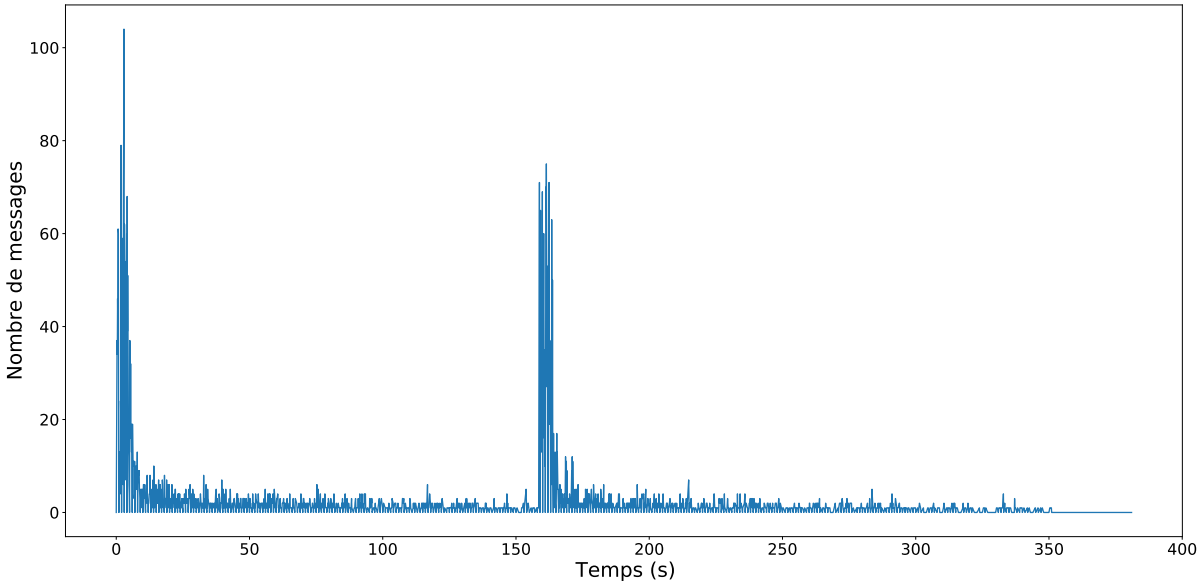
faussant la représentation en quartiles.

Pour résumer, l'augmentation du nombre de caractéristiques et désirs manipulés par les agents ABSG réduit leur probabilité de trouver des accointances jugées de bonne qualité. Ce manque d'accointances induit une plus grande difficulté à former des coalitions et entraîne une réduction importante de leur qualité et du temps d'exécution du système. Cet effet indésirable peut cependant être évité en adaptant la définition de ce qu'est une accointance intéressante en fonction du nombre de caractéristiques des agents et ainsi leur permettre d'élargir leur champ de recherche.

7.3.7 L'adaptabilité face à un système variable



(a) Évolution de la qualité des coalitions formées par le système



(b) Nombre de messages dans le système

Figure 7.15 – Évaluation de la variabilité dans un système composé de 50 agents. Modification aléatoire des caractéristiques et désirs des agents à 150 secondes

Dans cette expérience de 60 agents, nous voulons voir comment le système réagit si les agents

changent aléatoirement leurs caractéristiques et leurs désirs. Contrairement aux expériences précédentes, nous ne connaissons pas la valeur optimale qu'une coalition peut avoir. À environ 100s, l'observateur demande à tous les agents de modifier aléatoirement les valeurs de leurs caractéristiques et de leurs désirs. Conformément à la règle 8, lorsque les agents reçoivent de nouvelles caractéristiques, ils doivent les envoyer à leurs accointances, ce qui entraîne l'explosion des messages émis (figure 7.15 *(b)*). La qualité des coalitions est affectée par ces modifications (figure 7.15 *(a)* à environ 160 secondes). Lorsqu'ils reçoivent les nouvelles caractéristiques de leurs accointances, les agents mettent à jour toutes leurs valeurs d'attraction et se réorganisent, ce qui augmente la moyenne des qualités de coalition. Seules les coalitions détériorées par les changements de caractéristiques et de désirs des agents se réorganisent. Si une coalition reste de bonne qualité après la modification des agents, celle-ci n'est pas modifiée. Le système recherche des solutions de manière incrémentale et peut s'adapter à des modifications de ses agents sans avoir à reconstruire les coalitions en partant de zéro. En outre, notons que le système peut retourner une solution à n'importe quel moment.

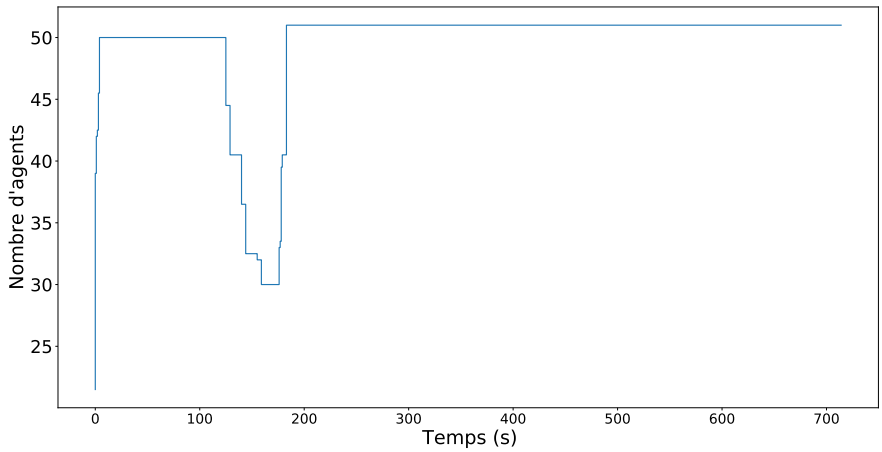
7.3.8 L'adaptabilité face à un système ouvert

Nous voulons voir comment le système réagit lorsque nous ajoutons ou retirons des agents. La figure 7.16 *(c)* présente la qualité moyenne des coalitions lors de l'exécution d'un système de 50 agents. Du fait de que des agents sont introduits durant le temps d'exécution de l'expérience, il est difficile de prédire à l'avance la qualité de la coalition optimale. C'est pourquoi V(C) est ici exprimé comme une valeur brute et non comme un pourcentage de l'optimal comme dans les expériences précédentes.

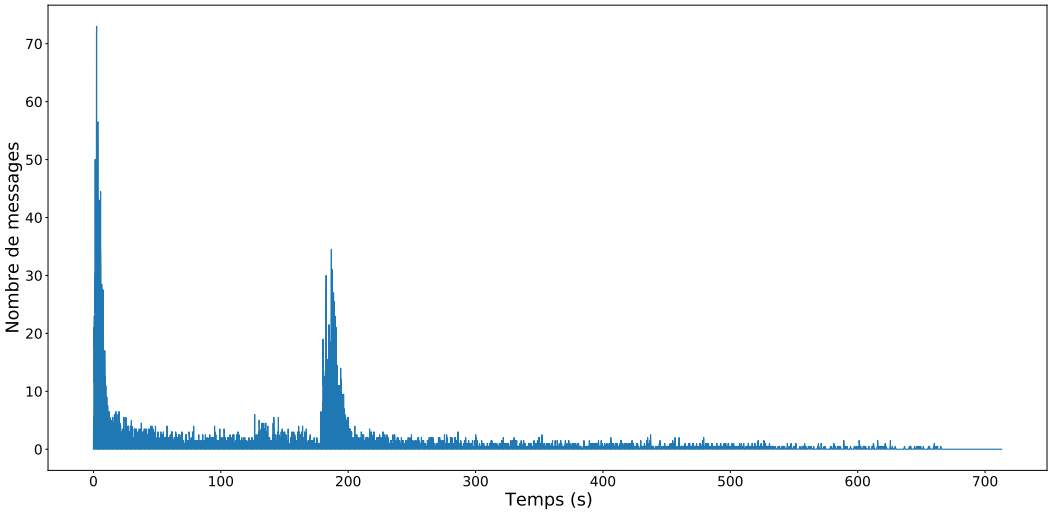
Après 100 secondes écoulées, le processus observateur demande à 20 agents aléatoires du système de s'en retirer. Nous ne pouvons observer un effet notable de leur départ sur la qualité des coalitions. Cependant l'ajout de 21 nouveaux agents quelques secondes plus tard semble faire drastiquement diminuer leur qualité aux alentours de 180 secondes. L'ajout de nouveaux membres entraîne une diminution importante de la qualité moyenne car au moment de leur entrée dans le système, ceux-ci créent de nouvelles petites coalitions qui n'ont pas eu le temps de se développer. L'ajout de nombreuses coalitions de piètre qualité dégrade nettement la moyenne de la qualité des coalitions. Cependant les coalitions se remplissent avec le temps et la qualité moyenne augmente à nouveau jusqu'à se stabiliser.

La figure 7.16 *(b)* montre le nombre de messages émis durant l'expérience. Comme pour les autres expériences, le premier pic représente le moment où les agents échangent leurs caractéristiques et leurs désirs. Le nombre de messages est ensuite peu élevé jusqu'au moment où les agents quittent le système. À environ 120 secondes une suite de petits pics montrent que des agents préviennent leurs accointances de leur départ de leurs coalitions. Lorsque les agents entrent dans le système un second grand pic apparaît aux alentours de 200 secondes qui est dû aux échanges des caractéristiques et désirs entre les nouveaux agents et ceux qui étaient déjà présents.

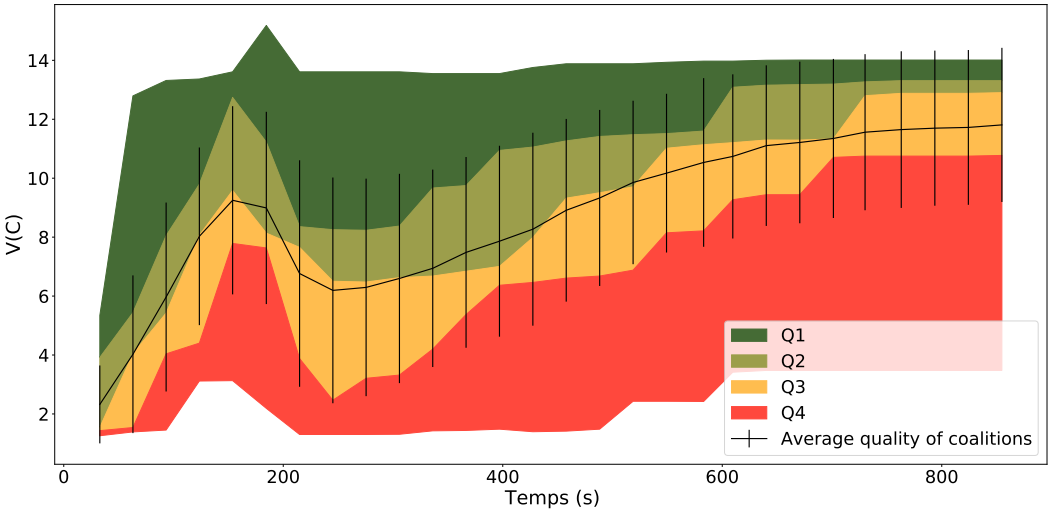
Cette évaluation montre la capacité du système à s'adapter dynamiquement à un environnement ouvert. Les agents se montrent capables de se réorganiser lorsqu'un évènement inattendu apparaît.



(a) Nombre d'agents dans le système en fonction du temps



(b) Nombre de messages émis dans le temps



(c) Mesure de la qualité des coalitions dans le temps

Figure 7.16 – Évaluation de l'ouverture du système par la suppression et l'ajout d'agents à respectivement 120 et 180 secondes

7.3.9 Synthèse et discussion

Notre approche se positionne entre les méthodes de programmation dynamique qui obtiennent des solutions exactes mais dont le passage à l'échelle est pour le moment impossible, et les algorithmes de clustering [Farinelli2017] qui fonctionnent avec plusieurs centaines – parfois milliers – d'agents dans des temps très courts mais où la qualité des solutions peut être moins importante. Les évaluations ont montré que notre système est capable de trouver des solutions proches de l'optimal malgré un grand nombre d'agents. Nous avons évalué les effets des principaux paramètres des agents ABSG et observé que certains d'entre eux pouvaient avoir un impact important sur la qualité des résultats et le temps d'exécution du système. Le nombre de caractéristiques ou le type de contraintes utilisées influencent grandement le comportement du système et des agents. Il semble donc important pour le concepteur d'un système d'agents ABSG de réfléchir au préalable au type de produits que celui-ci veut réutiliser. Si les produits possèdent un nombre très hétérogène de caractéristiques, la fonction de seuils doit être adaptée afin de permettre aux agents les plus détaillés de tout de même trouver des accointances avec lesquelles former des coalitions. Au contraire, si le cas applicatif ne nécessite que très peu de contraintes, il faudrait plutôt adapter la fonction de seuil de façon à ce que les agents deviennent plus sélectifs et diminuent leur nombre d'accointances jugées intéressantes pour réduire le temps d'exécution du système.

L'un des avantages que propose notre approche est sa capacité à prendre en compte des contraintes réalistes comme l'ouverture et la variabilité du système. Comme nous l'avons vu dans les sections 7.3.7 et 7.3.8, notre approche permet cette flexibilité et les agents de notre système se montrent capables de se réorganiser de manière incrémentale en cas de modification de l'un d'eux.

7.4 Évaluation sur le cas d'étude

Tandis que la partie précédente s'efforçait d'évaluer notre approche à travers un cas général de formation de coalitions, cette partie se concentre sur la validation de l'architecture ABSG dans le cas d'étude construit dans le chapitre 6. Contrairement au scénario générique de la partie précédente, les composants de ce cas d'étude ne sont pas tous inter-connectés. Un agent du système n'a donc pas nécessairement le désir d'ajouter toutes ses accointances dans sa coalition. En outre, l'objectif des agents des systèmes générés dans cette partie sera de concevoir des coalitions représentant des lave-linge composés de neufs agents contrairement aux systèmes précédents qui ne comptaient que cinq agents au maximum. Nous verrons dans cette partie comment ces changements influencent notre système multi-agent.

7.4.1 Modification de certains paramètres d'ABSG

La réduction de l'interconnexion des agents dans ce cas d'étude complexifie la formation de coalitions complètes. En effet, les agents n'ont connaissance que de leur objectif individuel – s'associer avec les agents qu'ils désirent – et non de l'objectif global du système. Dans le cas d'étude du lave-linge, un agent *Pompe de Vidange* sait qu'il doit s'affilier avec des agents *Cuve* et *Châssis*,

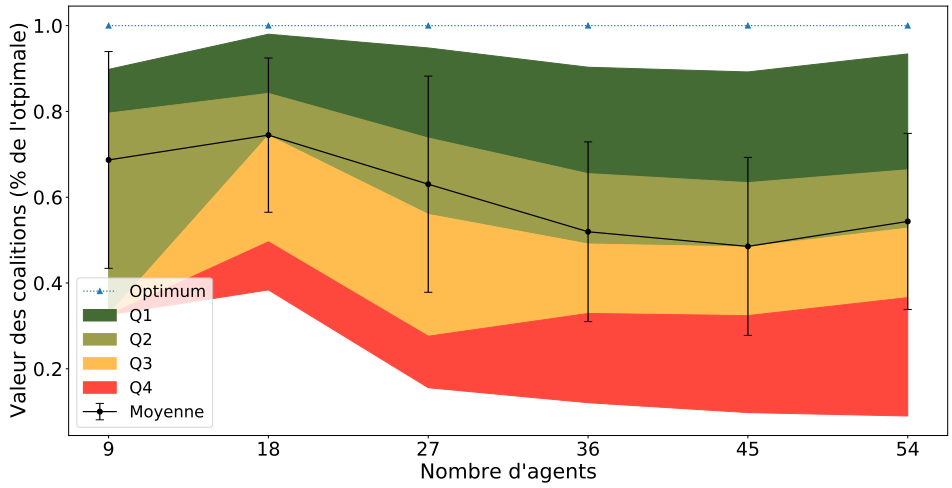


Figure 7.17 – Qualité des coalitions formées en fonction du nombre d'agents sur le cas d'étude présenté dans le chapitre 6.

mais n'a pas connaissance de l'existence des agents qui ne lui sont pas associables comme les agents *Moteurs* ou *Programmateur*. Contrairement à la partie précédente, la seule chance qu'ont des agents de former une coalition complète est d'ajouter comme membre à leur groupe les agents qui possèdent des désirs différents des leurs. La fonction d'attraction ne doit donc plus se focaliser sur l'attractivité des accointances mais sur le principe de complémentarité permettant aux agents de trouver attractifs les agents qui ont des désirs différents des leurs. Par conséquent, nous modifions les poids de la fonction d'attraction en surpondérant la complémentarité (*poids(similarité) = 1*, *poids(attractivité) = 1*, *poids(complémentarité) = 5*).

Pour cette même raison, une coalition composée de deux membres *Tambour* et *Roulements* peut difficilement s'agrandir si les accointances de type *Moteur* et *Cuve* ne sont pas jugées très attractives. Nous adaptons donc également les seuils de formation et d'affiliation aux coalitions pour agrandir le rayon de recherche d'accointances et éviter d'entraver le développement des coalitions les plus petites. Les seuils utilisés pour ces expériences sont calculés avec la même équation que celle présentée dans le chapitre 5 et la partie présentant les conditions expérimentales. Nous modifions cependant le paramètre *maxout* à *8* (environs doublé par rapport aux expériences de la partie précédente) et fixons le seuil de formation de coalition à *3*.

7.4.2 De l'effet du nombre d'agents

Nous évaluons dans cette section l'effet du nombre d'agents sur le système multi-agent. L'évolution de la qualité en fonction du nombre d'agents est présentée sur la figure 7.17. Nous constatons que celle-ci est moins importante, beaucoup plus dispersée sur l'ensemble du spectre des résultats possibles (les qualités des coalitions vont d'environ 10% dans les pires cas à 97% dans les meilleurs) et plus dépendante du nombre d'agents que dans la partie précédente. Sur des systèmes à petite échelle, la première partie montrait les coalitions du quartile 1 comme allant au-delà des 95% de qualité de la coalition optimale alors que dans cette expérience, la qualité descend jus-

qu'à 80%. Elle descend également jusqu'à 60% sur les systèmes un peu plus peuplés (de 27 à 54 agents), là où elle se stabilisait à environ 90% sur le cas d'étude générique. La figure montre également des coalitions de très mauvaise qualité dans le Q4, descendant jusqu'à environ 10% de la qualité optimale trouvable. Ces différences de résultats s'expliquent par la faible connectivité des composants de notre cas d'étude contrairement à celui de la partie précédente. Dans cette dernière, un agent *A* pouvait ajouter à sa coalition les agents *C* , *D* et *E* si un bon agent *B* était introuvable ce qui permettait un développement rapide des coalitions et améliorait leur qualité. Dans notre cas d'étude *lave-linge*, un agent *Programmateur* ne peut ajouter personne dans sa coalition à part un agent *Moteur* et un agent *Châssis*. Si l'un d'eux n'est pas présent dans sa coalition et qu'il ne le trouve pas attractif, alors le groupe réduit fortement ses chances de se compléter. Nous voyons que dans ce contexte, le développement des coalitions est plus complexe à opérer. Toutes les coalitions créées et incapables de s'agrandir sont au final très mal évaluées par le processus observateur qui prend en compte la taille du groupe pour lui affecter un score de qualité. C'est pourquoi cette expérience montre des coalitions de mauvaise qualité représentées dans le quartile 4 et c'est pourquoi la moyenne des qualités des coalitions est si faible. Une solution à ce problème consisterait à faire décroître les seuils d'affiliation et de départ d'une coalition permettant à celles de moindre qualité de se dissoudre et ainsi de faire augmenter la qualité moyenne. Il serait alors intéressant de se demander quand appliquer la décroissance des seuils et si cette décroissance ne pourrait pas rentrer en conflit avec la capacité du système multi-agent à s'adapter au contexte variable du système d'aide à la décision. Par exemple, faire décroître les seuils au moment où un nouvel agent entre pourrait pousser les agents déjà présents du système à ignorer le nouvel arrivant, même s'il était adapté à une des coalitions. Mais si les agents pouvaient mettre en place une stratégie d'adaptation dynamique de ces seuils en fonction de leur situation individuelle, cela permettrait une amélioration de la qualité moyenne des coalitions. Par exemple, les rendre capable de diminuer la valeur des seuils lorsque le système autour d'eux se stabilise (à travers la diminution du nombre de messages reçus), et les ré-augmenter lorsqu'un nouvel agent se présente à eux.

Mais même sans ce mécanisme d'adaptation, augmenter le seuil d'affiliation à une coalition permet aux agents d'être moins sélectifs sur la qualité de leurs accointances et de développer leurs coalitions pour atteindre une meilleure qualité. En contrepartie, le temps d'exécution du système est augmenté lorsque le seuil est modifié. En effet, les agents sélectionnent moins leurs accointances et passent donc plus de temps à interagir avec elles pour les intégrer à leurs coalitions (figure 7.18). Le mécanisme d'adaptation dynamique du seuil d'affiliation discuté dans le paragraphe précédent pourrait là aussi avoir un intérêt. Diminuer le seuil d'affiliation petit à petit au cours de l'exécution pourrait éventuellement diminuer le temps d'exécution du système en retirant quelques possibilités d'affiliation à l'agent en plus d'améliorer la qualité des coalitions.

Un autre facteur influe sur les temps d'exécutions : la taille des coalitions à former. Les expériences précédentes avaient pour objectif de former des coalitions de cinq agents ce qui est intuitivement plus rapide que de former des coalitions de neuf agents. D'autant plus quand le protocole de formation de coalitions interdit aux membres d'un même groupe de le modifier en même temps. Concevoir un protocole permettant de paralléliser l'ajout ou la suppression de membres d'une même coalition en même temps diminuerait probablement les temps que nous observons.

Comme pour la première partie, nous représentons l'évolution des messages en fonction du nombre d'agents dans la figure 7.19. La fonction tracée semble être une fonction logarithmique.

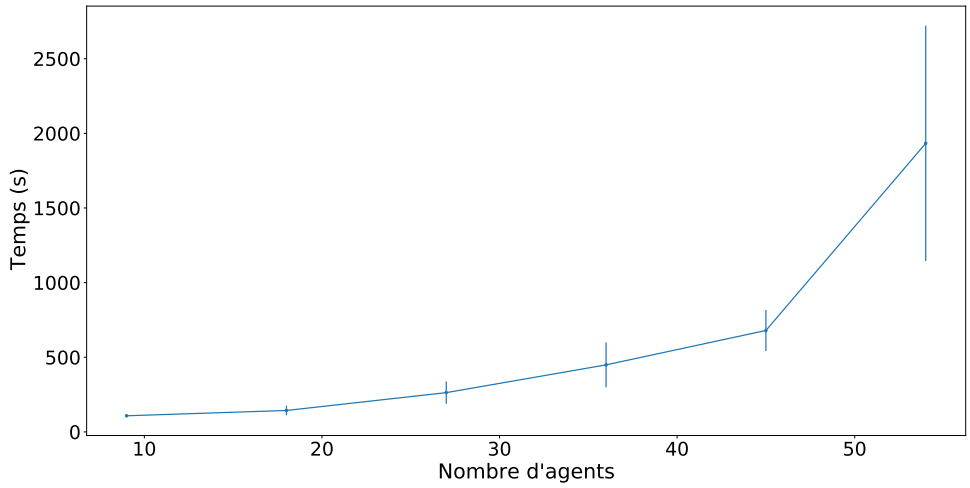


Figure 7.18 – Temps d'exécution du système en fonction du nombre d'agents

Le nombre de messages ne semble donc pas exploser en augmentant le nombre d'agents. Il faut toutefois être prudent puisqu'en augmentant de manière significative le nombre d'agents, il est possible que ceux-ci puissent communiquer avec un nombre beaucoup plus important d'accointances et que leur protocole d'introduction au moment d'entrer dans le système puisse faire augmenter de manière plus brutale le nombre de messages émis par le système.

Pour conclure cette section, le cas d'étude proposé dans le chapitre précédent est très différent de celui utilisé pour évaluer notre approche. La faible connectivité des composants d'un lave-linge induit une plus grande difficulté pour les agents à former des coalitions et donc une baisse de leur qualité et une augmentation du temps d'exécution. Cependant, 25% des coalitions formées atteignent des qualités au-delà de 70% de la qualité optimale sur des instances de systèmes allant de 9 à 54 agents. Bien que la qualité moyenne des coalitions est diminuée par le peu de connectivité inter-composants, la mise en place d'une stratégie dynamique des agents dans leur façon de calculer leur seuil d'affiliation à une coalition pourrait permettre une augmentation de la qualité moyenne en dissolvant les groupes de moins bonne qualité. En outre, la conception d'un protocole parallélisant les modifications faites à une coalition pourrait significativement diminuer le temps d'exécution. Comme pour la partie précédente, la section suivante évalue les principes de l'attraction sur la qualité des coalitions formées. Nous utiliserons cette fois le cas d'étude *lave-linge* pour essayer de déterminer l'influence de ces différents facteurs.

7.4.3 De l'effet des principes de l'attraction

Cette section présente les effets des principes de l'attraction avec leur nouvelle pondération dans le cas d'étude *lave-linge*. Étant donné que les poids des principes de l'attractivité et de la complémentarité ont été modifiés, cette expérience n'a pas vocation a être comparée avec celle de la partie précédente. Son objectif est plutôt de montrer leurs effets sur le temps d'exécution du système et la qualité des coalitions formées et de proposer une explication à ces effets.

Pour cette évaluation, nous configurons la fonction d'attraction de cinq manières différentes

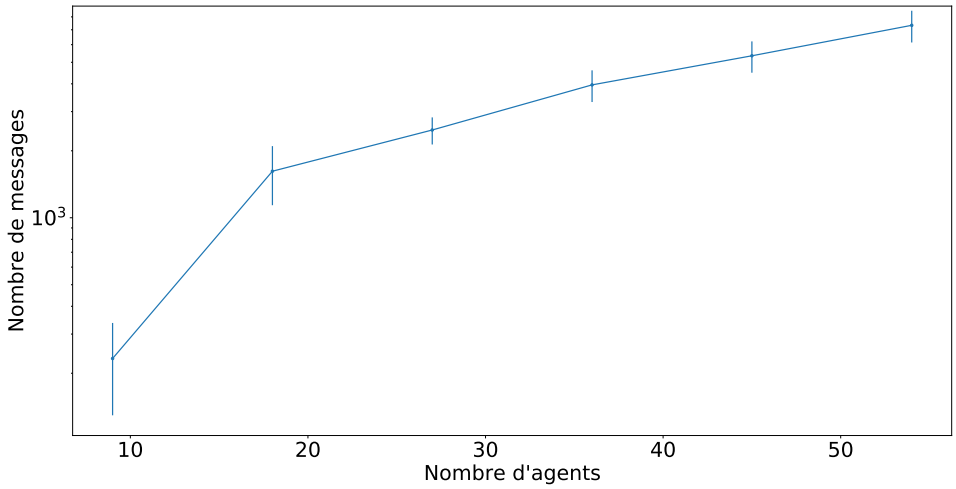


Figure 7.19 – Évolution du nombre de messages en fonction du nombre d'agents

de manière à isoler les principes que nous souhaitons évaluer puis nous exécutons chaque système multi-agent avec 36 agents ce qui permet d'avoir le même nombre d'agents de chaque type tout en gardant un temps d'exécution raisonnable. Les configurations expérimentées dans cette section sont les suivantes :

1. Nous retirons la similarité de la fonction pour en déterminer l'impact sur la qualité moyenne et maximum des coalitions formées. Nous évaluons donc la fonction sur la base des **principes d'attractivité physique, de complémentarité** et sur le minimax ;
2. Pour les mêmes raisons nous retirons uniquement la complémentarité. Restent **la similarité, l'attractivité physique** et le minimax ;
3. Nous retirons l'attractivité physique tout en gardant **la similarité, la complémentarité** et le minimax ;
4. Seule la **complémentarité** et le minimax sont préservés.
5. Cette dernière configuration représente la fonction dans son ensemble **sans modification de façon à comparer les résultats** des autres configuration à celle-ci.

Ces résultats ont été obtenus après un moyennage sur 10 expériences dans chacune des configurations présentées. Pour chaque expérience, nous avons recueilli la qualité des coalitions après stabilisation du système et en avons fait un jeu de données pour comparer nos différentes configurations. Pour cela, nous avons utilisé le test statistique de Shapiro-Wilk permettant de montrer la distribution normale de nos données parmi les jeux. Chacun des jeux de données a présenté une distribution normale avec un *p-valeur* supérieure à 0.05. Les variances étant inégales pour les jeux de données, nous avons vérifié l'égalité de leur moyenne à l'aide d'un test t de Welch.

Comme nous le décrivions en début de partie, la complémentarité joue un rôle important dans ce cas d'étude en permettant aux agents d'améliorer leur attraction en présence d'accointances possédant des objectifs d'appariement différents des leurs. Cette évaluation aborde donc le principe de complémentarité comme un élément central dans le processus de formation de coalition et l'isole des autres principes pour mieux comprendre ses effets. La figure 7.20 présente la qualité moyenne

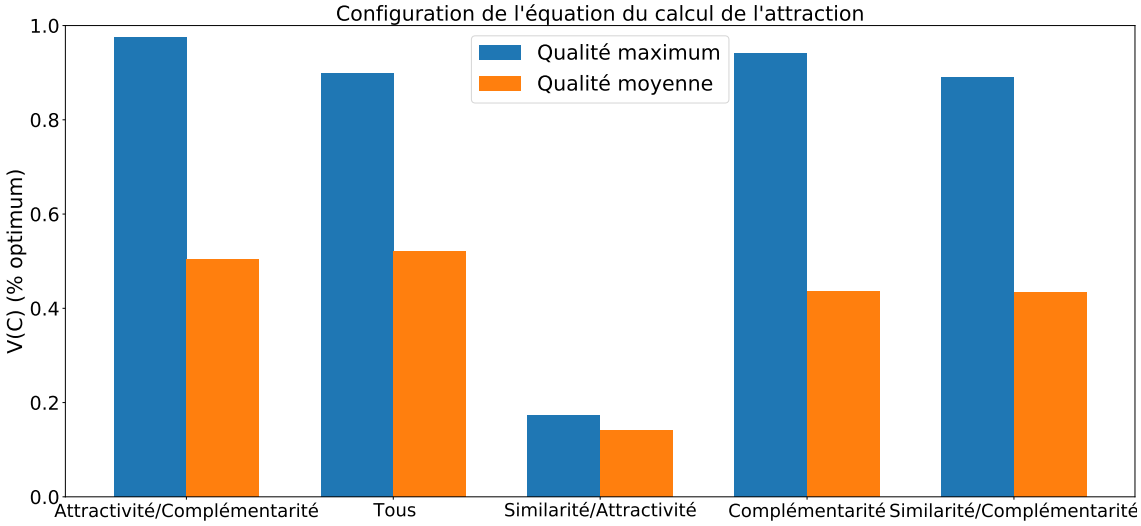


Figure 7.20 – Effet des principes de l'attraction sur la qualité des coalitions

des coalitions formées dans les cinq configurations précédemment présentées. La configuration *Complémentarité* montre que ce principe permet à lui seul d'obtenir un peu plus de 40% de qualité moyenne et des maximums se rapprochant de la qualité optimale (>90% de qualité). Lorsque nous ajoutons à ce principe la *similarité* dans la configuration *Similarité/Complémentarité*, la qualité moyenne n'augmente pas et la qualité maximum diminue légèrement. Cependant, le test t de Welch montre une *p-valeur* de 0.64 ne permettant pas de différencier les deux jeux de données et nous supposons donc que la similarité n'a pas d'effet notable dans ce cas d'étude sur la qualité des coalitions formées. La configuration *Similarité/Attractivité* montre que la complémentarité est un principe important dans le cadre de notre cas d'étude puisque sans ce principe la qualité moyenne des coalitions ne dépasse pas 20% de la qualité optimale. Mais l'attractivité n'est pas pour autant un facteur inutile puisque la configuration *Attractivité/Complémentarité* montre une qualité moyenne supérieure à la complémentarité seule (avec une *p-valeur* de 0.004). Enfin, la dernière configuration *Tous* qui rassemble tous les principes de l'attraction montre une qualité encore légèrement supérieure sur la moyenne. Cependant ce résultat ne peut être avéré puisque le *p-valeur* entre les jeux de données des deux configurations est de 0.308 ce qui ne permet pas d'affirmer une réelle différence. Nonobstant ce résultat confirme le fait que la similarité ne semble pas jouer de rôle majeur dans ce contexte applicatif.

7.4.4 De l'effet des transformations

La figure 7.21 présente l'évolution de la qualité moyenne des coalitions en fonction de la probabilité de la génération d'une transformation. Comme dans la partie précédente, $x = 0$ signifie que les caractéristiques n'ont aucune transformation disponible,$x = 0.5$ qu'ils ont une chance sur deux de se voir affecter une transformation, *etc*. Cette expérience ne montre pas non plus de fort effet des transformations sur la qualité des coalitions. En outre, les transformations sont limitées sur ce cas d'étude aux agents *Tambour*, *Cuve* et *Contrepoids* tels que décrits dans le chapitre 6.

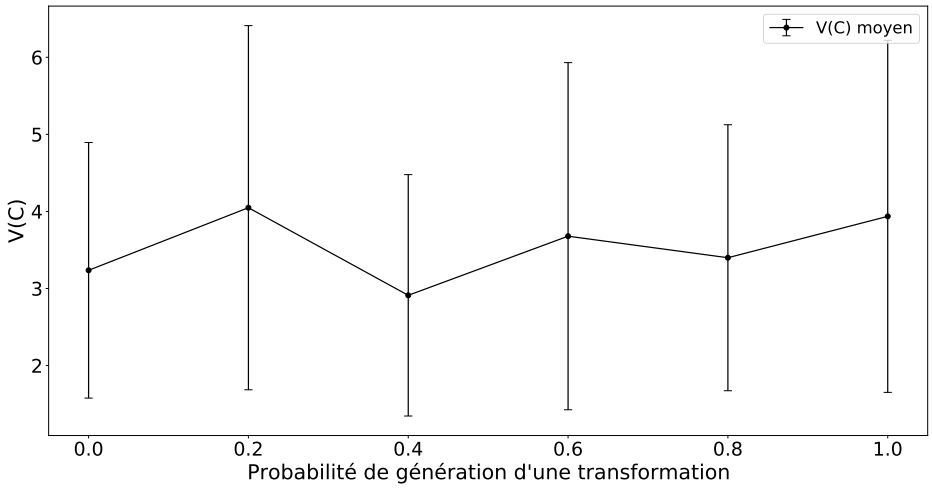


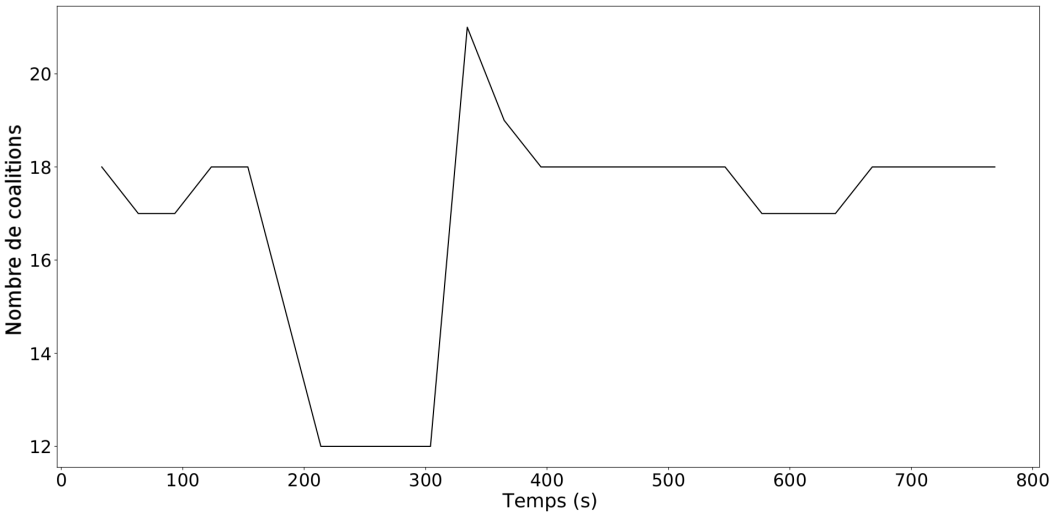
Figure 7.21 – Évolution de la qualité moyenne des coalitions en fonction de la probabilité de génération d'une transformation

7.4.5 L'adaptabilité face à un système ouvert

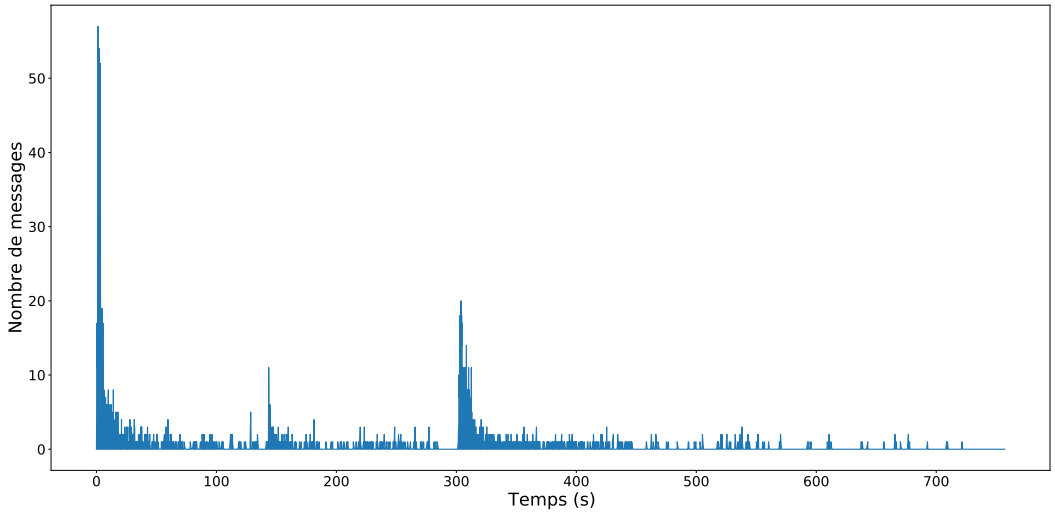
La figure 7.22 présente l'évaluation de l'ouverture dans le cadre de notre cas d'étude *lave-linge*. Le système est composé de 54 agents (6 agents de chaque type) et se voit retirer 18 agents aléatoires à 120 secondes puis ajouter 18 nouveaux à 300 secondes. Le graphique *(a)* montre l'évolution du nombre de coalitions dans le système à chaque instant de l'expérience, le graphique *(b)* le nombre de messages émis et le graphique *(c)* la qualité moyenne des coalitions. La suppression des agents influe sur le nombre de groupes puisque ceux-ci sont forcés de quitter leurs coalitions. Pour quitter leurs coalitions les agents sont obligés de prévenir les autres membres ce qui induit une augmentation du nombre de messages émis dans le système à environ 150 secondes. En quittant leurs groupes, certaines coalitions perdent légèrement en qualité ce qui fait stagner la qualité moyenne des coalitions.

Il n'est pas étonnant de constater que lorsque des agents sont ajoutés au système à 300 secondes, les nouveaux entrants interagissent avec les agents déjà présents et le nombre de groupes augmente. De la même manière qu'à l'initialisation du système, les nouveaux agents échangent leurs caractéristiques et désirs avec leurs accointances ce qui provoque un pic de messages à environ 300 secondes. Mais contre intuitivement la qualité diminue une seconde fois au moment de leur introduction. En effet, les coalitions se créent toujours à partir de deux membres puis évoluent lorsque les membres originaux contactent leurs accointances. Ces coalitions de deux membres sont de très mauvaises qualité puisqu'elles ne répondent pas aux désirs de leurs membres (notamment à travers leur incomplétude) et leur ajout au calcul de la qualité moyenne des coalitions fait diminuer cette dernière. Le temps passant, ces coalitions deviennent de plus en plus importantes en ajoutant de nouveaux membres et leur qualité augmente ce qui permet aussi l'augmentation de la qualité moyenne des coalitions du système.

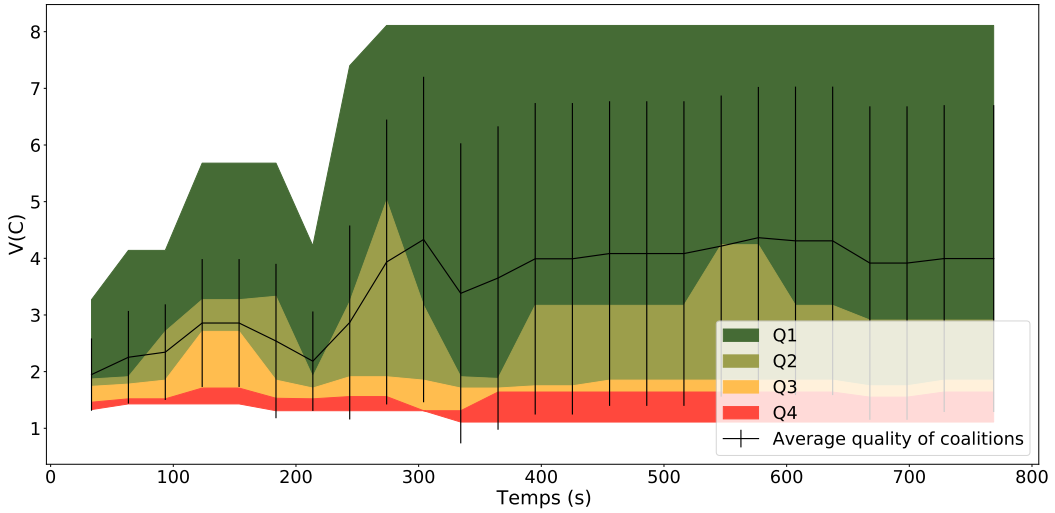
Cette évaluation montre que les agents sont capables de s'adapter face à l'ouverture du système se réorganisant lorsqu'une de leur accointance quitte subitement une coalition, mais aussi en



(a) Nombre de coalitions dans le système en fonction du temps



(b) Nombre de messages émis en fonction du temps



(c) Qualité des coalitions en fonction du temps

Figure 7.22 – Évaluation de l'ouverture du système par la suppression et l'ajout d'agents à respectivement 120 et 300 secondes

ajoutant de nouveaux membres lorsque ceux-ci entrent dans le système. La section suivante évalue une capacité similaire à l'ouverture, celle des agents à s'adapter à un environnement variable, et observer la réaction du système lorsque ses composants sont modifiés.

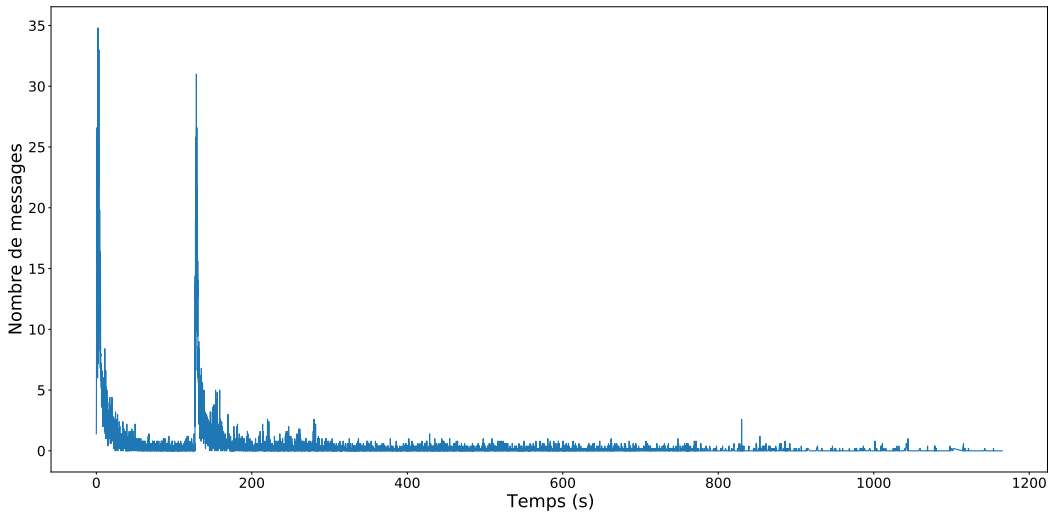
7.4.6 L'adaptabilité face à un système variable

La figure 7.23 présente un système de 54 agents dans lequel chacun d'entre eux modifie ses caractéristiques et désirs de manière aléatoire lorsque le processus observateur le leur demande, 120 secondes après le lancement du système. Jusqu'au moment où le signal est envoyé, le graphique *(c)* montre que les coalitions gagnent en qualité. Au moment où les agents reçoivent le signal de l'observateur, ils se voient attribuer de nouvelles caractéristiques et désirs puis en informent les autres agents. Le moment où tous les agents échangent de nouveau leurs caractéristiques correspond au second pic de messages à environ 150 secondes sur le graphique *(a)*. Lorsque tous les agents connaissent les nouvelles caractéristiques et désirs de leurs accointances, ceux-ci forment rapidement de nouveaux groupes ce qui peut s'observer sur le graphique *(b)* à environ 200 secondes. Comme dans l'expérience sur l'ouverture du système, la création de nouveaux groupes impacte la qualité moyenne des coalitions (graphique *(c)*). Dans le même temps, les agents remettent en question les coalitions formées avant les changement de caractéristiques en réevaluant leur attraction pour elles. Les plus mauvaises d'entre elles sont quittées par les agents faisant passer le nombre de coalitions d'environ 106 à 200 secondes, à environ 95 à la fin de l'expérience. Outre le fait que certain agents ne se retrouvent plus dans certaines de leurs coalitions, leur départ peut également engendrer le départ de leurs pairs qui, faute d'avoir un agent qui répond à leur désir, ne trouvent plus la coalition suffisamment intéressante et la quitte à leur tour. Ainsi, la désertion des coalitions se fait étape par étape jusqu'à leur dissolution complète.

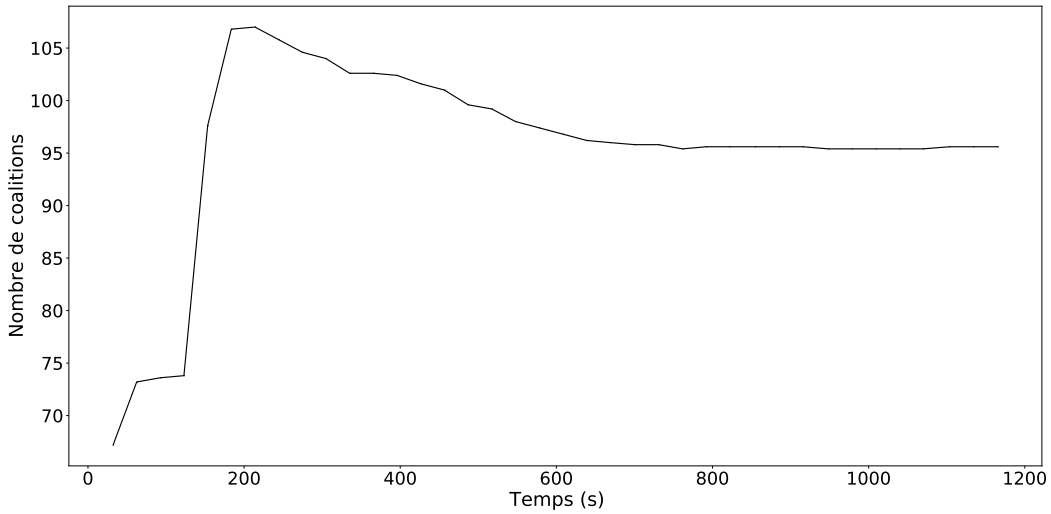
Cette évaluation montre la capacité du système multi-agent à se réorganiser lorsqu'un ou plusieurs agents du système se modifient au cours de son exécution. La réorganisation peut prendre un certain temps durant lequel les agents doivent demander le jeton à leur coalition pour ajouter un nouveau membre ou la quitter. Si les modifications à apporter à une coalition sont nombreuses comme dans le cas de ces systèmes où tous les agents ont été modifiées au même moment, tous les changement peuvent prendre un certain temps à se faire. Le temps est d'autant plus long que la coalition compte un grand nombre d'agents. Le temps pourrait être significativement écourté si les agents utilisaient un protocole leur permettant de modifier leur coalition de manière simultanée et rendre le protocole de demande de jeton caduc permettrait également la réduction du nombre de messages émis par le système.

7.4.7 L'explicabilité

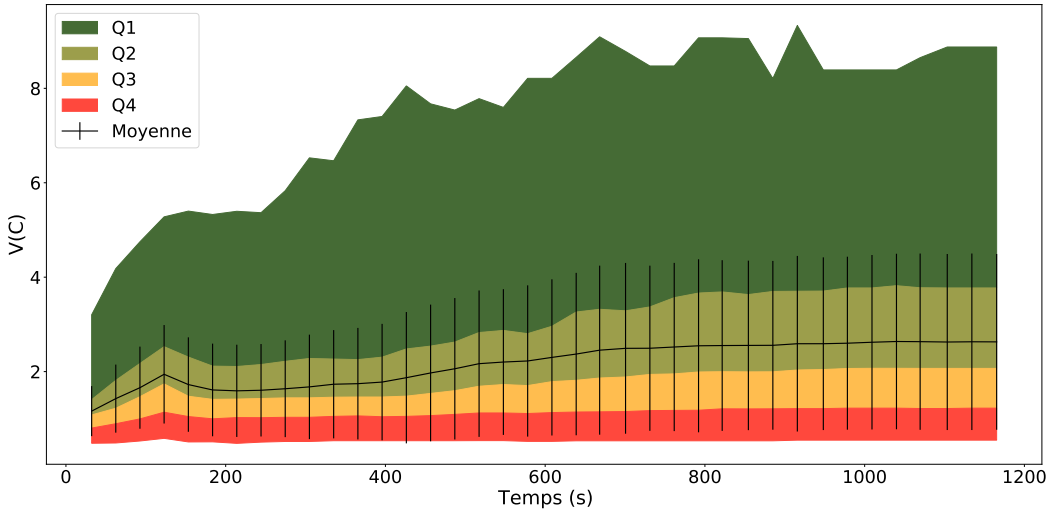
L'objectif de cette section n'est pas la formation de coalitions ni même de travailler sur un cas d'étude parfaitement réaliste, mais plutôt de montrer les capacités d'explicabilité offertes par les agents ABSG. Cette expérience est donc exécutée avec 18 agents pour faciliter la visualisation des coalitions et des agents. La figure 7.24 *(b)* présente une visualisation des coalitions dont un agent *Cuve* est membre. Les agents sont normalement identifiés par un numéro qui leur est propre au sein du système. Sur ces visualisations, nous ajoutons à ce numéro unique le type de l'agent en question pour faciliter la lecture et comprendre comment les coalitions sont compo-



(a) Nombre de messages émis dans le système en fonction du temps



(b) Nombre de coalitions formées durant l'exécution du système



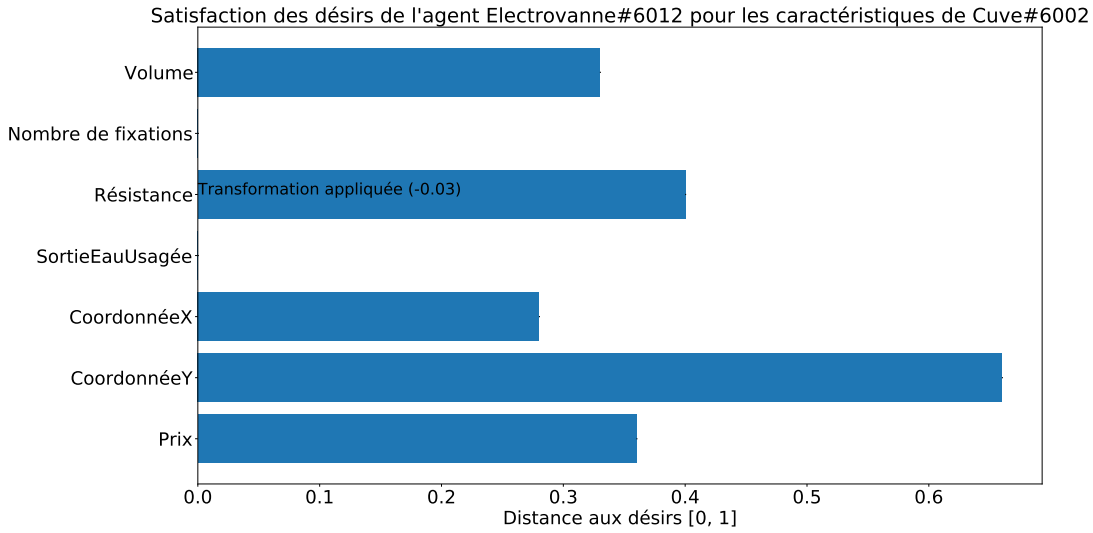
(c) Qualité des coalitions formées en fonction du temps

Figure 7.23 – Évaluation de la variabilité du système par la modification aléatoire des caractéristiques et désirs des agents à 120 secondes

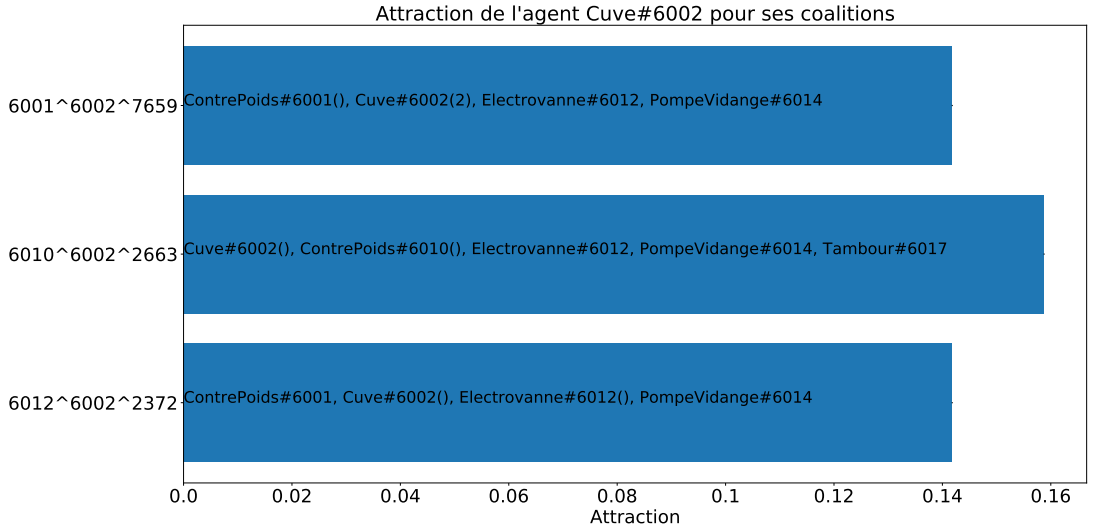
sées. Ainsi, cette première figure montre que nous nous intéressons à l'agent *6002* se trouvant être du type *Cuve* et qu'elle est membre de trois coalitions. Les coalitions se trouvent sur la gauche du graphique à barres, elles possèdent également un identifiant unique correspondant à l'association des deux numéros de ses créateurs (par exemple *6001* et *6002* dans la première coalition) suivis d'un nombre généré aléatoirement (*7659*). Les barres bleues représentent l'attraction que l'agent *Cuve#6002* a pour ses coalitions. Au centre de chaque barre bleue se trouvent les identifiants des membres de chaque coalition. Les transformations appliquées aux agents membres sont juxtaposées aux identifiants entre parenthèses. Pour ne pas surcharger la visualisation, les transformations sont représentées par un chiffre décrivant le numéro de la caractéristique de l'agent qui a été transformée. Par exemple, dans le premier groupe, l'agent *Cuve#6002* s'est vu appliquer une transformation sur sa deuxième caractéristique. Plus de détails peuvent être trouvés sur les transformations sur d'autres vues. Enfin, notons que certains numéros de membres sont adjacents à des parenthèses vides. Il s'agit d'agents en capacité de posséder des transformations mais qui n'ont pas eu besoin de les appliquer pour entrer dans la coalition. Au contraire, les identifiants d'agent non suivis de parenthèses ne peuvent être transformés (comme décrit dans le cas d'étude). Nous pouvons observer sur la figure 7.24 *(a)* la distance des caractéristiques de l'agent *Cuve#6002* aux désirs de l'agent *Electrovanne#6012*, tous deux présents dans la coalition *6001ˆ6002ˆ7659*. Le graphique permet de lire que le nombre de fixations et le diamètre de sortie des eaux usagées de la *Cuve#6002* correspondent aux attentes de l'agent *6012*, mais que ce composant se situe sur un site lointain (*coordonnéeY* et que la résistance n'était pas celle recherchée malgré la transformation appliquée. Cependant, l'ensemble de ces caractéristiques sont tout de même suffisantes pour l'intégration de l'*Électrovanne* dans la coalition.

Alors que la transformation était sur la figure précédente décrite très brièvement, cette figure présente le nom de la caractéristique transformée ainsi que l'ampleur de cette modification en noir sur la barre bleue associée. Ainsi, l'utilisateur peut plus facilement comprendre les raisons qui poussent un agent à en ajouter ou ignorer un autre dans le processus de formation de coalitions.

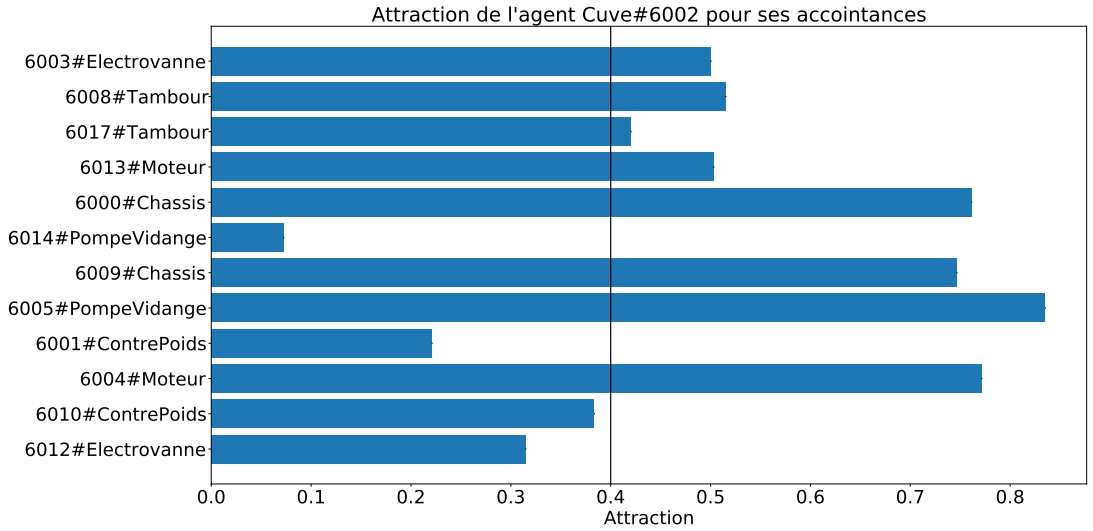
La première figure (7.24 *(b)*) montre que l'agent *6002* est membre de trois coalitions de qualités relativement proches. Parmi elles, deux coalitions (la première et la dernière) possèdent exactement les mêmes membres et les mêmes valeurs d'attraction. La seule différence réside en la transformation appliquée à l'agent *Cuve#6002*. Puisque l'agent que nous observons est celui qui a été transformé et puisque l'attraction est un agrégat de l'attraction qu'un agent a pour les autres membres, il est donc normal que l'attraction ressentie pour ses deux coalitions soit identique. Bien que le second groupe (*6010ˆ6002ˆ2663* soit plus complet que les autres d'un type de composant, il possède aussi une attraction plus élevée et donc de moins bonne qualité que les deux autres coalitions. La valeur d'attraction étant une moyenne de l'attraction pour les accointances du groupe, la présence d'une accointances de très mauvaise qualité peut dégrader la qualité globale de la coalition (du point de vue d'un agent uniquement) sans nécessairement remettre en question la coalition. Cet écart souligne la difficulté à évaluer ce qu'est une bonne coalition ainsi que la subjectivité apportée par le concepteur du système dans le choix de la méthode d'évaluation. Vaut-il mieux avoir une coalition complète dont les agents (et par extension les composants physiques du point de vue du système d'aide à la décision) ne s'assortissent pas convenablement, ou vaut-il mieux avoir une coalition incomplète dans laquelle tous les agents vont parfaitement bien ensemble ? La réponse est très probablement entre les mains de l'utilisateur du système en fonc-



(a) Vue de la satisfaction des désirs d'un agent à travers les caractéristiques de son accointance



(b) Vue de l'attraction qu'un agent a pour ses coalitions



(c) Vue de l'attraction qu'un agent a pour ses accointances

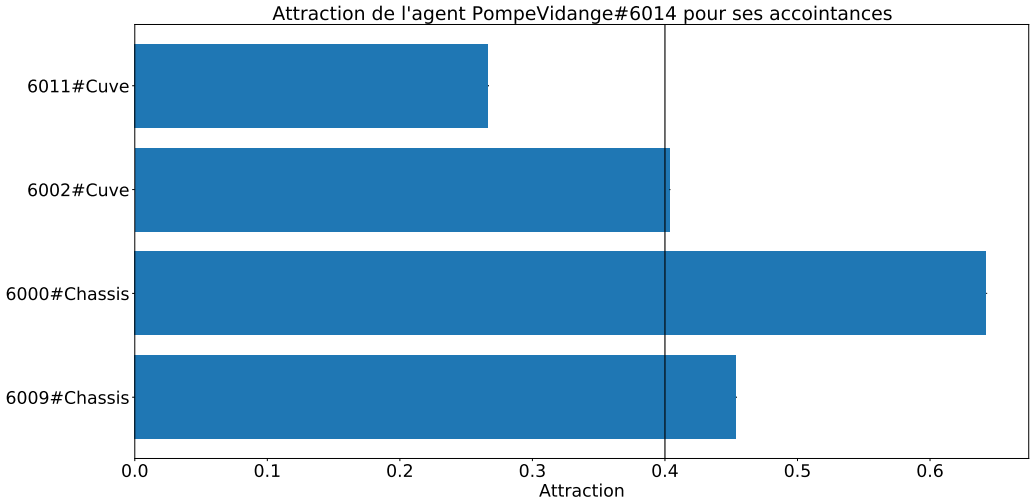
Figure 7.24 – Vues présentant les états internes des agents ABSG

tion de son objectif au moment de la conception de son produit. Tels que les agents ABSG ont été conçus dans ces expériences, nous avons essayé de faire un compromis entre les deux extrêmes ce qui peut expliquer la faible différence d'attraction entre les coalitions de différentes tailles.

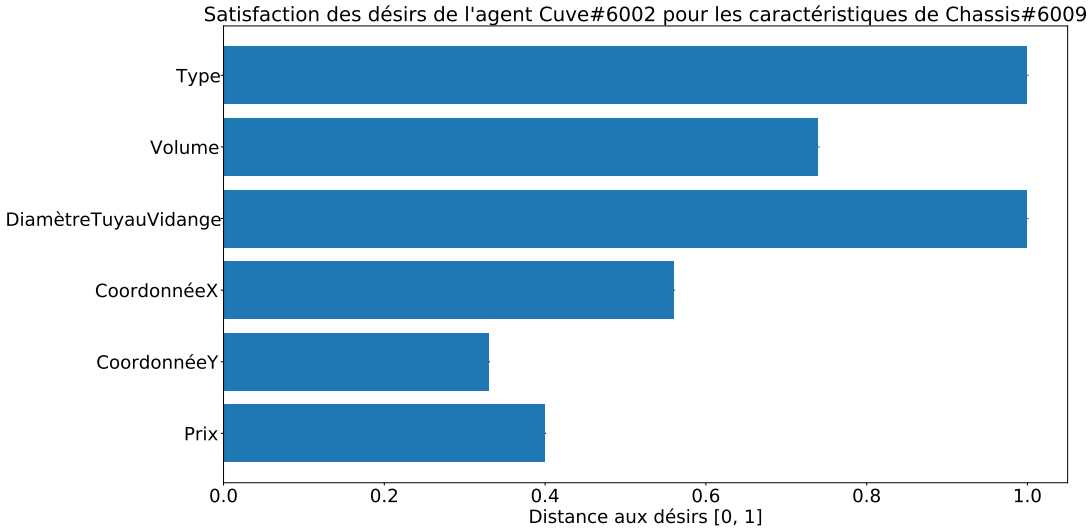
Bien que ce graphique permette de visualiser les coalitions d'un agent, il n'explique pas pourquoi les coalitions sont telles qu'elles sont. Par exemple, pourquoi aucune de ces coalitions ne présente d'agent *Châssis*. Le *Châssis* est l'un des composants associé au *Châssis* avec la *PompeVidange*. Pour mieux comprendre l'absence de *Châssis*, nous allons nous concentrer sur l'agent que nous suivons (*Cuve#6002*) et allons observer les valeurs d'attraction qu'il a pour ses accointances. La figure 7.24 *(c)* présente ces valeurs d'attraction pour l'agent *6002*. Sur la gauche se trouvent les identifiants des accointances de l'agent *6002*, les barres bleues représentent la valeur d'attraction et la barre verticale noire à 0.40 représente le seuil de formation ou de demande d'affiliation à une coalition pour cette expérience. Ainsi, toutes les accointances ayant une valeur d'attraction dépassant cette barre verticale noire ne peuvent pas être contactées pour être ajoutées à une coalition. Or nous pouvons voir que tous les agents *Châssis* possèdent des valeurs d'attraction très hautes ce qui explique leur absence dans les coalitions *6001^6002^7659* et *6012^6002^2372* de la figure précédente. Enfin, pour s'en assurer nous pouvons également observer les valeurs d'attraction de l'agent *PompeVidange#6014* pour ses accointances (figure 7.25). Nous pouvons y voir le même phénomène, aucune accointance *Châssis* n'est de suffisamment bonne qualité pour être ajoutée aux coalitions.

Prenant connaissance de ces informations, l'utilisateur du système peut : soit se résigner à acheter les composants de l'une de ces coalitions en décidant d'acheter un châssis neuf correspondant à son besoin, soit essayer de mieux comprendre pourquoi les châssis du système ne sont pas sélectionnés par le système d'aide à la décision. Les figures 7.25 *(b)* et 7.25 *(c)* présentent les distances entre chaque caractéristique des agents châssis et les désirs de l'agent *Cuve#6002*. Ces graphiques montrent que les distances entre les désirs de l'agent *6002* et les agents *Châssis* sont en moyenne supérieures au seuil de formation et affiliation à un groupe, fixé à environ *0.40* dans cette expérience. Or l'un des facteurs principaux dans le calcul de l'attraction étant cette distance, il paraît normal que les châssis ne semblent pas intéressants pour l'agent *Cuve*. La figure 7.25 *(b)* montre que le type du châssis et le diamètre du tuyau de sortie ne sont pas jugés intéressants. Cette distance élevée aux désirs de *6002* explique pourquoi les cuves ne sont pas intégrées au groupe.

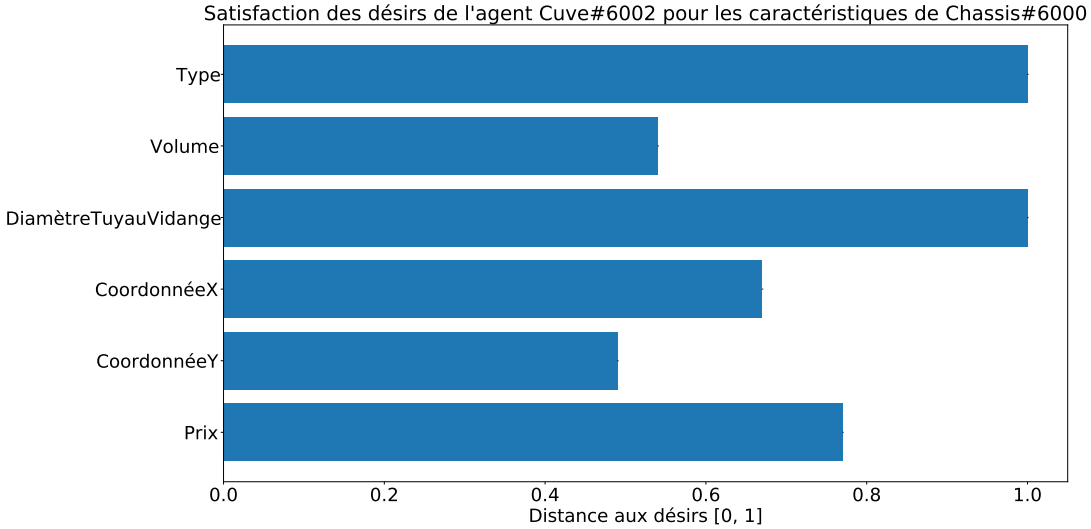
Ces dernières visualisations permettent de comprendre quelles caractéristiques permettent ou non aux agents d'entrer dans des coalitions. Cependant, l'adéquation des désirs à des caractéristiques est décrite sous forme numérique. Cette forme permet en effet de voir ce qui va ou ne va pas chez l'accointance d'un agent. Par exemple dans la figure 7.25 *(b)*, on voit que le type du chassis n'est pas celui attendu. Si l'utilisateur avait demandé un chassis *top*, on comprendrait alors que le chassis en question présenté sur le graphique est un chassis *front* car seuls deux types coexistent. La déduction est cependant impossible lorsque l'on aborde des types possédant un intervalle de valeur plus large comme le diamètre du tuyau de vidange. Nous pouvons constater que le diamètre n'est pas le bon, mais pas savoir exactement lequel est utilisé par l'agent *Chassis#6009*. Or, cette connaissance serait nécessaire à l'utilisateur pour réadapter sa demande. Les caractéristiques des agents étant uniquement représentées sous forme numérique dans leurs connaissances, elle ne peut être affichée dans le cadre de ce travail où nous n'avons implémenté que le système multi-agent. Cependant, cette information pourrait être obtenue grâce au système d'information qui contient



(a) Vue de l'attraction qu'un agent a pour ses accointances



(b) Vue de la satisfaction des désirs d'un agent à travers les caractéristiques de son accointance



(c) Vue de la satisfaction des désirs d'un agent à travers les caractéristiques de son accointance

Figure 7.25 – Vues présentant les états internes des agents ABSG

toutes les connaissances fournies par les acteurs du système d'aide à la décision. Une implémentation complète du système d'aide à la décision permettrait donc une amélioration de la qualité de ces visualisations et de l'explicabilité des choix du système.

7.4.8 Passage à l'échelle

Nous tentons d'évaluer dans cette section la capacité du système à s'adapter à un grand nombre de composants afin de le tester dans des conditions réalistes pour une application industrielle. Nous pouvons grossièrement estimer qu'une petite entreprise pratiquant le remanufacturing garde en stock une centaine de composants de chaque type de pièce pour un même appareil électroménager. Dans le cas d'étude des lave-linge par exemple, nous pouvons estimer que les pièces encombrantes telles que les châssis et les cuves soient présentes en nombre moins important mais ce sera contre-balancé par le grand nombre de petites pièces telles que les programmateurs, les électrovannes ou les pompes de vidange. Nous supposons donc que le nombre de pièces total pour ce type d'appareil est d'environ 900 composants. Il n'est cependant pas possible d'exécuter un si grand nombre d'agents dans les conditions expérimentales qui sont les nôtres, d'une part parce qu'il serait extrêmement long de chercher la coalition optimale avec un algorithme brute force avec autant d'agents, mais également parce que les capacités de calcul de l'ordinateur utilisé ne sont pas suffisantes pour un tel passage à l'échelle. Pour ces deux raisons, nous limitons dans cette expérience le nombre d'agents à 198, soit 22 agents de chaque type. Pour ces même raisons l'expérience n'est exécutée qu'une fois et limitée à 1300 secondes de temps d'exécution. Notre objectif ici est de montrer que, malgré les limites imposées par les conditions matérielles, un passage à l'échelle est envisageable. Afin de limiter au mieux le coût computationnel de cette expérience, nous avons paramétré le seuil de formation de coalition à *2.5* permettant aux agents de limiter le nombre de coalitions générées.

Les figures 7.26 présentent les résultats de cette expérience. La figure *(a)* montre le nombre de messages émis par le système durant son exécution, la figure *(b)* le nombre de groupes formés et la figure *(c)* une répartition de la qualité de ces coalitions par rapport à l'optimal. Au début de l'expérience, les agents se présentent et échangent leurs caractéristiques ce qui forme un pic de messages. Dans les expériences précédentes le pic était un évènement presque ponctuel dans le temps, mais le grand nombre d'agents présents dans ce système ralentit l'exécution de chaque individu ce qui étale le pic sur la première minute d'exécution. Les agents proposent ensuite à leurs meilleures accointances de former un groupe avec eux. Les coalitions se forment durant les 600 premières secondes d'expérience, pourtant le processus de formation n'est pas bloqué par un système de jeton comme l'est le processus d'affiliation. Cela est probablement dû au fait que les agents peinent à traiter les messages entrants et à y répondre rapidement. La qualité moyenne des coalitions augmente entre 500 et 600 secondes ce qui montre que celles-ci commencent à se développer à ce moment-là. Les coalitions formées continuent de s'agrandir jusqu'à la fin du temps imparti où les meilleures atteignent jusqu'à 35% de la valeur optimale. Cette expérience nous permet de voir que le système multi-agent est capable de fonctionner dans un contexte applicatif à plus large échelle mais que le grand nombre d'agents ralentit tout le système en raison de la charge computationnelle de chaque individu. En théorie, si l'on mettait de coté le besoin computationnel de notre système, l'un des facteurs les plus déterminants de la génération de coalitions devrait

être le nombre de composants du produit à concevoir puisque notre protocole de communication impose l'utilisation d'un jeton empêchant les agents d'ajouter plusieurs membres simultanément. Au contraire le nombre d'agents dans le système ne devrait pas être le facteur impactant le plus le temps d'exécution puisque les coalitions peuvent émerger simultanément dans le système sans besoin de synchronisation. Nous pouvons donc supposer que modifier le protocole de communication afin d'éviter l'utilisation d'un sémaphore faciliterait un passage à l'échelle.

7.4.9 Synthèse

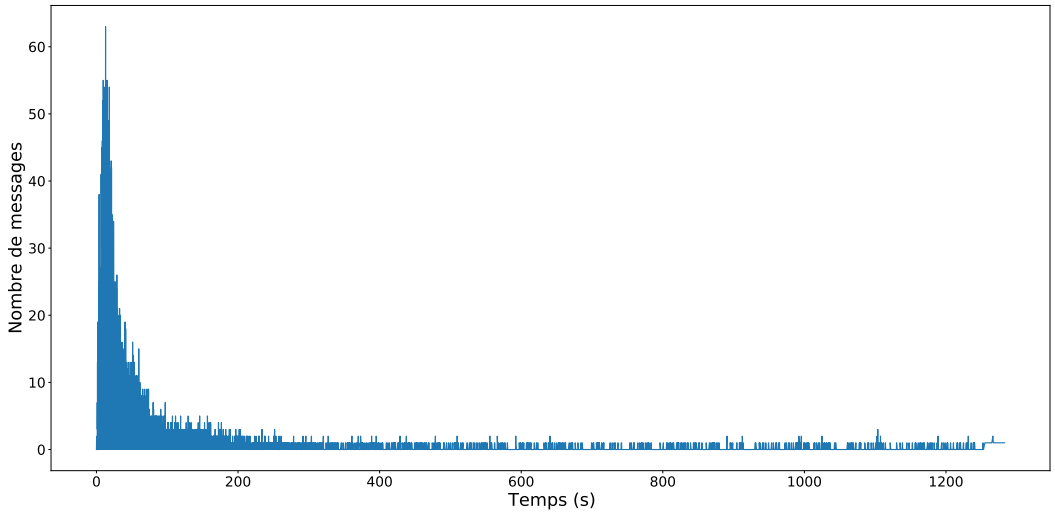
Cette partie a évalué les performances du système multi-agent et essayé de mesurer l'influence des paramètres de l'architecture ABSG sur le cas d'étude proposé dans le chapitre 6. Les principes de l'attraction que nous avons utilisés dans ces évaluations ne se sont pas tous révélés efficaces. Nous avons montré et justifié l'intérêt de l'attractivité physique, la complémentarité et le minimax mais la similarité s'est finalement révélée être sans effet sur la qualité et le temps d'exécution du système.

Nous avons présenté les capacités d'explicabilité du système multi-agent à travers des vues de l'état des agents permettant de mieux comprendre le processus de formation de coalitions. Ces vues permettent à l'utilisateur de voir les membres d'une coalition, les transformations appliquées aux agents, l'attraction que les agents ont pour leurs accointances et l'adéquation des caractéristiques des accointances aux désirs des agents. Grâce à elles un utilisateur pourrait être capable de négocier avec le système en adaptant les spécifications de son besoin à l'état des agents disponibles.

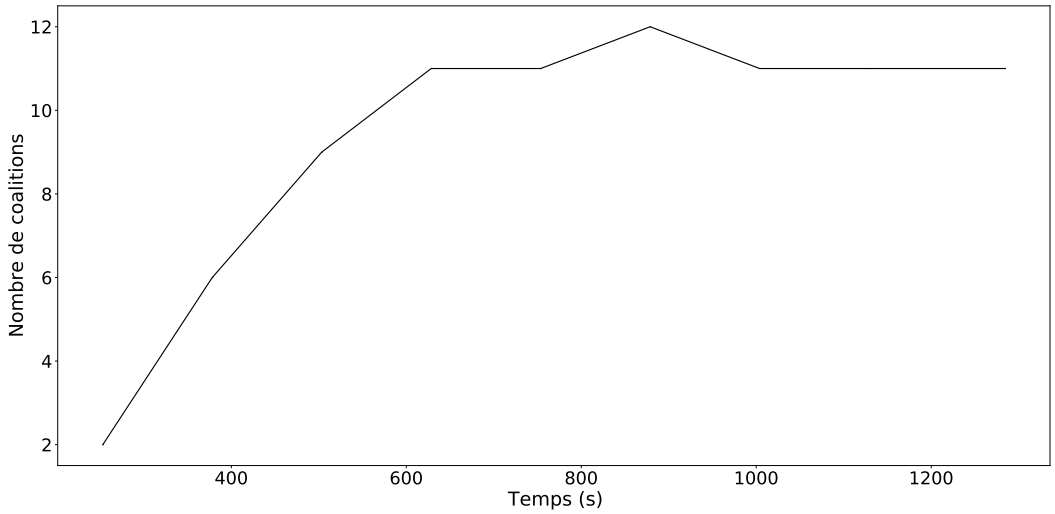
L'ouverture et la variabilité du système multi-agent ont également été évaluées et ont permis de constater que les agents sont capables de s'adapter à un environnement changeant durant l'exécution du système. Ils ont montré leur capacité à se réorganiser lorsque des agents entrent ou sortent du système, mais aussi lorsque leurs accointances changent de caractéristiques ou que leur objectif personnel change. Le système multi-agent se révèle donc être adapté à la dynamicité qu'impose le système d'aide à la décision.

7.5 Synthèse et discussion

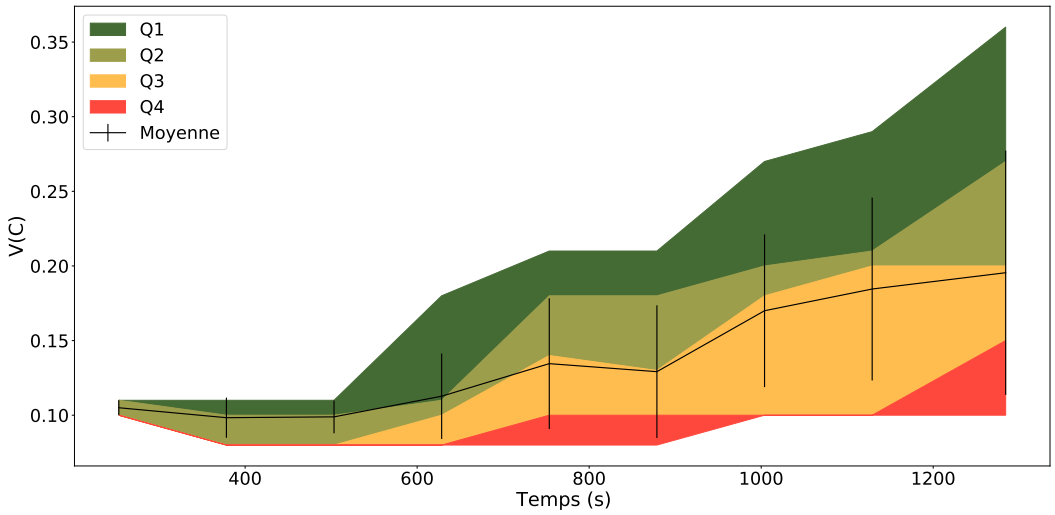
Ce chapitre a présenté les métriques utilisées pour mener à bien les évaluations : la qualité des coalitions formées, le temps d'exécution du système et le nombre de messages émis. Nous avons également présenté le paramétrage des agents, les détails de l'implémentation de l'architecture ABSG, les conditions expérimentales dans lesquelles sont exécutées les systèmes multi-agents et les évaluations associées à notre approche et à l'architecture. L'approche multi-agent a été comparée à d'autres méthodes de la littérature et l'architecture ABSG a été évaluée dans un cas d'étude générique et un cas d'étude plus réaliste construit à partir d'une collaboration avec des acteurs du remanufacturing. Les effets des divers paramètres de l'architecture ont été investigués et nous avons tenté d'expliquer et analyser les conséquences de la variation de ces paramètres. Nous avons



(a) Nombre de messages échangés durant l'exécution du système



(b) Nombre de coalitions formées au cours de l'expérience



(c) Qualité des coalitions formées dans le système au cours du temps

Figure 7.26 – Résultats non moyennés d'un système lancé avec 198 agents pendant 1300 secondes

pu observer que l'augmentation du nombre d'agents avait tendance à diminuer la qualité des résultats et augmenter le temps d'exécution du système dans les deux cas d'études testés. Nous avons également pu voir qu'augmenter le nombre de caractéristiques manipulées par les agents pouvait paradoxalement diminuer le temps d'exécution du système mais réduisait la qualité des coalitions. De la même manière l'ajout de contraintes montre les mêmes résultats en raison de la diminution de l'espace de recherche des agents. La mesure du nombre de messages émis dans le système montre sans surprise une évolution corrélée au nombre d'agents, l'augmentation étant d'autant plus importante que les messages d'introduction des agents sont pris en compte et que chacun d'entre eux a tendance à se présenter à toutes ses accointances à son arrivée dans le système. Nous avons également confirmé la capacité de notre approche et de notre système multi-agent à s'adapter à un contexte dynamique à travers l'ajout et la suppression d'agents mais aussi lors de leur modification à n'importe quel moment.

Parmi les paramètres les plus importants que nous avons pu déterminer dans ce chapitre à propos du contexte applicatif se trouvent le nombre de pièces qui composent le produit à concevoir ainsi la densité de la connectivité entre ces composants. Il n'est pas étonnant de constater que plus un produit contient de composants plus le temps d'exécution de notre système est long pour concevoir une coalition complète. La qualité des solutions est également fortement impactée par la connectivité des agents. Plus les pièces d'un produit sont connectées les unes aux autres et plus il est facile pour le système d'aide à la décision de former des coalitions complètes. En effet, les agents agissant de manière égoïste avec une connaissance limitée des autres agents, ils n'ajoutent à leur coalition que ceux dont ils ont besoin. Or si le système devait concevoir un assemblage de poupées russes et que l'une d'elle de taille *N* venait à manquer, la poupées de taille *N-1* ne pourrait pas former une coalition complète car elle n'aurait pas la volonté de contacter la poupée de taille *N+1*. Cette limite n'a pas été prise en compte dans le système d'aide à la décision car elle peut être évitée au moment de la modélisation du contexte applicatif, par exemple en fusionnant deux pièces qui ont une faible connectivité avec les autres pour ne les voir que comme un unique composant. C'est ce qui a involontairement été réalisé dans le cas d'étude *lave-linge* en fusionnant les pièces du système de roulement présentes entre le moteur et le tambour. Plutôt que d'avoir une chaîne d'éléments dont l'absence d'un chaînon pourrait entraver la formation d'une coalition complète, nous avons rassemblé ces éléments en un unique composant *Système de roulement*.

Les principes de l'attraction que nous avons utilisés dans ces évaluations ne se sont pas tous révélés efficaces. Nous avons montré et justifié l'intérêt de l'attractivité physique, la complémentarité et le minimax au cours des évaluations sur un contexte applicatif très général où le produit fictif est composé de 5 agents avec un nombre de caractéristiques et désirs fixes et initialisés aléatoirement, et un contexte applicatif plus réaliste dont le produit représentait un lave-linge composé de 9 pièces avec des caractéristiques et désirs propres à chacun des agents représentant les composants. La similarité s'est finalement révélée être sans effet sur la qualité et le temps d'exécution du système dans nos deux contextes applicatifs. Nous avons également mis en exergue l'importance de la pondération des principes de l'attraction dans la fonction d'attraction en fonction du contexte applicatif. Tandis qu'une faible connectivité des composants nécessite de surpondérer la complémentarité pour permettre la fabrication de coalitions complètes, une forte connectivité des pièces facilite la mise en relation des agents et leur regroupement ce qui rend alors plus intéressant de surpondérer l'attractivité physique pour accentuer l'importance de l'adéquation des désirs des

agents aux caractéristiques de leurs accointances.

Comme nous l'avons noté dans les parties précédentes, le protocole permettant l'affiliation des agents à une coalition peut ralentir le système lors du processus de formation de coalitions. En effet, en plus d'obliger les membres à échanger un jeton pour modifier leur coalition, il force les coalitions à se modifier étape par étape et empêche toute parallélisation de l'ajout d'agents. Les temps d'exécutions ne sont pas suffisamment élevés pour que ce mécanisme soit gênant sur des instances du système composées d'un faible nombre d'agents mais peut se révéler plus embarrassant pour un passage à l'échelle du système. Des recherches plus poussées pour remplacer ce protocole permettrait aux agents d'être plus robustes face à des modifications simultanées de leurs groupes.

Nous avons estimé qu'un système devrait prendre un compte environ 900 composants pour imiter les stocks d'une petite entreprise de remanufacturing sur un unique type de produit électroménager. Nos capacités matérielles ne permettant pas un tel passage à l'échelle, nous avons tout de même exécuté le système avec 200 agents. Bien que ce ne soit pas suffisant pour se dire réaliste, notre expérience laisse penser qu'un passage à l'échelle est envisageable avec une meilleure capacité de calcul. En outre, nous arguons que la représentation de lots de composants par un unique agent pourrait faciliter le passage à l'échelle. Nous estimions qu'une société pouvait stocker une centaine de composants de chaque type d'un appareil électroménager, mais tous les composants ne sont pas forcément différents une fois prêts au réemploi. Par exemple les châssis *top* possèdent tous les mêmes dimensions et la disposition des pièces est la même pour chaque machine utilisant ce châssis. Nous pourrions donc imaginer regrouper nos composants *châssis top* sous le même agent. En généralisant ce procédé aux autres pièces de l'appareil, le nombre d'agents à utiliser pourrait être considérablement réduit, au même titre que la charge computationnelle du système. Selon la manière dont les fournisseurs décrivent les caractéristiques de leurs pièces, celles-ci pourraient être regroupées avec plus ou moins de facilité. Notre cas d'étude a limité la réelle richesse des caractéristiques des composants utilisés. Il ne serait pas raisonnable de simplifier un moteur dans une application industrielle à un nombre de fils et un type de courroie. Mais même si le système multi-agent ne comportait que 10 agents *moteurs*, ce sont 10 catégories d'agents *moteur* utilisables dans lesquelles des centaines de pièces pourraient être répertoriées et manipulée comme étant un unique agent par le système multi-agent.

8 Conclusion

Sommaire

8.1 La problématique

La conception d'un système d'aide à la décision pour l'économie circulaire nous a amené à nous pencher sur la question de la formation de coalitions. Celui-ci doit concevoir des produits à partir d'un ensemble de composants variables en nombre mais aussi en caractéristiques et en fonction de la demande d'un utilisateur ; par exemple concevoir un produit électroménager en prenant en compte les besoins spécifiques d'une personne (chassis top, capacité 5kg, *etc.*). Le système doit être capable de faire émerger des groupes de composants représentant les produits qui répondent le mieux au besoin de l'utilisateur. Nous avons rapproché ce problème de celui de la génération de structures de coalitions (CSG) où chaque coalition est une conception de produit comblant plus ou moins les désirs de l'utilisateur. Le système d'aide à la décision peut ajouter ou rendre indisponibles des composants à n'importe quel moment en fonction des stocks des fournisseurs de pièces, notre problème de génération de structures de coalitions est donc un problème variable et ouvert dans lequel les agents que nous traitons peuvent entrer ou sortir à tout moment. De plus, les composants possèdent des caractéristiques altérables dans le temps comme leur état physique (*e.g.* qualité d'un composant électronique qui peut se dégrader avec le temps) ou leur état logistique (*e.g.* prix, endroit de stockage, *etc.*) ce qui rend notre problème de CSG variable en plus d'être ouvert. Les conceptions de produits n'étant pas toutes vouées à être choisies par l'opérateur humain, leurs composants peuvent appartenir à plusieurs groupes afin de tenter de faire émerger des coalitions répondant au besoin utilisateur ce qui ajoute également à notre problème de CSG la spécificité d'avoir des coalitions chevauchantes.

Enfin, en plus d'essayer de répondre à un problème de CSG ouvert, variable avec des coalitions chevauchantes, nous souhaitons que l'utilisateur puisse comprendre les conceptions de produit que le système lui propose ce qui nécessite une méthode de résolution du problème adaptée à l'explicabilité des décisions prises par le système.

8.2 Contribution à l'approche multi-agents

Ce problème de CSG fait intervenir de multiples contraintes issues d'un contexte applicatif réaliste. Celles-ci rendent difficile l'utilisation de nombreuses approches traditionnellement utilisées pour résoudre ce type de problèmes. Nous avons proposé l'utilisation d'une approche multi-agent inspirée par les dynamiques de groupes. Cette approche possède les propriétés requises pour concevoir notre système d'aide à la décision, à savoir la capacité à s'adapter à un système ouvert et variable, la possibilité de travailler avec des coalitions chevauchantes et la possibilité de concevoir une méthode permettant l'explicabilité des solutions du système. En outre l'inspiration des dynamiques de groupes nous permet de proposer une méthode pertinente et innovante fondée sur l'étude de l'humain pour la formation de groupes.

Les évaluations ont montré que notre approche a bien permis la conception d'un système d'aide à la décision ouvert et variable à travers la l'élaboration d'une architecture d'agent rendant le SMA capable de s'adapter à un environnement changeant. Les agents ont été capables de

s'adapter à l'entrée et la sortie d'agents durant l'exécution du système et ont pu se réorganiser en conséquence. De la même manière, la variabilité a été évaluée et les agents ont su restructurer leurs coalitions lorsque leurs accointances ont modifié leurs caractéristiques. L'architecture d'agent étant fondée sur une représentation des connaissances symbolique, celle-ci a également rendu possible l'explicabilité des choix du système afin d'aider l'utilisateur dans la conception de produits. Les performances du système multi-agent ont été évaluées sur un cas d'étude réaliste pour la conception d'appareils électroménagers construit avec la collaboration d'acteurs industriels et les évaluations ont montré des performances intéressantes par rapport à la littérature, notamment dans la qualité des solutions proposées.

8.3 Vers une application industrielle

Ce manuscrit a présenté une architecture d'agent capable de fonctionner dans un outil d'aide à la décision ouvert et variable pour la conception de produits à partir de composants usagés. Outre la contribution au domaine multi-agent, ce travail est fortement ancré dans un contexte industriel. Il ne souhaite pas seulement proposer une nouvelle approche de résolution d'un problème de génération de structures de coalitions, mais aussi présenter la façon dont on peut l'utiliser pour s'attaquer à des enjeux écologiques et économiques à travers un système d'aide à la décision. C'est pour cette raison qu'il nous semblerait important d'approfondir la recherche sur le passage à l'échelle d'un tel système afin de l'intégrer au mieux dans un cadre industriel. Bien que les premiers résultats montrent la possibilité d'utiliser ce système dans un contexte plus réaliste avec de nombreux agents et bien que nous proposions des réponses au frein que représente la charge computationnelle, il est encore nécessaire d'investiguer sur certains mécanismes du système tels que les protocoles d'interaction avant de penser à une implémentation pour une application industrielle. Le système d'aide à la décision n'ayant pas besoin de temps réel, son temps d'exécution ne représente pas une résistance majeure à son utilisation dans l'industrie. En outre, l'un des avantages apporté par la conception à l'aide du paradigme multi-agent est la capacité de notre système à se distribuer sur plusieurs machines ce qui réduit considérablement la problématique de son coût computationnel. Une perspective intéressante à ce travail serait l'évaluation du fonctionnement de notre système multi-agent lorsqu'il est distribué sur plusieurs machines de façon à mieux appréhender ses performances et à l'adapter à un contexte industriel.

Nous avons exposé la possibilité pour le système d'expliquer ses choix à travers une interface présentant les états internes des agents ABSG. L'interface actuelle peut paraître sommaire et peu évidente à comprendre pour une personne qui n'est pas familière avec le fonctionnement des agents ABSG. Dans un contexte industriel, l'interface devrait être accessible aux clients d'une entreprise et ne pourrait pas présenter de détails techniques comme les identifiants des groupes ou des agents. Le concept d'attraction pourrait également être caché ou adapté de façon à ce qu'il soit intuitivement compréhensible et n'entrave pas l'utilisation des capacités d'explicabilité du système.

Enfin, la législation interférant avec les capacités des acteurs industriels à transformer des composants pour leur réintégration dans un nouveau type ou modèle d'appareil, nous n'avons pas particulièrement développé cette facette du métier dans les mécanismes comportementaux de nos agents. Mais devant le développement de l'économie circulaire, il est fort probable que les lois se développent et laissent plus de place aux entreprises dans leur façon de réutiliser des pièces usagées. Ce type de mécanisme devra alors faire l'objet d'études plus approfondies en collaboration directe avec les acteurs industriels afin d'adapter le comportement du système à ces nouvelles possibilités.

8.4 Vers un système multi-niveau

L'aspect qui nous paraît aujourd'hui être le plus intéressant en vue du développement du système multi-agent et de son intégration dans une application industrielle est la conception d'une extension du système actuel en un système d'aide à la décision multi-niveau dans lequel les utilisateurs et les agents peuvent travailler et interagir sur plusieurs niveaux d'abstraction [Occello2019]. Un tel système fonctionnerait avec les mêmes mécanismes que ceux que nous avons présentés dans ce travail. Celui-ci pourrait adapter son fonctionnement au niveau d'expertise de ses utilisateurs tout en leur permettant de travailler à plusieurs au même instant sur la même instance du système. Nous présentons succinctement l'aspect multi-niveau et son intérêt dans les sections suivantes.

8.4.1 Une description multi-niveau des composants

Ce manuscrit a présenté un outil d'aide à la décision dans lequel les utilisateurs ne peuvent décrire leurs besoins qu'en terme de caractéristiques physiques. Les composants et par extension les agents sont de la même manière décrits par ces mêmes caractéristiques ce qui permet leur regroupement en les comparant au besoin utilisateur. Toutefois, un néophyte qui pour la première fois se penche sur la conception d'un lave-linge n'a que très peu de chances de correctement définir son besoin. L'une des facettes du multi-niveau pourrait être la création de plusieurs couches d'abstraction de la description des composants. Nous trouverions par exemple la couche classique que nous avons utilisé tout au long de ce travail, à savoir les caractéristiques brutes des composants. Une autre couche pourrait être plus abstraite et décrire la fonction des composants et enfin une troisième couche pourrait décrire la façon dont ils s'intègrent dans l'appareil final. De tels niveaux d'abstraction sur les composants ont comme intérêt d'une part de permettre à tout type d'utilisateur de comprendre le rôle des pièces utilisées et de le guider dans la spécification de son besoin et d'autre part d'utiliser les informations fonctionnelles des composants manipulés par l'outil d'aide à la décision. Un moteur a par exemple pour rôle de transmettre un mouvement de rotation à un objet. Décrire les moteurs d'appareils électroménagers comme tels pourrait, par un traitement de haut niveau des connaissances expertes par les agents, leur permettre de réutiliser des moteurs pour une nouvelle application pour laquelle ils n'étaient pas originellement conçus.

8.4.2 La description multi-niveau du besoin utilisateur

Grâce à une description de haut niveau des composants, les utilisateurs n'ont plus à savoir comment un appareil doit être assemblé pour spécifier leur besoin. L'objectif de la description multi-niveau du besoin utilisateur est de leur permettre de décrire leur désir sur le niveau d'abstraction qui leur convient le mieux. À savoir entrer dans les détails des caractéristiques de chaque composant pour les utilisateurs experts, décrire brièvement les types des composants souhaités pour les utilisateurs intermédiaires ou simplement un besoin fonctionnel pour les novices. Un besoin fonctionnel représente la description des fonctionnalités requises par l'utilisateur, par exemple : *"un lave-linge écologique pour un foyer de 3 personnes"*.

8.4.3 Une aide à la décision multi-niveau

Enfin, l'aide à la décision multi-niveau représente les couches composant le système multi-agent lui même pendant le processus d'émergence de coalitions. Tel que présenté dans cette étude, nous pouvons visualiser le système multi-agent comme une unique couche sur laquelle des agents ABSG représentant divers composants interagissent et se regroupent. Cette couche ne tient pas compte du type des agents qui travaillent dessus. Un fournisseur de pièces de vélo pourrait décider d'intégrer ses pièces détachées alors même que les autres fournisseurs proposent des composants de lave-linge. Le premier intérêt de l'ajout de couches est d'éviter les interactions entre les agents qui n'ont aucun lien entre eux en triant les composants qui ont un bénéfice à travailler ensemble sur les différentes couches afin de réduire la charge computationnelle. En outre, le système que nous avons présenté travaille avec un unique utilisateur. Plusieurs d'entre eux avec des besoins différents ne pourraient travailler sur la même couche sans influencer les conceptions de produit formées pour l'autre. L'intérêt du multi-niveau est donc également de paralléliser les tâches d'aide à la décision pour chaque personne. La figure 8.1 schématise cet aspect multi-niveau de l'aide à la décision en présentant des niveaux repartis sur deux dimensions : verticalement les systèmes qui fonctionnent avec des types de composants spécifiques (application aux lave-linge, théières ou fours) et horizontalement les systèmes qui fonctionnent avec des utilisateurs différents. Chaque couche représente une instance d'un système multi-agents fonctionnant avec un utilisateur et une catégorie de composants spécifiques. Les points de couleurs représentent les agents et chaque couleur est un rôle définissant le type d'application dans laquelle ils peuvent s'intégrer. Ainsi, le système d'aide à la décision fonctionne simultanément avec une multitude d'utilisateurs qui ne peuvent se gêner les uns les autres car ils travaillent sur des niveaux différents. Toutefois, les agents présents sur chaque niveau vertical possèdent les mêmes agents car leur rôle est de concevoir le même type de produit. Si l'un des clients achète des composants à la suite d'une proposition du système, les composants disparaissent sur les niveaux des autres utilisateurs.

Un outil d'aide à la décision multi-niveau apporterait davantage de liberté et de facilité d'utilisation à ses usagers. En plus de leur offrir la possibilité de travailler en simultané sur diverses demandes, il augmenterait la capacité du système d'aide à la décision à réutiliser des composants usagés selon la stratégie de réutilisation nommée *repurposing*, à savoir l'intégration de pièces réutilisables dans une application pour laquelle elles n'étaient à l'origine pas conçues. Plus que d'apporter du confort aux utilisateurs et plus que d'améliorer l'intégration de l'outil d'aide à la décision à l'industrie, ce système s'intègrerait d'avantage au modèle économique de l'économie

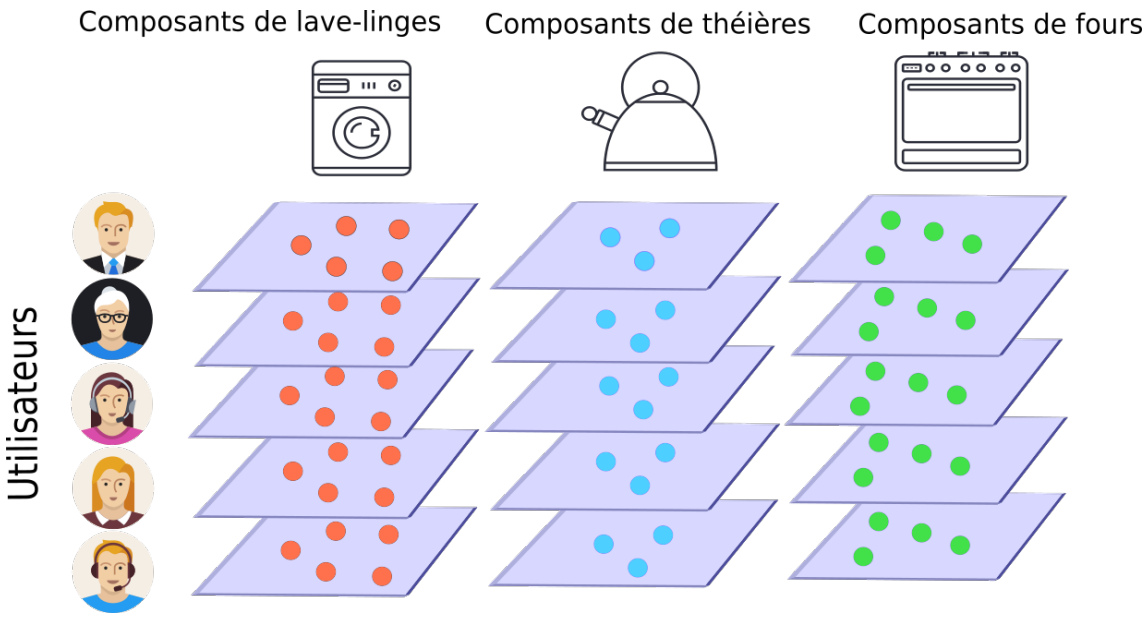


Figure 8.1 – Schéma du système d'aide à la décision multi-niveaux

circulaire et serait plus adapté aux enjeux écologiques des systèmes industriels du futur.

Bibliographie

[Agah1997] Arvin Agah et George A Bekey. *Cognitive architecture for robust adaptive control of robots in a team*. Journal of Intelligent and Robotic Systems, vol. 20, no. 2, pages 251–273, 1997.

[Aknine2004] Samir Aknine, Suzanne Pinson et Melvin F Shakun. *A multi-agent coalition formation method based on preference models*. Group Decision and Negotiation, vol. 13, no. 6, pages 513–538, 2004.

[Albus2005] James S Albus et Anthony J Barbera. *RCS : A cognitive architecture for intelligent multi-agent systems*. Annual Reviews in Control, vol. 29, no. 1, pages 87–99, 2005.

[Anderson1997] John R Anderson, Michael Matessa et Christian Lebiere. *ACT-R : A theory of higher level cognition and its relation to visual attention*. Human–Computer Interaction, vol. 12, no. 4, pages 439–462, 1997.

[Anshelevich2021] Elliot Anshelevich et Wennan Zhu. *Ordinal approximation for social choice, matching, and facility location problems given candidate positions*. ACM Transactions on Economics and Computation (TEAC), vol. 9, no. 2, pages 1–24, 2021.

[Back1951] Kurt W Back. *Influence through social communication.* The Journal of Abnormal and Social Psychology, vol. 46, no. 1, page 9, 1951.

[Baldoni2018] Matteo Baldoni, Cristina Baroglio, Olivier Boissier, Jomi F Hübner et Roberto Micalizio. *Norm-aware and Norm-oriented Programming*. 2018.

[Bennis2013] Mehdi Bennis, Meryem Simsek, Andreas Czylwik, Walid Saad, Stefan Valentin et Merouane Debbah. *When cellular meets WiFi in wireless small cell networks*. IEEE communications magazine, vol. 51, no. 6, pages 44–50, 2013.

[Bettinelli2020a] Mickael Bettinelli, Damien Genthial et Michel Occello. *Cohesion as a Tool for Maintaining the Functional Integrity of a Multi-agent System.* In ICAART (2), pages 445–452, 2020.

[Bettinelli2020b] Mickael Bettinelli, Occello Michel et Genthial Damien. *Coalition Formation Problem : a Group Dynamics Inspired Swarming Method*. HAL preprint hal-02903531, 2020.

[Bettinelli2020c] Mickaël Bettinelli, Michel Occello, Damien Genthial et Daniel Brissaud. *A decision support framework for remanufacturing of highly*

variable products using a collective intelligence approach. Procedia CIRP, vol. 90, pages 594–599, 2020.

[Bicchieri2005] Cristina Bicchieri. The grammar of society : The nature and dynamics of social norms. Cambridge University Press, 2005.

[Bistaffa2017] Filippo Bistaffa, Alessandro Farinelli, Jesús Cerquides, Juan Rodríguez-Aguilar et Sarvapali D Ramchurn. *Algorithms for graph-constrained coalition formation in the real world*. ACM Transactions on Intelligent Systems and Technology (TIST), vol. 8, no. 4, pages 1–24, 2017.

[Bollen1990] Kenneth A Bollen et Rick H Hoyle. *Perceived cohesion : A conceptual and empirical examination*. Social forces, vol. 69, no. 2, pages 479–504, 1990.

[Bornstein1989] Robert F Bornstein. *Exposure and affect : overview and meta-analysis of research, 1968–1987*. Psychological bulletin, vol. 106, no. 2, page 265, 1989.

[Brandt2016] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang et Ariel D Procaccia. Handbook of computational social choice. Cambridge University Press, 2016.

[Busoniu2006] Lucian Busoniu, Robert Babuska et Bart De Schutter. *Multi-agent reinforcement learning : A survey*. In 2006 9th International Conference on Control, Automation, Robotics and Vision, pages 1–6. IEEE, 2006.

[Buton2006] Fabrice Buton, Paul Fontayne et Jean-Philippe Heuzé. *La cohésion des groupes sportifs : évolutions conceptuelles, mesures et relations avec la performance*. Movement Sport Sciences, no. 3, pages 9–45, 2006.

[Carron1985] Albert V Carron. *The Development of an Instrument to Assess Cohesion in Sport Teams : The Group Environment Questionnaire*. Journal of Sport Psychology, vol. 7, pages 244–266, 1985.

[Carron2003] Albert V Carron, Heather A Hausenblas et Paul A Estabrooks. The psychology of physical activity, volume 1. McGraw-Hill Companies, 2003.

[Cartwright1956] Dorwin Cartwright et Frank Harary. *Structural balance : a generalization of Heider's theory.* Psychological review, vol. 63, no. 5, page 277, 1956.

[Casey-Campbell2009] Milly Casey-Campbell et Martin L Martens. *Sticking it all together : A critical assessment of the group cohesion–performance literature*. International Journal of Management Reviews, vol. 11, no. 2, pages 223–246, 2009.

[Changder2020] Narayan Changder, Samir Aknine, Sarvapali D Ramchurn et Animesh Dutta. *ODSS : Efficient Hybridization for Optimal Coalition Structure Generation.* In AAAI, pages 7079–7086, 2020.

[Chikhaoui2009] Belkacem Chikhaoui, Hélene Pigot, Mathieu Beaudoin, Guillaume Pratte, Philippe Bellefeuille et Fernando Laudares. *Learning a song : an ACT-R model*. In Proceedings of the international conference on computational intelligence, pages 405–410. Citeseer, 2009.

[Chung2019] Ching-jung Chung, Gwo-jen Hwang et Chiu-lin Lai. *A review of experimental mobile learning research in 2010–2016 based on the activity theory framework*. Computers & education, vol. 129, pages 1–13, 2019.

[Colorni1991] Alberto Colorni, Marco Dorigo, Vittorio Maniezzo*et al. Distributed optimization by ant colonies*. In Proceedings of the first European conference on artificial life, volume 142, pages 134–142. Paris, France, 1991.

[Conte2014] Rosaria Conte, Giulia Andrighetto et Marco Campennl. Minding norms : Mechanisms and dynamics of social order in agent societies. Oxford University Press, 2014.

[Delorme2020] Maxence Delorme, Sergio García, Jacek Gondzio, Joerg Kalcsics, David Manlove et William Pettersson. *Stability in the hospitals/residents problem with couples and ties : Mathematical models and computational studies*. Omega, page 102386, 2020.

[Di Marzo Serugendo2005] Giovanna Di Marzo Serugendo, Marie-Pierre Gleizes et Anthony Karageorgos. *Self-organization in multi-agent systems*. Knowledge Engineering Review, vol. 20, no. 2, pages 165–189, 2005.

[Dion2000] Kenneth L Dion. *Group cohesion : From" field of forces" to multidimensional construct.* Group Dynamics : Theory, research, and practice, vol. 4, no. 1, page 7, 2000.

[Duan2012] Junhua Duan, Yi-an Zhu et Shujuan Huang. *Stigmergy agent and swarm-intelligence-based multi-agent system*. In Proceedings of the 10th World Congress on Intelligent Control and Automation, pages 720–724, 2012.

[Ducheneaut2006] Nicolas Ducheneaut, Nicholas Yee, Eric Nickell et Robert J Moore. *" Alone together ?" Exploring the social dynamics of massively multiplayer online games*. In Proceedings of the SIGCHI conference on Human Factors in computing systems, pages 407–416, 2006.

[Ebbesen1976] Ebbe B Ebbesen, Glenn L Kjos et Vladimir J Konečni. *Spatial ecology : Its effects on the choice of friends and enemies*. Journal of Experimental Social Psychology, vol. 12, no. 6, pages 505–518, 1976.

[Faghihi2011] Usef Faghihi, Philippe Fournier-Viger et Roger Nkambou. *Implementing an efficient causal learning mechanism in a cognitive tuto-*

ring agent. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pages 27–36. Springer, 2011.

[Falcone2009] Rino Falcone et Cristiano Castelfranchi. *Socio-cognitive model of trust*. In Human Computer Interaction : Concepts, Methodologies, Tools, and Applications, pages 2316–2323. IGI Global, 2009.

[Farinelli2017] Alessandro Farinelli, Manuele Bicego, Filippo Bistaffa et Sarvapali D Ramchurn. *A hierarchical clustering approach to large-scale near-optimal coalition formation with quality guarantees*. Engineering Applications of Artificial Intelligence, vol. 59, pages 170–185, 2017.

[Ferber1997] Jacques Ferber. *Les systèmes multi-agents : un aperçu général*. Techniques et sciences informatiques, vol. 16, no. 8, 1997.

[Ferguson1992] Innes A Ferguson. *TouringMachines : An architecture for dynamic, rational, mobile agents*. Rapport technique, University of Cambridge, Computer Laboratory, 1992.

[Festinger1950] Leon Festinger. *Informal social communication.* Psychological review, vol. 57, no. 5, page 271, 1950.

[Finin1994] Tim Finin, Richard Fritzson, Don McKay et Robin McEntire. *KQML as an agent communication language*. In Proceedings of the third international conference on Information and knowledge management, pages 456–463, 1994.

[Forsyth2010] Donelson R Forsyth. Group dynamics. Cengage Learning, 2010.

[Franklin2006] Stan Franklin et FG Patterson Jr. *The LIDA architecture : Adding new modes of learning to an intelligent, autonomous, software agent*. pat, vol. 703, pages 764–1004, 2006.

[Fulantelli2015] Giovanni Fulantelli, Davide Taibi et Marco Arrigo. *A framework to support educational decision making in mobile learning*. Computers in human behavior, vol. 47, pages 50–59, 2015.

[Gale1962] David Gale et Lloyd S Shapley. *College admissions and the stability of marriage*. The American Mathematical Monthly, vol. 69, no. 1, pages 9–15, 1962.

[Gan2017] Kim Soon Gan, Kim On Chin, Patricia Anthony et Abdul Razak Hamdan. *A FIPA-ACL Ontology in enhancing interoperability multi-agent communication*. In International Conference on Computational Science and Technology, pages 151–160. Springer, 2017.

[Gazi2005] Veysel Gazi. *Swarm aggregations using artificial potentials and sliding-mode control*. IEEE Transactions on Robotics, vol. 21, no. 6, pages 1208–1214, 2005.

[Gemrot2010] Jakub Gemrot, Cyril Brom, Rudolf Kadlec, Michal Bída, Ondřej Burkert, Michal Zemčák, Radek Píbil et Tomáš Plch. *Pogamut 3–Virtual humans made simple*. Advances in Cognitive Science, pages 211–243, 2010.

[Gieryn2000] Thomas F Gieryn. *A space for place in sociology*. Annual review of sociology, vol. 26, no. 1, pages 463–496, 2000.

[Goertzel2007] Ben Goertzel et Cassio Pennachin. *The Novamente artificial intelligence engine*. In Artificial general intelligence, pages 63–129. Springer, 2007.

[Goertzel2013] Ben Goertzel, Shujing Ke, Ruiting Lian, Jade O'Neill, Keyvan Sadeghi, Dingjie Wang, Oliver Watkins et Gino Yu. *The cogprime architecture for embodied artificial general intelligence*. In 2013 IEEE symposium on computational intelligence for human-like intelligence (CIHLI), pages 60–67. IEEE, 2013.

[Gonçalves2010] Enyo José Tavares Gonçalves, Mariela Inés Cortés, Gustavo AL de Campos et Viviane Torres da Silva. *Extending MAS-ML to Model Proactive and Reactive Sotware Agents.* In ICEIS (2), pages 75–84, 2010.

[Hanna2012] Nader Hanna et Deborah Richards. *A framework for a multi-agent collaborative virtual learning environment (MACVILLE) based on activity theory*. In Pacific Rim Knowledge Acquisition Workshop, pages 209–220. Springer, 2012.

[Hardy2005] James Hardy, Mark A Eys et Albert V Carron. *Exploring the potential disadvantages of high cohesion in sports teams*. Small group research, vol. 36, no. 2, pages 166–187, 2005.

[Hector2005] Aaron Hector et V Lakshmi Narasimhan. *A new classification scheme for software agents*. In Third International Conference on Information Technology and Applications (ICITA'05), volume 1, pages 191–196. IEEE, 2005.

[Heider1946] Fritz Heider. *Attitudes and cognitive organization*. The Journal of psychology, vol. 21, no. 1, pages 107–112, 1946.

[Henningsen2013] David Dryden Henningsen, Mary Lynn Miller Henningsen et Paul Booth. *Predicting social and personal attraction in task groups*. Groupwork, vol. 23, no. 1, pages 73–93, 2013.

[Heuzé2002] Jean-Philippe Heuzé et Paul Fontayne. *Questionnaire sur l'Ambiance du Groupe : A French Language Instrument for Measuring Group Cohesion*. Human Kinetics Publishers, vol. Journal of Sport & Exercise Psychology, no. 24, pages 42–67, 2002.

[Homans1950] George C Homans. The human group. Routledge, 1950.

[Homans1974] George C Homans. *Social behavior : Its elementary forms*. 1974.

[Hornof1997] Anthony J Hornof et David E Kieras. *Cognitive modeling reveals menu search in both random and systematic*. In Proceedings of the ACM SIGCHI Conference on Human factors in computing systems, pages 107–114, 1997.

[Houhamdi2020] Zina Houhamdi et Belkacem Athamena. *Collaborative Team Construction in Open Multi-Agents System*. In 2020 21st International Arab Conference on Information Technology (ACIT), pages 1–7. IEEE, 2020.

[Hudlicka2002] Eva Hudlicka. *This time with feeling : Integrated model of trait and state effects on cognition and behavior*. Applied Artificial Intelligence, vol. 16, no. 7-8, pages 611–641, 2002.

[Inácio2019] Fabrício R Inácio, Douglas G Macharet et Luiz Chaimowicz. *PSO-based strategy for the segregation of heterogeneous robotic swarms*. Journal of Computational Science, vol. 31, pages 86–94, 2019.

[Janovsky2016] Pavel Janovsky et Scott A DeLoach. *Multi-agent simulation framework for large-scale coalition formation*. In 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI), pages 343–350. IEEE, 2016.

[Karavas2015] Christos-Spyridon Karavas, George Kyriakarakos, Konstantinos G Arvanitis et George Papadakis. *A multi-agent decentralized energy management system based on distributed intelligence for the design and control of autonomous polygeneration microgrids*. Energy Conversion and Management, vol. 103, pages 166–179, 2015.

[Kayal2019] Paridhika Kayal et Jörg Liebeherr. *Distributed service placement in fog computing : An iterative combinatorial auction approach*. In 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pages 2145–2156. IEEE, 2019.

[Kirchherr2017] Julian Kirchherr, Denise Reike et Marko Hekkert. *Conceptualizing the circular economy : An analysis of 114 definitions*. Resources, conservation and recycling, vol. 127, pages 221–232, 2017.

[Kirchherr2018] Julian Kirchherr, Laura Piscicelli, Ruben Bour, Erica Kostense-Smit, Jennifer Muller, Anne Huibrechtse-Truijens et Marko Hekkert. *Barriers to the circular economy : Evidence from the European Union (EU)*. Ecological Economics, vol. 150, pages 264–272, 2018.

[Klijn2014] Flip Klijn et Ayşe Yazıcı. *A many-to-many 'rural hospital theorem'*. Journal of Mathematical Economics, vol. 54, pages 63–73, 2014.

[Kotseruba2018] Iuliia Kotseruba et John K Tsotsos. *40 years of cognitive architectures : core cognitive abilities and practical applications*. Artificial Intelligence Review, vol. 53, no. 1, pages 17–94, 2018.

[Kristof-Brown2005] Amy Kristof-Brown, Murray R Barrick et Cynthia Kay Stevens. *When opposites attract : a multi-sample demonstration of comple-*

mentary person-team fit on extraversion. Journal of personality, vol. 73, no. 4, pages 935–958, 2005.

[Kumar2010] M. Kumar, D. P. Garg et V. Kumar. *Segregation of Heterogeneous Units in a Swarm of Robotic Agents*. IEEE Transactions on Automatic Control, vol. 55, no. 3, pages 743–748, 2010.

[Kumova2017] Bora İ Kumova et Samuel Bacha Heye. *A survey of robotic agent architectures*. In 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), pages 1–8. Ieee, 2017.

[Laird1987] John E Laird, Allen Newell et Paul S Rosenbloom. *Soar : An architecture for general intelligence*. Artificial intelligence, vol. 33, no. 1, pages 1–64, 1987.

[Laird2015] John E Laird et Clare Bates Congdon. *The Soar User's Manual Version 9.5. 0*. Rapport technique, Computer Science and Engineering Department, University of Michigan, 2015.

[Lane2019] Peter CR Lane, Anthony Sykes et Fernand Gobet. *Combining low-level perception with expectations in CHREST*. In Proceedings of EuroCogSci 03, pages 205–210. Routledge, 2019.

[Linson2015] Adam Linson, Chris Dobbyn, George E Lewis et Robin Laney. *A subsumption agent for collaborative free improvisation*. Computer Music Journal, vol. 39, no. 4, pages 96–115, 2015.

[Lott1965] Albert J Lott et Bernice E Lott. *Group cohesiveness as interpersonal attraction : A review of relationships with antecedent and consequent variables.* Psychological bulletin, vol. 64, no. 4, page 259, 1965.

[MacQueen1967] James MacQueen. *Some methods for classification and analysis of multivariate observations*. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[Manlove2008] David F Manlove. *Hospitals/residents problem*. 2008.

[McLeod2013] Janet McLeod et Kathryn Von Treuer. *Towards a cohesive theory of cohesion*. International Journal of Business and Social Research, vol. 3, no. 12, pages 1–11, 2013.

[Mendoza2021] E Mendoza, J Andramuño, J Núñez et I Benítez. *Deliberative architecture for smart sensors in the filtering operation of a water purification plant*. In Journal of Physics : Conference Series, volume 1730, page 012088. IOP Publishing, 2021.

[Michalak2010] Tomasz Michalak, Jacek Sroka, Talal Rahwan, Michael Wooldridge, Peter McBurney et Nicholas Jennings. *A distributed algorithm for anytime coalition structure generation*. volume 1, pages 1007–1014, 2010.

[Michalak2016] Tomasz Michalak, Talal Rahwan, Edith Elkind, Michael Wooldridge et Nicholas R Jennings. *A hybrid exact algorithm for complete set partitioning*. Artificial Intelligence, vol. 230, pages 14–50, 2016.

[Mikalachki1969] A. Mikalachki. *Group cohesion reconsidered.* 1969. University of Western Ontario.

[Miller2007] RS Miller, D Perlman et SS Brehm. *The building blocks of relationships*. Intimate Relationships, pages 3–39, 2007.

[Moreland1993] Richard L Moreland, John M Levine et Marie A Cini. *Group socialization : The role of commitment.* 1993.

[Morge2017] Maxime Morge et Antoine Nongaillard. *Affectation distribuée d'individus à des activités avec des préférences additivement séparables*. In Journées Francophones sur les Systèmes Multi-Agents, pages 19–28. Cépaudès édition, 2017.

[Morris-Martin2019] Andreasa Morris-Martin, Marina De Vos et Julian Padget. *Norm emergence in multiagent systems : a viewpoint paper*. Autonomous Agents and Multi-Agent Systems, vol. 33, no. 6, pages 706–749, 2019.

[Mrissa2015] Michael Mrissa, Lionel Médini, Jean-Paul Jamont, Nicolas Le Sommer et Jérôme Laplace. *An avatar architecture for the web of things*. IEEE Internet Computing, vol. 19, no. 2, pages 30–38, 2015.

[Nakashima1998] Hideyuki Nakashima et Itsuki Noda. *Dynamic subsumption architecture for programming intelligent agents*. In Proceedings International Conference on Multi Agent Systems (Cat. No. 98EX160), pages 190–197. IEEE, 1998.

[Newcomb1960] Theodore M Newcomb. *Some varieties of interpersonal attraction.* 1960.

[Newcomb1963] Theodore M Newcomb. *Stabilities underlying changes in interpersonal attraction.* The Journal of Abnormal and Social Psychology, vol. 66, no. 4, page 376, 1963.

[Newcomb1979] Theodore M Newcomb. *Reciprocity of interpersonal attraction : A nonconfirmation of a plausible hypothesis*. Social Psychology Quarterly, pages 299–306, 1979.

[Newell1994] Allen Newell. Unified theories of cognition. Harvard University Press, 1994.

[Nixon1979] Howard L Nixon. The small group. Prentice Hall, 1979.

[Occello2019] Michel Occello, Jean-Paul Jamont, Choukri-Bey Ben-Yelles et Thi Thanh Ha Hoang. *A multi-level generic multi-agent architecture for supervision of collective cyber-physical systems*. International journal of autonomous and adaptive communications systems, vol. 12, no. 2, pages 109–128, 2019.

[Perumal2014] Thinagaran Perumal, Md Nasir Sulaiman, Norwati Mustapha, Ahmad Shahi et R Thinaharan. *Proactive architecture for Internet of Things (IoTs) management in smart homes*. In 2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE), pages 16–17, 2014.

[Plass2015] Jan L Plass, Bruce D Homer et Charles K Kinzer. *Foundations of game-based learning*. Educational Psychologist, vol. 50, no. 4, pages 258–283, 2015.

[Premarathne2020] US Premarathne et S Rajasingham. *Trust based multi-agent cooperative load balancing system (TCLBS)*. Future Generation Computer Systems, vol. 112, pages 185–192, 2020.

[Pynadath2014] David V Pynadath, Paul S Rosenbloom et Stacy C Marsella. *Reinforcement learning for adaptive theory of mind in the sigma cognitive architecture*. In International Conference on Artificial General Intelligence, pages 143–154. Springer, 2014.

[Qasim2021] Awais Qasim, Areeba Bader et Adeel Munawar. *Efficient Contract-Net Protocol For Formal Modeling Of Multi-Agent Systems*. International Journal of Computing and Digital Systems, vol. 10, 2021.

[Rahwan2007] Talal Rahwan et Nicholas R Jennings. *An algorithm for distributing coalitional value calculations among cooperating agents*. Artificial Intelligence, vol. 171, no. 8-9, pages 535–567, 2007.

[Rahwan2009] Talal Rahwan, Sarvapali D Ramchurn, Nicholas R Jennings et Andrea Giovannucci. *An anytime algorithm for optimal coalition structure generation*. Journal of artificial intelligence research, vol. 34, pages 521–567, 2009.

[Ramos2013] Gabriel De O. Ramos, Burguillo Juan C. et Ana L.C. Bazzan. *Self-Adapting Coalition Formation Among Electric Vehicles in Smart Grids*. In 2013 IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems, pages 11–20, 2013.

[Ricci2002] Alessandro Ricci, Andrea Omicini et Enrico Denti. *Activity Theory as a framework for MAS coordination*. In International Workshop on Engineering Societies in the Agents World, pages 96–110. Springer, 2002.

[Ritter2012] Frank E Ritter, Jennifer L Bittner, Sue E Kase, Rick Evertsz, Matteo Pedrotti et Paolo Busetta. *CoJACK : A high-level cognitive architecture with demonstrations of moderators, variability, and implications for situation awareness*. Biologically Inspired Cognitive Architectures, vol. 1, pages 2–13, 2012.

[Rohrer2012] Brandon Rohrer. *BECCA : Reintegrating AI for natural world interaction*. In 2012 AAAI Spring Symposium Series, 2012.

[Rovio2009] Esa Rovio, Jari Eskola, Stephen A Kozub, Joan L Duda et Taru Lintunen. *Can high group cohesion be harmful ? A case study of a junior*

ice-hockey team. Small Group Research, vol. 40, no. 4, pages 421–435, 2009.

[Saad2010] Walid Saad, Zhu Han, Tamer Basar, Mérouane Debbah et Are Hjorungnes. *Hedonic coalition formation for distributed task allocation among wireless agents*. IEEE Transactions on Mobile Computing, vol. 10, no. 9, pages 1327–1344, 2010.

[Sacerdote2005] Bruce Sacerdote et David Marmaros. *How do friendships form ?* Rapport technique, National Bureau of Economic Research, 2005.

[Santos2014] Vinicius Graciano Santos et Luiz Chaimowicz. *Cohesion and segregation in swarm navigation.* Robotica, vol. 32, no. 2, pages 209–223, 2014.

[Schermerhorn2006] Paul W Schermerhorn, James F Kramer, Christopher Middendorff et Matthias Scheutz. *DIARC : A Testbed for Natural Human-Robot Interaction.* In AAAI, volume 6, pages 1972–1973, 2006.

[Segal1974] Mady W Segal. *Alphabet and attraction : An unobtrusive measure of the effect of propinquity in a field setting.* Journal of Personality and Social Psychology, vol. 30, no. 5, page 654, 1974.

[Shi2011] Hong Shi et Guangming Xie. *Collective dynamics of swarms with a new attraction/repulsion function*. Mathematical Problems in Engineering, vol. 2011, 2011.

[Sibson1973] Robin Sibson. *SLINK : an optimally efficient algorithm for the single-link cluster method*. The computer journal, vol. 16, no. 1, pages 30–34, 1973.

[Siebold2007] Guy L Siebold. *The essence of military group cohesion*. Armed forces & society, vol. 33, no. 2, pages 286–295, 2007.

[Singh2015] Munindar P Singh. *Cybersecurity as an Application Domain for Multiagent Systems.* In AAMAS, pages 1207–1212. Citeseer, 2015.

[Skinner1953] Burrhus Frederic Skinner. Science and human behavior. Numeéro 92904. Simon and Schuster, 1953.

[Slam2015] Nady Slam, Wenjun Wang, Guixiang Xue et Pei Wang. *A framework with reasoning capabilities for crisis response decision–support systems*. Engineering Applications of Artificial Intelligence, vol. 46, pages 346–353, 2015.

[Sun1999] Ron Sun, Todd Peterson et Edward Merrill. *A hybrid architecture for situated learning of reactive sequential decision making*. Applied Intelligence, vol. 11, no. 1, pages 109–127, 1999.

[Takalloo2020] Mahdi Takalloo, Aigerim Bogyrbayeva, Hadi Charkhgard et Changhyun Kwon. *Solving the winner determination problem in combinatorial auctions for fractional ownership of autonomous vehicles*. International Transactions in Operational Research, 2020.

[Thibaut1959] John Thibaut et Harold Kelley. The social psychology of groups. New York : John Wiley and Sons, 1959.

[Tobin2008] Kimberly Tobin. Gangs : An individual and group perspective. Pearson/Prentice Hall, 2008.

[Van Dang2018] Chien Van Dang, Mira Jun, Yong-Bin Shin, Jae-Won Choi et Jong-Wook Kim. *Application of modified Asimov's laws to the agent of home service robot using state, operator, and result (Soar)*. International Journal of Advanced Robotic Systems, vol. 15, no. 3, page 1729881418780822, 2018.

[Van Den Berg2011] Jur Van Den Berg, Stephen J Guy, Ming Lin et Dinesh Manocha. *Reciprocal n-body collision avoidance*. In Robotics research, pages 3–19. Springer, 2011.

[Vig2006] Lovekesh Vig et Julie A Adams. *Market-based multi-robot coalition formation*. In Distributed Autonomous Robotic Systems 7, pages 227–236. Springer, 2006.

[Vygotsky1978] Lev Semenovich Vygotsky. Mind in society : The development of higher psychological processes. Harvard university press, 1978.

[Walster1966] Elaine Walster, Vera Aronson, Darcy Abrahams et Leon Rottman. *Importance of physical attractiveness in dating behavior.* Journal of personality and social psychology, vol. 4, no. 5, page 508, 1966.

[Wang2016] Tianyu Wang, Lingyang Song, Zhu Han et Walid Saad. *Overlapping coalition formation games for emerging communication networks*. IEEE Network, vol. 30, no. 5, pages 46–53, 2016.

[Wang2020] Jingwen Wang, Xuyang Jing, Zheng Yan, Yulong Fu, Witold Pedrycz et Laurence T Yang. *A survey on trust evaluation based on machine learning*. ACM Computing Surveys (CSUR), vol. 53, no. 5, pages 1–36, 2020.

[Winch1958] Robert F Winch. *Mate-selection ; a study of complementary needs.* 1958.

[Ye2013] Dayong Ye, Minjie Zhang et Danny Sutanto. *Self-adaptation-based dynamic coalition formation in a distributed agent network : A mechanism and a brief survey*. IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 5, pages 1042–1051, 2013.

[Yeh1986] D Yun Yeh. *A dynamic programming approach to the complete set partitioning problem*. BIT Numerical Mathematics, vol. 26, no. 4, pages 467–474, 1986.

[Yu2013] Han Yu, Zhiqi Shen, Cyril Leung, Chunyan Miao et Victor R Lesser. *A survey of multi-agent trust management systems*. IEEE Access, vol. 1, pages 35–50, 2013.

[Zachary1993] Wayne W Zachary, Allen L Zaklad, James H Hicinbothom, Joan M Ryder et Janine A Purcell. *COGNET representation of tactical decision-making in Anti-Air Warfare*. In Proceedings of the Human Factors and Ergonomics Society Annual Meeting, volume 37, pages 1112–1116. SAGE Publications Sage CA : Los Angeles, CA, 1993.

[Zhang2000] Bo Zhang, Xiaoping Chen, Guiquan Liu et Qingsheng Cai. *Agent architecture : a survey on RoboCup-99 simulator teams*. In Proceedings of the 3rd World Congress on Intelligent Control and Automation (Cat. No.00EX393), volume 1, pages 194–198 vol.1, 2000.

[Zia2009] Tehseen Zia, R. Lang, H. Boley, D. Bruckner et G. Zucker. *An Autonomous Adaptive Multiagent Model for Building Automation*. IFAC Proceedings Volumes, vol. 42, pages 250–254, 2009.

Une approche d'intelligence collective pour la conception d'un système d'aide à la décision appliqué à l'économie circulaire

A collective intelligence approach for the conception of a decision support system applied to the circular economy

Résumé

Ce travail s'inscrit dans le projet Circular qui a pour objectif de mettre en place les outils et les conditions nécessaires à la conception d'un nouveau système industriel d'aide à la décision pour l'économie circulaire. Ce système industriel devra être capable de transformer des produits usés en fin de vie en produits neufs grâce à l'utilisation de stratégies de reconditionnement basées sur la réutilisation de leurs composants. Cette thèse est centrée sur la conception de ce système d'aide à la décision à l'aide du paradigme multi-agent. Les composants usés sont avatarisés en agents virtuels faisant partie du système multi-agent et ayant la capacité de manipuler des connaissances, de communiquer et de prendre des décisions. Autrement dit, nous associons à chaque composant physique un agent virtuel possédant les mêmes caractéristiques (physiques, logistiques, etc.). De la même manière qu'un produit est un ensemble de composants assemblés les uns aux autres, nous souhaitons que nos agents s'auto-organisent en coalitions afin de former au mieux les produits demandés par un utilisateur du système d'aide à la décision. Ce problème est très proche d'un problème de formation de coalitions dans la littérature informatique. De plus, nous faisons l'analogie entre les agents de notre système multi-agent et des individus. Grâce à celle-ci, nous proposons d'aborder le problème de formation de coalitions en nous inspirant d'un champ de la sociologie, les dynamiques des groupes. Le travail abordé dans cette thèse se concentre sur la conception d'une architecture d'agent inspirée par des mécanismes sociologiques capables de résoudre le problème de formation de coalitions et permettant de répondre aux besoins du système d'aide à la décision.

Mots-clés : SMA, architecture d'agent, économie circulaire, système d'aide à la décision, Dynamiques des Groupes

Abstract

This work is part of the Circular project, which aims to set up the tools and the necessary conditions for the design of a new industrial decision support system for the circular economy. This industrial system will have to be able to transform post-used products at the end of their life into new products through the use of reconditioning strategies based on the reuse of their components. This thesis focuses on the design of this decision support system using the multi-agent paradigm. The post-used components are avatarized into virtual agents that are part of the multi-agent system. Virtual agents have the ability to manipulate knowledge, communicate and make decisions. In the same way that a product is a set of components assembled together, we want our agents to self-organize into coalitions in order to form the products requested by a user of the decision support system. This problem is very similar to a coalition formation problem in the computer science literature. Moreover, we do the analogy between the agents of our multi-agent system and individuals. Thanks to this analogy, we propose to tackle the coalition formation problem by drawing inspiration from a field of sociology, group dynamics. The work addressed in this thesis focuses on the design of an agent architecture inspired by sociological mechanisms capable of solving the coalition formation problem and allowing to meet the needs of the decision support system.

Keywords : MAS, agent architecture, circular economy, decision support system, Group Dynamics