Optimisation de l'utilité des données lors d'un processus de k-anonymisation

Clémence Mauger

Thèse de Doctorat

Mention Informatique

présentée à l'École Doctorale en Sciences, Technologie, Santé (ED 585)

à l'Université de Picardie Jules Verne

par

Clémence Mauger

pour obtenir le grade de Docteur de l'Université de Picardie Jules Verne

Optimisation de l'utilité des données lors d'un processus de k-anonymisation

Soutenue le 6 décembre 2021 après avis des rapporteurs, devant le jury d'examen :

M. Michaël Krajecki, Professeur, URCA	Président du jury
M^me^ Maryline Laurent, Professeur, Telecom SudParis	Rapporteur
M. Benjamin Nguyen, Professeur, INSA Centre Val de Loire	Rapporteur
M^me^ Claire Delaplace, Maître de Conférences, UPJV	Examinateur
M. Mathieu Cunche, Maître de Conférences HDR, INSA Lyon	Examinateur
M. Gilles Dequen, Professeur, UPJV	Directeur de thèse
M. Gaël Le Mahec, Maître de Conférences, UPJV	Co-encadrant

Remerciements

Avant toute chose, je tiens à remercier Mme Maryline Laurent et M. Benjamin Nguyen pour le temps qu'ils ont consacré à relire mon manuscrit de thèse et pour leurs remarques pertinentes. Je les remercie d'avoir étudié en détails mon travail de recherche et d'avoir approuvé ma soutenance de thèse. Je remercie également Mme Claire Delaplace, M. Mathieu Cunche et M. Michaël Krajecki pour avoir accepté de faire partie de mon jury de thèse en tant qu'examinateurs.

Il va sans dire que je suis extrêmement reconnaissante à Gilles et à Gaël pour le soutien et l'écoute dont ils ont fait preuve durant ces trois dernières années. Ils ont su me conseiller, me rassurer et surtout me faire relativiser quand ça n'allait pas aussi bien que je le voulais. Un merci spécial à Gaël pour sa gentilles et sa bienveillance, notamment quand il était question de débrouiller mes algorithmes Python et mes résultats d'expérimentations... :)

Je souhaite remercier les membres du MIS que j'ai cotoyés, de près ou de loin. Une mention spéciale à mes collègues de bureau avec lesquels j'ai passé de bons moments dans la bonne humeur : Romuald, Monika, Fabien, Olivier, Pierre et Sébastien. Je remercie également les membres de l'entreprise Evolucare avec lesquels j'ai pu échanger au cours de ma thèse.

Un grand merci à Alice, que je connais depuis huit ans déjà, et avec laquelle je peux discuter (et me plaindre) de tout.

J'exprime ma très grande reconnaissance à mes parents. Ils me soutiennent depuis toutes ses années et ont toujours été présents pour moi. Je ne serais pas où j'en suis aujourd'hui sans eux. Je remercie également mes trois frères et ma petite sœur préférée.

Je ne peux pas finir ces remerciements sans mentionner mon compagnon Damien. Il a été présent tout au long de ma thèse et a été d'un soutien et d'une aide inimaginables. Dans les bons moments, il était là pour m'écouter et m'encourager. Dans les moments plus difficiles, il a toujours su me remonter le moral et me pousser à continuer à avancer...

Table des matières

Introduction

Depuis les années 1990, le nombre de données échangées et stockées a explosé. Cela est notamment dû à l'essor des capacités de stockage et de gestion de ces données. Sur internet, les données des utilisateurs sont récoltées partout et par de nombreux organismes. Par exemple, les données médicales sont conservées par les hôpitaux et servent au suivi du patient. Les moteurs de recherche quant à eux exploitent les historiques de recherches internet des utilisateurs pour leur proposer des publicités ciblées sur leurs centres d'intérêt.

Avec l'émergence de la *data science*, ou science des données, qui vise à exploiter ces gigantesques bases de données brutes, des questions sur la protection de la vie privée des individus concernés par ces données se sont posées. De plus, des failles de sécurité et des vols massifs de données ont accentué la méfiance des utilisateurs envers ceux qui récoltent et qui conservent leurs données. En France, de nombreux hôpitaux et laboratoires médicaux ont subi des piratages et des vols de données médicales de milliers de patients. Le dernier événement de ce genre date de 2020 et concerne les Hôpitaux de Paris : les données personnelles d'environ 1,4 million de personnes auraient été dérobées. De par le monde, des réglementations ont donc vu le jour pour fixer un cadre légal à la collecte, la conservation et l'exploitation des données à caractère personnel. Dans l'Union Européenne, le Règlement Général sur la Protection des Données (RGPD) est le texte faisant office de loi sur la protection des données.

Afin de continuer à exploiter les données sans compromettre la vie privée des individus, il est nécessaire de masquer l'identité ou les données sensibles des individus dans la base de données. Ce processus est appelé *anonymisation* [10]. Des chercheurs ont développé des modèles d'anonymisation permettant de publier ou d'analyser les données en garantissant une certaine sécurité aux propriétaires des données. Dans les domaines de recherche Privacy-Preserving Data Publishing (PPDP) [17] et Privacy-Preserving Data Mining (PPDM) [1] les objectifs sont différents.

En PPDP, les modèles proposés ont pour objectif de faire en sorte que la publication des données ne permettent pas d'associer un individu à un enregistrement de la base ni d'en apprendre plus sur les données sensibles de l'individu. Par exemple, en cas de publication d'une base de données médicales, un adversaire ne doit pas être en mesure d'associer une pathologie à une identité. La k-anonymité [41] est un exemple de modèles d'anonymisation proposés en PPDP. Elle garantit que tout enregistrement de la base de données est indistinguable d'au moins $k-1$ autres enregistrements par rapport à un ensemble d'attributs. Bien que ce modèle ne protège par entièrement contre la divulgation de nouvelles informations sur les individus, il a été très étudié et empêche de lier de manière unique un individu à un enregistrement de la base de données.

En PPDM, les modèles proposés visent à ce que les résultats de tâches de *data mining* sur une base de données soient les mêmes qu'on utilise les données brutes ou les données modifiées. Le modèle d'anonymisation le plus couramment utilisé en PPDM est la confidentialité différentielle [15]. Ce modèle garantit que la présence ou l'absence d'un individu dans une base de données ne change pas le résultat d'une tâche de *data mining* effectuée sur la base. Dans [25], les auteurs présentent des exemples d'application de la confidentialité différentielle dans l'industrie.

Alors que le cadre d'exploitation des données après publication n'a pas besoin d'être connu dans un contexte de PPDP, l'éditeur des données doit être capable de comprendre la tâche de *data mining* qui sera appliquée aux données anonymisées en PPDM. De plus, en PPDM, les modifications des données lors de l'anonymisation rendent généralement leur lecture peu claire pour le grand public. Ce sont généralement les résultats de la tâche de *data mining* qui sont publiés. En PPDP, les modèles d'anonymisation ne touchent pas à l'intégrité des données au-delà d'un ajout d'imprécision et n'ajoutent pas d'enregistrements fictifs à la base de données. Bien qu'elles aient subi des modifications, les données sont toujours compréhensibles après publication.

Malgré ces différences, des chercheurs ont proposé des méthodes pour relier les modèles de PPDP et de PPDM. Par exemple, dans [21] et [30], les auteurs font le lien entre k-anonymité et confidentialité différentielle. Dans [11], les auteurs explorent les relations entre t-proximité et confidentialité différentielle.

Dans le contexte qui a motivé cette thèse, l'objectif était de proposer des solutions pour stocker et publier des données médicales en respectant la vie privée des individus dans la cadre du projet PSPC PIA3 SmartAngel.

SmartAngel est un projet d'aide au suivi et à l'accompagnement des patients porté par l'entreprise Evolucare. D'après [16], leur objectif est de surveiller et d'accompagner les patients lors de leur parcours de soin et de leur retour à domicile à l'aide d'un éco-système numérique.

Aucune tâche de *data mining* n'ayant été prévue sur les données une fois anonymisées, nous nous sommes focalisés sur le domaine PPDP et nous n'avons pas exploré la confidentialité différentielle. Plus précisément, nous nous sommes intéressés au modèle de k-anonymité. Étant l'un des premiers modèles d'anonymisation proposés et d'une relative facilité de mise en œuvre, la k-anonymité permet de répondre aux problématiques posées par le projet SmartAngel.

Dans un premier temps, nous avons étudié certains des premiers algorithmes de k-anonymisation tels que *Incognito* [27] et *Mondrian* [26]. Nous nous sommes aperçus que les métriques utilisées pour comparer la qualité des productions des algorithmes ne sont pas toujours les mêmes d'un article à l'autre. Elles sont également difficilement comparables car exprimées de manières différentes et peu homogènes. Nous avons donc axé la première partie de nos travaux sur la recherche d'une métrique permettant à la fois de donner un critère de qualité pour classer les tables k-anonymes et de construire les meilleures tables k-anonymes possibles en termes d'utilité des données.

Après avoir mené une étude comparative de plusieurs métriques dans le cadre d'une k-anonymisation, nous nous sommes penchés sur l'une des faiblesses de la k-anonymité. Ne donnant aucune contrainte sur la répartition des valeurs de l'attribut sensible dans la table k-anonyme, les tables résultant d'une k-anonymisation sont parfois sujettes à un manque de diversité dans les valeurs de l'attribut sensible de leurs classes d'équivalence. Par exemple, imaginons qu'une table k-anonyme contenant pour données sensibles des maladies soit publiée. Dans cette table, une classe d'équivalence pourrait très bien contenir k femmes habitants dans la Somme ayant un cancer pour maladie. Nous ne savons pas précisément qui sont ces k femmes, en revanche, si un adversaire connaît une femme habitant dans la Somme et qu'il sait qu'elle est dans la table, il peut aisément conclure qu'elle souffre d'un cancer.

Afin de prévenir ce genre de situation, nous avons intégré dans le processus de k-anonymisation des conditions sur la répartition des valeurs de l'attribut sensible dans la table. Nous avons utilisé deux autres modèles d'anonymisation de PPDP : la l-diversité [34] et la t-proximité [29]. La deuxième partie de nos travaux a été consacrée à la recherche de stratégies à utiliser dans un algorithme de k-anonymisation glouton permettant de garder un contrôle sur l'altération des données et sur la répartition des valeurs de l'attribut sensible dans les classes d'équivalence de la table k-anonyme.

Dans les deux premières parties de nos travaux de recherche, nous avons utilisé un algorithme de k-anonymisation glouton. Pour construire une version k-anonyme d'une table, une succession de fusion de classes d'équivalence est effectuée dans la table jusqu'à ce que chaque classe contienne au moins k enregistrements. Néanmoins, nous nous sommes vite aperçus que cet algorithme ne fournit par les meilleures tables k-anonymes en termes de limitation de l'altération des données. Nous avons donc cherché à améliorer les tables k-anonymes produites avec l'algorithme glouton. En étudiant des exécutions de l'algorithme glouton, nous avons constaté que certaines classes d'équivalence des tables k-anonymes produites contiennent bien plus de k enregistrements. De plus, sur de petits exemples joués, nous avons remarqué que ces classes d'équivalence peuvent souvent être décomposées en plusieurs classes de plus petites tailles respectant toujours la k-anonymité. La troisième partie de nos travaux a été consacrée à la recherche d'une procédure permettant de trouver de meilleurs k-partitionnements des classes d'équivalence de grandes tailles dans une table k-anonyme. Pour cela, nous avons étudié la formulation même du problème de k-anonymisation d'une table.

La suite de ce manuscrit est organisée de la manière suivante.

Nous reviendrons dans le chapitre 1 page 5 sur le domaine PPDP et sur les attaques, modèles et techniques d'anonymisation s'y rattachant. Nous présenterons notamment trois modèles d'anonymisation : la k-anonymité, la l-diversité et la t-proximité.

Dans le chapitre 2 page 13, nous poserons les notations et définitions nécessaires à la présentation de nos travaux de recherche. Nous présenterons en particulier les hiérarchies de généralisation associées aux attributs quasi-identifiants car la technique d'anonymisation que nous utiliserons sera la généralisation.

Dans le chapitre 3 page 25, nous nous intéresserons aux métriques de perte d'infomation permettant d'évaluer la quantité d'information perdue dans les données lors d'un processus d'anonymisation par généralisation. Nous étant aperçus que les définitions des métriques dans la littérature n'utilisent souvent pas les mêmes notations, notre première contribution consistera à proposer un modèle unifiant l'écriture des métriques et simplifiant leur utilisation. Nous définirons pour cela une matrice pour chaque attribut quasi-identifiant contenant les coûts de généralisation des nœuds de la hiérarchie de généralisation. Après avoir présenté trois métriques de la littérature, nous définirons une nouvelle métrique issue de nos travaux, la *Lost Leaves Metric*, ainsi que trois de ses variantes. Nous comparerons ensuite les performances des métriques de perte d'information lors d'un processus de k-anonymisation en utilisant un algorithme de k-anonymisation glouton. Dans cet algorithme,

des fusions de classes d'équivalence de la table sont effectuées jusqu'à ce que la table soit k-anonyme. Les fusions de classes d'équivalence sont choisies de telle sorte à minimiser l'altération de la table pour une métrique. Nous confronterons les métriques sur deux tables publiques et déterminerons la ou les métriques permettant de produire les tables k-anonymes de meilleure qualité au regard de trois critères.

Dans le chapitre 4 page 55, nous essaierons de limiter l'un des points faibles de la k-anonymité. Ne tenant pas compte de la répartition des valeurs de l'attribut sensible dans sa définition, les tables k-anonymes souffrent parfois d'un manque de diversité dans les valeurs de l'attribut sensible de leurs classes d'équivalence. Nous nous intéresserons donc aux modèles de l-diversité et de t-proximité. Ces deux modèles d'anonymisation donnent des contraintes sur la répartition des valeurs de l'attribut sensible dans les classes d'équivalence. Notre principale contribution dans ce chapitre sera la présentation de stratégies à utiliser dans un algorithme d'anonymisation glouton pour guider les fusions de classes d'équivalence à effectuer. Ces stratégies chercheront à la fois à limiter l'altération des données et à optimiser les valeurs de l-diversité et de t-proximité de la table à chaque fusion effectuée. Nous comparerons les performances de ces stratégies lors d'un processus de k-anonymisation en menant des expérimentations sur des données réelles et des données simulées. Nous montrerons qu'utiliser certaines de ces nouvelles stratégies permet de garder un contrôle sur l'altération des données et sur la répartition des valeurs de l'attribut sensible dans les classes d'équivalence des tables k-anonymes produites.

L'algorithme de k-anonymisation glouton utilisé dans les deux chapitres précédents n'est pas optimal en termes de limitation de l'altération des données. Dans le chapitre 5 page 87, nous nous attacherons donc à améliorer les tables k-anonymes produites avec cet algorithme.

Notre première contribution consistera à donner une nouvelle formulation du problème de k-anonymisation d'une table. Pour cela, nous définirons les groupes de généralisation d'une table qui regroupent les enregistrements en fonction des généralisations possibles de leurs valeurs quasi-identifiantes. Nous verrons ensuite que les groupes de généralisation d'une table peuvent se représenter sous la forme d'un hypergraphe. La matrice d'incidence de cet hypergraphe nous fournira une formulation du problème de k-anonymisation d'une table.

Notre seconde contribution dans ce chapitre sera la présentation d'une procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$. En nous appuyant sur la formulation du problème de k-anonymisation précédente, nous expliquerons comment obtenir une table k-anonyme en cherchant de nouveaux partitionnements des enregistrements des classes d'équivalence d'une table k'-anonyme. Cette procédure nous servira à développer cinq algorithmes de construction d'une table k-anonyme par partionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$. Nous comparerons l'altération des données dans les tables k-anonymes produites avec ces cinq algorithmes et l'altération des données dans les tables k-anonymes produites avec l'algorithme glouton. Nous montrerons ainsi qu'il est possible d'améliorer la qualité des tables k-anonymes produites avec l'algorithme glouton.

Le chapitre 5.5 page 129 sera consacrée à la conclusion de ce manuscrit. Nous présenterons également une collaboration avec Fabien Viton, doctorant au MIS, et Jean-Luc Guérin, MCF au MIS, qui a abouti sur la proposition d'une nouvelle représentation des données à utiliser dans le cadre d'une tâche de *data mining*. De plus, nous évoquerons le travail de stage de seconde année de Master de Arsème Vadèle Djeufack Nanfack sur la parallélisation de l'algorithme de k-anonymisation glouton.

Chapitre 1

Privacy-Preserving Data Publishing : attaques, modèles et techniques d'anonymisation

Sommaire du présent chapitre

1.1 Privacy-Preserving Data Publishing

Depuis plusieurs décennies, les collectes de données à caractère personnel se multiplient. L'utilisateur est régulièrement sollicité pour partager ses données, le cadre d'exploitation de celles-ci n'étant pas toujours clairement défini. Les données collectées peuvent ensuite être utilisées à des fins publicitaires. Par exemple, Google et Facebook se basent sur les différentes recherches du consommateur sur leur site pour lui proposer des publicités ciblées en fonction de ses centres d'intérêt. Les données peuvent aussi être exploitées à des fins d'aide à la décision. Par exemple, les données médicales d'un patient sont comparées à une base de données contenant des milliers de données sur d'autres individus pour aider au diagnostic des médecins.

Face à la grande quantité de données échangées et publiées, des politiques de protection des données ont vu le jour ces dernières années. Le RGPD est un texte faisant office de loi sur la protection des données à caractère personnel dans l'Union Européenne [8]. En France, la Commission Nationale de l'Information et des Libertés (CNIL) est chargée de faire respecter ce réglement.

Le domaine de recherche PPDP [17] consiste à proposer des outils ou des méthodes pour publier des données dans un environnement dans lequel un adversaire cherche à découvrir des informations sur un individu. Deux éléments essentiels sont à prendre en compte avant de publier un ensemble de données : les données sont-elles encore utilisables ? La vie privée des individus concernés par les données est-elle protégée ?

Généralement, trois types de protagonistes interviennent dans la collecte et la publication de données. L'*éditeur des données* est chargé de collecter les données auprès des *propriétaires des données*. L'éditeur des données publie ensuite la base de données à destination du *bénéficiaire des données* qui peut être un expert en *data mining* ou un public plus large.

Une *base de données* est un ensemble d'enregistrements collectés auprès de propriétaires de données par l'éditeur de données. Généralement, les enregistrements comportent des attributs de quatre types : *identifiant*, *quasi-identifiant*, *sensible* et *non-sensible*. Dans ce manuscrit, une base de données fera réference à une base simple constituée d'une seule table ou vue comme la jointure de plusieurs tables.

Un attribut identifiant est un lien unique entre un individu et un enregistrement de la base de données. Il s'agit, par exemple, du nom de l'individu ou de son numéro de sécurité sociale. Afin de donner des garanties de

respect de la vie privée aux propriétaires des données, l'éditeur des données doit veiller à *anonymiser* les données avant de les publier. En gardant à l'esprit que des études statistiques vont être menées sur les données sensibles de la base, l'anonymisation [10] consiste à masquer l'identité ou les données sensibles des propriétaires des données dans la table. La première étape d'une anonymisation, appelée *pseudonymisation* ou encore *de-identification*, consiste à supprimer les attributs identifiants de la base de données ou à les remplacer par des identifiants uniques aléatoires.

Les attributs *quasi-identifiants* [10] sont les attributs sur lesquels des modifications vont être effectuées pour atteindre un certain niveau d'anonymisation. Pris séparemment, ces attributs ne constituent pas un risque pour la vie privée des individus de la table. On peut citer par exemple l'âge, le code postal ou le genre d'un individu. L. Sweeney a cependant montré que ces attributs constituent une faille dans le respect de la vie privée s'ils sont considérés dans leur ensemble. Dans [44], Sweeney s'appuie sur un registre de votants et un ensemble de données médicales d'une compagnie d'assurance dans lequels les identifiants ont été supprimés pour retrouver l'enregistrement médical du gouverneur du Massachussets de l'époque, William Weld. La *linkage attack*, nom donné à ce type d'attaque, met en lumière qu'un adversaire disposant d'assez de sources d'information, même pseudonymisées, peut découvrir des informations personnelles sur un individu. Par exemple, imaginons qu'un adversaire cherche à découvrir l'identité d'un homme né en Allemagne en 1879. Cette description correspond à environ 900 000 individus. Imaginons ensuite que l'adversaire apprenne que cet homme d'origine allemande a obtenu la nationalité américaine en 1940. Le nombre d'individus concernés se réduit à quelques milliers. Finalement, l'adversaire obtient d'une autre source d'information que l'individu recherché a obtenu un prix Nobel de physique. Le doute n'est plus permis, il s'agit d'Albert Einstein. Cet exemple montre que la connaissance de données quasi-identifiantes (genre, date et lieu de naissance, nationalité,...) conduisent parfois à l'identification d'un unique individu.

Un attribut sensible est un attribut à protéger lors de la publication de la table par l'éditeur de la base de données. Ce type d'attribut contient les données à intérêt statistique : les valeurs de ces attributs ne sont donc pas vouées à être modifiées. Cependant, les individus concernés pas ces données personnelles ne veulent pas que ces informations soient connues par tous. Une maladie ou un salaire sont des exemples d'attributs sensibles.

Un attribut non-sensible est un attribut n'entrant dans aucune des trois catégories précédentes.

En parallèlle du développement de PPDP a émergé le domaine PPDM [1]. Le premier objectif de PPDM était de faire en sorte que les outils de fouille de données (ou *data mining*) donnent les mêmes résultats s'ils sont appliqués sur des données brutes ou sur des données modifiées.

Trois principales différences sont à noter entre les domaines PPDP et PPDM [17]. 1) Dans le contexte de PPDM, l'éditeur des données doit être en mesure de comprendre l'outil de *data mining* qui sera utilisé sur les données modifiées. En effet, l'éditeur des données modifie les données pour que les résultats de la tâche de *data mining* soient les mêmes que sur les données brutes. 2) Par ailleurs, l'éditeur des données publie souvent les résultats d'une tâche de *data mining* et pas les données modifiées. Si les données modifiées sont publiées, elles peuvent ne pas être compréhensibles par le lecteur à cause des modifications apportées sur les données. Par exemple, la *randomization* [22] est une technique répandue en PPDM consistant à ajouter du bruit dans les enregistrements de la bases de données. Cette technique est souvent utilisée pour masquer les valeurs d'un attribut sensible numérique (le salaire par exemple) : une valeur aléatoire r est ajoutée à une valeur sensible s et la valeur publiée est alors $s + r$. 3) En PPDP, l'objectif est d'empêcher un adversaire de relier un individu à un enregistrement de la table. En PPDM, il s'agit plutôt de masquer les données sensibles.

1.2 Attaques et modèles d'anonymisation

Après publication d'une base de données, un adversaire peut chercher à obtenir de nouvelles informations sur un individu à partir des données publiées.

Lorsqu'un adversaire connaît les valeurs des attributs quasi-identifiants d'un individu, il peut chercher à
- relier l'individu à un enregistrement de la base de données : *attaque de divulgation d'identité*
- relier l'individu à une valeur d'un attribut sensible : *attaque de divulgation d'attribut*
- déterminer si la victime appartient à la table publiée : *attaque de divulgation d'appartenance.*

Dans les attaques de divulagation d'identité et d'attribut, l'adversaire sait déjà que la victime appartient à la base de données publiée [18].

Pour lutter contre la divulgation d'identité et la divulgation d'attribut, des modèles d'anonymisation ont vu le jour depuis les années 2000.

En 2001, Samarati a proposé dans [41] la *k-anonymité* pour prévenir de la divulgation d'identité. Une table est dite k-anonyme si chacun de ses enregistrements est indistinguable d'au moins $k-1$ autres enregistrements par rapport à l'ensemble des attributs quasi-identifiants. Dans une base de données, les enregistrements sont

regroupés en *classes d'équivalence* selon leurs valeurs pour les attributs quasi-identifiants : deux enregistrements sont dans la même classe d'équivalence si leurs valeurs pour les attributs quasi-identifiants sont les mêmes. Ainsi, la k-anonymité demande à ce que chaque classe d'équivalence de la table soit de taille supérieure à k. D'une table k-anonyme, un adversaire ne peut déduire qu'avec une probabilité $\frac{1}{k}$ l'enregistrement correspondant à un individu s'il connaît la classe d'équivalence dans laquelle il se trouve.

En jouant avec le paramètre k, deux objectifs peuvent être atteints. D'une part, en augmentant la valeur de k, la probabilité pour un adversaire de découvrir l'enregistrement correspondant à un individu diminue : en effet, la probabilité $\frac{1}{k}$ est d'autant plus petite que k est grand. Dans la table k-anonyme, les classes d'équivalence seront plus grandes et donc l'enregistrement correspondant à un individu plus difficilement identifiable par un adversaire. D'autre part, en diminuant la valeur de k, l'utilité des données augmente. En effet, moins de modifications des valeurs des attributs quasi-identifiants équivalent à un plus grand nombre d'informations conservées dans la table. Ces deux objectifs sont contradictoires. Quand k augmente, beaucoup de modifications sur les quasi-identifiants sont généralement nécessaires pour amener les classes d'équivalence à contenir au moins k enregistrements, ce qui entraîne une baisse de l'utilité des données. Quand k diminue, les classes d'équivalence sont plus petites et un adversaire a une plus grande probabilité d'identifier l'enregistrement d'un individu parmi les autres enregistrements de la classe. Il est donc plus facile d'obtenir des informations sensibles sur l'individu.

Une des faiblesses de la k-anonymité est qu'elle ne tient pas compte des attributs sensibles dans sa définition. Cela peut entraîner un manque de diversité dans les valeurs des attributs sensibles. Par exemple, dans une table k-anonyme à un attribut sensible, les enregistrements d'une classe d'équivalence peuvent partager la même valeur pour l'attribut sensible. Dans ce cas, si un adversaire sait dans quelle classe d'équivalence se trouve l'enregistrement d'un individu, il est sûr de connaître la valeur sensible de l'individu.

Pour pallier cet inconvénient, Machanavajjhala et co-auteurs ont défini la l-diversité en 2006 dans [34]. Contrairement à la k-anonymité, la l-diversité est un modèle d'anonymisation tenant compte de la répartition des valeurs des attributs sensibles dans les classes d'équivalence. La l-diversité vise à donner une protection contre les attaques de divulgation d'attribut. La l-diversité garantit que, dans chaque classe d'équivalence, au moins $l \geqslant 2$ valeurs sensibles sont « bien représentées ». On dit qu'une classe d'équivalence est « bien représentée » par l valeurs sensibles s'il existe au moins $l \geqslant 2$ valeurs sensibles dans la classe d'équivalence telles que les l valeurs sensibles les plus fréquentes aient une fréquence d'apparition comparable. Dans une table l-diverse, la recherche de la valeur sensible d'un individu est compliquée par la diversité des valeurs sensibles dans la classe d'équivalence, même si l'adversaire a des infomations sur les valeurs des attributs quasi-identifiants.

Cependant, les auteurs de [29] pointent trois faiblesses de la l-diversité. Tout d'abord, il peut ne pas être nécessaire ou difficile d'appliquer la l-diversité sur une table. Dans le cas où une table à un attribut sensible à deux valeurs possibles, demander que la table soit l-diverse obligerait à avoir les deux valeurs sensibles en proportions égales dans chaque classe d'équivalence. Cela entraînerait une chute drastique de l'utilité des données.

Ensuite, une *attaque asymétrique* est possible sur une table l-diverse. Supposons qu'une table ait un attribut sensible à deux valeurs possibles. Supposons que l'on connaisse la répartition des deux valeurs sensibles dans la table : la première valeur est présente dans 99% des enregistrements et la seconde dans 1% des enregistrements. Après publication d'une version l-diverse de la table, supposons qu'un adversaire sache dans quelle classe d'équivalence se trouve l'enregistrement d'un individu. Comme mentionné dans le premier point, 50% des enregistrements de la classe ont la première valeur sensible et 50% des enregistrements ont la seconde valeur sensible. Ainsi, alors qu'un adversaire n'avait qu'une probabilité de 0,1 de savoir que l'individu avait la première valeur sensible dans la table d'origine, il connaît avec une probabilité 0,5 la valeur sensible d'un individu dans la table 2-diverse.

Finalement, si les valeurs sensibles sont sémantiquement proches, un adversaire peut tenter une *attaque par similarité*. Dans une table l-diverse, bien que les valeurs sensibles soient équitablement réparties au sein d'une classe d'équivalence, il peut arriver que les valeurs sensibles soient sémantiquement proches (par exemple, l'infarctus du myocarde, l'insuffisance cardiaque et les accidents vasculaires cérébraux sont trois maladies cardiovasculaires). Dans ce cas, la connaissance de l'appartenance d'un individu à cette classe d'équivalence engendrerait une divulgation d'informations sensibles.

Dans l'exemple 1.2.1, nous illustrons la faiblesse de la k-anonymité évoquée précédemment et nous donnons un exemple d'attaque par similarité d'une table l-diverse.

Exemple 1.2.1

La table présentée en figure 1.1a page suivante a huit enregistrements et cinq attributs. L'attribut *Prénom* est un attribut identifiant. Les attributs *Genre*, *Code postal* et *Age* sont des attributs quasi-identifiants et l'attribut *Maladie* est un attribut sensible.

La table en figure 1.1b page suivante est une version 2-anonyme de la table en figure 1.1a page suivante. Une étape de pseudonymisation a été effectuée : l'attribut identifiant *Prénom* a été supprimé. De plus, les enregistrements de la table ont été regroupés en quatre classes d'équivalence contenant chacune deux enregistrements. La

Identifiant	Quasi-identifiants			Sensible
Prénom	*Genre*	*Code postal*	*Age*	*Maladie*
Ana	F	53713	21	Cancer de l'estomac
Béa	F	53712	29	Cancer de l'estomac
Carole	F	53713	32	Gastrite
Daphné	F	53712	36	Ulcère de l'estomac
Éric	M	53701	25	Maladie de cœur
Fred	M	53700	39	Maladie de cœur
Gui	M	53701	42	Infection virale
Hervé	M	53700	50	Gastrite

(a) Une table à cinq attributs : un identifiant, trois quasi-identifiants et un sensible.

	Quasi-identifiants			Sensible
	Genre	*Code postal*	*Age*	*Maladie*
Classe 1	F	5371*	[20,30]	Cancer de l'estomac
	F	5371*	[20,30]	Cancer de l'estomac
Classe 2	F	5371*	[30,40]	Gastrite
	F	5371*	[30,40]	Ulcère de l'estomac
Classe 3	*	5370*	[20,40]	Maladie de cœur
	*	5370*	[20,40]	Maladie de cœur
Classe 4	M	5370*	[40,50]	Infection virale
	M	5370*	[40,50]	Gastrite

(b) Une table 2-anonyme à quatre classes d'équivalence. Les enregistrements de la première classe d'équivalence ont la même valeur pour l'attribut *Maladie*.

	Quasi-identifiants			Sensible
	Genre	*Code postal*	*Age*	*Maladie*
Classe 1	F	5371*	[20,40]	Cancer de l'estomac
	F	5371*	[20,40]	Cancer de l'estomac
	F	5371*	[20,40]	Gastrite
	F	5371*	[20,40]	Ulcère de l'estomac
Classe 2	*	5370*	[20,50]	Maladie de cœur
	*	5370*	[20,50]	Maladie de cœur
	*	5370*	[20,50]	Infection virale
	*	5370*	[20,50]	Gastrite

(c) Une table 3-diverse. Dans les deux classes d'équivalence, il y a trois valeurs distinctes de l'attribut sensible bien représentées. En revanche, dans la classe d'équivalence orange, les valeurs sensibles des enregistrements font toutes référence à une maladie de l'estomac.

FIGURE 1.1 – Exemples de tables illustrant les modèles de k-anonymité et de l-diversité et leurs limites

technique utilisée est la généralisation (cf. section 1.3 page ci-contre). Les valeurs de l'attribut sensible *Maladie* n'ont pas été modifiées.

Bien que la table en figure 1.1b soit 2-anonyme, on constate un manque de diversité dans les valeurs de l'attribut sensible *Maladie*. En effet, les deux premiers enregistrements de la table forment une classe d'équivalence mais leur valeur pour l'attribut *Maladie* sont identiques. Si un adversaire sait que l'enregistrement d'Ana appartient à cette classe d'équivalence (car il a connaissance de ses valeurs pour les attributs quasi-identifiants), il pourra en déduire qu'Ana souffre d'un cancer de l'estomac. La classe d'équivalence contenant les enregistrements 5 et 6 illustre le même manque de diversité dans les valeurs de l'attribut sensible.

La table en figure 1.1c est une version 3-diverse de la table en figure 1.1a. En effet, dans chacune de ses deux classes d'équivalence, on trouve trois valeurs sensibles distinctes. Le problème de manque de diversité est donc résolu.

Bien que la table en figure 1.1c soit 3-diverse, elle illustre une des faiblesses de la l-diversité. En effet, une attaque par similarité est possible sur cette table. Appelons C_1 la classe d'équivalence de la table en figure 1.1c regroupant les quatre premiers enregistrements. On constate que les valeurs de l'attribut sensible *Maladie* dans C_1 font tout référence à une maladie gastrique. Si un adversaire sait que l'enregistrement d'un individu est dans la classe C_1, il pourra en déduire que l'individu souffre d'une maladie touchant l'estomac.

Pour remédier aux faiblesses de la l-diversité, Li et co-auteurs ont présenté dans [29] la t-proximité. Une

connaissance de la répartition des valeurs sensibles dans la table d'origine est indispensable pour utiliser la t-proximité. A l'instar de la l-diversité, la t-proximité fait en sorte que la répartition des valeurs sensibles dans les classes d'équivalence soient homogènes. Elle est en revanche plus contraignante : la distance entre la proportion d'une valeur de l'attribut sensible dans une classe d'équivalence et la proportion de cette valeur dans la table originale ne doit pas dépasser un seuil t. On dit qu'une classe d'équivalence a une t-proximité si chacune de ses valeurs pour l'attribut sensible respecte la contrainte précédente. Une table a une t-proximité si toutes ses classes d'équivalence ont une t-proximité. Ainsi, plus t est petit, plus la protection de la vie privée est importante. Dans [17], trois inconvénients de la t-proximité sont exposés. Premièrement, le niveau de protection est le même pour toutes les valeurs sensibles. Deuxièmement, la mesure de proximité entre les répartitions des valeurs sensibles présentée dans [29] n'est pas adaptée pour lutter contre l'attaque de divulgation d'attribut sur des attributs sensibles numériques. Troisièmement, la t-proximité engendre trop de modifications des données de la table et donc une forte diminution de l'utilité des données de la table.

1.3 Techniques d'anonymisation

Dans cette section, nous allons présenter plusieurs techniques permettant de satisfaire les contraintes d'un modèle d'anonymisation.

1.3.1 La généralisation

Lors de la *généralisation* d'une table [41], les valeurs des attributs quasi-identifiants sont remplacées par des valeurs moins spécifiques afin de regrouper les enregistrements en classes d'équivalence de plus grandes tailles. Pour appliquer la technique de généralisation, il est nécessaire de déterminer préalablement l'ensemble des attributs quasi-identifiants de la table. Pour chaque attribut quasi-identifiant, il faut ensuite construire une hiérarchie de généralisation. Cela n'est pas toujours évident quand le propriétaire de la table ne maîtrise pas le contenu des données de la table. Par exemple, quelqu'un d'étranger à la biologie végétale aura du mal à classifier de manière pertinente différentes espèces de plantes.

Il y a plusieurs façons d'appliquer la technique de généralisation sur une table. Le *recodage global* contraint tous les enregistrements de la table d'origine à appartenir à la même classe d'équivalence généralisée. Au contraire, dans un contexte de *recodage local*, les enregistrements identiques peuvent se retrouver dans des classes d'équivalence généralisées différentes. Par ailleurs, pour chaque attribut quasi-identifiant, un même niveau de généralisation peut être appliqué à tous les enregistrements de la table : on parle de *recodage unidimensionnel*. Quand des niveaux de généralisation différents peuvent être appliqués aux enregistrements pour un attribut quasi-identifiant, on parle de *recodage multi-dimensionnel*.

L'algorithme *Incognito*, développé par Lefevre et co-auteurs [27], est un exemple d'algorithme de k-anonymisation se basant sur une technique de généralisation globale et unidimensionnelle. L'objectif est de déterminer toutes les versions k-anonymes d'une table possibles avec les contraintes de ce type de généralisation. *Mondrian*, développé par Lefevre et co-auteurs [26], est un algorithme de k-anonymisation appliquant le recodage global multi-dimensionnel. Ses performances en termes de conservation de l'utilité des données sont meilleures que celles d'*Incognito*. L'algorithme *KACA* [28] utilise une généralisation de type local multi-dimensionnel. Les enregistrements identiques sont considérés ensemble au début de l'algorithme mais, par des étapes de division de classes d'équivalence, peuvent se retrouver séparés au cours du processus. Les algorithmes se basant sur des méthodes de clustering comme *k-member* [7], *OKA* [33] ou encore *GCCG* [39] utilisent un recodage local multi-dimensionnel. Les classes d'équivalence de la table d'origine ne sont pas prises en compte et chaque enregistrement est traité indépendamment. Ces algorithmes ont de meilleures performances en conservation de l'utilité des données que les algorithmes de recodage global unidimensionnel.

La table de la figure 1.1b page ci-contre illustre l'utilisation de la technique de généralisation. La table est 2-anonyme. Le recodage effectué est global multi-dimensionnel. En effet, tous les enregistrements issus d'une même classe d'équivalence de la table d'origine de la figure 1.1a page précédente sont dans la même classe généralisée de la table 2-anonyme de la figure 1.1b page ci-contre (recodage global). De plus, les valeurs de l'attribut *Genre* ne sont pas toutes généralisées au même niveau : certaines sont restées intactes alors que d'autres ont été généralisées en * (recodage multi-dimensionnel).

1.3.2 La bucketisation et le slicing

La technique de *bucketisation* [48] consiste à regrouper les enregistrements de la table en paquets. Les attributs sont séparés en deux groupes : les attributs quasi-identifiants d'un côté et les attributs sensibles de l'autre. Puis, dans chaque paquet, les valeurs sensibles des enregistrements sont permutées. Comme les valeurs

des attributs quasi-identifiants ne subissent aucune modification, la bucketisation ne fournit aucune protection contre la divulagation d'appartenance. De plus, la séparation entre attributs quasi-identifiants et attributs sensibles doit être clairement déterminée. Finalement, en permutant les valeurs des attributs sensibles dans chaque paquet, les corrélations entre valeurs quasi-identifiantes et valeurs sensibles sont perdues.

Pour remédier aux problèmes précédents, Li et co-auteurs proposent le *slicing* dans [31]. Cette technique effectue un partitionnement vertical sur les attributs et un partitionnement horizontal sur les enregistrements. Les attributs sont regroupés en colonne. Pour calculer le niveau de corrélation entre les attributs, on peut utiliser le coefficient de corrélation de Pearson pour les attributs numériques ou le coefficient de contingence quadratique moyen pour les attributs à catégories. Le partitionnement horizontal regroupe les enregistrements en paquets. Dans [31], les auteurs proposent de modifier l'algorithme *Mondrian* et de l'utiliser pour déterminer les paquets d'enregistrements. Une fois les deux partitionnements effectués, dans chaque paquet, les valeurs des colonnes sont permutées pour casser les corrélations entre les colonnes. Un avantage de la technique de slicing sur la bucketisation est que la séparation entre les types d'attributs n'a pas besoin d'être claire : des attributs quasi-identifiants et sensibles peuvent se retrouver dans la même colonne. De plus, le slicing fournit une protection contre la divulgation d'appartenance.

Dans [45], les auteurs proposent une technique d'anonymisation combinant généralisation et bucketisation.

1.3.3 La suppression et la relocation

La technique de suppression, présentée par Sweeney [43] consiste à supprimer des enregistrements complets ou certaines valeurs des enregistrements de la table. Ces enregistrements contiennent souvent des valeurs uniques ou éloignées sémantiquement de la majorité des enregistrements de la table. Lors de la généralisation d'une table, ces enregistrements entraînent d'importantes généralisations pour respecter les contraintes de la k-anonymité par exemple. En effet, ces valeurs aberrantes sont souvent trop peu nombreuses pour être regroupées dans la même classe d'équivalence et trop éloignées sémantiquement des autres valeurs pour être intégrées à une classe d'équivalence. Dans des ensembles de données avec beaucoup d'attributs, une suppression d'un attribut complet peut engendrer une grande baisse de l'utilité des données.

D'autre part, la *relocation* consiste à modifier les valeurs aberrantes par des valeurs plus proches sémantiquement des valeurs des autres enregistrements. Dans [38], une méthode de généralisation hybride est proposée. Tout d'abord, une étape de *relocation* est effectuée pour effacer les valeurs aberrantes. Pour ce faire, les algorithmes *Incognito* et *Mondrian* peuvent être utilisés. Ensuite, la table est généralisée pour respecter un modèle d'anonymisation.

1.3.4 La micro-agrégation

Lors de l'anonymisation d'une table en utilisant la technique de micro-agrégation [12], des clusters sont construits puis, dans chaque cluster, les valeurs des attributs sont remplacées par des agrégats. L'algorithme *MDAV-generic*, proposé dans [12], donne une méthode pour calculer des clusters de taille supérieure à un entier k. Les auteurs présentent également des opérateurs de moyenne et des formules de distance pour les attributs numériques, ordinaux et nomimaux. Les opérateurs de moyenne servent à calculer les agrégats dans les clusters d'enregistrements, les formules de distance permettent de déterminer les enregistrements dont les valeurs sont proches pour créer les clusters. Par exemple, pour les attributs numériques, la moyenne arithmétique est utilisée comme opérateur de moyenne et la distance Euclidienne est utilisée comme formule de distance. Au contraire de la généralisation, la micro-agrégation ne demande pas de créer de nouvelles catégories pour les attributs ; les valeurs de remplacement sont calculées automatiquement. De plus, la micro-agrégation n'entraîne pas de suppressions de valeurs. Ainsi, les tables k-anonymes produites restent facilement exploitables lors de tâches d'analyse de données.

1.3.5 Exemple d'anonymisation avec différentes techniques

Les tables en figure 1.2 page suivante sont des anonymisations de la table T en figure 1.1a page 8 obtenues avec différentes techniques d'anonymisation.

La table en figure 1.2a page suivante est une anonymisation de T obtenue par bucketisation. Les enregistrements ont été regroupés en deux paquets de quatre enregistrements chacun. Dans chacun des paquets, les valeurs de l'attribut sensible *Maladie* ont été permutées.

La table en figure 1.2b page ci-contre est une anonymisation de T obtenue par slicing. Les attributs ont été regroupés en deux colonnes : *Genre* et *Code postal* d'un côté et *Age* et *Maladie* de l'autre. On voit que le type des attributs n'est pas pris en compte avec cette technique. Les enregistrements ont été regroupés en deux

	Quasi-identifiants			Sensible
	Genre	*Code postal*	*Age*	*Maladie*
Paquet 1	F	53713	21	Cancer de l'estomac
	F	53712	29	Gastrite
	F	53713	32	Ulcère de l'estomac
	F	53712	36	Cancer de l'estomac
Paquet 2	M	53701	25	Gastrite
	M	53700	39	Infection virale
	M	53701	42	Maladie de cœur
	M	53700	50	Maladie de cœur

(a) Une table anonymisée avec la bucketisation : deux paquets sont construits et les valeurs de l'attribut sensible sont permutées.

	Colonne 1 (*Genre*, *Age*)	Colonne 2 (*Code postal*, *Maladie*)
Paquet 1	(F, 21)	(53712, Ulcère de l'estomac)
	(F, 29)	(53713, Gastrite)
	(F, 32)	(53713, Cancer de l'estomac)
	(F, 36)	(53712, Cancer de l'estomac)
Paquet 2	(M, 25)	(53701, Infection virale)
	(M, 39)	(53701, Maladie de cœur)
	(M, 42)	(53700, Maladie de cœur)
	(M, 50)	(53700, Gastrite)

(b) Une table anonymisée avec la technique de *slicing* : deux colonnes d'attributs et deux paquets d'enregistrements sont construits puis les valeurs de la seconde colonne sont permutées.

	Quasi-identifiants			
	Genre	Code postal	Age	Maladie
Cluster 1	F	53712	29,5	Cancer de l'estomac
	F	53712	29,5	Cancer de l'estomac
	F	53712	29,5	Cancer de l'estomac
	F	53712	29,5	Cancer de l'estomac
Cluster 2	M	53700	39	Maladie de cœur
	M	53700	39	Maladie de cœur
	M	53700	39	Maldie de cœur
	M	53700	39	Maladie de cœur

(c) Une table anonymisée avec la micro-agrégation : deux clusters sont construits et les valeurs quasi-identifiantes des clusters sont remplacées par des agrégats.

FIGURE 1.2 – Exemples de tables anonymisées avec différentes techniques d'anonymisation

paquets de quatre enregistrements chacun. Finalement, les valeurs de la seconde colonne ont été permutées. Les corrélations entre les colonnes ont été supprimées lors du slicing.

La table en figure 1.2c est une version 4-anonyme de T obtenue par micro-agrégation. On considère pour cet exemple que tous les attributs sont des quasi-identifiants. Les enregistrements ont été regroupés en deux clusters. Puis, les valeurs des attributs des clusters ont été remplacées par des agrégats. Les valeurs de l'attribut *Genre* sont identiques pour tous les enregistrements d'un même cluster, elles ne sont donc pas modifiées. L'attribut *Code postal* est ordinal. On remplace donc ses valeurs par la valeur médiane dans chaque cluster. Dans cet exemple, il n'y a que deux codes postaux présents en même quantité dans chaque cluster. On remplace les codes postaux par le plus petit des codes postaux dans chaque cluster (pour le second cluster, on met 53700 pour les quatre enregistrements). L'attribut *Age* est numérique. On remplace donc ses valeurs par la valeur moyenne dans chaque cluster. Dans le premier cluster, on fait la moyenne arithmétique des valeurs 21, 29, 32 et 36 : l'âge est remplacé par 29,5 dans les quatre enregistrements. L'attribut *Maladie* est nominal. On remplace donc ses valeurs par la valeur la plus fréquente dans chaque cluster. Pour le premier cluster, on remplace les maladies par Cancer de l'estomac et pour le second cluster, on remplace les maladies par Maladie de cœur.

Chapitre 2

Définitions et notations

Sommaire du présent chapitre

Dans ce chapitre, nous allons revenir sur les notions évoquées dans le chapitre 1 page 5. Ces définitions et notations nous serviront dans la suite de ce manuscrit. Nous commencerons par donner une définition d'attribut, d'enregistrement et de table en section 2.1. Puis nous reviendrons sur la technique de généralisation et nous présenterons les hiérarchies de généralisation en section 2.2 page 15. Dans la section 2.3 page 18, nous définirons la notion de table généralisée. Nous expliciterons également les notions de généralisation d'enregistrements et de généralisation de sous-ensembles d'une table. Finalement, en section 2.4 page 22, nous présenterons deux tables publiques sur lesquelles nous mènerons des expérimentations dans les chapitres suivants.

2.1 Attributs, table et classes d'équivalence

Dans cette première section, nous allons revenir sur les définitions d'attribut, d'enregistrement et de table.

2.1.1 Attributs

Définition 2.1.1 (Attribut et ensemble d'attributs)
Un *attribut* est un ensemble de valeurs qualitatives ou quantitatives.

Soit A un attribut, on note $n_A \in \mathbb{N}^*$ le cardinal de A et $a_1, \ldots, a_{n_A}$ les éléments de A.

Soient $A_1, \ldots, A_m$ $m \in \mathbb{N}^*$ attributs, on note $\mathcal{A}$ l'ensemble de ces attributs.

Par exemple, l'attribut $A_1 = \{Chat, Lion, Chien, Dauphin\}$ est un attribut qualitatif contenant des races d'animaux et $A_2 = [\![0, 100]\!]$ est un attribut quantitatif représentant un âge entre 0 et 100.

Comme mentionné dans le chapitre 1 page 5, il existe trois types d'attributs : identifiant, quasi-identifiant et sensible. Les attributs identifiants sont supprimés lors d'un processus d'anonymisation. Nous ne les ferons donc pas apparaître dans un ensemble d'attributs. Nous noterons $\mathcal{A} = \{Q_1, \ldots, Q_m, S_1, \ldots, S_t\}$ avec $Q_1, \ldots, Q_m$ $m \in \mathbb{N}^*$ attributs quasi-identifiants et $S_1, \ldots, S_t$ $t \in \mathbb{N}^*$ attributs sensibles. Pour simplifier la suite des travaux, nous considérons que l'ensemble des attributs sensibles est représenté par un unique attribut S (chaque valeur de l'attribut S sera un t-uplet de valeurs sensibles des S_i). Nous aurons donc $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$.

Quand le type des attributs n'est pas indispensable, nous considérerons $\mathcal{A} = \{A_1, \ldots, A_m\}$.

T	Q	S
E^1	q_3	s_1
E^2	q_2	s_2
E^3	q_3	s_2
E^4	q_1	s_1
E^5	q_1	s_2

FIGURE 2.1 – Représentation d'une table sur $\mathcal{A} = \{Q, S\}$.

Définition 2.1.2 (Enregistrement)
Soit $\mathcal{A} = \{A_1, \dots, A_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs.

Un *enregistrement sur* $\mathcal{A}$ est un m-uplet, noté $E = (e_1, \dots, e_m)$, tel que l'élément e_j soit une valeur de A_j pour tout j dans $[\![1, m]\!]$.

On note $\mathcal{E}_\mathcal{A} = \{(e_1, \dots, e_m) : e_j \in A_j \ \forall j \in [\![1, m]\!]\}$ l'ensemble des enregistrements sur $\mathcal{A}$.

Remarque 2.1.1
Pour $\mathcal{A} = \{A_1, \dots, A_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs, $\mathcal{E}_\mathcal{A}$ a pour cardinal $|\mathcal{E}_\mathcal{A}| = \prod_{j=1}^{m} n_{A_j}$.

Par exemple, considérons l'ensemble d'attributs $\mathcal{A} = \{A_1, A_2\}$ avec $A_1 = \{\textit{Chat}, \textit{Lion}, \textit{Chien}, \textit{Dauphin}\}$ et $A_2 = [\![0, 100]\!]$. $(\textit{Dauphin}, 4)$ et $(\textit{Lion}, 25)$ sont des enregistrements sur $\mathcal{A}$ correspondant à un dauphin de 4 ans et à un lion de 25 ans. Le nombre d'enregistrements sur $\mathcal{A}$ est de

$$|\mathcal{E}_\mathcal{A}| = \prod_{j=1}^{2} n_{A_j} = 4 \times 101 = 404.$$

2.1.2 Table et classes d'équivalence

Définition 2.1.3 (Table sur $\mathcal{A}$)
Soit $\mathcal{A} = \{A_1, \dots, A_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs.

Une *table sur* $\mathcal{A}$ est une combinaison avec répétition d'enregistrements sur $\mathcal{A}$.

Soit $n \in \mathbb{N}^*$. On note $\mathcal{T}_\mathcal{A}^n$ l'ensemble des tables sur $\mathcal{A}$ à n éléments.

Quand il n'y a pas d'ambiguïté, nous parlerons simplement de *table* sans préciser l'ensemble d'attributs considéré.

Remarque 2.1.2
Avec les notations de la définition 2.1.3, $\mathcal{T}_\mathcal{A}^n$ a pour cardinal $|\mathcal{T}_\mathcal{A}^n| = \binom{|\mathcal{E}_\mathcal{A}|+n-1}{n} = \binom{|\mathcal{E}_\mathcal{A}|+n-1}{|\mathcal{E}_\mathcal{A}|-1}$.

Exemple 2.1.1
Soit $\mathcal{A} = \{Q, S\}$ un ensemble d'attributs avec $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant et $S = \{s_1, s_2\}$ un attribut sensible. La figure 2.1 est une représentation d'une table T sur $\mathcal{A} = \{Q, S\}$. La table T a cinq enregistrements E^1, E^2, E^3, E^4 et E^5 tels que $E^1 = (q_3, s_1)$, $E^2 = (q_2, s_2)$, $E^1 = (q_3, s_2)$, $E^1 = (q_1, s_1)$ et $E^1 = (q_1, s_2)$.

Définition 2.1.4 (Projection de $\mathcal{E}_\mathcal{A}$ dans $\mathcal{E}_\mathcal{Q}$)
Soit $\mathcal{A} = \{Q_1, \dots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit $\mathcal{Q} = \{Q_1, \dots, Q_m\}$.

On définit la projection $\pi_\mathcal{Q}$ de $\mathcal{E}_\mathcal{A}$ dans $\mathcal{E}_\mathcal{Q}$ par

$$\begin{array}{rccc} \pi_\mathcal{Q} : & \mathcal{E}_\mathcal{A} & \longrightarrow & \mathcal{E}_\mathcal{Q} \\ & (q_1, \dots, q_m, s) & \longmapsto & (q_1, \dots, q_m) \end{array}.$$

Dans l'exemple 2.1.1, la projection dans $\mathcal{E}_{\{Q\}}$ de l'enregistrement $E^1 = (q_3, s_1)$ de la table T est l'enregistrement sur $\{Q\}$ (q_3). De même, la projection sur $\{Q\}$ de $E^3 = (q_3, s_2)$ est (q_3).

La projection $\pi_\mathcal{Q}$ restreint les enregistrements aux attributs quasi-identifiants.

Définissons maintenant l'ensemble des classes d'équivalence d'une table (cf. définition 2.1.5).

Définition 2.1.5 (Relation d'équivalence)
Soit $\mathcal{A} = \{Q_1, \dots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit T une table sur $\mathcal{A}$ à $n \in \mathbb{N}^*$ enregistrements.

Soient $E = (e_1, \ldots, e_m, s)$ et $E' = (e'_1, \ldots, e'_m, s')$ deux enregistrements de T.
On définit la relation d'équivalence $\sim$ sur T comme suit :

$$E \sim E' \Leftrightarrow \pi_{\mathcal{Q}}(E) = \pi_{\mathcal{Q}}(E').$$

On note $\mathcal{C}(T) = T/_{\sim}$ le quotient de T par la relation d'équivalence $\sim$.

Démonstration : Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit T une table sur $\mathcal{A}$ à $n \in \mathbb{N}^*$ enregistrements.

Montrons que la relation $\sim$ est une relation d'équivalence sur T. Il faut montrer que $\sim$ est réfexive, symétrique et transitive.

Réflexive. Soit $E = (e_1, \ldots, e_m, s)$ un enregistrement de T. On a $\pi_{\mathcal{Q}}(E) = (e_1, \ldots, e_m)$. Donc $E \sim E$.

Symétrique. Soient E et E' deux enregistrements de T. Supposons que $E \sim E'$ et montrons que $E' \sim E$. $E \sim E'$ donc $\pi_{\mathcal{Q}}(E) = \pi_{\mathcal{Q}}(E')$. On a alors $\pi_{\mathcal{Q}}(E') = \pi_{\mathcal{Q}}(E)$ et donc $E' \sim E$.

Transitive. Soient E, E' et E'' trois enregistrements de T. Montrons que si $E \sim E'$ et $E' \sim E''$ alors $E \sim E''$. On a $E \sim E'$ donc $\pi_{\mathcal{Q}}(E) = \pi_{\mathcal{Q}}(E')$. De même, on a $E' \sim E''$ donc $\pi_{\mathcal{Q}}(E') = \pi_{\mathcal{Q}}(E'')$. Par la transitivité de l'égalité, on a $\pi_{\mathcal{Q}}(E) = \pi_{\mathcal{Q}}(E'')$ et donc $E \sim E''$. ■

La relation $\sim$ sur une table T regroupe les enregistrements de T en fonction de leurs valeurs pour les attributs quasi-identifiants. Les valeurs des attributs sensibles ne sont pas considérées lors de la construction des classes d'équivalence de cette relation.

Une classe d'équivalence est un sous-ensemble d'enregistrements de T.

Dans la suite du manuscrit, quand nous parlerons de classes d'équivalence d'un ensemble d'enregistrements, nous sous-entendrons qu'il s'agit de la relation $\sim$.

Nous allons maintenant associer à chaque classe d'équivalence un unique enregistrement sur l'ensemble des attributs quasi-identifants. Cet enregistrement sera appelé *représentant sur l'ensemble des quasi-identifiants* de la classe d'équivalence et permettra de caractériser la classe d'équivalence.

Définition 2.1.6 (Représentant sur $\mathcal{Q}$ d'une classe d'équivalence)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit T une table sur $\mathcal{A}$ à $n \in \mathbb{N}^*$ enregistrements.

On définit l'application $\text{repr}_{\mathcal{Q}} : \mathcal{C}(T) \to \mathcal{E}_{\mathcal{Q}}$ qui à toute classe d'équivalence de T associe son représentant sur $\mathcal{Q}$:

$$\begin{array}{rccc} \text{repr}_{\mathcal{Q}} : & \mathcal{C}(T) & \longrightarrow & \mathcal{E}_{\mathcal{Q}} \\ & C & \longmapsto & \text{repr}_{\mathcal{Q}}(C) \end{array},$$

avec $\text{repr}_{\mathcal{Q}}(C)$ tel que, pour tout $E \in C$, $\pi_{\mathcal{Q}}(E) = \text{repr}_{\mathcal{Q}}(C)$.

Exemple 2.1.2
Soit $\mathcal{A} = \{Q, S\}$ un ensemble d'attributs avec $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant et $S = \{s_1, s_2\}$ un attribut sensible. On pose $\mathcal{Q} = \{Q\}$. Soit T une table sur $\mathcal{A}$ dont une représentation est donnée en figure 2.1 page ci-contre. La table T a cinq enregistrements E^1, E^2, E^3, E^4 et E^5 tels que $E^1 = (q_3, s_1)$, $E^2 = (q_2, s_2)$, $E^3 = (q_3, s_2)$, $E^4 = (q_1, s_1)$ et $E^5 = (q_1, s_2)$.

L'ensemble des classes d'équivalence de T est $\mathcal{C}(T) = \{C_{q_1}, C_{q_2}, C_{q_3}\}$ avec $C_{q_1} = \{E^4, E^5\}$, $C_{q_2} = \{E^2\}$ et $C_{q_3} = \{E^1, E^3\}$. Le représentant sur $\mathcal{Q}$ de C_{q_1} est $\text{repr}_{\mathcal{Q}}(C_{q_1}) = (q_1)$. Cela signifie que tout enregistrement dans C_{q_1} a q_1 pour valeur pour l'attribut Q. Le représentant sur $\mathcal{Q}$ de C_{q_2} est $\text{repr}_{\mathcal{Q}}(C_{q_2}) = (q_2)$ et le représentant sur $\mathcal{Q}$ de C_{q_3} est $\text{repr}_{\mathcal{Q}}(C_{q_3}) = (q_3)$.

Sur la figure 2.2 page suivante, nous avons représenté la table T en mettant en évidence les classes d'équivalence à l'aide de couleurs. Les enregistrements de la classe d'équivalence C_{q_1} sont en bleu, les enregistrements de C_{q_2} sont en vert et les enregistrements de C_{q_3} sont en rouge.

2.2 Hiérarchies de généralisation

Dans cette section, nous allons présenter la généralisation, une technique permettant d'anonymiser des tables. La généralisation consiste à remplacer les valeurs des attributs quasi-identifiants des enregistrements jusqu'à ce que la table respecte un modèle d'anonymisation. Par exemple, pour atteindre la k-anonymité, les valeurs des quasi-identifiants des enregistrements doivent être modifiées pour que chaque classe d'équivalence de la table soit de taille supérieure à k. La généralisation ne modifie pas les valeurs des attributs sensibles et conserve l'intégrité de la table (il n'y a pas de suppression d'enregistrements ni d'ajout d'enregistrements bruités).

T	Q	S
E^1	q_3	s_1
E^2	q_2	s_2
E^3	q_3	s_2
E^4	q_1	s_1
E^5	q_1	s_2

FIGURE 2.2 – Représentation d'une table sur $\mathcal{A}$ avec mise en évidence des classes d'équivalence. C_{q_1} est en bleu, C_{q_2} est en vert et C_{q_3} est en rouge.

Les valeurs généralisées sont déterminées sémantiquement par le propriétaire de la table en fonction de l'attribut quasi-identifiant. Pour représenter les généralisations possibles d'une valeur d'un attribut quasi-identifiant, le propriétaire de la table construit une *hiérarchie de généralisation*. Cette hiérarchie de généralisation est un arbre dont les feuilles sont les valeurs de l'attribut dans la table d'origine et la racine symbolise la perte totale d'information (représentée par * par exemple). Plus une valeur de la hiérachie est proche de la racine, plus son niveau de généralisation est élevé et moins elle contient d'information.

Dans la suite de cette section, nous ferons un rappel de théorie des graphes [4] pour définir une hiérarchie de généralisation d'un attribut quasi-identifiant. Nous définirons la hauteur d'une hiérarchie de généralisation ainsi que le niveau d'un nœud. Nous définirons également la notion de table sur un couple constitué d'un ensemble d'attributs et d'un vecteur de hiérarchies de généralisation des attributs quasi-identifiants.

Rappel 2.2.1 (Théorie des graphes)
Un *arbre* est un graphe acyclique et connexe. Les sommets d'un arbre sont appelés *nœuds*. Une *feuille* d'un arbre est un nœud de degré 1. Une *arborescence* est un arbre comportant un nœud particulier r nommé *racine* de l'arborescence, à partir duquel il existe un chemin unique vers tous les autres nœuds.

Définition 2.2.1 (Arborescence)
Soit $\mathscr{A}$ une arborescence.

On appelle *hauteur de l'arborescence* $\mathscr{A}$, notée $h_{\mathscr{A}}$, le nombre maximal de nœuds dans l'ensemble des chemins de $\mathscr{A}$.

Pour $l \in [\![0, h_{\mathscr{A}} - 1]\!]$, on appelle *niveau l de l'arborescence* $\mathscr{A}$ l'ensemble de nœuds de $\mathscr{A}$ défini de la manière itérative suivante : le niveau 0 de l'arborescence $\mathscr{A}$ est l'ensemble des feuilles de $\mathscr{A}$; si $l \geqslant 1$, le niveau l de l'arborescence $\mathscr{A}$ est l'ensemble des nœuds possédant au moins un enfant au niveau $l-1$ et ses enfants dans les niveaux 0 à $l-1$.

Pour $l \in [\![0, h_{\mathscr{A}} - 1]\!]$, le nombre de nœuds du niveau l de l'arborescence est noté $\mathrm{nn}(l)$.

Définissons maintenant une hiérarchie de généralisation d'un attribut quasi-identifiant (cf. définition 2.2.2).

Définition 2.2.2 (Hiérarchie de généralisation)
Soit Q un attribut quasi-identifiant.

Une *hiérarchie de Q* est une arborescence dont les feuilles sont exactement les valeurs de Q et telle que pour tout niveau l allant de 1 à la hauteur de l'arborescence - 1, pour tout nœud v du niveau l, v est tel que pour tout nœud v' dans le sous-arbre enraciné en v, le nombre de feuilles du sous-arbre enraciné en v est strictement supérieur au nombre de feuilles du sous-arbre enraciné en v'.

On note $\mathcal{H}_Q$ l'ensemble des hiérarchies de Q.

Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. On note $\mathcal{H}_{\mathcal{Q}} = \{(H_1, \ldots, H_m) : H_j \in \mathcal{H}_{Q_j}\ \forall j \in [\![1, m]\!]\}$ l'ensemble des hiérarchies de $\mathcal{Q}$.

Remarque 2.2.1
Les conditions de la définition 2.2.2 de hiérarchie de généralisation imposent que tout nœud interne a au moins deux fils.

En effet, si un nœud v n'a qu'un fils v' alors le nombre de feuilles dans le sous-arbre enraciné en v est égal au nombre de feuilles dans le sous-arbre enraciné en v'.

Sur la figure 2.3 page ci-contre, nous proposons un exemple de ce que nous voulons éviter dans la construction des hiérarchies de généralisation. Le racine r de l'arbre n'a qu'un fils v. Le nœud v a deux fils, f_1 et f_2, qui sont des feuilles de l'arbre. Cet arbre ne correspond pas à notre définition d'une hiérarchie de généralisation.

Définition 2.2.3 (Hauteur, niveaux et nœuds)
Soient Q un attribut quasi-identifiant et $H \in \mathcal{H}_Q$.

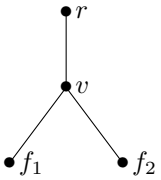


FIGURE 2.3 – Exemple d'arbre qui n'est pas une hiérarchie de généralisation.

On appelle *hauteur de la hiérarchie* H, notée h_H, la hauteur de l'arborescence associée.

Pour $l \in [\![0, h_H - 1]\!]$, on appelle *niveau* l *de la hiérarchie* H le niveau l de l'arborescence associée.

Pour $l \in [\![0, h_H - 1]\!]$, le nombre de nœuds du niveau l de la hiérarchie H est noté $\mathrm{nn}(l)$.

Pour chaque niveau $l \in [\![0, h_H - 1]\!]$, on construit un vecteur de nœuds $(v^j_{l,1}, \ldots, v^j_{l,\mathrm{nn}(l)})$ en lisant les nœuds du niveau l de gauche à droite.

Le *niveau d'un nœud* v de H est noté $\mathrm{niv}(v)$.

Exemple 2.2.1
Soit $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant. Considérons H une hiérarchie de généralisation de Q dont une représentation est donnée en figure 2.4.

H est constituée de trois feuilles (q_1, q_2 et q_3), d'un nœud de niveau 1 ($q_{1,2}$ relié à q_1 et q_2) et d'une racine ($q_{1,2,3}$ relié à q_3 et $q_{1,2}$). Sa hauteur est $h_H = 3$.

Au niveau 1 de la H, il y a un nœud donc $\mathrm{nn}(1) = 1$. Le vecteur de nœuds du niveau 0 de H est (q_1, q_2, q_3). Le niveau du nœud $q_{1,2,3}$ est $\mathrm{niv}(q_{1,2,3}) = 2$.

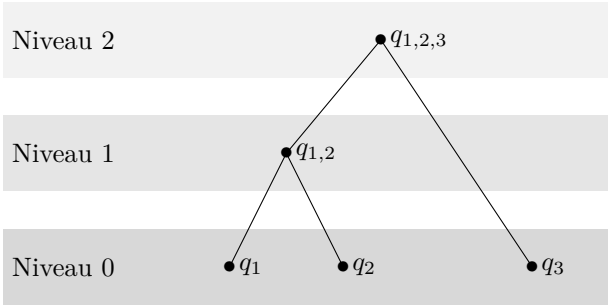


FIGURE 2.4 – Représentation d'une hiérarchie sur $Q = \{q_1, q_2, q_3\}$.

Grâce à la notion de hiérarchie de généralisation d'un attribut quasi-identifiant, nous définissons les enregistrements généralisés comme des enregistrements pouvant avoir comme valeurs quasi-identifiantes des nœuds internes des hiérarchies de généralisation (cf. définition 2.2.4).

Définition 2.2.4 (Enregistrement généralisé)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$.

Un *enregistrement généralisé sur* $(\mathcal{A}, \mathcal{H})$ est un $(m+1)$-uplet, noté $F = (f_1, \ldots, f_m, s)$, tel que l'élément f_j soit un nœud de H_j pour tout j dans $[\![1, m]\!]$ et s soit une valeur de S.

On note $\mathcal{F}_{(\mathcal{A},\mathcal{H})} = \{(f_1, \ldots, f_m, s) : f_j \in H_j \ \forall j \in [\![1, m]\!] \text{ et } s \in S\}$ l'ensemble des enregistrements généralisés sur $(\mathcal{A}, \mathcal{H})$.

Remarque 2.2.2
Avec les notations des définitions 2.1.2 page 14 et 2.2.4, on a l'inclusion $\mathcal{E}_\mathcal{A} \subset \mathcal{F}_{(\mathcal{A},\mathcal{H})}$.

En effet, les valeurs d'un attribut quasi-identifiant sont exactement les feuilles de ses hiérarchies de généralisation. Ainsi, tout enregistrement sur $\mathcal{A}$ est un enregistrement sur $(\mathcal{A}, \mathcal{H})$.

Définition 2.2.5 (Table sur $(\mathcal{A}, \mathcal{H})$)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$.

Une *table sur* $(\mathcal{A}, \mathcal{H})$ est une combinaison avec répétition d'enregistrements généralisés sur $(\mathcal{A}, \mathcal{H})$.

Pour $n \in \mathbb{N}^*$, $\mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ est l'ensemble des tables sur $(\mathcal{A}, \mathcal{H})$ à n enregistrements.

Soit T une table sur $(\mathcal{A}, \mathcal{H})$ à $n \in \mathbb{N}^*$ enregistrements. On note $\mathcal{C}(T) = T/_\sim$ le quotient de T par la relation d'équivalence $\sim$ donnée dans le définition 2.1.5 page 14.

T	Q	S
F^1	$q_{1,2}$	s_1
F^2	q_1	s_1
F^3	q_3	s_2

FIGURE 2.5 – Représentation d'une table sur $(\mathcal{A}, \mathcal{H})$.

Exemple 2.2.2
Soit $\mathcal{A} = \{Q, S\}$ un ensemble d'attributs avec $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant et $S = \{s_1, s_2\}$ un attribut sensible. Considérons la hiérarchie H de Q donnée en la figure 2.4 page précédente. H est constituée de trois feuilles (q_1, q_2 et q_3), d'un nœud de niveau 1 ($q_{1,2}$ relié à q_1 et q_2) et d'une racine ($q_{1,2,3}$ relié à q_3 et $q_{1,2}$).

La figure 2.5 est une représentation d'une table T sur $(\mathcal{A}, \mathcal{H})$. T contient trois enregistrements généralisés sur $(\mathcal{A}, (H))$ F^1, F^2 et F^3 tels que $F^1 = (q_{1,2}, s_1)$, $F_2 = (q_1, s_1)$ et $F^3 = (q_3, s_2)$.

Remarque 2.2.3
Avec les notations de la définition 2.2.5 page précédente et la remarque 2.2.2 page précédente, on a

$$\mathcal{T}^n_{\mathcal{A}} \subseteq \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}.$$

En effet, soit $T = \{E^1, \ldots, E^n\}$ une table sur $\mathcal{A}$ à $n \in \mathbb{N}^*$ enregistrements. D'après la remarque 2.2.2 page précédente, si $E \in \mathcal{E}_{\mathcal{A}}$ alors $E \in \mathcal{F}_{(\mathcal{A},\mathcal{H})}$. Donc pour tout i dans $[\![1, n]\!]$, $E^i \in \mathcal{F}_{(\mathcal{A},\mathcal{H})}$. Donc T est une table sur $(\mathcal{A}, \mathcal{H})$.

Comme pour les tables sur un ensemble d'attributs, nous avons défini les classes d'équivalence de la relation $\sim$ sur une table sur $(\mathcal{A}, \mathcal{H})$ avec $\mathcal{A}$ un ensemble d'attributs et $\mathcal{H}$ un vecteur de hiérarchies des attributs quasi-identifiants. Nous voulons maintenant définir de la même façon le représentant sur $\mathcal{Q}$ d'une classe d'équivalence, avec $\mathcal{Q}$ l'ensemble des attributs quasi-identifiants de $\mathcal{A}$. Nous commençons par étendre la projection $\pi_{\mathcal{Q}}$ de la définition 2.1.4 page 14 de $\mathcal{F}_{(\mathcal{A},\mathcal{H})}$ dans $\mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ (cf. définition 2.2.6).

Définition 2.2.6 (Projection de $\mathcal{F}_{(\mathcal{A},\mathcal{H})}$ dans $\mathcal{F}_{(\mathcal{Q},\mathcal{H})}$)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit $\{Q_1, \ldots, Q_m\}$.

On étend la définition de la projection $\pi_{\mathcal{Q}}$ de $\mathcal{F}_{(\mathcal{A},\mathcal{H})}$ dans $\mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ par

$$\begin{array}{rccc} \pi_{\mathcal{Q}} : & \mathcal{F}_{(\mathcal{A},\mathcal{H})} & \longrightarrow & \mathcal{F}_{(\mathcal{Q},\mathcal{H})} \\ & (q_1, \ldots, q_m, s) & \longmapsto & (q_1, \ldots, q_m) \end{array}.$$

Définition 2.2.7 (Représentant sur $\mathcal{Q}$ d'une classe d'équivalence)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$.

On définit l'application $\text{repr}_{\mathcal{Q}} : \mathcal{C}(T) \to \mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ qui à toute classe d'équivalence de T associe son représentant sur $\mathcal{Q}$:

$$\begin{array}{rccc} \text{repr}_{\mathcal{Q}} : & \mathcal{C}(T) & \longrightarrow & \mathcal{F}_{(\mathcal{Q},\mathcal{H})} \\ & C & \longmapsto & \text{repr}_{\mathcal{Q}}(C) \end{array},$$

avec $\text{repr}_{\mathcal{Q}}(C)$ tel que, pour tout $E \in C$, $\pi_{\mathcal{Q}}(E) = \text{repr}_{\mathcal{Q}}(C)$.

2.3 Table généralisée et généralisation de sous-ensembles d'enregistrements

Dans cette section, nous allons définir la notion de table généralisée sur une table d'origine : une telle table contiendra des généralisations des enregistrements de la table d'origine (cf. section 2.3.1 page suivante). De plus, dans la section 2.3.2 page 21, nous allons expliquer comment généraliser des sous-ensembles d'enregistrements d'une table en utilisant la notion de plus petit ancêtre commun. Ces méthodes seront utilisées dans des algorithmes d'anonymisation se basant sur la technique de généralisation que nous présenterons dans les chapitres 3 page 25 et 4 page 55.

2.3.1 Table généralisée

Avant de définir une table généralisée, nous définissons la généralisation d'un nœud (cf. définition 2.3.1) et la généralisation d'un enregistrement (cf. définition 2.3.2). Dans l'optique de réaliser des démonstrations dans le chapitre 3 page 25 traitant des métriques de perte d'information, nous montrerons également que la relation *généralisation* est transitive pour des nœuds et pour des enregistrements (cf. propriétés 2.3.1 et 2.3.2).

Définition 2.3.1 (Généralisation d'un nœud)
Soient Q un attribut quasi-identifiant et $H \in \mathcal{H}_Q$ une hiérarchie sur Q. Soient v_1 et v_2 deux nœuds de H.

On dit que v_2 *est une généralisation de* v_1 si v_2 appartient au chemin reliant v_1 à la racine r de H.

On dit aussi que v_1 peut se généraliser en v_2.

Propriété 2.3.1 (Transitivité de la relation *généralisation de* pour des nœuds)
Soient Q un attribut quasi-identifiant et $H \in \mathcal{H}_Q$ une hiérarchie sur Q. Soient v, v' et v'' trois nœuds de H.

Si v'' est une généralisation de v' et v' est une généralisation de v alors v'' est une généralisation de v.

Démonstration : Montrons que v'' est une généralisation de v c'est-à-dire v'' appartient au chemin reliant v à la racine r de H.

v'' est une généralisation de v' donc v'' appartient au chemin reliant v' à r et v' est une généralisation de v donc v' appartient au chemin reliant v à r. Par unicité du chemin reliant un nœud à la racine dans une arborescence, v'' appartient au chemin reliant v à r. ■

Définition 2.3.2 (Généralisation d'un enregistrement)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soient $F = (f_1, \ldots, f_m, s)$ et $F' = (f'_1, \ldots, f'_m, s')$ deux enregistrements généralisés sur $(\mathcal{A}, \mathcal{H})$.

On dit que F' *est une généralisation de* F si pour tout j dans $[\![1, m]\!]$, f'_j est une généralisation de f_j et $s' = s$.

Propriété 2.3.2 (Transitivité de la relation *généralisation de* pour des enregistrements)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soient $F = (f_1, \ldots, f_m, s)$, $F' = (f'_1, \ldots, f'_m, s')$ et $F'' = (f''_1, \ldots, f''_m, s'')$ trois enregistrements généralisés sur $(\mathcal{A}, \mathcal{H})$.

Si F'' est une généralisation de F' et F' est une généralisation de F alors F'' est une généralisation de F.

Démonstration : Montrons que F'' est une généralisation de F c'est-à-dire f''_j est une généralisation de f_j pour tout $j \in [\![1, m]\!]$ et $s'' = s$.

Soit $j \in [\![1, m]\!]$. Comme F'' est une généralisation de F', f''_j est une généralisation de f'_j et comme F' est une généralisation de F, f'_j est une généralisation de f_j. Par la propriété 2.3.1 de transitivité de la relation *généralisation de* pour des nœuds, f''_j est une généralisation de f_j.

Comme F'' est une généralisation de F', $s'' = s'$ et comme F' est une généralisation de F, $s' = s$ donc $s'' = s$. ■

Définition 2.3.3 (Table généralisée)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. On pose $T = \{E^1, \ldots, E^n\}$.

Une table T^{gen} sur $(\mathcal{A}, \mathcal{H})$ est une *table généralisée sur* $(T, \mathcal{H})$ si $|T^{gen}| = n$ et pour tout i dans $[\![1, n]\!]$, l'enregistrement F^i de T^{gen} est une généralisation de l'enregistrement E^i de T.

On note $\mathcal{T}^{gen}_{(T,\mathcal{H})} = \{T^{gen} : T^{gen}$ soit une table généralisée sur $(T, \mathcal{H})\}$ l'ensemble des tables généralisées sur $(T, \mathcal{H})$.

Propriété 2.3.3 (Transitivité de la relation *table généralisée sur*)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soient T_1, T_2 et T_3 trois tables sur $(\mathcal{A}, \mathcal{H})$ de cardinaux respectifs $n_1, n_2, n_3 \in \mathbb{N}^*$.

Si T_3 est table généralisée sur $(T_2, \mathcal{H})$ et T_2 est une table généralisée sur $(T_1, \mathcal{H})$ alors T_3 est une table généralisée sur $(T_1, \mathcal{H})$.

Démonstration : Posons $T_1 = \{F_1^1, \ldots, F_1^{n_1}\}$, $T^2 = \{F_2^1, \ldots, F_2^{n_2}\}$ et $T_3 = \{F_3^1, \ldots, F_3^{n_3}\}$. Montrons que T_3 est une table généralisée sur $(T_1, \mathcal{H})$ c'est-à-dire $n_3 = n_1$ et pour tout i dans $[\![1, n_1]\!]$, F_3^i est une généralisation de F_1^i.

T^{gen}	Q	S
F^1	$q_{1,2,3}$	s_1
F^2	$q_{1,2,3}$	s_2
F^3	q_3	s_2
F^4	$q_{1,2}$	s_1
F^5	$q_{1,2}$	s_2

FIGURE 2.6 – Représentation d'une table généralisée sur $(T, \mathcal{H})$ avec mise en évidence des classes d'équivalence. C_1^{gen} est en rouge, C_2^{gen} est en vert et C_3^{gen} est en bleu.

Comme T_3 est une table généralisée sur $(T_2, \mathcal{H})$, $n_3 = n_2$ et comme T_2 est une table généralisée sur $(T_1, \mathcal{H})$, $n_2 = n_1$ donc $n_3 = n_1$.

Soit $i \in [\![1, n_1]\!]$. Comme T_3 est une table généralisée sur $(T_2, \mathcal{H})$, F_3^i est une généralisation de F_2^i et comme T_2 est une table généralisée sur $(T_1, \mathcal{H})$, F_2^i est une généralisation de F_1^i. Par la propriété 2.3.2 page précédente de transitivité de la relation *généralisation de* pour des enregistrements, F_3^i est une généralisation de F_1^i. ■

Une table généralisée est en particulier une table sur $(\mathcal{A}, \mathcal{H})$ avec $\mathcal{A}$ un ensemble d'attributs et $\mathcal{H}$ un vecteur de hiérarchies des attributs quasi-identifiants. Elle possède donc des classes d'équivalence pour la relation $\sim$. À chacune des classes d'équivalence de la table généralisée, nous allons associer le sous-ensemble d'enregistrements de la table d'origine qui ont été regroupés pour créer la classe d'équivalence (cf. définition 2.3.4).

Définition 2.3.4 (Classes d'équivalence de T^{gen} et sous-ensembles de T associés)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. Soit T^{gen} une table généralisée sur $(T, \mathcal{H})$.

Soit $\mathcal{C}(T^{gen}) = \{C_1^{gen}, \ldots, C_p^{gen}\}$ l'ensemble des classes d'équivalence de T^{gen} pour $p \in \mathbb{N}^*$.

Pour tout $i \in [\![1, p]\!]$, on associe à C_i^{gen} le sous-ensemble C_i de T tel que $C_i = \{E^j \in T : j \in [\![1, n]\!] \text{ et } F^j \in C_i^{gen}\}$.

Exemple 2.3.1
Soit $\mathcal{A} = \{Q, S\}$ un ensemble d'attributs avec $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant et $S = \{s_1, s_2\}$ un attribut sensible. On pose $\mathcal{Q} = \{Q\}$.

Considérons la hiérarchie H de Q donnée en la figure 2.4 page 17. H est constituée de trois feuilles (q_1, q_2 et q_3), d'un nœud de niveau 1 ($q_{1,2}$ relié à q_1 et q_2) et d'une racine ($q_{1,2,3}$ relié à q_3 et $q_{1,2}$).

Soit T une table sur $(\mathcal{A}, \mathcal{H})$ dont une représentation est donnée en figure 2.1 page 14. La table T a cinq enregistrements E^1, E^2, E^3, E^4 et E^5 tels que $E^1 = (q_3, s_1)$, $E^2 = (q_2, s_2)$, $E^3 = (q_3, s_2)$, $E^4 = (q_1, s_1)$ et $E^5 = (q_1, s_2)$.

La figure 2.6 est une représentation d'une table généralisée T^{gen} sur $(T, (H))$. Nous avons mis en évidence les classes d'équivalence de T^{gen} avec des couleurs.

Les classes d'équivalence de T^{gen} sont $\mathcal{C}(T^{gen}) = \{C_1^{gen}, C_2^{gen}, C_3^{gen}\}$ avec $C_1^{gen} = \{F^1, F^2\}$ de représentant sur $\mathcal{Q}$ $\text{repr}_\mathcal{Q}(C_1^{gen}) = (q_{1,2,3})$, $C_2^{gen} = \{F^3\}$ de représentant sur $\mathcal{Q}$ $\text{repr}_\mathcal{Q}(C_2^{gen}) = (q_3)$ et $C_3^{gen} = \{F^4, F^5\}$ de représentant sur $\mathcal{Q}$ $\text{repr}_\mathcal{Q}(C_3^{gen}) = (q_{1,2})$.

Le sous-ensemble de T associé à C_1^{gen} est $C_1 = \{E^1, E^2\}$. En effet, les enregistrements E^1 et E^2 de T ont été regroupés pour créer la classe C_1^{gen}. De même, le sous-ensemble de T associé à C_2^{gen} est $C_2 = \{E^3\}$ et le sous-ensemble de T associé à C_3^{gen} est $C_3 = \{E^4, E^5\}$.

Pour toute table T sur $(\mathcal{A}, \mathcal{H})$ avec $\mathcal{A}$ un ensemble d'attributs et $\mathcal{H}$ un vecteur de hiérarchies des attributs quasi-identifiants de $\mathcal{A}$, nous définissons la table généralisée sur T dans laquelle tous les enregistrements ont pour chaque valeur quasi-identifiante la racine de la hiérarchie de généralisation de l'attribut quasi-identifiant correspondant (cf. définition 2.3.5). Cette table généralisée correspond à la perte totale de l'information contenue dans T.

Définition 2.3.5 (Table T^*)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$.

La table T^* est la table généralisée sur $(T, \mathcal{H})$ telle que pour tout $F = (f_1, \ldots, f_m, s) \in T^*$, $f_j = r_j$ avec r_j la racine de la hiérarchie H_j pour tout $j \in [\![1, m]\!]$.

2.3.2 Généralisation de sous-ensembles d'enregistrements

Dans cette section, nous allons décrire la généralisation de sous-ensembles d'enregistrements. Pour cela, nous définissons le plus petit ancêtre commun de deux nœuds d'une hiérarchie de généralisation, noté LCA pour Lowest Common Ancestor, en nous basant sur la définition de l'article [3] (cf. définition 2.3.6).

Définition 2.3.6 (LCA)
Soient Q un attribut quasi-identifiant et $H \in \mathcal{H}_Q$. Soient v_1 et v_2 deux nœuds de H.

On dit qu'un nœud v de H est un *ancêtre commun des nœuds* v_1 *et* v_2 si v est une généralisation de v_1 et de v_2.

On définit l'application LCA : $H \times H \to H$ qui à tout couple de nœuds de H associe leur plus petit ancêtre commun :

$$\begin{array}{rccc} \text{LCA} : & H \times H & \longrightarrow & H \\ & (v, v') & \longmapsto & \text{LCA}(v, v') \end{array} ,$$

avec $\text{LCA}(v, v')$ l'ancêtre commun de v et v' le plus éloigné de la racine de H.

Par exemple, dans la hiérarchie H de l'attribut $Q = \{q_1, q_2, q_3\}$ de la figure 2.4 page 17, nous observons que les ancêtres communs de q_1 et q_2 sont $q_{1,2}$ et $q_{1,2,3}$. Le LCA de q_1 et q_2 est $q_{1,2}$.

Remarque 2.3.1
Soient Q un attribut quasi-identifiant et H est une hiérarchie de généralisation de Q. Soient v_1, v_2 et v_3 des nœuds de H.

L'application LCA est associative et symétrique :

$$\begin{aligned} &\text{LCA}(v_1, v_2) = \text{LCA}(v_2, v_1) \\ &\text{LCA}(\text{LCA}(v_1, v_2), v_3) = \text{LCA}(v_1, LCA(v_2, v_3)) \end{aligned}$$

Nous avons défini le LCA de deux nœuds d'une hiérarchie. Or il est possible de déterminer le plus petit ancêtre commun de plus de deux nœuds (cf. définition 2.3.7).

Définition 2.3.7 (Extension du LCA)
Soient Q un attribut quasi-identifiant et H une hiérarchie de généralisation de Q. Notons $\mathcal{N}_H$ l'ensemble des nœuds de H.

La remarque 2.3.1 nous permet d'étendre la définition de plus petit ancêtre commun à un sous-ensemble de nœuds de H.

On définit l'application LCA : $\mathscr{P}(\mathcal{N}_H) - \emptyset \to \mathcal{N}_H$ qui a tout sous-ensemble $\{v_1, \ldots, v_p\}$ de nœuds de H avec $p \in \mathbb{N}^*$ associe $\text{LCA}(\{v_1, \ldots, v_p\})$ l'ancêtre commun aux nœuds $v_1, \ldots, v_p$ le plus éloigné de la racine.

Grâce à la notion de plus petit ancêtre commun d'un ensemble de nœuds, nous pouvons définir la généralisation d'un ensemble d'enregistrements (cf. définition 2.3.8). À un ensemble d'enregistrements, nous associons un enregistrement F tel que F soit une généralisation de tout enregistrement de l'ensemble et que les niveaux de généralisation des valeurs quasi-identifiantes de F soient les plus petits possibles.

Définition 2.3.8 (Généralisation d'enregistrements)
Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. On note $\mathscr{P}(\mathcal{F}_{(\mathcal{Q},\mathcal{H})})$ l'ensemble des parties de $\mathcal{F}_{(\mathcal{Q},\mathcal{H})}$.

On définit l'application $\overline{gen} : \mathscr{P}(\mathcal{F}_{(\mathcal{Q},\mathcal{H})}) - \emptyset \to \mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ comme suit :

$$\begin{array}{rccc} \overline{gen} : & \mathscr{P}(\mathcal{F}_{(\mathcal{Q},\mathcal{H})}) - \emptyset & \longrightarrow & \mathcal{F}_{(\mathcal{Q},\mathcal{H})} \\ & \{F^1, \ldots, F^p\} & \longmapsto & (\text{LCA}(\{f_1^1, \ldots, f_1^p\}), \ldots, \text{LCA}(\{f_m^1, \ldots, f_m^p\})) \end{array} ,$$

avec avec $p \in \mathbb{N}^*$ et $F^i = (f_1^i, \ldots, f_m^i)$ pour tout $i \in [\![1, p]\!]$.

L'application $\overline{gen}$ associe à un ensemble non vide Λ d'enregistrements de $\mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ l'enregistrement de $\mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ correspondant à la généralisation des enregistrements de Λ.

Définition 2.3.9 (Généralisation effective d'un p-uplet de sous-ensembles de T)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. On pose $T = \{E^1, \ldots, E^n\}$.

Soient $p \in \mathbb{N}^*$ et $\Lambda = (\Lambda_1, \ldots, \Lambda_p)$ un p-uplet de sous-ensembles de T non vides ($\forall i \in [\![1, p]\!]$, $\Lambda_i \in \mathscr{P}(T) - \emptyset$) tels que, pour tous i et l dans $[\![1, p]\!]$, $\Lambda_i \cap \Lambda_l = \emptyset$.

Pour tout $i \in [\![1, p]\!]$, on note $\mathcal{C}(\Lambda_i)$ l'ensemble des classes d'équivalence de Λ_i. On note $\text{Repr}(\Lambda_i) = \{\text{repr}_\mathcal{Q}(C) : C \in \mathcal{C}(\Lambda_i)\}$ l'ensemble des représentants des classes d'équivalence de Λ_i.

T	Q	S
E^1	q_3	s_1
E^2	q_2	s_2
E^3	q_3	s_2
E^4	q_1	s_1
E^5	q_1	s_2

FIGURE 2.7 – Représentation d'une table sur $\mathcal{A}$ avec mise en évidence des sous-ensembles $\Lambda_1 = \{E^1, E^4\}$ et $\Lambda_2 = \{E^2, E^5\}$ à généraliser dans T. Λ_1 est en bleu et Λ_2 est en rouge.

On définit l'application $gen_T : (\mathscr{P}(T) - \emptyset)^p \to \mathcal{T}^{gen}_{(T,\mathcal{H})}$ par $gen_T(\Lambda) = T^{gen}$ avec $T^{gen} = \{F^1, \ldots, F^n\}$ telle que pour tout $l \in [\![1, n]\!]$,

$$\begin{cases} F^l = (g_1, \ldots, g_m, s^l) & \text{s'il existe } i \in [\![1, p]\!] \text{ tel que } E^l = (e^l_1, \ldots, e^l_m, s^l) \in \Lambda_i \text{ et } \overline{gen}(\text{Repr}(\Lambda_i)) = (g_1, \ldots, g_m) \\ F^l = E^l & \text{sinon} \end{cases}$$

L'application gen_T généralise les enregistrements de chaque sous-ensemble Λ_i pour qu'ils appartiennent à une même classe d'équivalence de T^{gen} de représentant sur $\mathcal{Q}$ $\overline{gen}(\text{Repr}(\Lambda_i))$. On dira également que l'on *fusionne* les enregistrements de chaque sous-ensemble Λ_i.

Exemple 2.3.2
Soit $\mathcal{A} = \{Q, S\}$ un ensemble d'attributs avec $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant et $S = \{s_1, s_2\}$ un attribut sensible.

Considérons la hiérarchie H de Q donnée en la figure 2.4 page 17. H est constituée de trois feuilles (q_1, q_2 et q_3), d'un nœud de niveau 1 ($q_{1,2}$ relié à q_1 et q_2) et d'une racine ($q_{1,2,3}$ relié à q_3 et $q_{1,2}$).

Soit T une table sur $(\mathcal{A}, \mathcal{H})$. La table T a cinq enregistrements E^1, E^2, E^3, E^4 et E^5 tels que $E^1 = (q_3, s_1)$, $E^2 = (q_2, s_2)$, $E^3 = (q_3, s_2)$, $E^4 = (q_1, s_1)$ et $E^5 = (q_1, s_2)$.

Soient $\Lambda = (\Lambda_1, \Lambda_2)$ avec $\Lambda_1 = \{E^1, E^4\}$ et $\Lambda_2 = \{E^2, E^5\}$ deux sous-ensembles de T. On a $\Lambda_1 \cap \Lambda_2 = \emptyset$. Nous voulons généraliser les enregistrements de Λ_1 et les enregistrements de Λ_2. En figure 2.7, nous donnons une représentation de la table T mettant en évidence les sous-ensembles Λ_1 et Λ_2 avec des couleurs.

Suivons la définition 2.3.9 page précédente. Commençons par calculer les classes d'équivalence de Λ_1 et de Λ_2 ainsi que leur ensemble de représentants des classes d'équivalence.

L'ensemble des classes d'équivalence de Λ_1 est $\mathcal{C}(\Lambda_1) = \{\{E^1\}, \{E^4\}\}$ et l'ensemble des représentants sur $\mathcal{Q}$ des classes d'équivalence de Λ_1 est $\text{Repr}(\Lambda_1) = \{(q_3), (q_1)\}$.

L'ensemble des classes d'équivalence de Λ_2 est $\mathcal{C}(\Lambda_2) = \{\{E^2, E^5\}\}$ et l'ensemble des représentants sur $\mathcal{Q}$ des classes d'équivalence de Λ_2 est $\text{Repr}(\Lambda_2) = \{(q_2), (q_1)\}$.

On a $gen_T(\Lambda) = T^{gen}$ avec $T^{gen} = \{F^1, F^2, F^3, F^4, F^5\}$ une table généralisée sur $(T, \mathcal{H})$ telle que
— $F^1 = (q_{1,2,3}, s_1)$ car $E^1 = (q_3, s_1) \in \Lambda_1$ et $\overline{gen}(\text{Repr}(\Lambda_1)) = (q_{1,2,3})$
— $F^2 = (q_{1,2}, s_2)$ car $E^2 = (q_2, s_2) \in \Lambda_2$ et $\overline{gen}(\text{Repr}(\Lambda_2)) = (q_{1,2})$
— $F^3 = E^3$ car $E^3 \notin \Lambda_1 \bigcup \Lambda_2$
— $F^4 = (q_{1,2,3}, s_1)$ car $E^4 = (q_1, s_1) \in \Lambda_1$ et $\overline{gen}(\text{Repr}(\Lambda_1)) = (q_{1,2,3})$
— $F^5 = (q_{1,2}, s_2)$ car $E^5 = (q_1, s_2) \in \Lambda_2$ et $\overline{gen}(\text{Repr}(\Lambda_2)) = (q_{1,2})$

Une représentation de T^{gen} sur $(\mathcal{A}, \mathcal{H})$ est donnée en figure 2.8 page suivante.

L'ensemble des classes d'équivalence de T^{gen} est $\mathcal{C}(T^{gen}) = \{C^{gen}_1, C^{gen}_2, C^{gen}_3\}$ avec $C^{gen}_1 = \{F^1, F^4\}$, $C^{gen}_2 = \{F^2, F^5\}$ et $C^{gen}_3 = \{F^3\}$.

Vérifions que les enregistrements de T^{gen} correspondant aux enregistrements de Λ_1 et de Λ_2 se trouvent dans les mêmes classes d'équivalence.

Les enregistrements de T^{gen} correspondant aux enregistrements de Λ_1 sont F^1 et F^4. Ils appartiennent à la même classe d'équivalence de T^{gen}, il s'agit de C^{gen}_1.

Les enregistrements de T^{gen} correspondant aux enregistrements de Λ_2 sont F^2 et F^5. Ils appartiennent à la même classe d'équivalence de T^{gen}, il s'agit de C^{gen}_2.

2.4 Tables pour les expérimentations

Dans cette section, nous allons présenter les deux tables sur lesquelles nous mènerons nos expérimentations dans les chapitres suivants. Elles sont publiques et disponibles au téléchargement en ligne.

T^{gen}	Q	S
F^1	$q_{1,2,3}$	s_1
F^2	$q_{1,2}$	s_2
F^3	q_3	s_2
F^4	$q_{1,2,3}$	s_1
F^5	$q_{1,2}$	s_2

FIGURE 2.8 – Représentation d'une table généralisée sur $(T, \mathcal{H})$ avec mise en évidence des classes d'équivalence. C_1^{gen} est en bleu, C_2^{gen} est en rouge et C_3^{gen} est en vert.

Nom	Nombre de valeurs distinctes	Hauteur de la hiérarchie	Nombre de nœuds dans la hiérarchie
Age	72	5	105
Genre	2	2	3
Race	5	2	6
Statut marital	7	3	10
Éducation	16	4	22
Pays de naissance	41	3	45
Catégorie professionnelle	8	3	12
Occupation	14	3	17
Salaire	2	2	3

TABLEAU 2.1 – Caractéristiques des attributs retenus pour *Adult data set*

2.4.1 *Adult data set*

Adult data set est une table couramment utilisée dans le domaine de l'anonymisation des données. Elle est publique et disponible en ligne à l'adresse suivante [46]. L'ensemble d'attributs considérés est de taille 14. La table comporte 48 842 enregistrements.

Pour notre étude, nous allons considérer neuf attributs parmi les quatorze proposés. Les attributs retenus sont *Age*, *Genre*, *Race*, *Statut marital*, *Éducation*, *Pays de naissance*, *Catégorie professionnelle*, *Occupation* et *Salaire*.

Après avoir supprimé les enregistrements dont certaines valeurs sont manquantes, nous obtenons une table de 30 162 enregistrements. Si nous considérons que tous les attributs sont quasi-identifiants, la table a 19 502 classes d'équivalence. La taille minimale des classes est 1 et la taille maximale est 45. Nous observons que les classes d'équivalence de taille 1 représentent près de 80% de l'ensemble des classes d'équivalence.

Pour chaque attribut retenu, nous avons construit une hiérarchie de généralisation se basant sur le sens sémantique des valeurs de l'attribut. Par exemple, pour l'attribut *Éducation*, nous nous sommes référés au système éducatif des États-Unis pour construire la hiérarchie de généralisation. Les hiérarchies de généralisation des neuf attributs de *Adult data set* considérés sont en section A.1 page 137 de l'annexe A page 137. Pour chaque attribut, nous précisons dans le tableau 2.1 son nom, son nombre de valeurs distinctes dans la table d'origine et la hauteur et le nombre de nœuds de la hiérarchie de généralisation que nous avons créée.

2.4.2 *Florida*

Le registre des votants de l'état de Floride aux États-Unis est public et disponible en ligne [20]. Au 31 décembre 2020, 15 085 402 personnes étaient enregistrées comme votant dans l'un des comtés de Floride. L'ensemble des attributs considérés est de taille 38. La table que nous avons téléchargée, appelée *Florida*, date du 31 mai 2020 et contient 14 826 104 enregistrements.

Pour notre étude, nous considérons cinq attributs parmi les trente-huit proposés. Les attributs retenus sont *Code postal* (uniquement les cinq premiers digits), *Genre*, *Race*, *Année de naissance* (dérivée de l'attribut *Date de naissance*) et *Parti politique*.

Comme la table contient un trop grand nombre d'enregistrements, nous avons aléatoirement tiré 30 162 enregistrements dans *Florida*. Nous avons obtenu un extrait de *Florida* appelé *florida_30162*. En considérant que tous les attributs sont quasi-identifiants, *florida_30162* a 28 896 classes d'équivalence. La taille minimale des classes est 1 et la taille maximale est 6. Les classes d'équivalence de taille 1 représentent près de 96% de l'ensemble des classes d'équivalence.

Nom	Nombre de valeurs distinctes	Hauteur de la hiérarchie	Nombre de nœuds dans la hiérarchie
Code postal	941	5	1167
Genre	2	2	3
Race	8	2	9
Année de naissance	92	4	106
Parti politique	9	2	10

TABLEAU 2.2 – Caractéristiques des attributs retenus pour *florida_30162*

Comme pour les attributs de *Adult data set*, nous avons créé une hiérarchie de généralisation pour chacun des attributs de *Florida* retenu en nous basant sur le sens sémantique des valeurs de l'attribut. Par exemple, pour l'attribut *Année de naissance*, nous avons regroupé les années par décennies puis par siècle pour finir par une année quelconque symbolisé par ****. Les hiérarchies de généralisation des cinq attributs de *florida_30162* considérés sont en section A.2 page 139 de l'annexe A page 137. Pour chaque attribut, nous précisons dans le tableau 2.2 son nom, son nombre de valeurs distinctes dans la table d'origine et la hauteur et le nombre de nœuds de la hiérarchie de généralisation que nous avons créée.

Chapitre 3

Comparaison de métriques de perte d'information

Sommaire du présent chapitre

D'une table donnée, le nombre de tables k-anonymes pouvant en être dérivées est, en fonction de k et du nombre d'enregistrements de la table, de nature exponentielle. Considérons une table à n enregistrements. Le nombre de tables k-anonymes potentielles, correspondant au nombre de partitionnements de n éléments en sous-ensembles de taille supérieure à k, est supérieur à $\sum_{i=1}^{k} \binom{n}{k}$ qui est de l'ordre de 2^n. Plus précisément, il s'agit du nombre de Stirling k-associé de second espèce $\left\{ {n \atop m} \right\}_{\geq k}$ représentant le nombre de façons de partitionner n éléments étiquetés en m sous-ensembles non étiquetés contenant au moins k éléments ; des détails sur ce nombre peuvent être lus dans l'article[9]. Il est ainsi important de pourvoir classifier ces tables k-anonymes en fonction de leur qualité en termes d'utilité des données.

Dans la littérature, la notion de métriques de perte d'information est souvent utilisée pour évaluer la qualité des tables anonymes. Une métrique de perte d'information est une application qui, à une table anonymisée grâce à la technique de généralisation, associe un réel traduisant la quantité d'information perdue par rapport à la table d'origine. Il est à noter que les métriques de perte d'information servent à évaluer l'utilité des données d'une table anonyme, elles ne mesurent pas le niveau de protection de la vie privée de la table anonyme. Bien que le principe général reste le même, plusieurs métriques de perte d'information ont été proposées [24, 2, 28, 7, 50]. La plupart du temps, les notations utilisées pour définir ces métriques de perte d'information ne sont pas les mêmes d'un article à l'autre. De plus, peu de justifications sont données pour le choix d'une métrique plutôt qu'une autre.

Dans ce chapitre, nous allons mener une étude comparative de métriques de perte d'information. L'objectif est, dans un premier temps, de proposer une écriture unifiée et facilement utilisable des métriques de perte d'information. Une nouvelle définition ainsi qu'un modèle à base de matrices seront présentés. Nous proposerons également quatre nouvelles métriques de perte d'information. Dans un second temps, nous tâcherons d'évaluer les performances de plusieurs métriques de perte d'information lorsqu'elles sont utilisées dans un algorithme de k-anonymisation. Cet algorithme, présenté en section 3.3.2 page 37, produit une table k-anonyme en fusionnant

les classes d'équivalence de la table jusqu'à ce qu'elles soient de taille supérieure à k. Dans cet algorithme, les fusions à effectuer sont déterminées grâce à une métrique de perte d'information.

Dans la suite de ce chapitre, nous donnerons en section 3.1 une définition aux métriques de perte d'information et nous présenterons notre modèle pour simplifier l'utilisation des métriques. Nous détaillerons également le coût de généralisation de deux classes d'équivalence d'une table pour une métrique ainsi que le coût de généralisation d'une table généralisée par rapport à une table d'origine pour une métrique. Nous définirons également l'altération pour une métrique d'une table généralisée. Dans la section 3.2 page 32, nous présenterons trois métriques de perte d'information issues de la littérature ainsi que quatre métriques issues de nos travaux. Dans la section 3.3 page 36, nous reviendrons sur la définition de table k-anonyme et nous en proposerons une formulation se basant sur les notations introduites dans le chapitre 2 page 13. Nous présenterons également un algorithme de k-anonymisation. Enfin, dans la section 3.4 page 40, nous présenterons le protocole expérimental nous permettant de comparer les performances des métriques lors d'un processus de k-anonymisation. Nous proposerons trois critères d'évaluation de qualité des tables k-anonymes et nous étudierons les résultats obtenus.

Les contributions de ce chapitre sont à retrouver dans l'article [35].

3.1 Métrique de perte d'information

Dans la littérature, de nombreuses métriques de perte d'information sont proposées pour évaluer l'utilité des données d'une table anonyme. En revanche, leur définition utilisent des notations différentes d'un article à l'autre. Pour pallier ce problème, nous introduisons un modèle permettant d'unifier l'écriture des métriques de perte d'information et de faciliter leur utilisation. Premièrement, nous proposons une définition de métrique de perte d'information se basant sur des poids à mettre sur les arêtes des hiérarchies de généralisation des attributs quasi-identifiants de la table (cf. définition 3.1.1). Deuxièmement, pour une métrique donnée, nous définissons une matrice pour chaque attribut quasi-identifiant de la table. Cette matrice est labellisée par les nœuds de la hiérarchie de généralisation du quasi-identifiant en ligne et en colonne (cf. définition 3.1.3 page suivante). Une valeur de la matrice correspondra au coût de généralisation du nœud labellisant la ligne en l'ancêtre commun des nœuds labellisant la ligne et la colonne pour la métrique de perte d'information choisie.

Définition 3.1.1 (Métrique de perte d'information)
Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. On pose $\mathcal{H} = (H_1, \ldots, H_m)$.

Pour tout $j \in [\![1, m]\!]$, on pondère les arêtes de H_j avec des valeurs dans $\mathbb{R}^+$. Pour deux nœuds x et x' de H_j, s'il existe une arête (x, x') dans H_j, on note $\omega(x, x')$ le poids de l'arête (x, x').

On appelle *métrique sur* $\mathcal{H}$ un m-uplet d'ensembles de poids sur les hiérarchies de $\mathcal{H}$:

$$\mu = (\mu_1, \ldots, \mu_m),$$

avec $\mu_j = \{\omega(x, x') : (x, x') \text{ arête de } H_j\}$ pour $j \in [\![1, m]\!]$.

Définir une métrique de perte d'information revient donc à définir un ensemble de poids sur les arêtes des hiérarchies de généralisation des attributs quasi-identifiants. Les valeurs des poids peuvent être définies à partir d'une formule ou de manière arbitraire.

Dans la suite du manuscrit, tout emploi du terme « métrique » fera référence à une métrique de perte d'information.

Avant de présenter les matrices des coûts associées à une métrique de perte d'information, nous définissons le coût de généralisation pour une métrique d'un nœud d'une hiérarchie en une de ses généralisations. Rappelons qu'un nœud v' est une généralisation d'un nœud v dans une hiérarchie si v' est sur le chemin de v à la racine de la hiérarchie (cf. définition 2.3.1 page 19).

Définition 3.1.2 (Coût de généralisation d'un nœud en une de ses généralisations)
Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. On pose $\mathcal{H} = (H_1, \ldots, H_m)$. Soit $\mu = (\mu_1, \ldots, \mu_m)$ une métrique sur $\mathcal{H}$.

Pour tout $j \in [\![1, m]\!]$, l'application $\Omega_{H_j} : H_j \times H_j \to \mathbb{R}$ associe à tout couple de nœuds de H_j :

$$\begin{array}{rccl} \Omega_{H_j} : & H_j \times H_j & \longrightarrow & \mathbb{R} \\ & (v, v') & \longmapsto & \begin{cases} \displaystyle\sum_{(x,x') \in \mathcal{E}(v,v')} \omega(x, x') & \text{si } v' \text{ est une généralisation de } v \\ 0 & \text{sinon} \end{cases} \end{array}$$

avec $\mathcal{E}(v, v')$ l'ensemble des arêtes sur le chemin de v à v', $\omega(x, x') \in \mu_j$ pour tout $(x, x') \in \mathcal{E}(v, v')$ et r_j la racine de H_j.

Si v' est une généralisation de v dans H_j, $\Omega_{H_j}(v, v')$ calcule le coût pour μ du chemin de v à v'. En d'autres termes, il représente le coût pour μ de généraliser v en v'. Si v' n'est pas une généralisation de v dans H_j, $\Omega_{H_j}(v, v')$ vaut 0.

Remarque 3.1.1
En reprenant les notations de la définition 3.1.2 page précédente, nous avons

$$\Omega_{H_j}(v, v) = 0$$

car l'ensemble des arêtes sur le chemin de v à v est vide.

L'application Ω_{H_j} représente le coût de généralisation d'un nœud en un autre si le second nœud est une généralisation du premier nœud. Par exemple, supposons qu'une hiérarchie H comporte trois feuilles *Chat*, *Lion* et *Chien*, un nœud intermédiaire *Félin* auquel sont reliés *Chat* et *Lion* et une racine *Mammifère* à laquelle sont reliés *Félin* et *Chien*. $\Omega_H(\textit{Chat}, \textit{Félin})$ est différent de 0 car *Félin* est un ancêtre de *Chat* dans H. En revanche, $\Omega_H(\textit{Chien}, \textit{Félin})$ vaut 0 car *Félin* n'est pas un ancêtre de *Chien* dans H.

La matrice des coûts définit le coût de généralisation de tout couple de nœuds d'une hiérarchie. Les lignes et les colonnes de la matrice étant labellisées par les nœuds de la hiérarchie, tout coefficient représente le coût de généralisation du nœud labellisant la ligne en le plus petit ancêtre commun du nœud labellisant la ligne et du nœud labellisant la colonne. Ainsi, en reprenant l'exemple de hiérarchie précédent, le coût de généralisation de *Chien* en *Félin* sera le coût de généralisation de *Chien* en *Mammifère* et le coût de généralisation de *Félin* en *Chien* sera le coût de généralisation de *Mammifère*. Rappelons que le plus petit ancêtre commun de deux nœuds v et v' d'une hiérarchie est noté $\mathrm{LCA}(v, v')$ (cf. définition 2.3.6 page 21).

Définition 3.1.3 (Matrice des coûts d'une hiérarchie pour une métrique de perte d'information)
Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. On pose $\mathcal{H} = (H_1, \ldots, H_m)$. Soit μ une métrique sur $\mathcal{H}$.

Pour tout $j \in [\![1, m]\!]$, la *matrice des coûts* de H_j pour la métrique μ notée M_{μ,H_j} est définie comme suit
— les lignes et les colonnes de M_{μ,H_j} sont labellisées par les nœuds de H_j
— pour tout couple de nœuds (v, v') de H_j, $M_{\mu,H_j}(v, v') = \Omega_{H_j}(v, \mathrm{LCA}(v, v'))$

Remarque 3.1.2
Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. On pose $\mathcal{H} = (H_1, \ldots, H_m)$. Soit μ une métrique sur $\mathcal{H}$. Soit $j \in [\![1, m]\!]$.

Dans le cas général, pour v et v' deux nœuds de H_j, $\Omega_{H_j}(v, \mathrm{LCA}(v, v'))$ est différent de $\Omega_{H_j}(v', \mathrm{LCA}(v', v))$. La matrice des coûts M_{μ,H_j} n'est donc pas symétrique.

Exemple 3.1.1
Soit $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant dont une hiérarchie H est donnée en figure 2.4 page 17.

Soit μ une métrique sur H définie par les poids sur les arêtes de H suivants : $\omega(q_1, q_{1,2}) = 1$, $\omega(q_2, q_{1,2}) = 2$, $\omega(q_{1,2}, q_{1,2,3}) = 3$ et $\omega(q_3, q_{1,2,3}) = 4$. La figure 3.1 page suivante montre la hiérarchie H avec les poids de la métrique μ.

Nous construisons la matrice des coûts $M_{\mu,H}$ de H pour μ.

$M_{\mu,H}$ est une matrice 5×5 car la hiérarchie H a cinq nœuds. D'après la définition 3.1.3, chaque coefficient de la matrice est défini par $\Omega_H(v, LCA(v, v'))$ avec v le nœud de la ligne correspondante et v' le nœud de la colonne correspondante.

Tous les coefficients de la forme $M_{\mu,H}(v, v)$ avec v un nœud de H valent 0 car $\Omega_H(v, LCA(v, v)) = \Omega_H(v, v) = 0$ d'après la remarque 3.1.1. De plus, tous les coefficients de la forme $M_{\mu,H}(v, v')$ avec v et v' deux nœuds de H tels que v est une généralisation de v' valent 0 d'après la définition 3.1.2 page précédente.

Pour les autres coefficients, nous appliquons la formule de la définition 3.1.2 page ci-contre pour deux nœuds v et v' de H avec v' une généralisation de v. Par exemple, nous obtenons

$$M_{\mu,H}(q_1, q_2) = \Omega_H(q_1, \mathrm{LCA}(q_1, q_2)) = \Omega_H(q_1, q_{1,2}) = \omega(q_1, q_{1,2}) = 1$$

et

$$M_{\mu,H}(q_2, q_{1,2,3}) = \Omega_H(q_2, \mathrm{LCA}(q_2, q_{1,2,3})) = \Omega_H(q_2, q_{1,2,3}) = \omega(q_2, q_{1,2}) + \omega(q_{1,2}, q_{1,2,3}) = 2 + 3 = 5.$$

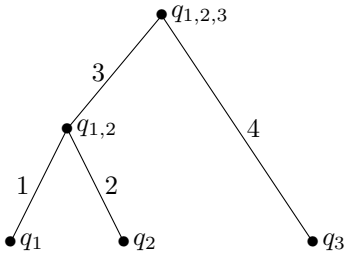


FIGURE 3.1 – Représentation d'un hiérarchie de Q avec des poids sur les arêtes.

La matrice des coûts de H pour la métrique μ est :

$$M_{\mu,H} = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \\ q_{1,2} \\ q_{1,2,3} \end{array} \begin{array}{c} \begin{array}{ccccc} q_1 & q_2 & q_3 & q_{1,2} & q_{1,2,3} \end{array} \\ \begin{pmatrix} \Omega_H(q_1,q_1) & \Omega_H(q_1,q_{1,2}) & \Omega_H(q_1,q_{1,2,3}) & \Omega_H(q_1,q_{1,2}) & \Omega_H(q_1,q_{1,2,3}) \\ \Omega_H(q_2,q_{1,2}) & \Omega_H(q_2,q_2) & \Omega_H(q_2,q_{1,2,3}) & \Omega_H(q_2,q_{1,2}) & \Omega_H(q_2,q_{1,2,3}) \\ \Omega_H(q_3,q_{1,2,3}) & \Omega_H(q_3,q_{1,2,3}) & \Omega_H(q_3,q_3) & \Omega_H(q_3,q_{1,2,3}) & \Omega_H(q_3,q_{1,2,3}) \\ \Omega_H(q_{1,2},q_{1,2}) & \Omega_H(q_{1,2},q_{1,2}) & \Omega_H(q_{1,2},q_{1,2,3}) & \Omega_H(q_{1,2},q_{1,2}) & \Omega_H(q_{1,2},q_{1,2,3}) \\ \Omega_H(q_{1,2,3},q_{1,2,3}) & \Omega_H(q_{1,2,3},q_{1,2,3}) & \Omega_H(q_{1,2,3},q_{1,2,3}) & \Omega_H(q_{1,2,3},q_{1,2,3}) & \Omega_H(q_{1,2,3},q_{1,2,3}) \end{pmatrix} \end{array}$$

$$M_{\mu,H} = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \\ q_{1,2} \\ q_{1,2,3} \end{array} \begin{array}{c} \begin{array}{ccccc} q_1 & q_2 & q_3 & q_{1,2} & q_{1,2,3} \end{array} \\ \begin{pmatrix} 0 & 1 & 4 & 1 & 4 \\ 2 & 0 & 5 & 2 & 5 \\ 4 & 4 & 0 & 4 & 4 \\ 0 & 0 & 3 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Grâce à la représentation des métriques de perte d'information sous forme de matrices des coûts, nous pouvons utiliser notre modèle pour calculer la quantité d'information perdue pour une métrique dans deux cas :

Cas 1 on cherche à connaître le coût engendré par la fusion de deux classes d'équivalence d'une table

Cas 2 on cherche à connaître le coût d'une table généralisée par rapport à la table d'origine

Dans la définition 3.1.4, nous donnons l'expression du coût de généralisation de deux classes d'équivalence d'une table pour une métrique de perte d'information. Cela sera notamment utilisé dans l'algorithme de k-anonymisation présenté en section 3.3.2 page 37 pour guider le choix des fusions de classes d'équivalence en fonction du coût pour une métrique engendré dans la table.

Définition 3.1.4 (Coût de généralisation de deux classes d'équivalence d'une table)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q}$. Soient μ une métrique sur $\mathcal{H}$ et M_{μ,H_j} la matrice des coûts de H_j pour μ pour tout $j \in [\![1, m]\!]$.

On définit l'application $\overline{\mu} : \mathcal{F}_{(\mathcal{A},\mathcal{H})} \times \mathcal{F}_{(\mathcal{A},\mathcal{H})} \to \mathbb{R}^+$ qui à deux enregistrements généralisés sur $\mathcal{F}_{(\mathcal{A},\mathcal{H})}$ associe le coût de généraliser ces deux enregistrements pour la métrique μ :

$$\begin{array}{rccl} \overline{\mu} : & \mathcal{F}_{(\mathcal{A},\mathcal{H})} \times \mathcal{F}_{(\mathcal{A},\mathcal{H})} & \longrightarrow & \mathbb{R}^+ \\ & (F, F') & \longmapsto & \sum_{j=1}^{m} M_{\mu,H_j}\left(f_j, f'_j\right) + M_{\mu,H_j}\left(f'_j, f_j\right) \end{array},$$

avec $F = (f_1, \ldots, f_m, s)$ et $F' = (f'_1, \ldots, f'_m, s')$.

Soit T une table sur $(\mathcal{A}, \mathcal{H})$. On étend la définition de $\overline{\mu}$ à deux classes d'équivalence de la table T. Soient C et C' deux classes d'équivalence de T de représentants sur $\mathcal{Q}$ respectifs $\text{repr}_{\mathcal{Q}}(C) = (c_1, \ldots, c_m)$ et $\text{repr}_{\mathcal{Q}}(C') = (c'_1, \ldots, c'_m)$.

On pose

$$\overline{\mu}(C, C') := \sum_{j=1}^{m} M_{\mu,H_j}\left(c_j, c'_j\right) \times |C| + M_{\mu,H_j}\left(c'_j, c_j\right) \times |C'|,$$

avec $|C|$ et $|C'|$ le nombre d'enregistrements dans C et C' respectivement, $M_{\mu,H_j}\left(c_j, c'_j\right)$ le coût de généraliser c_j en $\text{LCA}(c_j, c'_j)$ pour μ et $M_{\mu,H_j}\left(c'_j, c_j\right)$ le coût de généraliser c'_j en $\text{LCA}(c'_j, c_j)$ pour μ.

L'application $\overline{\mu}$ représente le coût pour une métrique de perte d'information μ de généraliser deux enregistrements ou deux classes d'équivalence d'une même table.

Dans la définition 3.1.5, nous exprimons le coût de généralisation pour une métrique de perte d'information d'une table généralisée par rapport à la table d'origine. Cela nous permettra, lors des expérimentations de la section 3.4 page 40 par exemple, de comparer plusieurs tables k-anonymes en fonction de leur coût de généralisation pour une métrique par rapport à une même table d'origine.

Définition 3.1.5 (Coût de généralisation d'une table généralisée par rapport à la table d'origine) Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. On pose $T = \{E^1, \ldots, E^n\}$. Soient μ une métrique sur $\mathcal{H}$ et M_{μ,H_j} la matrice des coûts de H_j pour μ pour tout $j \in [\![1, m]\!]$. Soit $T^{gen} = \{F^1, \ldots, F^n\}$ une table généralisée sur $(T, \mathcal{H})$.

On définit l'application $\mu_T : T^{gen} \to \mathbb{R}$ qui à tout enregistrement de T^{gen} associe son coût de généralisation par rapport à T :

$$\mu_T\left(F^i\right) = \overline{\mu}\left(F^i, E^i\right),$$

avec $i \in [\![1, n]\!]$.

On étend la définition de μ_T à T^{gen}. On pose

$$\mu_T\left(T^{gen}\right) = \sum_{i=1}^{m} \mu_T\left(F^i\right).$$

$\mu_T\left(T^{gen}\right)$ représente le coût de généralisation de T^{gen} par rapport à T.

L'application μ_T représente le coût de généralisation pour une métrique de perte d'information μ d'un enregistrement ou d'une table généralisée par rapport à la table d'origine T.

Le résultat présenté dans la proposition 3.1.1 page suivante lie table généralisée et coût de généralisation pour une métrique de perte d'information. Il permettra de justifier la pertinence des outils utilisés pour analyser les résultats des expérimentations menées dans la section 3.4 page 40.

Nous avons besoin du lemme 3.1.1 pour démontrer la proposition 3.1.1 page suivante.

Lemme 3.1.1
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q}$.

Soient F, F' et F'' des enregistrements généralisés sur $(\mathcal{A}, \mathcal{H})$.

Soit $\mu = (\mu_1, \ldots, \mu_m)$ une métrique sur $\mathcal{H}$.

Si F'' est une généralisation de F' et F' est une généralisation de F alors

$$\overline{\mu}\left(F', F\right) \leqslant \overline{\mu}\left(F'', F\right).$$

Démonstration : Par la propriété 2.3.2 page 19 de transitivité de la relation *généralisation de* pour des enregistrements, F'' est une généralisation de F.

Posons $F = (f_1, \ldots, f_m, s)$, $F' = (f'_1, \ldots, f'_m, s)$ et $F'' = (f''_1, \ldots, f''_m, s)$.

Par la définition 3.1.4 page ci-contre de $\overline{\mu}$ sur deux enregistrements, on a $\overline{\mu}\left(F', F\right) = \sum_{j=1}^{m} M_{\mu,H_j}\left(f'_j, f_j\right) + M_{\mu,H_j}\left(f_j, f'_j\right)$ et $\overline{\mu}\left(F'', F\right) = \sum_{j=1}^{m} M_{\mu,H_j}\left(f''_j, f_j\right) + M_{\mu,H_j}\left(f_j, f''_j\right)$.

Montrons dans un premier temps que $\forall j \in [\![1, m]\!]\ M_{\mu,H_j}\left(f'_j, f_j\right) = 0$ et $M_{\mu,H_j}\left(f''_j, f_j\right) = 0$. Soit $j \in [\![1, m]\!]$. Comme F' est une généralisation de F, f'_j est une généralisation de f_j donc par définition f'_j se trouve sur le chemin de f_j à la racine de H_j. Cela implique que soit $f_j = f'_j$ soit f_j n'est pas une généralisation de f'_j. Si $f_j = f'_j$ alors $\Omega_{H_j}\left(f'_j, f_j\right) = 0$ d'après la remarque 3.1.1 page 27. Si f_j n'est pas une généralisation de f'_j alors $\Omega_{H_j}\left(f'_j, f_j\right) = 0$ par la définition 3.1.2 page 26. Dans les deux cas, on obtient $M_{\mu,H_j}\left(f'_j, f_j\right) = 0$ par la définition 3.1.3 page 27. Le raisonnement est le même pour $M_{\mu,H_j}\left(f''_j, f_j\right)$.

Montrons dans un second temps que $\forall j \in [\![1, m]\!]\ M_{\mu,H_j}\left(f_j, f'_j\right) \leqslant M_{\mu,H_j}\left(f_j, f''_j\right)$. Soit $j \in [\![1, m]\!]$. Par la définition 3.1.3 page 27, on a $M_{\mu,H_j}\left(f_j, f'_j\right) = \sum_{(x,x') \in \mathcal{E}(f_j, f'_j)} \omega(x, x')$ et $M_{\mu,H_j}\left(f_j, f''_j\right) = \sum_{(x,x') \in \mathcal{E}(f_j, f''_j)} \omega(x, x')$. Comme F'' est une généralisation de F et de F', f''_j est une généralisation de f_j et de f'_j. De même, comme F' est une généralisation de F, f'_j est une généralisation de f_j. Donc l'ensemble des arêtes sur le chemin de f_j à f''_j est l'union des arêtes sur le chemin de f_j à f'_j et des arêtes sur le chemin de f'_j à f''_j c'est-à-dire

$\mathcal{E}(f_j, f_j'') = \mathcal{E}(f_j, f_j') \cup \mathcal{E}(f_j', f_j'')$. Donc

$$\begin{aligned}
M_{\mu,H_j}\left(f_j, f_j''\right) &= \sum_{(x,x')\in\mathcal{E}(f_j,f_j'')} \omega(x,x') \\
&= \sum_{(x,x')\in\mathcal{E}(f_j,f_j')\cup\mathcal{E}(f_j',f_j'')} \omega(x,x') \\
&= \sum_{(x,x')\in\mathcal{E}(f_j,f_j')} \omega(x,x') + \sum_{(x,x')\in\mathcal{E}(f_j',f_j'')} \omega(x,x') \text{ car } \mathcal{E}(f_j,f_j')\cap\mathcal{E}(f_j',f_j'') = \emptyset \\
&\geqslant \sum_{(x,x')\in\mathcal{E}(f_j,f_j')} \omega(x,x') \text{ car } \omega(x,x') \geqslant 0 \text{ pour tout } \omega(x,x') \in \mu_j \\
&= M_{\mu,H_j}\left(f_j, f_j'\right)
\end{aligned}$$

On déduit des deux points précédents que

$$\begin{aligned}
\overline{\mu}\left(F'', F\right) &= \sum_{j=1}^{m} M_{\mu,H_j}\left(f_j'', f_j\right) + M_{\mu,H_j}\left(f_j, f_j''\right) \text{ par la définition 3.1.4 de } \overline{\mu} \\
&= \sum_{j=1}^{m} M_{\mu,H_j}\left(f_j, f_j''\right) \text{ par le premier point de la démonstration} \\
&\geqslant \sum_{j=1}^{m} M_{\mu,H_j}\left(f_j, f_j'\right) \text{ par le second point de la démonstration et } M_{\mu,H_j}(v,v') \geqslant 0 \text{ pour tout } v, v' \in H_j \\
&= \overline{\mu}\left(F', F\right) \text{ par la définition 3.1.4 de } \overline{\mu} \text{ et le premier point de la démonstration}
\end{aligned}$$

■

Proposition 3.1.1 (Lien entre table généralisée et coût de généralisation pour une métrique) Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q}$.

Soient T_1, T_2 et T_3 trois tables sur $(\mathcal{A}, \mathcal{H})$ de cardinaux respectifs $n_1, n_2, n_3 \in \mathbb{N}^*$.

Soit μ une métrique sur $\mathcal{H}$.

Si T_3 est une table généralisée sur $(T_2, \mathcal{H})$ et T_2 est une table généralisée sur $(T_1, \mathcal{H})$ alors

$$\mu_{T_1}\left(T_2\right) \leqslant \mu_{T_1}\left(T_3\right).$$

Démonstration : Par la propriété 2.3.3 page 19 de transitivité de la relation *table généralisée sur*, T_3 est une table généralisée sur $(T_1, \mathcal{H})$. On en déduit que $n_3 = n_2 = n_1$. Posons $n = n_1$.

Posons $T_1 = \{F_1^1, \ldots, F_1^n\}$, $T^2 = \{F_2^1, \ldots, F_2^n\}$ et $T_3 = \{F_3^1, \ldots, F_3^n\}$.

Par la définition 2.3.3 page 19 de table généralisée, F_2^i et F_3^i sont des généralisations de F_1^i d'une part et F_3^i est une généralisation de F_2^i d'autre part pour tout $i \in [\![1, n]\!]$.

On obtient donc

$$\begin{aligned}
\mu_{T_1}\left(T_2\right) &= \sum_{i=1}^{n} \mu_{T_1}\left(F_2^i\right) \text{ par la définition 3.1.5} \\
&= \sum_{i=1}^{n} \overline{\mu}\left(F_2^i, F_1^i\right) \text{ par la définition 3.1.5} \\
&\leqslant \sum_{i=1}^{n} \overline{\mu}\left(F_3^i, F_1^i\right) \text{ par le lemme 3.1.1 et car } \overline{\mu}\left(Z, Z'\right) \geqslant 0 \text{ pour tout } Z, Z' \in \mathcal{F}_{(\mathcal{A},\mathcal{H})} \\
&= \sum_{i=1}^{n} \mu_{T_1}\left(F_3^i\right) \\
&= \mu_{T_1}\left(T_3\right)
\end{aligned}$$

■

Soit μ une métrique de perte d'information sur un vecteur de hiérarchies. Pour obtenir une estimation de l'écart entre le coût de généralisation pour μ d'une table généralisée et le coût pour μ de la table où toute l'information a été perdue, nous définissons l'*altération* d'une table généralisée pour μ. L'altération est le pourcentage du coût de généralisation pour μ de la table généralisée sur le coût de généralisation pour μ de la table généralisée à la racine (cf. définition 2.3.5 page 20). L'altération a le mérite d'être un pourcentage qui est une valeur facilement compréhensible quelque soit la métrique de perte d'information utilisée.

Définition 3.1.6 (Altération d'une table généralisée pour une métrique de perte d'information)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. Soit μ une métrique sur $\mathcal{H}$.

On définit l'application $\Lambda_{\mu,T} : \mathcal{T}^{gen}_{(T,\mathcal{H})} \to \mathbb{R}$ qui à toute table généralisée sur $(T, \mathcal{H})$ associe son *altération* pour μ :

$$\begin{array}{rccc} \Lambda_{\mu,T} : & \mathcal{T}^{gen}_{(T,\mathcal{H})} & \longrightarrow & \mathbb{R} \\ & T^{gen} & \longmapsto & \frac{\mu_T(T^{gen})}{\mu_T(T^*)} \times 100 \end{array},$$

avec T^* la table généralisée sur $(T, \mathcal{H})$ de la définition 2.3.5 page 20.

Exemple 3.1.2
Soit $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant.

Considérons H une hiérarchie associée à l'attribut Q et décrite dans la figure 2.4. La hiérarchie H est composée de trois feuilles (q_1, q_2 et q_3), d'un nœud de niveau 1 ($q_{1,2}$ qui est une généralisation de q_1 et q_2) et d'une racine ($q_{1,2,3}$).

Soit μ une métrique sur H définie par les poids des arêtes de H suivants : $\omega(q_1, q_{1,2}) = 1$, $\omega(q_2, q_{1,2}) = 2$, $\omega(q_{1,2}, q_{1,2,3}) = 3$ et $\omega(q_3, q_{1,2,3}) = 4$. La figure 3.1 page 28 montre la hiérarchie H avec les poids de la métrique μ.

Soit $T = \{E^1, E^2\}$ une table sur $\{Q\}$ définie par $E^1 = (q_1)$ et $E^2 = (q_3)$.

Soient $T^{gen}_1 = \{F^1_1, F^2_1\}$ et $T^{gen}_2 = \{F^1_2, F^2_2\}$ deux tables généralisées sur $(T, (H))$ définies par $F^1_1 = (q_{1,2})$, $F^2_1 = (q_3)$, $F^1_2 = (q_{1,2})$ et $F^2_2 = (q_{1,2,3})$.

Par la définition 2.3.5 page 20, $T^* = \{(q_{1,2,3}), (q_{1,2,3})\}$. Il s'agit de la table généralisée sur $(T, (H))$ dans laquelle chaque valeur quasi-identifiante a été généralisée à la racine de la hiérarchie.

Les tables sont à retouver en figure 3.2 page suivante.

Calculons les coûts de généralisation pour μ des tables T^{gen}_1, T^{gen}_2 et T^* par rapport à T :

$$\begin{aligned}
\mu_T\left(T^{gen}_1\right) &= \sum_{i=1}^{2} \mu_T\left(F^i_1\right) \text{ par la définition 3.1.5 de } \mu_T \\
&= \sum_{i=1}^{2} \overline{\mu}\left(F^i_1, E^i\right) \\
&= M_{\mu,H}\left(q_{1,2}, q_1\right) + M_{\mu,H}\left(q_1, q_{1,2}\right) + M_{\mu,H}\left(q_3, q_3\right) + M_{\mu,H}\left(q_3, q_3\right) \\
&= 0 + 1 + 0 + 0 \\
&= 1 \\
\mu_T\left(T^{gen}_2\right) &= \sum_{i=1}^{2} \mu_T\left(F^i_2\right) \\
&= \sum_{i=1}^{2} \overline{\mu}\left(F^i_2, E^i\right) \\
&= M_{\mu,H}\left(q_{1,2}, q_1\right) + M_{\mu,H}\left(q_1, q_{1,2}\right) + M_{\mu,H}\left(q_{1,2,3}, q_3\right) + M_{\mu,H}\left(q_3, q_{1,2,3}\right) \\
&= 0 + 1 + 0 + 4 \\
&= 5 \\
\mu_T\left(T^*\right) &= M_{\mu,H}\left(q_{1,2,3}, q_1\right) + M_{\mu,H}\left(q_1, q_{1,2,3}\right) + M_{\mu,H}\left(q_{1,2,3}, q_3\right) + M_{\mu,H}\left(q_3, q_{1,2,3}\right) \\
&= 0 + 4 + 0 + 4 \\
&= 8
\end{aligned}$$

T	Q
E^1	q_1
E^2	q_3

(a) Une table T sur $\{Q\}$ à deux enregistrements.

T_1^{gen}	Q
F_1^1	$q_{1,2}$
F_1^2	q_3

(b) Une table généralisée T_1^{gen} sur $(T,(H))$.

T_2^{gen}	Q
F_2^1	$q_{1,2}$
F_2^2	$q_{1,2,3}$

(c) Une table généralisée T_2^{gen} sur $(T,(H))$.

T^*	Q
F_*^1	$q_{1,2,3}$
F_*^2	$q_{1,2,3}$

(d) La table généralisée sur $(T,(H))$ correspondant à la perte totale de l'information de T.

FIGURE 3.2 – Une table T et trois de ses généralisations.

Ainsi, nous obtenons les altérations pour μ des tables T_1^{gen} et T_2^{gen} :

$$\begin{aligned}\Lambda_{\mu,T}(T_1^{gen}) &= \frac{\mu_T(T_1^{gen})}{\mu_T(T^*)} \times 100 \\ &= \frac{1}{8} \times 100 \\ &= 12{,}5\% \\ \Lambda_{\mu,T}(T_2^{gen}) &= \frac{\mu_T(T_2^{gen})}{\mu_T(T^*)} \times 100 \\ &= \frac{5}{8} \times 100 \\ &= 62{,}5\%\end{aligned}$$

Pour la métrique μ, les généralisations effectuées dans la table T_1^{gen} ne sont pas très importantes car l'altération pour cette dernière n'est que de 12,5%. En revanche, dans la table T_2^{gen}, les généralisations effectuées sont plus proches de celles effectuées dans la table T^* car l'altération pour μ de T_2^{gen} est de 62,5%. Pour μ, la table T_2^{gen} est plus altérée que la table T_1^{gen}.

3.2 Métriques étudiées

Dans cette section, nous allons présenter les métriques de perte d'information dont nous allons comparer les performances avec l'algorithme de k-anonymisation *GkAA* présenté en section 3.3.2 page 37. Nous commencerons par introduire trois métriques issues d'articles de la littérature. Puis, nous proposerons notre métrique de perte d'information, la *Lost Leaves Metric* ou *LLM*, ainsi que trois de ses variantes.

Pour définir les métriques, nous allons utiliser les mêmes notations et la définition 3.1.1 page 26. Nous allons donc expliciter les poids à mettre sur les arêtes des hiérarchies des attributs quasi-identifiants.

Les métriques étudiées reposent principalement sur trois notions : la hauteur d'un nœud dans la hiérarchie, le nombre de feuilles dans le sous-arbre enraciné en un nœud et des pondérations des attributs quasi-identifiants.

Commençons par poser des notations qui nous serviront à donner les poids sur les arêtes des hiérarchies pour les métriques étudiées.

Définition 3.2.1 (Nombre de feuilles)
Soient Q un attribut quasi-identifiant et H une hiérarchie de Q.

On définit l'application $\mathrm{nf} : \mathcal{N}_H \to \mathbb{N}$ qui à tout nœud de H associe le nombre de feuilles du sous-arbre enraciné en le nœud :

$$\begin{array}{rccc}\mathrm{nf}: & \mathcal{N}_H & \longrightarrow & \mathbb{N} \\ & v & \longmapsto & |V|\end{array},$$

avec $V = \{v' \in \mathcal{N}_H \,:\, \mathrm{niv}(v') = 0 \text{ et } v \text{ est une généralisation de } v'\}$.

Certaines métriques pondèrent les attributs quasi-identifiants en fonction de la hauteur de leur hiérarchie de généralisation. L'objectif est de préserver les informations contenues dans les attributs quasi-identifiants dont la hauteur de la hiérarchie de généralisation est petite. En effet, pour des attributs avec une telle hiérarchie, il y a peu de gradations dans les généralisations : les valeurs sont donc rapidement généralisées à la racine. En

donnant une forte pondération à un attribut, on peut espérer que les fusions de classes d'équivalence entraînant la généralisation des valeurs d'attributs avec de hautes hiérarchies seront préférées à des fusions entraînant la généralisation des valeurs de cet attribut.

Nous présentons deux pondérations dans la définition 3.2.2 : p_1 définie dans [40] et p_2 une pondération que nous proposons.

Définition 3.2.2 (Pondérations sur un ensemble de quasi-identifiants)
Soit $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m) \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies.

On définit l'application $p_1 : \mathcal{Q} \to \mathbb{R}$ qui à tout attribut quasi-identifiant de $\mathcal{Q}$ associe son poids pour p_1 :

$$\begin{array}{rccc} p_1 : & \mathcal{Q} & \longrightarrow & \mathbb{R} \\ & Q_j & \longmapsto & 1 - \frac{(h_{H_j}-1)^m}{\sum_{i=1}^{m}(h_{H_i}-1)^m} \end{array},$$

pour $j \in [\![1, m]\!]$.

On pose $h_{\max} = \max\limits_{1 \leqslant j \leqslant m} h_{H_j}$.

On définit l'application $p_2 : \mathcal{Q} \to \mathbb{R}$ qui à tout attribut quasi-identifiant de $\mathcal{Q}$ associe son poids pour p_2 :

$$\begin{array}{rccc} p_2 : & \mathcal{Q} & \longrightarrow & \mathbb{R} \\ & Q_j & \longmapsto & \frac{h_{\max}}{h_{H_j}} \end{array},$$

pour $j \in [\![1, m]\!]$.

Exemple 3.2.1
Considérons trois attributs quasi-identifiants Q_1, Q_2 et Q_3 auxquels sont associés les hiérarchies de généralisation H_1, H_2 et H_3. On pose $h_{H_1} = 5$, $h_{H_2} = 3$ et $h_{H_3} = 2$. La hiérarchie H_3 de l'attribut Q_3 est de hauteur 2 donc la seule option de généralisation possible est la perte totale de l'information contenue dans les feuilles de la hiérarchie.

Calculons les pondérations des trois attributs quasi-identifiants pour p_1 et p_2. On a $\sum_{i=1}^{m}(h_{H_i} - 1)^m = (5-1)^3 + (3-1)^3 + (2-1)^3 = 73$ et $h_{\max} = h_{H_1} = 5$.

$p_1(Q_1) = 1 - \frac{64}{73} \simeq 0{,}12$, $p_1(Q_2) = 1 - \frac{8}{73} \simeq 0{,}89$ et $p_1(Q_3) = 1 - \frac{1}{73} \simeq 0{,}99$.

$p_2(Q_1) = \frac{5}{5} = 1$, $p_2(Q_2) = \frac{5}{3} \simeq 1{,}67$ et $p_2(Q_3) = \frac{5}{2} = 2{,}5$.

Nous observons que le rapport entre la pondération de l'attribut à la plus basse hiérarchie et la pondération de l'attribut à la plus haute hiérarchie est de 8,25 pour p_1 et de 2,5 pour p_2. La pénalité donnée aux attributs dont les hiérarchies sont de petites tailles est donc plus importante pour p_1 que pour p_2.

3.2.1 Présentation des métriques étudiées

Pour définir les métriques utilisées dans notre étude, nous allons spécifier les poids à mettre sur les arêtes des hiérarchies des attributs quasi-identifiants. Les notations suivantes sont valables pour le reste de la section. Soit donc $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. On pose $h_{\max} = \max\limits_{1 \leqslant j \leqslant m} h_{H_j}$. Soit $j \in [\![1, m]\!]$. Soient $x, x' \in H_j$ tels qu'il existe une arête de x vers x' et $\mathrm{niv}(x) = \mathrm{niv}(x') - 1$. Soit r_j la racine de H_j.

3.2.1.1 Trois métriques de la littérature

Nous allons présenter trois métriques proposées dans des articles de la littérature : *Distortion*, *NCP* et *Total*. Pour les métriques *NCP* et *Total*, l'expression de leur définition avec notre modèle s'obtient trivialement à partir des formules présentées dans les articles [50] et [7]. En revanche, pour *Distortion*, l'opération est un peu plus délicate. Nous montrerons donc l'équivalence entre la définition originale de *Distortion* de l'article [28] et la définition que nous proposons.

Distortion La métrique *Distortion* a été introduite dans [28]. Elle tient compte du niveau des nœuds dans la hiérarchie. Nous allons utiliser la pondération p_1 comme cela a été fait dans [40].

Le poids à mettre sur l'arête (x, x') pour *Distortion* est :

$$\omega(x, x') = \frac{\frac{1}{h_{H_j} - \mathrm{niv}(x')}}{\sum_{i=1}^{h_{H_j}-1} \frac{1}{h_{H_j} - i}} \times p_1(Q_j). \tag{3.2.1}$$

Afin de faire le lien entre la définition originale de la métrique *Distortion* dans l'article [28] et la définition que nous proposons dans le paragraphe ci-dessus, nous allons reprendre les notations de l'article [28] puis nous partirons de l'expression de *Distortion* dans [28] et nous montrerons qu'elle est équivalente à l'équation 3.2.1 page précédente divisée par $p_1(Q_j)$. En effet, dans l'article [28], la pondération p_1 n'est pas utilisée.

Dans un premier temps, reprenons les notations de l'article [28]. Les niveaux de la hiérarchie H_j sont numérotés de 1 à h_{H_j} et sont tels que le niveau 1 contienne la racine de la hiérarchie et le niveau h_{H_j} contienne les feuilles de la hiérarchie. Il est à noter que cette numérotation des niveaux de la hiérarchie n'est pas la même que celle que nous utilisons. Pour rappel, avec les notations introduites dans le chapitre 2 page 13, les niveaux de H_j sont numérotées de 0 à $h_{H_j} - 1$ et sont tels que le niveau 0 contienne les feuilles de la hiérarchie et le niveau $h_{H_j} - 1$ contienne la racine de la hiérarchie. La relation suivante lie ces deux numérotations : si p est un niveau de H_j dans la numérotation de l'article [28] alors $h_{H_j} - p$ est le niveau de H_j correspondant dans notre numérotation. Inversement, si p est un niveau de H_j dans notre numérotation alors $h_{H_j} - p$ est le niveau de H_j correspondant dans la numérotation de l'article [28]. La numérotation des niveaux considérée est celle de l'article [28] sauf mention du contraire.

Dans la suite de l'article [28], les auteurs notent $w_{l,l-1}$ le poids entre les niveaux l et $l-1$ de H_j pour $l \in [\![2, h_{H_j}]\!]$. Plusieurs expressions sont proposées pour $w_{l,l-1}$, $l \in [\![2, h_{H_j}]\!]$. Nous avons choisi les *heigth weight* définis par $w_{l,l-1} = \frac{1}{(l-1)^\beta}$ avec $\beta = 1$ pour tout $l \in [\![2, h_{H_j}]\!]$.

Puis les auteurs définissent le coût de la transition du niveau p au niveau q, appelé *weighted hierarchical distance* et noté $\mathrm{WHD}(p, q)$, pour p et q deux niveaux de H_j tels que $p > q$ et $p, q \in [\![1, h_{H_j}]\!]$:

$$\mathrm{WHD}(p, q) = \frac{\sum_{l=q+1}^{p} w_{l,l-1}}{\sum_{l=2}^{h_{H_j}} w_{l,l-1}}.$$

Nous cherchons maintenant à obtenir le poids $\omega(x, x')$ à mettre sur l'arête (x, x') divisé par $p_1(Q_j)$ en partant de la définition de la weighted hierarchical distance ci-dessus. Avec les notations de [28], si on note p le niveau de x alors $p-1$ est le niveau de x'. Notre objectif est donc d'exprimer $\mathrm{WHD}(p, q)$ avec $q = p-1$ avec nos notations pour obtenir la formule $\frac{\omega(x,x')}{p_1(Q_j)} = \frac{\frac{1}{h_{H_j} - \mathrm{niv}(x')}}{\sum_{i=1}^{h_{H_j}-1} \frac{1}{h_{H_j}-i}}$.

$$\begin{aligned}
\mathrm{WHD}(p, p-1) &= \frac{\sum_{l=p}^{p} w_{l,l-1}}{\sum_{l=2}^{h_{H_j}} w_{l,l-1}} \\
&= \frac{w_{p,p-1}}{\sum_{l=2}^{h_{H_j}} w_{l,l-1}} \\
&= \frac{\frac{1}{p-1}}{\sum_{l=2}^{h_{H_j}} \frac{1}{l-1}} && \text{par définition des } w_{l,l-1} \\
&= \frac{\frac{1}{h_{H_j}-p'-1}}{\sum_{l'=0}^{h_{H_j}-2} \frac{1}{h-l'-1}} && \text{par les changements de variables } p' = h_{H_j} - p \text{ et } l' = h_{H_j} - l \\
&= \frac{\frac{1}{h_{H_j}-p'-1}}{\sum_{i=1}^{h_{H_j}-1} \frac{1}{h_{H_j}-i}} && \text{par le changement de variables } i = l' + 1
\end{aligned}$$

Pour obtenir la formule de l'équation 3.2.1 page précédente divisée par $p_1(Q_j)$, reste à montrer que $p' + 1 = \mathrm{niv}(x')$. Avec les notations de l'article [28], le niveau de nœud x' est $p-1$. Avec nos notations, le niveau du nœud x' est $\mathrm{niv}(x')$. Par la relation entre la numérotation des niveaux de H_j de l'article [28] et notre numérotation des niveaux de H_j, nous obtenons $p - 1 = h_{H_j} - \mathrm{niv}(x')$. Donc

$$\begin{aligned}
p' &= h_{H_j} - p && \text{par définition de } p' \\
&= h_{H_j} - p + 1 - 1 \\
&= h_{H_j} - (p-1) - 1 \\
&= \mathrm{niv}(x') - 1 && \text{car } p - 1 = h_{H_j} - \mathrm{niv}(x')
\end{aligned}$$

Pour conclure, nous obtenons l'égalité suivante

$$\mathrm{WHD}(p, p-1) = \frac{\frac{1}{h_{H_j} - \mathrm{niv}(x')}}{\sum_{i=1}^{h_{H_j}-1} \frac{1}{h_{H_j}-i}} = \frac{\omega(x, x')}{p_1(Q_j)},$$

qui montre le lien entre la définition de la métrique *Distortion* donnée dans l'article [28] et la définition des poids à mettre sur les arêtes proposée dans l'équation 3.2.1 page 33.

NCP La métrique *Normalized Certainty Penalty*, ou *NCP*, prend en compte le nombre de feuilles se généralisant en un nœud. Une étape de normalisation est effectuée en divisant par le nombre total de feuilles de la hiérarchie. Elle provient de [50].

Le poids à mettre sur l'arête (x, x') pour *NCP* est :

$$\omega(x, x') = \frac{\mathrm{nf}(x') - \mathrm{nf}(x)}{\mathrm{nf}(r_j)}.$$

Total Exposée dans [7], la métrique *Total* tient compte du niveau des nœuds dans la hiérarchie. Plus un nœud est proche de la racine, plus son coût de généralisation pour *Total* est élevé. Une étape de normalisation est effectuée en divisant par la hauteur de la hiérarchie moins 1.

Le poids à mettre sur l'arête (x, x') pour *Total* est :

$$\omega(x, x') = \frac{\mathrm{niv}(x') - \mathrm{niv}(x)}{h_{H_j} - 1}.$$

3.2.1.2 La métrique *LLM* et trois de ses variantes

Nous allons maintenant présenter la métrique de perte d'information *Lost Leaves Metric* issue de nos travaux ainsi que trois variantes de cette métrique. L'intérêt de présenter la métrique et trois variantes est d'étudier l'effet de la normalisation sur le coût du nœud et des pondérations sur les attributs quasi-identifiants.

LLM *LLM*, pour *Lost Leaves Metric*, est une métrique se basant sur le nombre de feuilles perdues quand une généralisation est effectuée. L'idée est la même que pour *NCP* à l'exception près que nous avons ajouté la pondération sur les attributs quasi-identifiants p_2 pour tenir compte des hauteurs des hiérarchies.

Le poids à mettre sur l'arête (x, x') pour *LLM* est :

$$\omega(x, x') = (\mathrm{nf}(x') - \mathrm{nf}(x)) \times p_2(Q_j).$$

NLLM La première variante de *LLM* est la *Normalized Lost Leaves Metric* ou *NLLM*. Une étape de normalisation est ajoutée en divisant par le nombre de feuilles dans la hiérarchie. Il s'agit de la métrique *NCP* à laquelle a été ajoutée la pondération sur les attributs quasi-identifiants p_2. L'objectif de cette variante est d'étudier les effets de la normalisation sur le coût de généralisation du nœud quand l'application de pondération sur les attributs quasi-identifiants utilisée est p_2.

Le poids à mettre sur l'arête (x, x') pour *NLLM* est :

$$\omega(x, x') = \frac{\mathrm{nf}(x') - \mathrm{nf}(x)}{\mathrm{nf}(r_j)} \times p_2(Q_j).$$

WLLM La deuxième variante de *LLM* est la *Wid Lost Leaves Metric* ou *WLLM*. L'application de pondérations sur les attributs quasi-identifiants utilisée est p_1. L'objectif est de comparer les performances des deux pondérations sur les attributs quasi-identifiants.

Le poids à mettre sur l'arête (x, x') pour *WLLM* est :

$$\omega(x, x') = (\mathrm{nf}(x') - \mathrm{nf}(x)) \times p_1(Q_j).$$

WNLLM La dernière variante de *LLM* est la *Wid Normalized Lost Leaves Metric* ou *WNLLM*. Dans cette métrique, une étape de normalisation sur le coût de généralisation du nœud est effectuée et l'application de pondérations sur les attributs quasi-identifiants est p_1. L'objectif de cette variante est d'étudier les effets de

	Distortion	*NCP*	*Total*	*LLM*	*NLLM*	*WLLM*	*WNLLM*
μ_{inter} dépend de la hauteur du nœud dans la hiérarchie	✓	×	✓	×	×	×	×
μ_{inter} dépend du nombre de feuilles perdues	×	✓	×	✓	✓	✓	✓
Normalisation sur μ_{inter}	✓	✓	✓	×	✓	×	✓
μ_{multi} pénalise les petites hiérarchies	p_1	×	×	p_2	p_2	p_1	p_1

TABLEAU 3.1 – Caractéristiques des deux phases pour les sept métriques étudiées

la normalisation sur le coût de généralisation du nœud quand l'application de pondération sur les attributs quasi-identifiants utilisée est p_1.

Le poids à mettre sur l'arête (x, x') pour *WNLLM* est :

$$\omega(x, x') = \frac{\mathrm{nf}(x') - \mathrm{nf}(x)}{\mathrm{nf}(r_j)} \times p_1(Q_j).$$

Pour mieux comprendre les différences dans les définitions des métriques, nous décomposons le calcul des poids sur les arêtes en deux phases. La première phase correspond au coût de généralisation du nœud. Elle est notée μ_{inter} pour μ une métrique. La seconde phase correspond à l'application d'une pondération sur les attributs quasi-identifiants. Elle est notée μ_{multi} pour μ une métrique.

Par exemple, pour la métrique *Distortion*, nous avons $Distortion_{inter} = \frac{\frac{1}{h_{H_j} - \mathrm{niv}(x')}}{\sum_{i=1}^{h_{H_j}-1} \frac{1}{h_{H_j}-i}}$ et $Distortion_{multi} = p_1(Q_j)$. Pour *NLLM*, nous avons $NLLM_{inter} = \frac{\mathrm{nf}(x') - \mathrm{nf}(x)}{\mathrm{nf}(r_j)}$ et $NLLM_{multi} = p_2(Q_j)$. Enfin, pour *NCP*, nous avons $NCP_{inter} = NLLM_{inter}$ et $NCP_{multi} = 1$.

Cela nous permet de caractériser les métriques étudiées en fonction de leurs deux phases. Le tableau 3.1 répertorie certaines caractéristiques des deux phases des métriques étudiées.

3.3 Table k-anonyme et algorithme de k-anonymisation

Dans ce chapitre, notre objectif est de comparer les performances de métriques de perte d'information quand elles sont utilisées pour guider un processus de k-anonymisation. Dans cette section, nous allons donc revenir sur les notions de table k-anonyme et de version k-anonyme d'une table (cf. section 3.3.1). Nous présenterons également en section 3.3.2 page suivante un algorithme permettant de k-anonymiser une table grâce à la technique de généralisation et dans lequel les fusions de classes d'équivalence à effectuer sont guidées par une métrique de perte d'information.

3.3.1 Table k-anonyme et version k-anonyme d'une table

D'après la définition de Samarati dans [41], une table est k-anonyme si chacun de ses enregistrements est indistinguable d'au moins $k-1$ autres enregistrements de la table par rapport à l'ensemble des attributs quasi-identifiants. Comme expliqué dans le chapitre 1 page 5, la k-anonymité vise à protéger contre la divulgation d'identité en regroupant les enregistrements de la table en ensembles dont le cardinal est supérieur à k.

Dans la définition 2.1.5 page 14 du chapitre 2 page 13, nous avons présenté une relation d'équivalence sur une table. Deux enregistrements d'une table sont dans la même classe d'équivalence s'ils ont les mêmes valeurs pour les attributs quasi-identifiants. Ainsi, une table est k-anonyme si toutes ses classes d'équivalence contiennent au moins k enregistrements. La définition 3.3.1 reprend les notations du chapitre 2 page 13 pour définir la notion de table k-anonyme.

Définition 3.3.1 (Table k-anonyme)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$. Soit $k \in \mathbb{N}^*$.

T est une *table k-anonyme* si toutes ses classes d'équivalence sont de tailles supérieures à k, c'est-à-dire :

$$\forall C \in \mathcal{C}(T), |C| \geqslant k.$$

Dans le chapitre 2 page 13, nous avons présenté la notion de table généralisée (cf. définition 2.3.3 page 19). Nous associons les notions de table généralisée et de table k-anonyme pour définir une version k-anonyme d'une table en définition 3.3.2.

Définition 3.3.2 (Table k-anonyme sur $(T, \mathcal{H})$)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. Soit T^{gen} une table généralisée sur $(T, \mathcal{H})$. Soit $k \in [\![1, n]\!]$.

On dit que T^{gen} est une *table k-anonyme sur* $(T, \mathcal{H})$ ou que T^{gen} est une *version k-anonyme* de T si toutes ses classes d'équivalence contiennent au moins k enregistrements, c'est-à-dire :

$$\forall C \in \mathcal{C}(T^{gen}), |C| \geqslant k.$$

A toute table T sur $(\mathcal{A}, \mathcal{H})$, nous associons un entier correspondant au k maximal pour lequel la table T respecte le modèle de k-anonymité.

Définition 3.3.3 (Paramètre de k-anonymité)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de m attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$. Soit $n \in \mathbb{N}^*$

On définit l'application $\kappa : \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} \to \mathbb{N}^*$ par

$$\begin{array}{rccl} \kappa : & \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} & \longrightarrow & \mathbb{N}^* \\ & T & \longmapsto & \min\{|C| \text{ pour } C \in \mathcal{C}(T)\} \end{array}.$$

Pour $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$, $\kappa(T)$ est appelé *paramètre de k-anonymité de T*.

L'application κ est bien définie car l'ensemble $\{|C| \text{ pour } C \in \mathcal{C}(T)\}$ est un sous-ensemble fini de $\mathbb{N}^*$ et $\mathbb{N}^*$ est totalement ordonné.

Remarques 3.3.1
Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table généralisée sur $(\mathcal{A}, \mathcal{H})$ à n enregistrements avec $\mathcal{A}$ un ensemble d'attributs et $\mathcal{H}$ un uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{A}$.

1. T est une table $\kappa(T)$-anonyme. En effet, si $C \in \mathcal{C}(T)$ est une classe d'équivalence de T alors, par définition de $\kappa(T)$, on a $|C| \geqslant \kappa(T)$ donc T est $\kappa(T)$-anonyme.
2. T est une table k-anonyme pour tout $k \in [\![1, \kappa(T)]\!]$. En effet, soit $k \in [\![1, \kappa(T)]\!]$. Montrons que T est k-anonyme. D'après la remarque précédente, T est $\kappa(T)$-anonyme donc pour toute classe d'équivalence C de T, on a $|C| \geqslant \kappa(T)$. Or $k \leqslant \kappa(T)$ donc $|C| \geqslant \kappa(T) \geqslant k$. Ainsi, $|C| \geqslant k$ pour toute classe d'équivalence C de T. T est k-anonyme.

3.3.2 Algorithme de k-anonymisation

Dans cette section, nous allons présenter un algorithme permettant de produire une version k-anonyme d'une table. Nommé *Greedy k-Anonymization Algorithm* et abrégé en *GkAA*, cet algorithme s'inspire de l'algorithme *KACA* proposé par Li dans [28]. L'idée générale est de fusionner les classes d'équivalence de la table jusqu'à ce que toutes ses classes d'équivalence soient de tailles supérieures à k. À chaque tour, la fusion de deux classes d'équivalence à effectuer est choisie de telle sorte que le coût de généralisation pour une métrique de la table dans laquelle la fusion a été effectuée soit minimisé.

L'algorithme 1 page suivante est une formalisation de *GkAA*. *GkAA* prend en entrées $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble d'attributs, $\mathcal{H}$ un vecteur de hiérachies de généralisation, T une table sur $(\mathcal{A}, \mathcal{H})$, μ une métrique de perte d'information sur $\mathcal{H}$ et k un entier. En sortie, l'algorithme founit une version k-anonyme de la table T.

Tant que la table T n'est pas k-anonyme, l'algorithme va effectuer la fusion de deux classes d'équivalence. La première classe d'équivalence, notée C_s, est choisie arbitrairement parmi les classes ne respectant pas la k-anonymité (c'est-à-dire parmi les classes de tailles strictement inférieures à k). Le choix de la seconde classe d'équivalence à fusionner, notée C, est déterminé par la métrique de perte d'information μ : il s'agira de l'une des classes d'équivalence de coût de généralisation avec C_s minimal pour μ. Les classes C_s et C sont ensuite fusionnées dans T.

Remarque 3.3.1
GkAA est un algorithme déterministe : deux exécutions avec les mêmes paramètres en entrées et avec les mêmes configurations donnent le même résultat en sortie. Cela est dû au choix de C_s à chaque tour. En effet, à chaque tour de *GkAA*, les classes d'équivalence sont triées en fonction de leur taille. La classe C_s choisie est alors la première classe de cette liste. Ainsi, si la façon de trier les classes d'équivalence ne changent pas entre deux exécutions, les mêmes classes C_s seront choisies à chaque tour des deux exécutions.

Algorithme 1 Greedy k-Anonymization Algorithm

Entrées: $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{A}$, T une table sur $(\mathcal{A}, \mathcal{H})$, μ une métrique sur $\mathcal{H}$, $k \in \mathbb{N}^*$

Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

1: **procédure** *GkAA*($\mathcal{A}$, $\mathcal{H}$, T, μ, k)
2: **tant que** $\kappa(T) < k$ **faire**
3: On choisit arbitrairement une classe d'équivalence C_s de T telle que $C_s = |\kappa(T)|$
4: On cherche C une classe d'équivalence de T différente de C_s telle que $\overline{\mu}(C_s, C)$ soit minimal
5: $T \leftarrow gen_T(C_s \cup C)$
6: **fin tant que**
7: Retourne T
8: **fin procédure**

T	Q
E^1	q_1
E^2	q_2
E^3	q_3
E^4	q_2

(a) État initial

T	Q
E^1	$q_{1,2,3}$
E^2	q_2
E^3	$q_{1,2,3}$
E^4	q_2

(b) Table à la fin du premier tour de l'exécution : la table est 2-anonyme

FIGURE 3.3 – État de la table à la fin de chaque tour d'une première exécution de *GkAA* pour $k = 2$

Exemple 3.3.1
Soit $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant.

Considérons H une hiérarchie associée à l'attribut Q et décrite dans la figure 2.4. La hiérarchie H est composée de trois feuilles (q_1, q_2 et q_3), d'un nœud de niveau 1 ($q_{1,2}$ qui est une généralisation de q_1 et q_2) et d'une racine ($q_{1,2,3}$).

Soit μ une métrique de perte d'information sur H définie par $\omega(q_1, q_{1,2}) = 1$, $\omega(q_2, q_{1,2}) = 2$, $\omega(q_{1,2}, q_{1,2,3}) = 3$ et $\omega(q_3, q_{1,2,3}) = 4$. La figure 3.1 page 28 montre la hiérarchie H avec les poids de la métrique μ.

Considérons la table sur $\{Q\}$ décrite par la figure 3.6 page 40 et notée T. T a quatre enregistrements $E^1 = (q_1)$, $E^2 = (q_2)$, $E^3 = (q_3)$ et $E^4 = (q_2)$.

Nous allons décrire les tours effectués lors de l'exécution $GkAA(\{Q\}, (H), T, \mu, 2)$. L'état de la table T à la fin de chaque tour de l'exécution est en figure 3.3 pour le premier cas présenté et en figure 3.4 page ci-contre pour le second.

Au premier tour de l'exécution, T a trois classes d'équivalence. Un tri est effectué pour les ranger dans l'ordre croissant de leur taille : $C_1 = \{E^3\}$, $C_2 = \{E^1\}$ et $C_3 = \{E^2, E^4\}$. Il y a deux classes de taille 1. La classe C_s sélectionnée est la première de la liste c'est-à-dire $C_1 = \{E^3\}$. On cherche ensuite la classe $C \in \{C_2, C_3\}$ telle que le coût de généralisation pour μ de C et C_1 soit minimal. On a $\overline{\mu}(C_1, C_2) = 4+4 = 8$ et $\overline{\mu}(C_1, C_3) = 4+5\times 2 = 14$. La fusion effectuée dans T est donc celle de C_1 et C_2. À la fin du premier tour, la table T obtenue a deux classes d'équivalence de taille 2 donc l'algorithme s'arrête (cf. figure 3.3b).

Supposons maintenant que le tri effectué pour ranger les classes d'équivalence dans l'ordre croissant de leur taille donne la liste suivante : $C_1 = \{E^1\}$, $C_2 = \{E^3\}$ et $C_3 = \{E^2, E^4\}$. La classe C_s sélectionnée est alors $C_1 = \{E^1\}$. On cherche ensuite la classe $C \in \{C_2, C_3\}$ telle que le coût de généralisation pour μ de C et C_1 soit minimal. Cette fois-ci, on obtient $\overline{\mu}(C_1, C_2) = 4 + 4 = 8$ et $\overline{\mu}(C_1, C_3) = 1 + 2 \times 2 = 5$. La fusion effectuée dans T est donc celle de C_1 et C_3. À la fin du premier tour, la table T obtenue a trois classes d'équivalence et n'est pas 2-anonyme, l'exécution se poursuit donc (cf. figure 3.4b page ci-contre). À la fin du second tour de cette exécution, la table obtenue contient une classe d'équivalence de taille 4 et est 2-anonyme (cf. figure 3.4c page suivante). Elle est cependant différente de la table obtenue à la fin de l'exécution avec le premier tri.

Dans l'exemple 3.3.1, nous avons illustré le fait que le tri des classes d'équivalence déterminant le choix de la classe C_s a un impact sur la table k-anonyme produite. Il est donc important de spécifier les configurations d'exécution de *GkAA* lors d'expérimentations afin que ces dernières soient reproductibles.

Remarque 3.3.2
Lors d'une exécution de *GkAA*, le choix de la classe d'équivalence C à fusionner avec C_s ne dépend pas de la valeur de k passée en entrées de *GkAA* car C est choisie parmi toutes les classes d'équivalence différentes de C_s.

T	Q
E^1	q_1
E^2	q_2
E^3	q_3
E^4	q_2

(a) État initial

T	Q
E^1	$q_{1,2}$
E^2	$q_{1,2}$
E^3	q_3
E^4	$q_{1,2}$

(b) Table à la fin du premier tour de l'exécution

T	Q
E^1	$q_{1,2,3}$
E^2	$q_{1,2,3}$
E^3	$q_{1,2,3}$
E^4	$q_{1,2,3}$

(c) Table à la fin du second tour de l'exécution : la table est 2-anonyme

FIGURE 3.4 – État de la table à la fin de chaque tour d'une seconde exécution de $GkAA$ pour $k = 2$

T	Q
E^1	q_1
E^2	q_2
E^3	q_3
E^4	q_2

(a) État initial

T	Q
E^1	$q_{1,2,3}$
E^2	q_2
E^3	$q_{1,2,3}$
E^4	q_2

(b) Table à la fin du premier tour de l'exécution : la table est 2-anonyme

T	Q
E^1	$q_{1,2,3}$
E^2	$q_{1,2,3}$
E^3	$q_{1,2,3}$
E^4	$q_{1,2,3}$

(c) Table à la fin du second tour de l'exécution : la table est 4-anonyme

FIGURE 3.5 – État de la table à la fin de chaque tour d'une exécution de $GkAA$ pour $k = 4$

Soient $\mathcal{A}$ un ensemble d'attributs, $\mathcal{H}$ un vecteur de hiérarchies de généralisation, T une table sur $(\mathcal{A}, \mathcal{H})$ et μ une métrique de perte d'information sur $\mathcal{H}$. Soient $k < k'$ deux entiers. Soit x_k le nombre de tours effectués lors de l'exécution de $GkAA$ pour k. Les x_k premiers tours de l'exécution $GkAA(\mathcal{A}, \mathcal{H}, T, \mu, k')$ sont les mêmes que les x_k tours de l'exécution $GkAA(\mathcal{A}, \mathcal{H}, T, \mu, k)$.

Plus particulièrement, la table obtenue à la fin de l'exécution $GkAA(\mathcal{A}, \mathcal{H}, T, \mu, k)$ est une table intermédiaire obtenue à la fin d'un tour de l'exécution $GkAA(\mathcal{A}, \mathcal{H}, T, \mu, k')$.

Notons $T_{\mu,k}$ et $T_{\mu,k'}$ les tables obtenues à la fin des exécutions $GkAA(\mathcal{A}, \mathcal{H}, T, \mu, k)$ et $GkAA(\mathcal{A}, \mathcal{H}, T, \mu, k')$ respectivement. $T_{\mu,k'}$ est une table généralisée sur $(T_{\mu,k}, \mathcal{H})$.

L'exemple 3.3.2 illustre la remarque 3.3.2 page ci-contre.

Exemple 3.3.2
Soit $Q = \{q_1, q_2, q_3\}$ un attribut quasi-identifiant dont une hiérarchie H est donnée en figure 2.4 page 17. Soit T une table sur Q décrite dans la figure 3.6 page suivante. Soit μ une métrique de perte d'information sur H définie par $\omega(q_1, q_{1,2}) = 1$, $\omega(q_2, q_{1,2}) = 2$, $\omega(q_{1,2}, q_{1,2,3}) = 3$ et $\omega(q_3, q_{1,2,3}) = 4$.

Nous allons décrire les tours effectués lors de l'exécution $GkAA(\{Q\}, (H), T, \mu, 2)$ et de l'exécution $GkAA(\{Q\}, (H), T, \mu, 4)$. L'état de la table T à la fin de chaque tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 2)$ est en figure 3.3 page précédente et l'état de la table T à la fin de chaque tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 4)$ est en figure 3.5.

Au premier tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 2)$, T a trois classes d'équivalence. Un tri est effectué pour les ranger dans l'ordre croissant de leur taille : $C_1 = \{E^3\}$, $C_2 = \{E^1\}$ et $C_3 = \{E^2, E^4\}$. Il y a deux classes de taille 1. La classe C_s sélectionnée est la première de la liste c'est-à-dire $C_1 = \{E^3\}$. On cherche ensuite la classe $C \in \{C_2, C_3\}$ telle que le coût de généralisation pour μ de C et C_1 soit minimal. On a $\overline{\mu}(C_1, C_2) = 4 + 4 = 8$ et $\overline{\mu}(C_1, C_3) = 4 + 5 \times 2 = 14$. La fusion effectuée dans T est donc celle de C_1 et C_2. À la fin du premier tour, la table T obtenue a deux classes d'équivalence de taille 2 donc l'algorithme s'arrête (cf. figure 3.3b page précédente).

Au premier tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 4)$, T a trois classes d'équivalence. Un tri est effectué pour les ranger dans l'ordre croissant de leur taille : $C_1 = \{E^3\}$, $C_2 = \{E^1\}$ et $C_3 = \{E^2, E^4\}$. Il y a deux classes de taille 1. La classe C_s sélectionnée est la première de la liste c'est-à-dire $C_1 = \{E^3\}$. On cherche ensuite la classe $C \in \{C_2, C_3\}$ telle que le coût de généralisation pour μ de C et C_1 soit minimal. On a $\overline{\mu}(C_1, C_2) = 4 + 4 = 8$ et $\overline{\mu}(C_1, C_3) = 4 + 5 \times 2 = 14$. La fusion effectuée dans T est donc celle de C_1 et C_2. À la fin du premier tour, la table T obtenue a deux classes d'équivalence de taille 2 (cf. figure 3.5b). Nous constatons que le déroulement du premier tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 4)$ est exactement le même que le déroulement du premier tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 2)$. $GkAA$ produit donc la table 2-anonyme résultat de l'exécution $GkAA(\{Q\}, (T, (H)), \mu, 2)$ lors de l'exécution $GkAA(\{Q\}, (H), T, \mu, 4)$.

À la fin du second tour de l'exécution $GkAA(\{Q\}, (H), T, \mu, 4)$, la table T obtenue a une classe d'équivalence et est 4-anonyme, l'algorithme s'arrête donc (cf. figure 3.5c).

T	Q
E^1	q_1
E^2	q_2
E^3	q_3
E^4	q_2

FIGURE 3.6 – Représentation d'une table sur Q.

3.4 Expérimentations

Dans cette section, nous allons étudier les performances de métriques de perte d'information lorsqu'elles sont utilisées dans un processus de k-anonymisation. Pour cela, nous allons revenir sur le protocole expérimental mis en place en section 3.4.1. Puis nous analyserons les résultats obtenus en section 3.4.2 page 45.

3.4.1 Protocole expérimental

Nous souhaitons comparer les performances de métriques de perte d'information lorsqu'elles sont utilisées pour guider un processus de k-anonymisation. Pour cela, nous présentons le protocole expérimental suivant.

Nous allons mener des expérimentations sur deux tables de 30 162 enregistrements : *Adult data set* et *florida_30162* (cf. section 2.4 page 22). Tous les attributs des tables seront considérés comme quasi-identifiants.

Soit $\mathcal{M} = \{Distortion, NCP, Total, LLM, NLLM, WLLM, WNLLM\}$ l'ensemble des métriques de perte d'information que nous allons étudier (cf. section 3.2 page 32).

Pour chaque table, pour chaque métrique de $\mathcal{M}$, nous allons appliquer l'algorithme de k-anonymité *GkAA* (cf. algorithme 1 page 38) pour 14 valeurs de k entre 3 et 15 000. En d'autres termes, pour une table T, pour une métrique $\mu \in \mathcal{M}$, pour un entier k, nous allons construire une version k-anonyme de T avec *GkAA* en guidant les fusions de classes d'équivalence à effectuer grâce à la métrique μ. La table k-anonyme obtenue est notée $T_{\mu,k}$. Il est à noter que le k demandé en entrées de *GkAA* n'est pas forcément égal au paramètre de k-anonymité de la table k-anonyme produite. L'algorithme *GkAA* s'arrête quand le paramètre de k-anonymité de la table est supérieur au k demandé.

Notons $K = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000, 10\,000, 15\,000]$ l'ensemble des valeurs de k choisies. En pratique, dans nos expérimentations, nous n'avons pas lancé l'algorithme *GkAA* pour les 14 valeurs de k, pour les sept métriques et pour les deux tables. Cela aurait représenté $14 \times 7 \times 2 = 56$ exécutions de *GkAA*. A partir des remarques 3.3.1 page 37 et 3.3.2 page 38 de la section 3.3.2 page 37, nous avons procédé comme suit : pour $T \in \{Adult, florida\_30162\}$, pour $\mu \in \mathcal{M}$, nous avons exécuté *GkAA* avec $k = \max(K)$ c'est-à-dire $k = 15\,000$. Lors de cette exécution, nous avons sauvegardé la première table k'-anonyme obtenue pour tout $k' \in K$.

Ainsi, nous n'avons à exécuter qu'une fois *GkAA* pour chaque couple (table, métrique) pour obtenir les 14 tables k-anonymes pour $k \in K$. Cela représente $7 \times 2 = 14$ exécutions de *GkAA* au total.

Pour comparer les tables k-anonymes obtenues en utilisant différentes métriques dans *GkAA*, nous nous intéressons à trois critères de qualité :

— l'altération moyenne de la table k-anonyme sur $\mathcal{M}$
— le pourcentage de valeurs généralisées sur le nombre total de valeurs
— le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs

En section 3.1 page 26, nous avons expliqué comment calculer l'altération d'une table pour une certaine métrique de perte d'information. Dans notre étude, nous allons construire des tables k-anonymes dont nous aimerions connaître la quantité de perte d'information par rapport à la table d'origine. Cependant, nous disposons d'un ensemble $\mathcal{M}$ de sept métriques de perte d'information. Calculer l'altération d'une table k-anonyme pour chacune des métriques donnerait alors sept valeurs d'altération : cela est difficilement représentable graphiquement de manière concise. Nous proposons donc, pour chaque table k-anonyme, de calculer la moyenne des altérations obtenues pour les métriques de $\mathcal{M}$. Ainsi, pour une table k-anonyme, nous obtenons une unique valeur représentant l'*altération moyenne* de cette table sur $\mathcal{M}$.

Définition 3.4.1 (Altération moyenne sur $\mathcal{M}$)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de m attributs quasi-identifiants et d'un attribut sensible. Soit $\mathcal{H}$ un m-uplet de hiérarchies de $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. Soit $\mathcal{M}$ un ensemble de métriques de perte d'information.

On définit l'application $\Lambda_{\mathcal{M},T} : \mathcal{T}^{gen}_{(T,\mathcal{H})} \to \mathbb{R}$ qui à toute table généralisée sur $(T, \mathcal{H})$ associe son *altération*

moyenne sur $\mathcal{M}$:

$$\begin{array}{rccc} \Lambda_{\mathcal{M},T} : & \mathcal{T}^{gen}_{(T,\mathcal{H})} & \longrightarrow & \mathbb{R} \\ & T^{gen} & \longmapsto & \frac{1}{|\mathcal{M}|} \times \sum_{\mu \in \mathcal{M}} \Lambda_{\mu,T}(T^{gen}) \end{array} .$$

Les définitions des métriques étudiées étant différentes sur certains points (cf. tableau 3.1 page 36), nous aimerions savoir si certaines métriques impliquent plus de généralisations des valeurs que d'autres lors du processus de k-anonymisation. Pour cela, nous allons calculer le pourcentage de valeurs généralisées dans une version k-anonyme d'une table sur le nombre total de valeurs.

Définition 3.4.2 (Pourcentage de valeurs généralisées)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de m attributs quasi-identifiants et d'un attribut sensible. Soit $\mathcal{H}$ un m-uplet de hiérarchies de $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$.

On définit l'application $v_{\text{gen},T} : \mathcal{T}^{gen}_{(T,\mathcal{H})} \to \mathbb{N}$ qui à toute table généralisée sur $(T, \mathcal{H})$ associe son *nombre de valeurs généralisées* par rapport à T :

$$\begin{array}{rccc} v_{\text{gen},T} : & \mathcal{T}^{gen}_{(T,\mathcal{H})} & \longrightarrow & \mathbb{N} \\ & T^{gen} & \longmapsto & |\{(i,j) \in [\![1,n]\!] \times [\![1,m]\!] \ : \ \text{niv}(f^i_j) > \text{niv}(e^i_j)\}| \end{array} ,$$

avec $T = \{E^1, \ldots, E^n\}$, $E^i = (e^i_1, \ldots, e^i_m, s^i)$ pour tout $i \in [\![1,n]\!]$ et $T^{gen} = \{F^1, \ldots, F^n\}$, $F^i = (f^i_1, \ldots, f^i_m, s^i)$ pour tout $i \in [\![1,n]\!]$.

Nous considérerons le pourcentage de valeurs généralisées par rapport à T sur le nombre total de valeurs :

$$p_{\text{gen},T}(T^{gen}) = \frac{v_{\text{gen},T}(T^{gen})}{n \times m} \times 100.$$

Le pourcentage de valeurs généralisées d'une table k-anonyme sur le nombre total de valeurs ne permet pas d'avoir une idée du nombre de valeurs pour lesquelles toute l'information a été perdue. En d'autres termes, nous aimerions connaître le nombre de valeurs d'une table qui ont été généralisées en la racine de la hiérarchie dans une version k-anonyme : nous dirons que ces valeurs sont *généralisées à la racine*. Nous calculons donc le pourcentage de valeurs généralisées à la racine dans une version k-anonyme d'une table sur le nombre total de valeurs.

Définition 3.4.3 (Pourcentages de valeurs généralisées à la racine)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de m attributs quasi-identifiants et d'un attribut sensible. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Notons r_j la racine de H_j pour tout $j \in [\![1,m]\!]$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$.

On définit l'application $v_{\text{racine},T} : \mathcal{T}^{gen}_{(T,\mathcal{H})} \to \mathbb{N}$ qui à toute table généralisée sur $(T, \mathcal{H})$ associe son *nombre de valeurs généralisées à la racine* par rapport à T :

$$\begin{array}{rccc} v_{\text{racine},T} : & \mathcal{T}^{gen}_{(T,\mathcal{H})} & \longrightarrow & \mathbb{N} \\ & T^{gen} & \longmapsto & |\{(i,j) \in [\![1,n]\!] \times [\![1,m]\!] \ : \ f^i_j = r_j\}| \end{array} ,$$

avec $T^{gen} = \{F^1, \ldots, F^n\}$, $F^i = (f^i_1, \ldots, f^i_m, s^i)$ pour tout $i \in [\![1,n]\!]$.

Nous considérerons le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs :

$$p_{\text{racine},T}(T^{gen}) = \frac{v_{\text{racine},T}(T^{gen})}{n \times m} \times 100.$$

L'altération moyenne sur $\mathcal{M}$, le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs sont des critères à minimiser. Pour une table T, pour une valeur de k, pour un critère *crit*, pour deux métriques de perte d'information μ et ν, si $crit(T_{\mu,k}) < crit(T_{\nu,k})$ alors la métrique μ a permis de produire une version k-anonyme de T de meilleure qualité pour le critère *crit* que la métrique ν.

Pour l'altération moyenne sur $\mathcal{M}$ et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs, si une table k-anonyme obtient un résultat de 0% pour un de ces critères, cela signifie qu'aucune information n'a été perdue par rapport à la table d'origine (aucune généralisation n'a été effectuée). Au contraire, si une table k-anonyme obtient un résultat de 100% pour un de ces critères, cela signifie que toute l'information a été perdue par rapport à la table d'origine (toutes les valeurs quasi-identifiantes de la table ont été généralisées à la racine, la table k-anonyme est donc la table T^* de la définition 2.3.5 page 20).

Pour le pourcentage de valeurs généralisées sur le nombre total de valeurs, si une table k-anonyme obtient un résultat de 0% pour ce critère, cela signifie à nouveau qu'aucune information n'a été perdue par rapport à la

table d'origine. En revanche, si une table k-anonyme obtient un résultat de 100% pour ce critère, cela signifie que toutes les valeurs quasi-identifiantes ont subi une généralisation mais pas forcément à la racine. Cette table k-anonyme peut contenir plus d'information qu'une autre table k-anonyme ayant obtenu un pourcentage de valeurs généralisées sur le nombre total de valeurs de 100%.

Nous avons réalisé un graphique pour chaque critère et pour chacune des deux tables : figure 3.7 page suivante pour *Adult data set* et figure 3.8 page 44 pour *florida_30162*. Sur chaque graphique, l'axe des abscisses représente les k demandés. La plage de valeurs étant étendue, l'échelle n'est volontairement pas respectée pour cet axe. Ainsi, les résultats pour les petites valeurs de k sont lisibles.

Sur chaque graphique, il y a sept courbes correspondant chacune à une métrique de perte d'information. Un point d'une courbe correspond au résultat du critère considéré pour la table k-anonyme obtenue avec *GkAA* en utilisant la métrique correspondant à la courbe. Par exemple, sur le graphique de la figure 3.7a page suivante, le premier point de la courbe orange est l'altération moyenne sur $\mathcal{M}$ de $Adult_{NLLM,3}$ (la version 3-anonyme de *Adult data set* obtenue avec *GkAA* en utilisant la métrique *NLLM*). De même, le cinquième point de la courbe verte est l'altération moyenne sur $\mathcal{M}$ de $Adult_{NCP,20}$ (la version 20-anonyme de *Adult data set* obtenue avec *GkAA* en utilisant la métrique *NCP*).

Pour chaque table $T \in \{Adult, florida\_30162\}$, pour chaque critère de qualité, pour chaque métrique $\mu \in \mathcal{M}$, nous aimerions synthétiser en une unique valeur les résultats obtenus pour ce critère par les tables k-anonymes produites avec *GkAA* en utilisant cette métrique. Pour cela, nous aimerions utiliser la notion d'*aire sous la courbe*. En effet, pour une fonction f continue sur un intervalle $[a, b] \subset \mathbb{R}$, la valeur moyenne de f sur $[a, b]$ est $m \in \mathbb{R}$ tel que

$$m = \frac{1}{b-a} \int_a^b f(x)dx.$$

Plusieurs méthodes existent pour approcher l'aire sous la courbe. Nous avons choisi la méthode des trapèzes qui est une méthode d'ordre 2 (cf. définition 3.4.4).

Définition 3.4.4 (La méthode des trapèzes)
Soient $a, b \in \mathbb{R}$. Soit f une fonction continue de $[a, b]$ dans $\mathbb{R}$. Soit $(x_i)_{i \in [\![0,n]\!]}$ une subdivision de l'intervalle $[a, b]$ en $n \in \mathbb{N}^*$ intervalles de longueur $h_i = x_{i+1} - x_i$ pour $i \in [\![0, n]\!]$.

On peut approcher la valeur de l'intégrale de f sur $[a, b]$ par

$$\int_a^b f(x)dx \simeq \frac{1}{2} \sum_{i=0}^{n} (f(x_i) + f(x_{i+1})) \times (x_{i+1} - x_i).$$

Dans notre étude, l'intervalle $[a, b]$ considéré est $[3, 15\,000]$. Chaque courbe comporte 14 points correspondant aux 14 valeurs de k choisies. Pour que la méthode des trapèzes donne une bonne approximation de l'aire sous la courbe, il faut pouvoir subdiviser l'intervalle en un nombre important de sous-intervalles. Or nous ne pouvons pas augmenter le nombre de sous-intervalles car nous n'avons pas de données pour toutes les valeurs de k comprises entre 3 et 15 000. En revanche, si la courbe est monotone, la méthode des trapèzes donnera tout de même une bonne approximation de l'aire sous la courbe. Nous allons donc vérifier si les courbes obtenues pour les trois critères de qualité sont monotones (cf. proposition 3.4.1)

Lemme 3.4.1
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies de $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $n \in \mathbb{N}^*$.

Soient $T_1 = \{F_1^1, \ldots, F_1^n\}$ et $T_2 = \{F_2^1, \ldots, F_2^m\}$ deux tables sur $(\mathcal{A}, \mathcal{H})$ de cardinal n. Pour tout $l \in [\![1, 2]\!]$, pour tout $i \in [\![1, n]\!]$, posons $F_l^i = (f_{l,1}^i, \ldots, f_{l,m}^i, s^i)$.

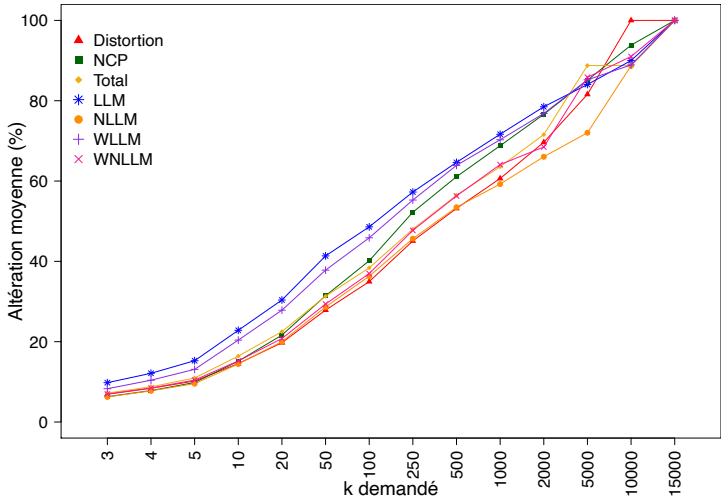
Si T_2 est une table généralisée sur $(T_1, \mathcal{H})$ alors

$$\forall i \in [\![1, n]\!], \forall j \in [\![1, m]\!], \text{niv}(f_{2,j}^i) \geqslant \text{niv}(f_{1,j}^i).$$

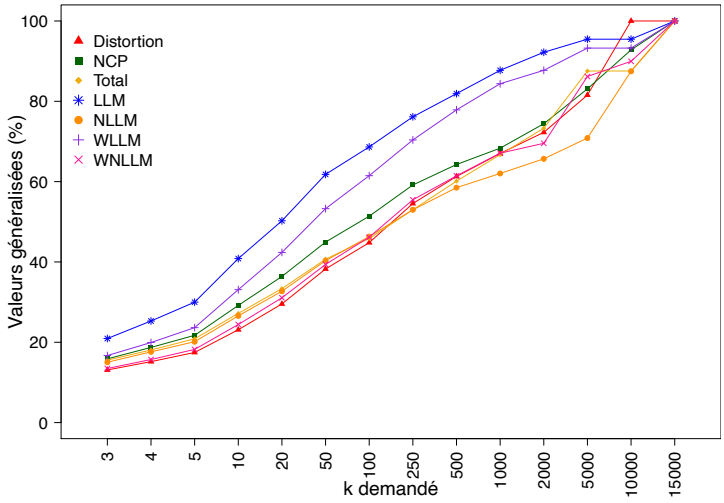
Démonstration : T_2 est une table généralisée sur $(T_1, \mathcal{H})$ donc, pour tout $i \in [\![1, n]\!]$, F_2^i est une généralisation de F_1^i d'après la définition 2.3.3 page 19. D'après la définition 2.3.2 page 19, pour tout $i \in [\![1, n]\!]$, pour tout $j \in [\![1, m]\!]$, $f_{2,j}^i$ est une généralisation de $f_{1,j}^i$. Donc, d'après la définition 2.3.1 page 19, $f_{2,j}^i$ appartient au chemin de $f_{1,j}^i$ à la racine de H_j. Par conséquent, pour tout $i \in [\![1, n]\!]$ et pour tout $j \in [\![1, m]\!]$, $\text{niv}(f_{2,j}^i) \geqslant \text{niv}(f_{1,j}^i)$. ■

Proposition 3.4.1
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit $\mathcal{H}$ un m-uplet de hiérarchies de $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$.

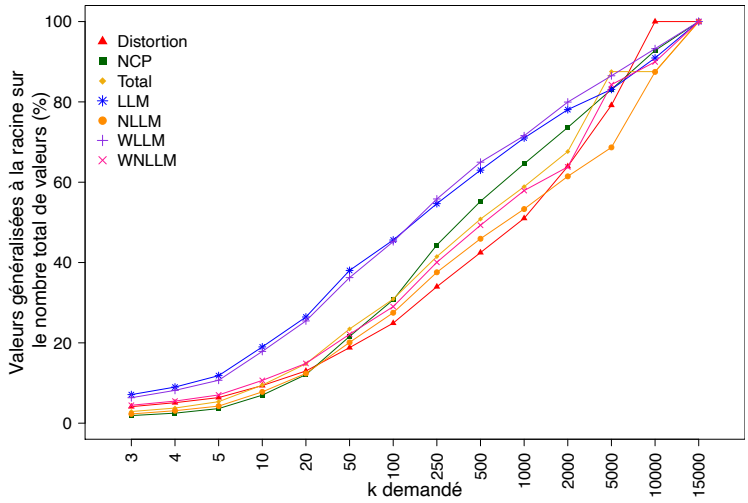
Soient T_1, T_2 et T_3 trois tables sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$.



(a) Altération moyenne

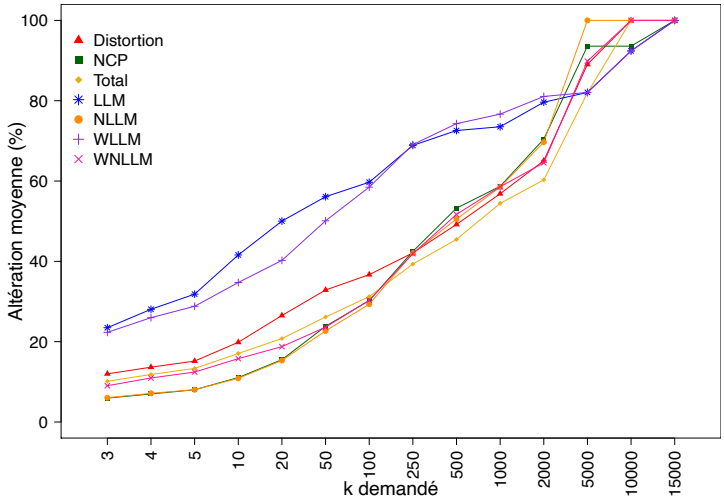


(b) Pourcentage de valeurs généralisées

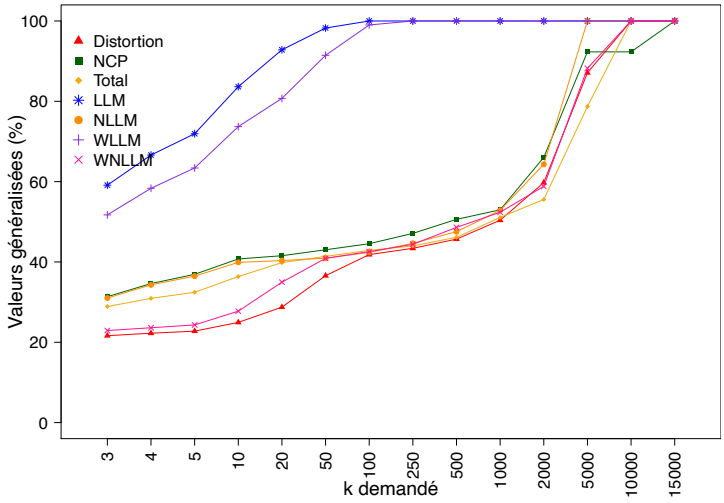


(c) Pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs

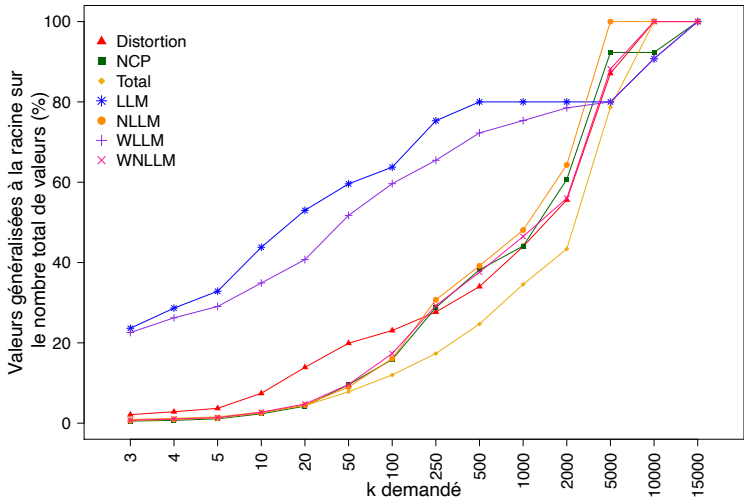
FIGURE 3.7 – Évolution de trois critères de qualité des tables k-anonymes obtenues avec les sept métriques pour *Adult data set*



(a) Altération moyenne



(b) Pourcentage de valeurs généralisées



(c) Pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs

FIGURE 3.8 – Évolution de trois critères de qualité des tables k-anonymes obtenues avec les sept métriques pour *florida_30162*

Si T_3 est table généralisée sur $(T_2, \mathcal{H})$ et T_2 est une table généralisée sur $(T_1, \mathcal{H})$ alors

1. $\Lambda_{\mathcal{M},T_1}(T_2) \leqslant \Lambda_{\mathcal{M},T_1}(T_3)$
2. $p_{\text{gen},T_1}(T_2) \leqslant p_{\text{gen},T_1}(T_3)$
3. $p_{\text{racine},T_1}(T_2) \leqslant p_{\text{racine},T_1}(T_3)$

Démonstration : Par la propriété 2.3.3 page 19, T_3 est une table généralisée sur $(T_1, \mathcal{H})$. Posons $T_l = \{F_l^1, \ldots, F_l^n\}$ avec $F_l^i = (f_{l,1}^i, \ldots, f_{l,m}^i, s^i)$ pour tout $i \in [\![1, n]\!]$ et pour tout $l \in [\![1, 3]\!]$.

1. Montrons dans un premier temps que $\forall \mu \in \mathcal{M}$, $\Lambda_{\mu,T_1}(T_2) \leqslant \Lambda_{\mu,T_1}(T_3)$.

$$\begin{aligned}\Lambda_{\mu,T_1}(T_2) &= \frac{\mu_{T_1}(T_2)}{\mu_{T_1}(T_1^*)} \times 100 \text{ par la définition 3.1.6} \\ &\leqslant \frac{\mu_{T_1}(T_3)}{\mu_{T_1}(T_1^*)} \times 100 \text{ par la proposition 3.1.1} \\ &= \Lambda_{\mu,T_1}(T_3)\end{aligned}$$

   On en déduit que

$$\begin{aligned}\Lambda_{\mathcal{M},T_1}(T_2) &= \frac{1}{|\mathcal{M}|} \times \sum_{\mu \in \mathcal{M}} \Lambda_{\mu,T_1}(T_2) \text{ par la définition 3.4.1} \\ &\leqslant \frac{1}{|\mathcal{M}|} \times \sum_{\mu \in \mathcal{M}} \Lambda_{\mu,T_1}(T_3) \text{ car } \Lambda_{\mu,T_1}(T_2) \geqslant 0 \text{ pour tout } \mu \in \mathcal{M} \\ &= \Lambda_{\mathcal{M},T_1}(T_3)\end{aligned}$$

2. Montrons dans un premier temps que $v_{\text{gen},T_1}(T_2) \leqslant v_{\text{gen},T_1}(T_3)$. Posons $P_2 = \{(i,j) \in [\![1,n]\!] \times [\![1,m]\!] : \text{niv}(f_{2,j}^i) > \text{niv}(f_{1,j}^i)\}$ et $P_3 = \{(i,j) \in [\![1,n]\!] \times [\![1,m]\!] : \text{niv}(f_{3,j}^i) > \text{niv}(f_{1,j}^i)\}$. Soit $(i,j) \in P_2$. On a $\text{niv}(f_{2,j}^i) > \text{niv}(f_{1,j}^i)$. D'après le lemme 3.4.1 page 42, comme T_3 est une table généralisée sur $(T_2, \mathcal{H})$, $\text{niv}(f_{3,j}^i) \geqslant \text{niv}(f_{2,j}^i)$. Donc $\text{niv}(f_{3,j}^i) \geqslant \text{niv}(f_{2,j}^i) > \text{niv}(f_{1,j}^i)$ donc $(i,j) \in P_3$. Ainsi $P_2 \subseteq P_3$ et donc $|P_2| \leqslant |P_3|$ c'est-à-dire $v_{\text{gen},T_1}(T_2) \leqslant v_{\text{gen},T_1}(T_3)$. Par conséquent, $p_{\text{gen},T_1}(T_2) = \frac{v_{\text{gen},T_1}(T_2)}{n \times m} \times 100 \leqslant \frac{v_{\text{gen},T_1}(T_3)}{n \times m} \times 100 = p_{\text{gen},T_1}(T_3)$.
3. Montrons dans un premier temps que $v_{\text{racine},T_1}(T_2) \leqslant v_{\text{racine},T_1}(T_3)$. Posons $P_2 = \{(i,j) \in [\![1,n]\!] \times [\![1,m]\!] : f_{2,j}^i = r_j\}$ et $P_3 = \{(i,j) \in [\![1,n]\!] \times [\![1,m]\!] : f_{3,j}^i = r_j\}$. Soit $(i,j) \in P_2$. On a $f_{2,j}^i = r_j$. D'après le lemme 3.4.1 page 42, comme T_3 est une table généralisée sur $(T_2, \mathcal{H})$, $\text{niv}(f_{3,j}^i) \geqslant \text{niv}(f_{2,j}^i)$. Or $f_{2,j}^i = r_j$ donc $\text{niv}(f_{3,j}^i) = \text{niv}(f_{2,j}^i)$ et $f_{3,j}^i = f_{2,j}^i = r_j$ donc $(i,j) \in P_3$. Ainsi $P_2 \subseteq P_3$ et donc $|P_2| \leqslant |P_3|$ c'est-à-dire $v_{\text{racine},T_1}(T_2) \leqslant v_{\text{racine},T_1}(T_2)$. Par conséquent, $p_{\text{racine},T_1}(T_2) = \frac{v_{\text{racine},T_1}(T_2)}{n \times m} \times 100 \leqslant \frac{v_{\text{racine},T_1}(T_3)}{n \times m} \times 100 = p_{\text{racine},T_1}(T_3)$.

■

Nous avons montré que les trois critères de qualité fournissent toujours des courbes croissantes quand la valeur de k augmente.

Pour l'altération moyenne sur $\mathcal{M}$, le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs, nous calculons donc une valeur moyenne des résultats du critère obtenus par les tables k-anonymes produites avec *GkAA* en utilisant une métrique en divisant l'aire sous la courbe par la longueur de l'intervalle $[3, 15\,000]$. Cette valeur est appelée *VMN*, pour *Valeur Moyenne Normalisée*, de cette métrique pour ce critère.

Les tableaux des tables 3.2 page suivante à 3.4 page suivante présentent les *VMN* des métriques obtenues pour les trois critères d'évaluation et les deux tables pour k compris entre 3 et 15000. Par exemple, le tableau 3.2a page suivante présente les *VMN* calculées sur $[3, 15\,000]$ des sept métriques pour l'altération moyenne sur *Adult data set*. Les résultats sont triés dans l'ordre croissant. Nous lisons que la valeur moyenne d'altération moyenne sur $\mathcal{M}$ des tables k-anonymes obtenues avec *GkAA* en utilisant *Distortion* est de 86,3662%.

3.4.2 Analyse des résultats

L'analyse des résultats se fera en deux parties. La première partie consiste à comparer les performances des métriques selon trois critères de qualité (cf. section 3.4.2.1 page 47). Nous allons étudier les résultats obtenus

Métriques	*VMN*
NLLM	79,5256
WNLLM	84,6481
Total	85,0889
WLLM	85,587
LLM	86,0094
Distortion	86,3662
NCP	86,9245

(a) Pour *Adult data set*

Métriques	*VMN*
Total	84,6116
LLM	87,0383
NCP	87,3524
WLLM	87,39
Distortion	87,4548
WNLLM	87,6634
NLLM	91,0093

(b) Pour *florida_30162*

TABLEAU 3.2 – Valeurs moyennes normalisées des sept métriques pour l'altération moyenne calculées sur l'intervalle $[3, 15\,000]$ sur les deux tables

Métriques	*VMN*
NLLM	79,2416
Total	85,0028
WNLLM	85,0253
NCP	86,0426
Distortion	87,4683
WLLM	92,0642
LLM	94,4266

(a) Pour *Adult data set*

Métriques	*VMN*
Total	83,1788
Distortion	85,8865
NCP	85,9162
WNLLM	86,2693
NLLM	90,1614
WLLM	99,9201
LLM	99,9684

(b) Pour *florida_30162*

TABLEAU 3.3 – Valeurs moyennes normalisées des sept métriques pour le pourcentage de valeurs généralisées calculées sur l'intervalle $[3, 15\,000]$ sur les deux tables

Métriques	*VMN*
NLLM	76,9205
WNLLM	82,598
Total	83,2866
Distortion	83,9603
NCP	85,0809
LLM	85,8756
WLLM	87,8855

(a) Pour *Adult data set*

Métriques	*VMN*
Total	79,4698
NCP	83,8459
Distortion	84,3048
WNLLM	84,8096
WLLM	85,7942
LLM	86,6208
NLLM	89,1916

(b) Pour *florida_30162*

TABLEAU 3.4 – Valeurs moyennes normalisées des sept métriques pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs calculées sur l'intervalle $[3, 15\,000]$ sur les deux tables

Métriques	*VMN*
NLLM	56,0653
Distortion	57,2953
WNLLM	59,2918
Total	60,0587
NCP	64,6506
WLLM	66,5998
LLM	68,121

(a) Pour *Adult data set*

Métriques	*VMN*
Total	50,4038
Distortion	53,9585
WNLLM	54,3272
NLLM	55,2454
NCP	56,1311
LLM	72,913
WLLM	74,4815

(b) Pour *florida_30162*

Tableau 3.5 – Valeurs moyennes normalisées des sept métriques pour l'altération moyenne calculées sur l'intervalle $[3, 2000]$ sur les deux tables

pour chaque métrique, pour chaque critère et pour chaque table. Dans la seconde partie, nous allons analyser les effets de trois caractéristiques des définitions des métriques sur les performances des métriques (cf. section 3.4.2.2 page 51).

3.4.2.1 Étude des performances des métriques selon trois critères de qualité

Étudions maintenant les résultats obtenus pour chaque critère.

Pour l'altération moyenne sur $\mathcal{M}$, nous constatons sur le graphique 3.7a page 43 que la métrique *NLLM* permet de produire des tables k-anonymes de meilleure qualité pour ce critère que les autres métriques sur *Adult data set*. La courbe correspondant à *NLLM* est en effet en-dessous des autres courbes pour la plupart des valeurs de k. Cette impression graphique est confirmée par la *VMN* calculée sur $[3, 15\,000]$ de *NLLM* de 79,5256% (tableau 3.2a page ci-contre) qui est nettement inférieure aux *VMN* calculées sur $[3, 15\,000]$ des autres métriques, la deuxième meilleure métrique pour l'altération moyenne sur $\mathcal{M}$ sur *Adult data set*, *WNLLM*, ayant une *VMN* calculée sur $[3, 15\,000]$ de 84,6481%.

En revanche, sur *florida_30162*, nous observons dans le tableau 3.2b page précédente que *NLLM* est la moins bonne métrique pour l'altération moyenne sur $\mathcal{M}$: sa *VMN* calculée sur $[3, 15\,000]$ pour ce critère est de 91,0093%, supérieure de plus de 3 points de celle de la seconde moins bonne métrique pour l'altération moyenne sur $\mathcal{M}$ sur *florida_30162*. Sur le graphique 3.8a page 44, nous observons que l'altération moyenne sur $\mathcal{M}$ des tables k-anonymes produites avec *NLLM* est de 100% pour $k \in \{5000, 10\,000, 15\,000\}$. Cela n'est pas le cas pour les autres métriques et peut expliquer que *NLLM* ait une *VMN* calculée sur $[3, 15\,000]$ plus élevée que les autres métriques. Concernant les résultats d'altération moyenne sur $\mathcal{M}$ des autres métriques sur *florida_30162*, *Distortion*, *NCP*, *LLM*, *WLLM* et *WNLLM* obtiennent des *VMN* calculées sur $[3, 15\,000]$ proches : elles sont comprises entre 87,0383% pour *LLM* et 87,6634% pour *WNLLM*. Seule la métrique *Total* se détache avec une *VMN* calculée sur $[3, 15\,000]$ pour l'altération moyenne sur $\mathcal{M}$ de 84,1661%. Nous observons effectivement sur le graphique 3.8a page 44 que la courbe correspondant à *Total* est en-dessous des autres courbes pour k compris entre 250 et 5000.

Lors du calcul des *VMN* sur l'intervalle $[3, 15\,000]$, la longueur de l'intervalle $[5000, 15\,000]$ représente environ 67% de la longueur de l'intervalle total. Les tables k-anonymes produites pour ces valeurs de k ont par ailleurs de mauvais résultats pour les trois critères de qualité pour la plupart des métriques. Cela peut être dû au fait que ces valeurs de k sont proches du nombre d'enregistrements des tables étudiées (*Adult data set* et *florida_30162* ont 30 162 enregistrements).

Afin d'étudier le comportement des métriques pour des valeurs de k raisonnables, c'est-à-dire pour des valeurs de k pour lesquelles la dégradation des données reste contenue pour la plupart des métriques, nous avons dans un premier temps calculé les *VMN* de chaque métrique pour les trois critères de qualité sur les deux tables sur l'intervalle $[3, 2000]$.

Les tableaux des tables 3.5 à 3.7 page suivante présentent les résultats de *VMN* calculées sur $[3, 2000]$ obtenus pour l'altération moyenne sur $\mathcal{M}$, le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs pour les deux tables.

En particulier, les tableaux de la table 3.5 correspondent aux *VMN* calculées sur $[3, 2000]$ des sept métriques pour l'altération moyenne sur $\mathcal{M}$ pour les deux tables. Pour *Adult data set* (tableau 3.5a), nous remarquons que *NLLM* obtient la meilleure *VMN* calculée sur $[3, 2000]$ pour l'altération moyenne sur $\mathcal{M}$. Pour *florida_30162* (tableau 3.5b), la *VMN* calculée sur $[3, 2000]$ de *NLLM* est dans le même ordre de grandeur que celles de *Distortion*, *NCP*, *Total* et *WNLLM* alors que la *VMN* calculée sur $[3, 15\,000]$ de *NLLM* était bien plus élevée que celles des autres métriques pour l'altération moyenne sur $\mathcal{M}$ sur *florida_30162*. Nous constatons également

Métriques	*VMN*
NLLM	59,6296
WNLLM	63,2421
Total	63,5907
Distortion	63,6637
NCP	66,2799
WLLM	80,0624
LLM	84,4295

(a) Pour *Adult data set*

Métriques	*VMN*
Total	49,7467
Distortion	50,0693
WNLLM	51,4271
NLLM	52,9203
NCP	54,3963
WLLM	99,3997
LLM	99,7624

(b) Pour *florida_30162*

TABLEAU 3.6 – Valeurs moyennes normalisées des sept métriques pour le pourcentage de valeurs généralisées calculées sur l'intervalle $[3, 2000]$ sur les deux tables

Métriques	*VMN*
Distortion	48,348
NLLM	49,7413
WNLLM	53,0956
Total	54,9588
NCP	59,6377
LLM	66,928
WLLM	68,0272

(a) Pour *Adult data set*

Métriques	*VMN*
Total	31,0317
Distortion	41,3234
WNLLM	42,5889
NCP	42,8535
NLLM	45,6376
WLLM	72,7058
LLM	77,7895

(b) Pour *florida_30162*

TABLEAU 3.7 – Valeurs moyennes normalisées des sept métriques pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs calculées sur l'intervalle $[3, 2000]$ sur les deux tables

que les métriques *LLM* et *WLLM* sont les moins bonnes en termes d'altération moyenne sur $\mathcal{M}$ pour des valeurs de k comprises entre 3 et 2000 sur les deux tables. L'altération moyenne sur $\mathcal{M}$ des versions k-anonymes de *florida_30162* produites avec *LLM* est en moyenne de 72,913% alors qu'elle est en moyenne de 56,1311% pour les versions k-anonymes de *florida_30162* produites avec *NCP*.

Pour le pourcentage de valeurs généralisées sur le nombre total de valeurs, nous observons dans le tableau 3.6a que *NLLM* a la meilleure *VMN* calculée sur $[3, 2000]$ sur *Adult data set*. Le pourcentage de valeurs généralisées sur le nombre total de valeurs des versions k-anonymes de *Adult data set* pour k entre 3 et 2000 produites avec *NLLM* est en moyenne de 59,6296%. Les métriques *Distortion*, *Total* et *WNLLM* ont des *VMN* calculées sur $[3, 2000]$ d'environ 63% pour le pourcentage de valeurs généralisées sur le nombre total de valeurs sur *Adult data set*. Les métriques *LLM* et *WLLM* sont les moins bonnes pour ce critère sur *Adult data set* : les pourcentages de valeurs généralisées sur le nombre total de valeurs des versions k-anonymes de *Adult data set* pour k entre 3 et 2000 produites par *LLM* et *WLLM* sont de 84,4295% et 80,0624% respectivement.

Sur *florida_30162* (tableau 3.6b), *LLM* et *WLLM* sont également les moins bonnes pour le pourcentage de valeurs généralisées sur le nombre total de valeurs. Le pourcentage de valeurs généralisées sur le nombre total de valeurs des versions k-anonymes de *florida_30162* pour k entre 3 et 2000 produites avec *LLM* et *WLLM* est de plus de 99% en moyenne. Cette observation est visible sur le graphique 3.8b page 44 : les courbes correspondant à *LLM* et *WLLM* atteignent 100% dès $k = 250$. Ces deux métriques sont les seules à ne pas avoir de normalisation sur le coût de généralisation des nœuds (cf. section 3.2 page 32, tableau 3.1 page 36). Les autres métriques ont des *VMN* calculées sur $[3, 2000]$ autour des 50% pour le pourcentage de valeurs généralisées sur le nombre total de valeurs sur *florida_30162*. Cela signifie que, en moyenne, dans les versions k-anonymes de *florida_30162* pour k entre 3 et 2000 produites avec ces métriques, une valeur quasi-identifiante sur deux a subi une généralisation.

Pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs (tableau 3.7), nous observons des résultats similaires à ceux obtenus pour le pourcentage de valeurs généralisées sur le nombre total de valeurs. *LLM* et *WLLM* sont les moins bonnes métriques pour ce critère sur les deux tables. Sur *Adult data set* (tableau 3.7a), la *VMN* calculée sur $[3, 2000]$ de *LLM* est d'environ 7 points supérieure à celle de la moins bonne des autres métriques ; sur *florida_30162* (tableau 3.7b), la *VMN* calculée sur $[3, 2000]$ de *WLLM* est d'environ 27 points supérieure à celle de la moins bonne des autres métriques. Sur *Adult data set* (tableau 3.7a), *NLLM* et *Distortion* sont les métriques ayant les meilleures *VMN* calculées sur $[3, 2000]$ pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs. Le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs des versions k-anonymes de *Adult data set* pour k entre 3 et 2000 produites avec

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
Distortion	11,2632	27,2877	51,9354	82,8841
NCP	11,1948	30,7754	59,5101	84,734
Total	12,3225	30,4953	55,0503	83,5345
LLM	17,1258	39,8982	63,614	83,7696
NLLM	10,8032	28,0649	51,9057	74,6198
WLLM	14,9816	36,8661	62,2493	83,5372
WNLLM	11,6113	28,7851	54,9417	82,2056

(a) Pour *Adult data set*

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
Distortion	16,3967	31,7986	48,7057	84,9719
NCP	8,8131	23,0483	50,4443	86,5026
Total	14,2257	25,8563	45,4066	80,6253
LLM	34,1841	54,9449	70,9416	83,9205
NLLM	8,7847	22,2407	49,092	90,9578
WLLM	30,0574	49,3818	72,4699	84,4126
WNLLM	13,1746	23,9632	49,6505	85,3059

(b) Pour *florida_30162*

TABLEAU 3.8 – Valeurs moyennes normalisées des sept métriques pour l'altération moyenne calculées sur plusieurs intervalles sur les deux tables

ces deux métriques est de moins de 50% en moyenne. Sur *florida_30162* (tableau 3.7b page ci-contre), *Total* a une *VMN* calculée sur $[3, 2000]$ pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs bien inférieure aux *VMN* des autres métriques. Le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs des versions k-anonymes de *florida_30162* pour k entre 3 et 2000 produites avec *Total* est d'environ 31% alors que pour *Distortion*, la deuxième meilleure métrique pour ce critère sur *florida_30162*, ce pourcentage est d'environ 41.

Pour conclure sur ce premier point, les métriques *LLM* et *WLLM* ont obtenu les moins bons résultats pour les trois critères de qualité en termes de *VMN* calculées sur $[3, 2000]$. Concernant les métriques ayant obtenu les meilleurs résultats, *NLLM* semble être la plus intéressante à utiliser sur *Adult data set* et *Total* parvient à produire de bonnes versions k-anonymes de *florida_30162* au regard des trois critères de qualité.

Dans un second temps, nous avons calculé les *VMN* sur quatre sous-intervalles de $[3, 15\,000]$ pour chaque métrique pour les trois critères de qualité et sur les deux tables. L'objectif est de comparer les performances des métriques pour des valeurs de k dans le même ordre de grandeur. Les sous-intervalles choisis sont $[3, 10]$, $[10, 100]$, $[100, 1000]$ et $[1000, 10\,000]$.

Les tableaux 3.8 à 3.10 page suivante présentent les *VMN* calculées sur chaque sous-intervalle de chaque métrique sur les deux tables pour l'altération moyenne sur $\mathcal{M}$, le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs. Pour un même sous-intervalle, les *VMN* sont classées de la moins élevée à la plus élevée : le classement est représenté par des couleurs du vert au rouge. Par exemple, dans le tableau 3.9b page suivante, nous lisons que la métrique *Distortion* a une *VMN* calculée sur $[3, 10]$ de 23,3922% pour le pourcentage de valeurs généralisées sur le nombre total de valeurs sur *florida_30162*. Il s'agit de la meilleure métrique pour ce critère sur cet intervalle et sur cette table car la case correspondante est vert foncé.

Pour l'altération moyenne sur $\mathcal{M}$, nous observons dans le tableau 3.8a que les résultats de *VMN* obtenus sur les sous-intervalles sur *Adult data set* sont similaires à ceux obtenus sur l'intervalle $[3, 2000]$. La métrique *NLLM* a les meilleures résultats pour ce crtitère et sur cette table suivie par *Distortion*, les métriques *LLM* et *WLLM* sont les moins bonnes quelque soit le sous-intervalle considéré.

Sur *florida_30162* (tableau 3.8b), nous observons que *NLLM* a les meilleures *VMN* calculées sur les sous-intervalles $[3, 10]$ et $[10, 100]$. Excepté pour la métrique *NCP*, les écarts avec les *VMN* des autres métriques sont de plus assez conséquents sur $[3, 10]$: la *VMN* calculée sur $[3, 10]$ de *NLLM* est d'environ 8% alors que la *VMN* calculée sur $[3, 10]$ de la troisième meilleure métrique est d'environ 13% pour ce critère sur *florida_30162*. Cependant, *NLLM* a la *VMN* calculée sur $[1000, 10\,000]$ pour l'altération moyenne sur $\mathcal{M}$ la plus élevée. *NLLM* produit donc de meilleures versions k-anonymes de *florida_30162* que les autres métriques en termes d'altération moyenne sur $\mathcal{M}$ quand k est inférieur à 100. Quand k dépasse 100, *NLLM* est moins performante que les autres métriques. Sur les intervalles $[100, 1000]$ et $[1000, 10\,000]$, *Total* obtient les meilleures *VMN* pour l'altération

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
Distortion	18,8403	37,2983	60,0064	83,8075
NCP	23,5411	43,9599	63,1876	83,0424
Total	22,3433	39,7388	59,1713	83,2171
LLM	32,5443	59,9718	81,1327	94,3118
NLLM	21,7094	39,5306	57,2407	73,8299
WLLM	25,9936	52,009	76,6436	91,5248
WNLLM	19,7355	38,5897	60,4112	82,4716

(a) Pour *Adult data set*

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
Distortion	23,3922	35,6488	46,1597	82,5693
NCP	37,5581	42,9945	49,9845	84,2877
Total	33,3755	41,1726	46,7644	77,9339
LLM	74,4329	96,7173	100,0	100,0
NLLM	36,9899	41,1471	47,934	89,4497
WLLM	65,5172	90,1839	99,9178	100,0
WNLLM	25,3395	39,3101	48,2103	82,9641

(b) Pour *florida_30162*

TABLEAU 3.9 – Valeurs moyennes normalisées des sept métriques pour le pourcentage de valeurs généralisées calculées sur plusieurs intervalles sur les deux tables

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
Distortion	7,0966	18,6903	41,4964	80,0083
NCP	4,5401	21,2062	53,3816	82,6627
Total	6,4236	22,8257	49,3261	81,5234
LLM	13,6468	36,5029	61,9245	83,4925
NLLM	5,2114	19,7615	44,5871	71,438
WLLM	12,5811	35,3148	63,1327	86,0901
WNLLM	7,919	21,7993	47,9436	79,8508

(a) Pour *Adult data set*

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
Distortion	4,7966	18,7663	34,4628	81,2866
NCP	1,4345	9,7325	35,8842	82,6086
Total	1,8996	7,9356	24,7433	74,3125
LLM	35,5014	58,402	77,601	82,9837
NLLM	1,6392	9,6018	37,8471	89,1854
WLLM	30,2774	50,5514	70,5614	82,3929
WNLLM	1,7729	10,3087	36,5213	82,009

(b) Pour *florida_30162*

TABLEAU 3.10 – Valeurs moyennes normalisées des sept métriques pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs calculées sur plusieurs intervalles sur les deux tables

moyenne sur $\mathcal{M}$ sur *florida_30162*.

Pour le pourcentage de valeurs généralisées sur le nombre total de valeurs, nous observons dans le tableau 3.9a page ci-contre que *Distortion* est la meilleure métrique pour ce critère sur *Adult data set* sur les intervalles $[3, 10]$ et $[10, 100]$. Pour de plus grandes valeurs de k, *NLLM* est la métrique permettant de produire des tables k-anonymes de meilleure qualité en termes de pourcentage de valeurs généralisées sur le nombre total de valeurs. Sur *florida_30162* (tableau 3.9b page précédente), *Distortion* obtient les meilleures *VMN* calculées sur trois des quatre sous-intervalles, *Total* étant la meilleure pour le sous-intervalle restant. Sur les deux tables, *LLM* et *WLLM* sont les métriques ayant les moins bons résultats de pourcentage de valeurs généralisées sur le nombre total de valeurs sur les quatre sous-intervalles étudiés. Nous observons notamment dans le tableau 3.9b page ci-contre que les versions k-anonymes de *florida_30162* pour k entre 3 et 10 produites avec *LLM* ont déjà un pourcentage de valeurs généralisées sur le nombre total de valeurs d'environ 75% en moyenne.

Pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs, les résultats obtenus sur les quatre sous-intervalles sont similaires aux résultats obtenus sur l'intervalle $[3, 2000]$. Sur *Adult data set*, *NLLM* est l'une des meilleures métriques pour ce critère sur tous les sous-intervalles. Sur *florida_30162*, *Total* a les *VMN* les moins élevées sur trois intervalles sur quatre. Elle a notamment une *VMN* calculée sur $[100, 1000]$ d'environ 10 points inférieure à celle de la deuxième métrique et une *VMN* calculée sur $[1000, 10\,000]$ d'environ 7 points inférieure à celle de la deuxième meilleure métrique. *LLM* et *WLLM* restent en général les moins bonnes métriques pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs sur les quatre sous-intervalles et les deux tables.

Pour conclure sur ce second point, nous avons observé que, quelque soit le sous-intervalle considéré, *LLM* et *WLLM* sont les métriques ayant les moins bons résultats de *VMN* pour les trois critères de qualité sur les deux tables étudiées. Sur *Adult data set*, *NLLM* produit des tables k-anonymes ayant de bons résultats pour les trois critères de qualité et sur les quatre sous-intervalles étudiés. Sur *florida_30162*, *Total* reste la meilleure métrique pour fournir des tables k-anonymes conservant une bonne utilité des données. Cependant, contrairement à ce que nous avions observé dans le premier point de l'analyse, la métrique *NLLM* parvient à fournir de meilleures versions k-anonymes de *florida_30162* que les autres métriques en termes d'altération moyenne sur $\mathcal{M}$ pour k inférieur à 100. Le choix de la métrique à utiliser pour k-anonymiser une table peut donc dépendre de la valeur de k choisie.

3.4.2.2 Étude des effets de trois caractéristiques des définitions des métriques sur leurs performances

Lors de la présentation des métriques étudiées en section 3.2 page 32, nous avons répertorié des caractéristiques des définitions des métriques. Pour définir une métrique, nous avons donné une formule calculant les poids à mettre sur les arêtes des hiérarchies de généralisation. Cette formule peut se décomposer en deux phases : le coût de généralisation du nœud et la pondération sur l'ensemble des attributs quasi-identifiants. Dans le tableau 3.1 page 36, trois principales caractéristiques se dégagent dans les définitions des métriques :

1. le coût de généralisation du nœud dépend de la hauteur de la hiérarchie ou de la largeur de la hiérarchie
2. la pondération sur l'ensemble des quasi-identifiants est p_1, p_2 ou aucune (cf. section 3.2 page 32)
3. il y a une étape de normalisation dans le calcul du coût de généralisation du nœud ou non

Pour chacun des points précédents, chaque métrique de $\mathcal{M}$ vérifie exactement une affirmation. Par exemple, pour la métrique *NCP*, le coût de généralisation du nœud dépend de la largeur de la hiérarchie (point 1) et comporte une étape de normalisation (point 3). Aucune pondération sur l'ensemble des attributs n'est utilisée pour la métrique *NCP* (point 2).

Pour chaque point, nous allons regarder si les différences dans les définitions des métriques ont des conséquences sur les résultats obtenus pour les trois critères de qualité et pour les deux tables. Nous allons étudier les *VMN* calculées sur $[3, 2000]$ (tables 3.5 page 47 à 3.7 page 48).

Pour le point 1, nous cherchons à savoir si la façon de calculer le coût de généralisation du nœud influence les performances des métriques. Nous distinguons deux groupes de métriques dans $\mathcal{M}$: celles pour lesquelles le coût de généralisation du nœud dépend de la hauteur de la hiérarchie (*Distortion* et *Total*) et celles pour lesquelles le coût de généralisation du nœud dépend de la largeur de la hiérarchie (*NCP*, *LLM*, *NLLM*, *WLLM* et *WNLLM*). Pour *Adult data set*, pour les trois critères (tableaux 3.5a page 47, 3.6a page 48 et 3.7a page 48), cette caractéristique ne semble pas avoir d'impact sur les performances des métriques. La métrique *NLLM* (calcul sur la largeur de la hiérarchie) a les *VMN* sur $[3, 2000]$ les plus basses pour deux critères et la métrique *Distortion* (calcul sur la hauteur de la hiérarchie) a les *VMN* sur $[3, 2000]$ les plus basses pour le dernier critère. En revanche, pour *florida_30162*, pour l'altération moyenne sur $\mathcal{M}$ (tableau 3.5b page 47), le pourcentage de valeurs généralisées (tableau 3.6b page 48) et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs (tableau 3.7b page 48), les métriques *Distortion* et *Total* ont les *VMN* sur $[3, 2000]$ les plus

basses. Ce sont les deux seules métriques pour lesquelles le coût de généralisation du nœud dépend de la hauteur de la hiérarchie. Bien que les écarts avec les autres métriques soient minces, les résultats concernent les trois critères de qualité étudiés.

Pour le point 2 page précédente, nous voulons étudier l'impact de l'application ou non d'une pondération sur les attributs quasi-identifiants dans la définition des métriques sur les performances des métriques. Nous avons présenté deux pondérations sur un ensemble de quasi-identifiants en section 3.2 page 32. La pondération p_1 est utilisée dans la définition des métriques *Distortion*, *WLLM* et *WNLLM*. La pondération p_2 est utilisée dans la définition des métriques *LLM* et *NLLM*. Dans la définition des métriques *Total* et *NCP*, aucune pondération n'est utilisée. Pour les deux tables, pour les trois critères, aucun groupe de métriques n'obtient de résultats significativement meilleurs que les deux autres. Par exemple, pour *Adult data set*, pour les critères d'altération moyenne (tableau 3.5a page 47) et de pourcentage de valeurs généralisées sur le nombre total de valeurs (tableau 3.6a page 48), les métriques *NLLM* et *LLM* ont respectivement la meilleure et la pire *VMN* sur $[3, 2000]$ alors que la pondération p_2 est utilisée dans la définition de ces deux métriques. De même, pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs (tableau 3.7a page 48), les métriques *Distortion* et *WLLM* ont respectivement la meilleure et la pire *VMN* sur $[3, 2000]$ alors que la pondération p_1 est utilisée dans la définition de ces deux métriques. L'application d'une pondération sur les attributs quasi-identifiants seule ne paraît pas être un facteur déterminant pour obtenir une métrique avec de bonnes performances.

Pour le point 3 page précédente, nous étudions les effets d'une étape de normalisation dans le calcul du coût de généralisation du nœud dans la définition des métriques sur les performances des métriques. Par exemple, dans la définition de la métrique *NCP*, le nombre de feuilles dans le sous-arbre enraciné en le nœud est divisé par le nombre de feuilles total de la hiérarchie. Dans les définitions des métriques de $\mathcal{M}$, soit nous effectuons une étape de normalisation dans le calcul du coût de généralisation du nœud (*Distortion*, *Total*, *NCP*, *NLLM* et *WNLLM*) soit nous n'en effectuons pas (*LLM* et *WLLM*). Pour les critères d'altération moyenne sur $\mathcal{M}$ (table 3.5 page 47), de pourcentage de valeurs généralisées sur le nombre total de valeurs (table 3.6 page 48) et de pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs (table 3.7 page 48), les métriques *LLM* et *WLLM* ont les *VMN* sur $[3, 2000]$ les plus élevées pour les deux tables. Les écarts avec les *VMN* sur $[3, 2000]$ des autres métriques sont importants (à l'exception de l'altération moyenne sur $\mathcal{M}$ sur *Adult data set*). Par exemple, pour *florida_30162*, nous notons un écart d'environ 27 points entre les *VMN* de *WLLM* et de *NLLM* pour le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs. Il semble donc que l'étape de normalisation dans le calcul du coût de généralisation du nœud influence grandement les performances de la métrique. Cette caractéristique paraît indispensable pour que la métrique permette de produire des tables k-anonymes de bonne qualité selon les trois critères étudiés.

3.5 Conclusion du chapitre

Dans ce chapitre, notre objectif a été de comparer les performances de métriques de perte d'information quand elles sont utilisées dans un algorithme de k-anonymisation.

Pour cela, nous avons défini en section 3.1 page 26 une métrique de perte d'information sur un ensemble de hiérarchies de généralisation comme un ensemble de poids mis sur les arêtes des hiérarchies. A partir de cette définition, nous avons présenté un nouveau modèle permettant de simplifier l'utilisation des métriques de perte d'information : nous avons défini les matrices des coûts associées à une métrique de perte d'information sur un ensemble de hiérarchies. Grâce aux matrices des coûts, nous avons explicité le coût de généralisation de deux classes d'équivalence d'une table et le coût de généralisation d'une table généralisée par rapport à sa table d'origine pour une métrique de perte d'information.

Une fois la définition de métrique de perte d'information clairement exposée, nous nous sommes intéressés à plusieurs métriques de perte d'information en section 3.2 page 32. Nous avons tout d'abord étudié trois métriques de la littérature : *Distortion*, *NCP* et *Total*. Puis nous avons proposé une nouvelle métrique, la *Lost Leaves Metric*, et trois variantes de cette métrique. Notre souhait en proposant ces quatre métriques était de pouvoir comparer les effets de certaines caractéristiques dans les définitions des métriques comme le choix de la pondération sur un ensemble d'attributs quasi-identifiants ou la présence d'une étape de normalisation dans le calcul du coût de généralisation du nœud (cf. tableau 3.1 page 36).

Pour pouvoir comparer les performances des sept métriques de perte d'information précédentes lors d'un processus de k-anonymisation, nous sommes revenus en section 3.3 page 36 sur les définitions de table k-anonyme et de version k-anonyme d'une table. Nous avons également présenté l'algorithme *GkAA*. Cet algorithme permet de produire une version k-anonyme d'une table grâce à des fusions de classes d'équivalence. Le choix de ces fusions de classes d'équivalence est déterminé grâce à une métrique de perte d'information : à chaque tour, la seconde classe d'équivalence à fusionner sera choisie de telle sorte à minimiser le coût de généralisation avec la première classe d'équivalence pour la métrique de perte d'information.

Dans la section 3.4 page 40, nous avons expliqué comment comparer les performances des sept métriques de perte d'information lors d'un processus de k-anonymisation et nous avons analysé les résultats obtenus. Nous avons évalué l'algorithme *GkAA* pour chacune des sept métriques, pour 14 valeurs de k et pour deux tables (*Adult data set* et *florida_30162*). Puis nous avons présenté trois critères pour juger de la qualité des tables k-anonymes produites : nous avons étudié l'altération moyenne, le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs.

Nous avons d'abord comparé les performances des métriques selon les trois critères de qualité. Nous avons constaté que les métriques ayant les meilleures performances ne sont pas les mêmes pour les deux tables : pour *Adult data set*, la métrique *NLLM* permet d'obtenir des tables k-anonymes de bonne qualité au regard des trois critères étudiés alors que pour *florida_30162*, il s'agit de la métrique *Total*. En revanche, les métriques *LLM* et *WLLM* donnent les moins bons résultats pour les deux tables. De plus, en étudiant les performances des métriques pour des valeurs de k dans le même ordre de grandeur, nous avons observé qu'une métrique peut fournir de bonnes tables k-anonymes pour une certaine plage de valeurs de k mais beaucoup dégrader les données des tables k-anonymes sur une autre plage de valeurs. Le choix de la métrique à utiliser pour k-anonymiser peut donc être motivé par le k demandé.

Nous avons ensuite étudié les effets de trois caractéristiques des définitions des métriques sur les performances des métriques : le coût de généralisation du nœud dépend de la hauteur ou de la largeur de la hiérarchie, la pondération sur les attributs quasi-identifiants utilisée est p_1, p_2 ou aucune et une étape de normalisation est présente dans le calcul du coût de généralisation du nœud ou non. En ce qui concerne le premier point, nous avons noté que les métriques dont le calcul du coût de généralisation dépend de la hauteur de la hiérarchie, *Distortion* et *Total*, ont les meilleurs résultats pour les trois critères de qualité sur *florida_30162*. En revanche, sur *Adult data set*, nous n'avons observé aucun effet notable de cette caractéristique sur les performances des métriques. Pour le deuxième point, nous avons observé que la pondération sur l'ensemble des attributs quasi-identifiants choisie n'était pas un facteur déterminant pour obtenir une métrique avec de bonnes performances : rien ne se dégage de l'analyse des résultats sur les deux tables. Pour le troisième point, nous avons mis en lumière qu'une étape de normalisation dans le calcul du coût de généralisation du nœud est indispensable pour obtenir une métrique avec de bonnes performances. En effet, les deux métriques n'effectuant pas une telle étape de normalisation dans leur définition, *LLM* et *WLLM*, ont obtenu les moins bons résultats pour les trois critères de qualité étudiés et pour les deux tables.

Perspectives. Au vu des résultats présentés, nous pensons que les performances d'une métrique de perte d'information dépendent de la table sur laquelle elle est utillisée. Pour poursuivre ce travail, il serait intéressant de construire une métrique de perte d'information optimale à partir de la table et des hiérarchies de généralisation considérées. L'optimalité de la métrique proposée pourrait être mesurée grâce la qualité des tables k-anonymes produites pour des critères tels que le pourcentage de valeurs généralisées sur le nombre total de valeurs ou le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs mais aussi grâce à l'altération pour une métrique de référence. L'expérimentation à mener serait la suivante. Considérons un ensemble d'attributs quasi-identifiants, un uplet de hiérarchies de généralisation et une table. Nous commençons par construire une métrique de perte d'information sur l'ensemble des hiérarchies de généralisation (les valeurs choisies pourraient par exemple être aléatoires). Nous appliquons *GkAA* sur la table en utilisant la métrique créée pour produire une table k-anonyme. Nous évaluons la qualité de la table k-anonyme grâce à des critères. Puis nous changeons une ou plusieurs valeurs de la métrique et nous appliquons à nouveau *GkAA* sur la table. En fonction de la qualité de cette seconde table k-anonyme, nous pourrions dégager des directions de recherche dans lesquelles la métrique créée permet de produire de bonnes tables k-anonymes. L'un des problèmes de cette expérimentation est que le temps de calcul serait potentiellement trop important. Dans les expérimentations que nous avons menées en section 3.4 page 40, une exécution de l'algorithme *GkAA* durait en moyenne 45 minutes. De plus, la métrique créée dépend de la valeur de k pour laquelle nous avons construit des tables k-anonymes ; rien ne permet d'affirmer que la métrique créée sera optimale pour une autre valeur de k.

Une seconde piste de recherche serait de s'intéresser à la topologie des hiérarchies de généralisation pour construire des tables k-anonymes. Dans notre étude, les hiérarchies de généralisation ont été construites en respectant un sens sémantique. Par exemple, la hiérarchie de l'attribut *Éducation* de *Adult data set* a été construite en suivant le système scolaire en vigueur aux États-Unis. Cependant, quel serait l'impact sur la qualité des tables k-anonymes produites si les hiérarchies de généralisation ne tenaient pas compte de la sémantique ? Nous pourrions construire des hiérarchies de généralisation optimales en termes de nombre de nœuds dans chacun des niveaux. Par exemple, considérons un attribut quasi-identifiant à 2^n valeurs possibles. Au premier niveau de la hiérarchie, nous regroupons les feuilles deux par deux. Nous obtenons donc 2^{n-1} nœuds au premier niveau. Dans ce cas idéal, la hiérarchie de généralisation serait de hauteur $n+1$ et chaque niveau j contiendrait 2^{n-j} nœuds. Pour évaluer les performances de telles hiérarchies « optimales », l'expérimentation serait la suivante.

Considérons un ensemble d'attributs quasi-identifiants et une table sur cet ensemble. Choisissons une métrique de perte d'information. Fixons un uplet de hiérarchies de généralisation qui servira de référence (par exemple, les hiérarchies avec un sens sémantique). Considérons le uplet de hiérarchies « optimales ». Pour plusieurs valeurs de k, nous appliquons l'algorithme *GkAA* sur la table en utilisant la métrique choisie et le uplet de hiérarchies « optimales ». Nous évaluons la qualité des tables k-anonymes produites avec des critères tels que le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs. Nous pouvons ensuite comparer les résultats obtenus avec ceux pour lesquels les hiérarchies avec un sens sémantique ont été utilisées.

Chapitre 4

Optimisation multi-critère pour la k-anonymisation

Sommaire du présent chapitre

Lors de notre étude comparative de métriques de perte d'information dans le chapitre 3 page 25, nous ne nous sommes pas intéressés à des tables contenant des attributs sensibles : nous n'avons k-anonymisé que des tables ne contenant que des attributs quasi-identifiants. Or, l'une des principales faiblesses de la k-anonymité est qu'un manque de diversité peut apparaître dans les valeurs de l'attribut sensible dans les classes d'équivalence de la table k-anonyme et potentiellement révéler des informations (cf. chapitre 1 page 5). Comme la k-anonymité ne tient pas compte des attributs sensibles, rien ne garantit que les classes d'équivalence d'une table k-anonyme ne présenteront pas des répartitions des valeurs des attributs sensibles très déséquilibrées. Intuitivement, plus la valeur de k demandée est grande, plus la probabilité d'une meilleure répartition des valeurs des attributs sensibles dans les classes d'équivalence de la table k-anonyme est importante. Cependant, comme le nombre de k-anonymisation d'une table est de l'ordre de 2^n, il convient de définir ce que signifie une « meilleure répartition » des valeurs des attributs sensibles dans les classes d'équivalence d'une table k-anonyme. Dans ce chapitre, nous étudions la l-diversité [34] et la t-proximité [29], deux modèles d'anonymisation contrôlant la répartition des valeurs des attributs sensibles dans les classes d'équivalence d'une table.

Notre objectif est toujours de produire des tables k-anonymes. L'algorithme *GkAA* utilisé pour les expérimentations du chapitre 3 page 25 construit une version k-anonyme d'une table en guidant les fusions de classes d'équivalence en se basant uniquement sur le coût de généralisation des deux classes pour une métrique de perte d'information. Dans ce chapitre, nous nous proposons de modifier l'algorithme *GkAA* afin d'également prendre en compte les modèles de l-diversité et de t-proximité lors du processus de k-anonymisation. Nous allons développer de nouvelles stratégies correspondant chacune à une manière de guider les fusions de classes d'équivalence dans *GkAA*. Grâce à ces stratégies mêlant coût de généralisation pour une métrique, l-diversité et t-proximité, nous espérons que les tables k-anonymes produites respectent d'autres modèles d'anonymisation que la k-anonymité (ou possèdent d'autres garanties de protection de la vie privée que celles apportées par la k-anonymité).

Dans la suite de ce chapitre, nous reviendrons en section 4.1 page suivante sur les définitions de l-diversité et de t-proximité et nous introduirons des mesures permettant d'évaluer les niveaux de l-diversité et de t-proximité d'une table. Dans la section 4.2 page 61, nous présenterons un algorithme d'anonymisation produisant une

table respectant un modèle d'anonymisation et dans lequel une stratégie permet de guider les fusions de classes d'équivalence. Ensuite, nous détaillerons dans la section 4.3 page 62 sept stratégies d'optimisation à utiliser dans l'algorithme d'anonymisation précédent pour la production de tables k-anonymes de bonne qualité en termes de l-diversité et de t-proximité. Enfin, dans la section 4.4 page 65, nous présenterons le protocole expérimental nous permettant de comparer les performances des stratégies proposées lors d'un processus de k-anonymisation. Pour nos expérimentations, nous utiliserons les tables *Adult data set* et *florida_30162* présentées en section 2.4 page 22. Si celles-ci ont l'avantage d'être le reflet de données « réelles », elles ne nous permettent que d'expérimenter sur des jeux de données limités avec des répartitions des valeurs des attributs sensibles fixées et spécifiques aux données représentées. Afin d'étendre notre étude à une plus grande variété de répartitions des données, nous utiliserons des données simulées suivant différentes lois de probabilité. Nous conclurons cette étude dans la section 4.5 page 85.

La plupart des résultats détaillés dans ce chapitre sont présentés dans l'article [36].

4.1 l-diversité, t-proximité et mesures

Dans cette section, nous allons revenir sur les définitions de l-diversité et de t-proximité déjà évoquées dans le chapitre 1 page 5. Nous définirons ensuite des mesures permettant d'évaluer la qualité d'une table en termes de l-diversité et de t-proximité.

Les modèles de l-diversité et de t-proximité s'intéressent à la répartition des valeurs d'un attribut sensible dans une table et dans les classes d'équivalence de la table. Nous introduisons donc les applications p et p_T représentant respectivement la proportion d'une valeur sensible dans une table et la proportion d'une valeur sensible dans une classe d'équivalence d'une table dans les définitions 4.1.1 et 4.1.2.

Définition 4.1.1 (Proportion d'une valeur d'un attribut sensible dans une table)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$.

On définit l'application $p : S \times \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} \to [0,1]$ qui à toute valeur de l'attribut S et à toute table sur $(\mathcal{A},\mathcal{H})$ associe la proportion de la valeur dans la table :

$$\begin{array}{rccl} p: & S \times \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} & \longrightarrow & [0,1] \\ & (s,T) & \longmapsto & p(s,T) := \frac{1}{n}|\{E = (e_1,\ldots,e_r,s_E) \in T \,:\, s_E = s\}| \end{array} .$$

Définition 4.1.2 (Proportion d'une valeur d'un attribut sensible dans une classe d'équivalence)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A},\mathcal{H})$.

On définit l'application $p_T : S \times \mathcal{C}(T) \to [0,1]$ qui à toute valeur de l'attribut S et à toute classe d'équivalence de T associe la proportion de la valeur dans la classe d'équivalence :

$$\begin{array}{rccl} p_T: & S \times \mathcal{C}(T) & \longrightarrow & [0,1] \\ & (s,C) & \longmapsto & p_T(s,C) := \frac{1}{|C|}|\{E = (e_1,\ldots,e_r,s_E) \in C \,:\, s_E = s\}| \end{array} .$$

Remarque 4.1.1
Avec les notations des définitions 4.1.1 et 4.1.2, on a $\sum_{s\in S} p(s,T) = 1$ et, pour toute classe d'équivalence C de T, on a $\sum_{s\in S} p_T(s,C) = 1$.

4.1.1 La l-diversité

4.1.1.1 Première définition de la l-diversité

Définie par Machanavajjhala et co-auteurs dans [34], la l-diversité est un modèle d'anonymisation garantissant que, dans chaque classe d'équivalence d'une table, un ensemble de valeurs de l'attribut sensible soient « bien représentées ». Il est dit qu'une classe d'équivalence C est « bien représentée » par l valeurs d'un attribut sensible S s'il existe au moins $l \geqslant 2$ valeurs de S dans C telles que les l valeurs les plus fréquentes dans C aient une fréquence d'apparition « comparable ».

En d'autres termes, l'idée de la l-diversité est de garantir que, dans chaque classe d'équivalence, il y ait au moins l valeurs différentes de l'attribut sensible présentes en quantité suffisante. Ainsi, même si un adversaire connaît la classe d'équivalence de la table anonyme à laquelle appartient un individu, la probabilité de lui

associer une valeur sensible en particulier est faible. On évite ainsi les situations où l'appartenance à une classe d'équivalence donne des indices forts sur les valeurs de l'attribut de sensible (par exemple une classe dans laquelle 99% des individus souffriraient d'une maladie cardiaque).

La définition 4.1.3 reprend les notations du chapitre 2 page 13 pour formuler la notion de classe d'équivalence « bien représentée » par l valeurs de l'attribut sensible.

Définition 4.1.3 (Classe d'équivalence bien représentée par l valeurs sensibles)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A}, \mathcal{H})$. Soit $\mathcal{C}(T)$ l'ensemble des classes d'équivalence de T. Soit $C \in \mathcal{C}(T)$.

Posons $S_C = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C : s_E = s\}$ l'ensemble des valeurs distinctes de l'attribut sensible S présentes dans les enregistrements de C.

Soient $l \geqslant 2$ et $\epsilon \in [0, 1]$.

On dit que C est bien représentée par l valeurs de l'attribut S pour une tolérance ϵ si les deux conditions suivantes sont vérifiées :

1. $|S_C| \geqslant l$
2. $\forall i, j \in [\![1, l]\!]$, $|p_T(s_i, C) - p_T(s_j, C)| \leqslant \epsilon$ où $s_1, \ldots, s_l$ sont les l éléments de S_C ayant les plus grandes proportions dans C

Remarque 4.1.2
La valeur ϵ est une tolérance définie arbitrairement. Si $\epsilon = 0$ alors les l valeurs de l'attribut S ayant les plus grandes proportions dans C sont présentes en même proportion dans C. Si $\epsilon = 1$ alors les l valeurs de l'attribut S ayant les plus grandes proportions dans C peuvent avoir des proportions dans C aussi éloignées que l'on veut.

En utilisant la définition de classe d'équivalence bien représentée par l valeurs de l'attribut sensible pour une tolérance ϵ (cf. définition 4.1.3), nous pouvons définir la notion de table l-diverse pour une tolérance ϵ.

Définition 4.1.4 (Table l-diverse)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A}, \mathcal{H})$. Soient $l \in \mathbb{N}^*$ et $\epsilon \in [0, 1]$.

On dit que T est *l-diverse* pour une tolérance ϵ si chaque classe d'équivalence de T est bien représentée par l valeurs de S pour une tolérance ϵ.

Exemple 4.1.1
Soit $\mathcal{A} = \{Q, S\}$ un ensemble d'attributs avec $Q = \{q_1, q_2\}$ un attribut quasi-identifiant et $S = \{s_1, s_2, s_3\}$ un attribut sensible. On considère la table T sur $\mathcal{A}$ décrite dans la figure 4.1 page suivante.

La table T est composée de deux classes d'équivalence $C_1 = \{E^1, E^2, E^3, E^4, E^5\}$ et $C_2 = \{E^6, E^7, E^8\}$.

Cherchons pour quelles valeurs de $l \geqslant 2$ et de $\epsilon \geqslant 0$ les classes C_1 et C_2 sont bien représentées par l valeurs de l'attribut S pour une tolérance ϵ. Pour cela, nous utilisons la défintion 4.1.3.

Pour la classe C_1, on a $S_{C_1} = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C_1 : s_E = s\} = \{s_1, s_2, s_3\}$. D'après la première condition de la définition 4.1.3, $l \leqslant |S_{C_1}| = 3$ donc C_1 peut être bien représentée par 2 ou 3 valeurs de l'attribut S.

Calculons maintenant la proportion dans C_1 de chaque valeur de S présente dans C_1 en utilisant la définition 4.1.2 page précédente : $p_T(s_1, C_1) = \frac{2}{5}$, $p_T(s_2, C_1) = \frac{2}{5}$ et $p_T(s_3, C_1) = \frac{1}{5}$.

Si $l = 2$, les deux valeurs de S ayant les plus grandes proportions dans C_1 sont s_1 et s_2. Comme $p_T(s_1, C_1) = p_T(s_2, C_1)$, C_1 est bien représentée par deux valeurs de S pour tout $\epsilon \in [0, 1]$.

Si $l = 3$, les trois valeurs de S présentes dans C_1 doivent vérifier la seconde condition de la définition 4.1.3. Or $|p_T(s_1, C_1) - p_T(s_2, C_1)| = 0$, $|p_T(s_1, C_1) - p_T(s_3, C_1)| = 0{,}2$ et $|p_T(s_2, C_1) - p_T(s_3, C_1)| = 0{,}2$. Donc, pour que C_1 soit bien représentée par trois valeurs de l'attribut S, il faut que la tolérance ϵ soit supérieure à 0,2.

Pour la classe C_2, on a $S_{C_2} = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C_2 : s_E = s\} = \{s_1, s_2, s_3\}$. D'après la première condition de la définition 4.1.3, $l \leqslant |S_{C_2}| = 3$ donc C_2 peut être bien représentée par 2 ou 3 valeurs de l'attribut S.

Calculons maintenant la proportion dans C_2 de chaque valeur de S présente dans C_2 en utilisant la définition 4.1.2 page ci-contre : $p_T(s_1, C_2) = \frac{1}{3}$, $p_T(s_2, C_2) = \frac{1}{3}$ et $p_T(s_3, C_2) = \frac{1}{3}$.

Comme $p_T(s_1, C_2) = p_T(s_2, C_2) = p_T(s_3, C_2)$, C_2 est bien représentée par trois valeurs (et *a fortiori* par deux valeurs) de S pour toute tolérance $\epsilon \in [0, 1]$.

Pour conclure, la table T est 2-diverse pour tout $\epsilon \in [0, 1]$ et 3-diverse pour tout $\epsilon \in [0{,}2, 1]$ d'après la définition 4.1.4.

T	Q	S
E^1	q_1	s_1
E^2	q_1	s_1
E^3	q_1	s_2
E^4	q_1	s_2
E^5	q_1	s_3
E^6	q_2	s_1
E^7	q_2	s_2
E^8	q_2	s_3

FIGURE 4.1 – Représentation d'une table sur $\mathcal{A}$.

4.1.1.2 La l-diversité entropique

La définition 4.1.4 page précédente de l-diversité proposée n'est pas simple à utiliser en pratique. Comme illustré dans l'exemple 4.1.1 page précédente, le choix de la tolérance ϵ multiplie les vérifications à effectuer pour connaître le niveau de l-diversité d'une table.

Dans l'article [34], les auteurs introduisent la notion de table *l-diverse entropique*.

Définition 4.1.5 (Table l-diverse entropique)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A}, \mathcal{H})$. Soit $l \in \mathbb{N}^*$.

On dit que T est *l-diverse entropique* si, pour toute classe d'équivalence C de T, on a

$$-\sum_{s \in S_C} p_T(s, C) \ln(p_T(s, C)) \geqslant \ln(l),$$

avec $S_C = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C : s_E = s\}$.

Nous allons maintenant relier les définitions de table l-diverse pour une tolérance $\epsilon = 1$ et de table l-diverse entropique. Pour cela, nous revenons sur la notion d'entropie définie par Claude Shannon dans [42] (cf. définition 4.1.6).

Définition 4.1.6 (Entropie de Shannon)
Soit X une variable aléatoire discrète à $n \in \mathbb{N}^*$ valeurs $\{x_1, \ldots, x_n\}$ avec p_i la probabilité de x_i pour tout $i \in [\![1, n]\!]$.

L'entropie de X est définie par

$$\mathrm{H}(X) = -\sum_{i=1}^{n} p_i \ln(p_i).$$

Nous énonçons maintenant deux résultats de l'entropie de Shannon qui nous permettront de démontrer la proposition 4.1.1 page suivante reliant les deux définitions de l-diversité.

Lemme 4.1.1 (Inégalité de Gibbs)
Soit X une variable aléatoire discrète à $n \in \mathbb{N}^*$ valeurs $\{x_1, \ldots, x_n\}$ avec p_i la probabilité de x_i pour tout $i \in [\![1, n]\!]$.

Pour toute distribution de probabilté $(q_i)_{1 \leqslant i \leqslant n}$ sur X, on a

$$\mathrm{H}(X) \leqslant -\sum_{i=1}^{n} p_i \ln(q_i).$$

Propriété 4.1.1
Soit X une variable aléatoire discrète à $n \in \mathbb{N}^*$ valeurs $\{x_1, \ldots, x_n\}$ avec p_i la probabilité de x_i pour tout $i \in [\![1, n]\!]$.

On a

$$\mathrm{H}(X) \leqslant \ln(n).$$

Démonstration : On applique l'inégalité de Gibbs avec la distribution de probabilité uniforme $q_i = \frac{1}{n}$ pour tout $i \in [\![1, n]\!]$.

On obtient

$$\begin{aligned}
\mathrm{H}(X) &= -\sum_{i=1}^{n} p_i \ln(p_i) \\
&\leqslant -\sum_{i=1}^{n} p_i \ln(q_i) && \text{inégalité de Gibbs} \\
&\leqslant -\sum_{i=1}^{n} p_i \ln\left(\frac{1}{n}\right) && \text{car } q_i = \frac{1}{n}\ \forall i \in [\![1,n]\!] \\
&\leqslant \ln(n) \sum_{i=1}^{n} p_i && \text{car } \ln\left(\frac{1}{n}\right) = -\ln(n) \\
&\leqslant \ln(n) && \text{car } (p_i)_{1\leqslant i\leqslant n} \text{est une distribution de probabilité}
\end{aligned}$$

■

Proposition 4.1.1
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A}, \mathcal{H})$. Soit $l \in \mathbb{N}^*$.

Si T est l-diverse entropique alors T est l-diverse au sens de la définition 4.1.4 page 57 pour une tolérance $\epsilon = 1$.

Démonstration : Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A}, \mathcal{H})$. Soit $l \in \mathbb{N}^*$.

Supposons que T soit l-diverse entropique c'est-à-dire $\forall C \in \mathcal{C}(T)$ avec $S_C = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C : s_E = s\}$, $-\sum_{s\in S_C} p_T(s, C) \ln(p_T(s, C)) \geqslant \ln(l)$ d'après la définition 4.1.5 page précédente.

Montrons que T est l-diverse au sens de la définition 4.1.4 page 57 pour une tolérance $\epsilon = 1$ c'est-à-dire $\forall C \in \mathcal{C}(T)$ avec $S_C = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C : s_E = s\}$, $|S_C| \geqslant l$ et $\forall i, j \in [\![1, l]\!]$, $|p_T(s_i, C) - p_T(s_j, C)| \leqslant \epsilon$ où $s_1, \ldots, s_l$ sont les l éléments de S_C ayant les plus grandes proportions dans C.

Soit $C \in \mathcal{C}(T)$. Pour montrer la première condition de la définition, on revient à la définition de l'entropie de Shannon. On considère l'ensemble S_C comme une variable aléatoire discrète avec $|S_C|$ valeurs, chaque valeur s ayant une probabilité de $p_T(s, C)$.

On a

$$\mathrm{H}(S_C) = -\sum_{s\in S_C} p_T(s, C) \ln(p_T(s, C)) \leqslant \ln(|S_C|)$$

d'après la propriété 4.1.1 page précédente.

Or par hypothèse de la l-diversité entropique, on a $-\sum_{s\in S} p_T(s, C) \ln(p_T(s, C)) \geqslant \ln(l)$, donc

$$\ln(S_C) \geqslant -\sum_{s\in S_C} p_T(s, C) \ln(p_T(s, C)) \geqslant \ln(l).$$

En passant à l'exponentielle qui est une fonction croissante sur $\mathbb{R}$, on obtient

$$|S_C| \geqslant l.$$

Comme $0 \leqslant p_T(s, C) \leqslant 1$ pour tout $s \in S_C$, on a $0 \leqslant |p_T(s, C) - p_T(s', C)| \leqslant 1$ pour tous les s, s' dans S_C. Donc la seconde condition de la définition est vérifiée pour $\epsilon = 1$. ■

Nous nous appuyons sur la proposition 4.1.1 pour définir une mesure de la l-diversité d'une table.

Définition 4.1.7 (Mesure de la l-diversité)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ à n enregistrements.

On définit l'application $l_{div,T} : \mathcal{C}(T) \to \mathbb{R}$ qui à toute classe d'équivalence de T associe :

$$\begin{aligned}
l_{div,T} : \ \mathcal{C}(T) &\longrightarrow \mathbb{R} \\
C &\longmapsto \exp\left(-\sum_{s\in S_C} p_T(s, C) \ln(p_T(s, C))\right)
\end{aligned}$$

avec $S_C = \{s \in S : \exists E = (e_1, \ldots, e_m, s_E) \in C : s_E = s\}$.

On définit l'application $l_{div} : \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} \to \mathbb{R}$ qui à toute table dans $\mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ associe sa *valeur de l-diversité* :

$$\begin{array}{rccc} l_{div} : & \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} & \longrightarrow & \mathbb{R} \\ & T & \longmapsto & \min\limits_{C \in \mathcal{C}(T)} l_{div,T}(C) \end{array} .$$

Remarque 4.1.3
Avec les notations de la définition 4.1.7 page précédente, dire que le valeur de l-diversité de T est $l_{div}(T)$ signifie que T est $l_{div}(T)$-diverse entropique et, par la proposition 4.1.1 page précédente, que T est $l_{div}(T)$-diverse pour une tolérance $\epsilon = 1$.

Montrons que T est $l_{div}(T)$-diverse entropique c'est-à-dire $\forall C \in \mathcal{C}(T)$, $-\sum_{s \in S_C} p_T(s, C) \ln(p_T(s, C)) \geqslant \ln(l_{div}(T))$ d'après la définition 4.1.5 page 58.

Soit $C \in \mathcal{C}(T)$. Par définition de $l_{div}(T)$, on a $l_{div}(T) \leqslant l_{div,T}(C)$. Par définition de $l_{div,T}(C)$, on a $l_{div}(T) \leqslant \exp\left(-\sum_{s \in S_C} p_T(s, C) \ln(p_T(s, C))\right)$. En appliquant la fonction ln, qui est une fonction croissante sur $\mathbb{R}^*_+$, à l'inégalité précédente, on obtient $\ln(l_{div}(T)) \leqslant -\sum_{s \in S_C} p_T(s, C) \ln(p_T(s, C))$.

4.1.2 La t-proximité

Définie par Li et co-auteurs dans [29], la t-proximité demande à ce que la répartition des valeurs de l'attribut sensible dans chaque classe d'équivalence de la table ne soit pas plus éloignée qu'un seuil t de la répartition des valeurs de l'attribut sensible dans la table entière. On dit alors que la classe d'équivalence a une t-proximité. La connaissance de la répartition des valeurs de l'attribut sensible dans la table entière est un prérequis à la mise en pratique de la t-proximité.

En d'autres termes, l'idée de la t-proximité est de garantir que, dans chaque classe d'équivalence, la répartition des valeurs de l'attribut sensible soit approximativement la même que la répartition des valeurs de l'attribut sensible dans toute la table. Ainsi, même en connaissant la classe d'équivalence de la table anonyme à laquelle appartient un individu, la probabilité de découvrir la valeur sensible d'un individu est la même que dans la table entière.

La définition 4.1.9 reprend les notations du chapitre 2 page 13 pour formuler la notion de t-proximité d'une classe d'équivalence. Afin de déterminer la distance entre deux répartitions, nous utilisons la distance associée à la norme vetorielle 1 définie dans la définition 4.1.8 (d'autres distances sont présentées dans l'article [29]).

Définition 4.1.8 (Distance associée à la norme 1 sur $\mathbb{R}^n$)
Soit $n \in \mathbb{N}^*$.

La norme 1 sur $\mathbb{R}^n$ est définie par :

$$\begin{array}{rccc} \|.\|_1 : & \mathbb{R}^n & \longrightarrow & \mathbb{R} \\ & x & \longmapsto & \sum_{i=1}^n |x_i| \end{array} ,$$

avec $x = (x_1, \ldots, x_n)$.

La distance associée à la norme 1 sur $\mathbb{R}^n$ est définie par :

$$\begin{array}{rccc} d_{\|.\|_1} : & \mathbb{R}^n \times \mathbb{R}^n & \longrightarrow & \mathbb{R} \\ & (x, y) & \longmapsto & \|x - y\|_1 \end{array} .$$

Définition 4.1.9 (t-proximité d'une classe d'équivalence)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible S. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. On pose $S = \{s_1, \ldots, s_q\}$ pour $q \in \mathbb{N}^*$. Soit $n \in \mathbb{N}^*$. Soit $T \in \mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ une table sur $(\mathcal{A}, \mathcal{H})$. Soit $\mathcal{C}(T)$ l'ensemble des classes d'équivalence de T. Soit $C \in \mathcal{C}(T)$. Soit $t \in [0, 1]$.

On note $P(S, T) = (p(s_1, T), \ldots, p(s_q, T))$ le vecteur des répartitions des valeurs de l'attribut S dans T et $P_T(S, C) = (p_T(s_1, C), \ldots, p_T(s_q, C))$ le vecteur des répartitions des valeurs de l'attribut S dans C.

On dit que C a une *t-proximité* pour la distance $d_{\|.\|_1}$ si la condition suivante est vérifiée :

$$d_{\|.\|_1}(P_T(S, C), P(S, T)) \leqslant t.$$

Définition 4.1.10 (t-proximité d'une table)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. Soit T une table sur $\mathcal{A}$. Soit $t \in \mathbb{R}$.

On dit que T a *une t-proximité* pour la distance $d_{\|.\|_1}$ si chaque classe d'équivalence de T a une t-proximité pour la distance $d_{\|.\|_1}$.

Nous nous appuyons sur la définition 4.1.10 page précédente pour définir une mesure de la t-proximité d'une table.

Définition 4.1.11 (Mesure de la *t*-proximité)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. On pose $S = \{s_1, \ldots, s_q\}$ pour $q \in \mathbb{N}^*$. Soit $n \in \mathbb{N}^*$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ à n enregistrements.

On définit l'application $t_{prox,T} : \mathcal{C}(T) \to \mathbb{R}$ qui à toute classe d'équivalence de T associe :

$$\begin{array}{rccl} t_{prox,T} : & \mathcal{C}(T) & \longrightarrow & \mathbb{R} \\ & C & \longmapsto & d_{\|.\|_1}(P_T(S,C), P(S,T)) \end{array},$$

avec $P(S,T) = (p(s_1,T), \ldots, p(s_q,T))$ et $P_T(S,C) = (p_T(s_1,C), \ldots, p_T(s_q,C))$.

On définit l'application $t_{prox} : \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} \to \mathbb{R}$ qui à tout table dans $\mathcal{T}^n_{(\mathcal{A},\mathcal{H})}$ associe sa *valeur de t-proximité* pour la distance $d_{\|.\|_1}$:

$$\begin{array}{rccl} t_{prox} : & \mathcal{T}^n_{(\mathcal{A},\mathcal{H})} & \longrightarrow & \mathbb{R} \\ & T & \longmapsto & \max\limits_{C \in \mathcal{C}(T)} t_{prox,T}(C) \end{array}.$$

Remarque 4.1.4
Avec les notations de la définition 4.1.11, dire que le valeur de t-proximité de T pour la distance $d_{\|.\|_1}$ est $t_{prox}(T)$ signifie que T a une $t_{prox}(T)$-proximité pour la distance $d_{\|.\|_1}$.

En effet, montrons que $\forall C \in \mathcal{C}(T)$, $d_{\|.\|_1}(P_T(S,C), P(S,T)) \leqslant t$ d'après la définition 4.1.10 page précédente.

Soit $C \in \mathcal{C}(T)$. Par définition de $t_{prox}(T)$, on a $t_{prox}(T) \geqslant t_{prox,T}(C)$. Par définition de $t_{prox,T}(C)$, on a $t_{prox}(T) \geqslant d_{\|.\|_1}(P_T(S,C), P(S,T))$.

4.2 Algorithme d'anonymisation

Dans cette section, nous allons présenter un algorithme d'anonymisation. Il s'agit d'une généralisation de l'algoritme *GkAA* présenté en section 3.3.2 page 37 qui produisait une table k-anonyme en optimisant les fusions de classes d'équivalence grâce à une métrique de perte d'information. Les modèles de k-anonymisation, de l-diversité et de t-proximité étant NP-difficiles (cf. les articles [37, 14], [49, 13] et [32] respectivement), l'algorithme proposé utilise une méthode heuristique pour guider la production d'une version anonyme d'une table.

Le *Greedy Anonymization Algorithm* ou *GAA* vise à produire une version d'une table respectant un modèle d'anonymisation en optimisant les fusions de classes d'équivalence selon une stratégie déterminée.

L'algorithme 2 est une formalisation de *GAA*. Il prend en entrées $\mathcal{A}$ un ensemble d'attributs, $\mathcal{H}$ un vecteur de hiérachies de généralisation, T une table sur $(\mathcal{A}, \mathcal{H})$, Φ un modèle d'anonymisation et *Strat* une stratégie permettant de sélectionner la fusion à effectuer à chaque tour de boucle.

Algorithme 2 Greedy Anonymization Algorithm

Entrées: $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{A}$, T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$, Φ un modèle d'anonymisation et *Strat* une stratégie de sélection des fusions
Sortie: Une table généralisée sur $(T, \mathcal{H})$ respectant le modèle Φ

```
1: procédure GAA(A, H, T, Φ, Strat)
2:     tant que T ne respecte pas le modèle Φ faire
3:         On choisit arbitrairement une classe d'équivalence C_s de T de taille minimale ne respectant pas le modèle Φ
4:         On cherche C une classe d'équivalence de T différente de C_s respectant les conditions de la stratégie Strat
5:         T ← gen_T(C_s ∪ C)
6:     fin tant que
7:     Retourne T
8: fin procédure
```

Par exemple, dans *GkAA*, le choix de la classe d'équivalence C à fusionner avec C_s est déterminé par le coût de généralisation pour une métrique de perte d'information μ. La stratégie correspondante à donner comme paramètre *Strat* à *GAA* a pour condition :

$$C \in \{C' \in \mathcal{C}(T) - C_s \,:\, \overline{\mu}(C_s, C') = \min_{C'' \in \mathcal{C}(T) - C_s} \overline{\mu}(C_s, C'')\}.$$

La condition précédente traduit le fait que la classe C est choisie de telle sorte que son coût de généralisation avec C_s pour la métrique μ soit minimal.

Les remarques 3.3.1 page 37 et 3.3.2 page 38 faites sur *GkAA* sont également valables pour *GAA*. *GAA* est un algorithme déterministe : deux exécutions avec les mêmes paramètres et les mêmes configurations donneront le même résultat en sortie. Pour $k < k'$ deux nombres entiers, l'exécution de *GAA* pour k' fournit la sortie de l'exécution de *GAA* pour k à la fin d'un tour de boucle.

4.3 Stratégies d'optimisation

Dans cette section, nous allons présenter sept stratégies d'optimisation mêlant coût de généralisation pour une métrique de perte d'information, valeur de l-diversité et valeur de t-proximité à utiliser dans l'algorithme d'anonymisation *GAA* comme paramètre *Strat*. Ces stratégies d'optimisation fournissent des conditions de sélection de la classe C à fusionner avec la classe C_s à chaque tour de *GAA* (cf. section 4.2 page précédente).

Chaque stratégie a pour but d'optimiser une ou plusieurs des mesures suivantes :

— le coût de généralisation représenté par $\overline{\mu}$ avec μ une métrique de perte d'information définie en section 3.1 page 26

— la valeur de l-diversité représentée par l_{div} définie en section 4.1.1 page 56

— la valeur de t-proximité représentée par t_{prox} définie en section 4.1.2 page 60

Deux approches sont possibles :

— deux mesures sont considérées successivement. La fusion sélectionnée est choisie parmi les fusions optimisant la seconde mesure parmi celles optimisant la première mesure

— la fusion sélectionnée est choisie parmi celles optimisant deux mesures en même temps (scalarisation)

Dans la première approche, nous évaluons la fusion de C_s avec toutes les autres classes d'équivalence selon une première mesure puis, parmi les classes qui ont obtenu le meilleur résultat pour la première mesure, nous choisissons une classe dont la fusion avec C_s donne le meilleur résultat pour la seconde mesure.

Par exemple, considérons la stratégie prenant compte du coût de généralisation pour une métrique μ puis de la valeur de l-diversité. A chaque tour de *GAA*, la classe C est choisie de telle sorte que sa fusion avec C_s soit parmi celles maximisant la valeur de l-diversité parmi celles minimisant le coût de généralisation pour μ. En d'autres termes, nous calculons les coûts de généralisation pour μ de C_s avec toutes les autres classes d'équivalence. Parmi les classes telles que le coût de généralisation avec C_s pour μ est minimal, nous choisissons une classe telle que la valeur de l-diversité de sa fusion avec C_s soit maximale.

Dans la seconde approche, l'objectif est d'optimiser les fusions de classes d'équivalence selon deux critères en même temps. Pour cela, nous allons définir deux applications : l'une mêlant le coût de généralisation et la valeur de l-diversité (cf. définition 4.3.1), l'autre mêlant le coût de généralisation et la valeur de t-proximité (cf. définition 4.3.2 page suivante). Rappelons que le coût de généralisation et la valeur de t-proximité sont à minimiser dans une table et que la valeur de l-diversité est à maximiser.

Définition 4.3.1 (Coût de généralisation et valeur de l-diversité)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_\mathcal{Q}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ à n enregistrements. Soit μ une métrique sur $\mathcal{H}$.

On définit l'application $\overline{\mu}_{l_{div}} : \mathcal{C}(T) \times \mathcal{C}(T) \to \mathbb{R}$ qui à tout couple de classes d'équivalence de T associe leur coût de généralisation divisé par la valeur de l-diversité de la table généralisée sur $(T, \mathcal{H})$ dans laquelle les deux classes ont été fusionnées :

$$\begin{array}{rccc} \overline{\mu}_{l_{div}} : & \mathcal{C}(T) \times \mathcal{C}(T) & \longrightarrow & \mathbb{R} \\ & (C, C') & \longmapsto & \frac{\overline{\mu}(C,C')}{l_{div}(gen_T(C \cup C'))} \end{array} .$$

Pour minimiser $\overline{\mu}_{l_{div}}$, nous pouvons minimiser $\overline{\mu}$ ou maximimiser la valeur de l-diversité. L'idée est de proposer un compromis entre coût de généralisation pour μ et valeur de l-diversité. En d'autres termes, nous autorisons un coût de généralisation pour μ x fois plus important si la fusion obtenue a une valeur de l-diversité de x.

Exemple 4.3.1
Soit T une table sur un ensemble d'attributs $\mathcal{A}$ contenant un ensemble $\mathcal{Q}$ d'attributs quasi-identifiants et un attribut sensible S. Soit μ une métrique sur un uplet de hiérarchies de $\mathcal{Q}$. Supposons que T ait trois classes d'équivalence C, C' et C'' telles que :

$$\overline{\mu}(C, C') = 3 \text{ et } l_{div}(gen_T(C \cup C')) = 2$$
$$\overline{\mu}(C, C'') = 12 \text{ et } l_{div}(gen_T(C \cup C'')) = 8.$$

On cherche la meilleure classe à fusionner avec C au regard de la mesure $\overline{\mu}_{l_{div}}$.

Pour rappel, le coût de généralisation pour μ est une mesure à minimiser et la valeur de l-diversité est une mesure à maximiser. Ainsi, la fusion de C et C'' dans T est moins intéressante pour le coût de généralisation pour μ mais est meilleure pour la valeur de l-diversité que la fusion de C et C' dans T.

On remarque que le coût de généralisation pour μ de C et C' est quatre fois moins élevé que le coût de généralisation pour μ de C et C'' ($\overline{\mu}(C, C'') = 4 \times \overline{\mu}(C, C')$). Or la valeur de l-diversité de la version de T dans laquelle C et C' ont été fusionnées est quatre fois moins élevée que la valeur de l-diversité de la table dans laquelle C et C'' ont été fusionnées ($l_{div}(gen_T(C \cup C'')) = 4 \times l_{div}(gen_T(C \cup C'))$).

On a

$$\overline{\mu}_{l_{div}}(C, C') = \frac{\overline{\mu}(C, C')}{l_{div}(gen_T(C \cup C'))} = \frac{3}{2}$$

et

$$\overline{\mu}_{l_{div}}(C, C'') = \frac{\overline{\mu}(C, C'')}{l_{div}(gen_T(C \cup C''))} = \frac{3}{2}.$$

Ainsi, les deux fusions sont équivalentes pour la mesure $\overline{\mu}_{l_{div}}$.

Définition 4.3.2 (Coût de généralisation et valeur de *t*-proximité)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit $n \in \mathbb{N}^*$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ à n enregistrements. Soit μ une métrique sur $\mathcal{H}$.

On définit l'application $\overline{\mu}_{t_{prox}} : \mathcal{C}(T) \times \mathcal{C}(T) \to \mathbb{R}$ qui à tout couple de classes d'équivalence de T associe leur coût de généralisation multiplié par la valeur de t-proximité de la table généralisée sur $(T, \mathcal{H})$ dans laquelle les deux classes ont été fusionnées :

$$\begin{array}{rccc} \overline{\mu}_{t_{prox}} : & \mathcal{C}(T) \times \mathcal{C}(T) & \longrightarrow & \mathbb{R} \\ & (C, C') & \longmapsto & \overline{\mu}(C, C') \times t_{prox}(gen_T(C \cup C')) \end{array}.$$

Pour maximiser $\overline{\mu}_{t_{prox}}$, nous pouvons minimiser $\overline{\mu}$ ou minimiser la valeur de t-proximité. L'idée est de proposer un compromis entre coût de généralisation pour μ et valeur de t-proximité. En d'autres termes, nous autorisons un coût de généralisation pour μ x fois plus important si la fusion obtenue a une valeur de t-proximité de $\frac{1}{x}$.

Exemple 4.3.2
Soit T une table sur un ensemble d'attributs $\mathcal{A}$ contenant un ensemble $\mathcal{Q}$ d'attributs quasi-identifiants et un attribut sensible S. Soit μ une métrique sur un uplet de hiérarchies sur $\mathcal{Q}$. Supposons que T ait trois classes d'équivalence C, C' et C'' telles que :

$$\overline{\mu}(C, C') = 3 \text{ et } t_{prox}(gen_T(C \cup C')) = 1$$
$$\overline{\mu}(C, C'') = 12 \text{ et } t_{prox}(gen_T(C \cup C'')) = 0{,}25.$$

On cherche la meilleure classe à fusionner avec C au regard de la mesure $\overline{\mu}_{t_{prox}}$.

Pour rappel, le coût de généralisation pour μ et la valeur de t-proximité sont des mesures à minimiser. Ainsi, la fusion de C et C'' dans T est moins intéressante pour le coût de généralisation pour μ mais est meilleure pour la valeur de t-proximité que la fusion de C et C' dans T.

On remarque que le coût de généralisation pour μ de C et C' est quatre fois moins élevé que le coût de généralisation pour μ de C et C'' ($\overline{\mu}(C, C'') = 4 \times \overline{\mu}(C, C')$). Or la valeur de t-proximité de la version de T dans laquelle C et C' ont été fusionnées est quatre fois plus élevée que la valeur de t-proximité de la table dans laquelle C et C'' ont été fusionnées ($t_{prox}(gen_T(C \cup C'')) = \frac{1}{4} \times t_{prox}(gen_T(C \cup C'))$).

On a

$$\overline{\mu}_{t_{prox}}(C, C') = \overline{\mu}(C, C') \times t_{prox}(gen_T(C \cup C')) = 3$$

et

$$\overline{\mu}_{t_{prox}}(C, C'') = \overline{\mu}(C, C'') \times t_{prox}(gen_T(C \cup C'')) = 3.$$

Ainsi, les deux fusions sont équivalentes pour la mesure $\overline{\mu}_{t_{prox}}$.

Pour alléger les descriptions des stratégies d'optimisation proposées, nous allons définir des sous-ensembles particuliers de classes d'équivalence respectant certaines propriétés.

Définition 4.3.3 (Sous-ensembles particuliers de classes d'équivalence)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. Soit $\mathcal{C}(T)$ l'ensemble des classes d'équivalence de T. Soit μ une métrique sur $\mathcal{H}$.

Soit $C \in \mathcal{C}(T)$. On définit cinq sous-ensembles de $\mathcal{C}(T)$ dépendant de C.

$\Delta_{\overline{\mu}}(C, \mathcal{C}(T))$ est l'ensemble des classes d'équivalence différentes de C telles que leur fusion avec C minimisent le coût de généralisation pour μ. Il est défini par :

$$\Delta_{\overline{\mu}}(C, \mathcal{C}(T)) = \{C' \in \mathcal{C}(T) - C \,:\, \overline{\mu}(C, C') = \min_{C'' \in \mathcal{C}(T)-C} \overline{\mu}(C, C'')\}.$$

$\Delta_{l_{div}}(C, \mathcal{C}(T))$ est l'ensemble des classes d'équivalence différentes de C telles que leur fusion avec C maximisent la valeur de l-diversité. Il est défini par :

$$\Delta_{l_{div}}(C, \mathcal{C}(T)) = \{C' \in \mathcal{C}(T) - C \,:\, l_{div}(gen_T(C \cup C')) = \max_{C'' \in \mathcal{C}(T)-C} l_{div}(gen_T(C \cup C''))\}.$$

$\Delta_{t_{prox}}(C, \mathcal{C}(T))$ est l'ensemble des classes d'équivalence différentes de C telles que leur fusion avec C minimisent la valeur de t-proximité. Il est défini par :

$$\Delta_{t_{prox}}(C, \mathcal{C}(T)) = \{C' \in \mathcal{C}(T) - C \,:\, t_{prox}(gen_T(C \cup C')) = \min_{C'' \in \mathcal{C}(T)-C} t_{prox}(gen_T(C \cup C''))\}.$$

$\Delta_{\overline{\mu}_{l_{div}}}(C, \mathcal{C}(T))$ est l'ensemble des classes d'équivalence différentes de C telles que leur fusion avec C minimisent $\overline{\mu}_{l_{div}}$. Il est défini par :

$$\Delta_{\overline{\mu}_{l_{div}}}(C, \mathcal{C}(T)) = \{C' \in \mathcal{C}(T) - C \,:\, \overline{\mu}_{l_{div}}(C, C') = \min_{C'' \in \mathcal{C}(T)-C} \overline{\mu}_{l_{div}}(C, C'')\}.$$

$\Delta_{\overline{\mu}_{t_{prox}}}(C, \mathcal{C}(T))$ est l'ensemble des classes d'équivalence différentes de C telles que leur fusion avec C minimisent $\overline{\mu}_{t_{prox}}$. Il est défini par :

$$\Delta_{\overline{\mu}_{t_{prox}}}(C, \mathcal{C}(T)) = \{C' \in \mathcal{C}(T) - C \,:\, \overline{\mu}_{t_{prox}}(C, C') = \min_{C'' \in \mathcal{C}(T)-C} \overline{\mu}_{t_{prox}}(C, C'')\}.$$

Nous définissons les sept stratégies suivantes par les conditions de sélection de la classe C à fusionner avec C_s dans GAA.

Définition 4.3.4 (Stratégies d'optimisation)
Soit $\mathcal{A} = \{Q_1, \ldots, Q_m, S\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants et d'un attribut sensible. On pose $\{Q_1, \ldots, Q_m\}$. Soit $\mathcal{H} \in \mathcal{H}_{\mathcal{Q}}$ un m-uplet de hiérarchies des attributs quasi-identifiants de $\mathcal{Q}$. Soit T une table sur $(\mathcal{A}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$. Soit $\mathcal{C}(T)$ l'ensemble des classes d'équivalence de T. Soit μ une métrique sur $\mathcal{H}$. Soit Φ un modèle d'anonymisation.

Soit C_s la classe d'équivalence de taille minimale choisie lors d'un tour donné de l'exécution de $GAA(\mathcal{A}, \mathcal{H}, T, \Phi, Strat)$.

Stratégie 1 C est choisie parmi les classes telles que leur fusion avec C_s minimisent le coût de généralisation pour μ. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{\overline{\mu}}(C_s, \mathcal{C}(T)).$$

Stratégie 2 C est choisie parmi les classes telles que leur fusion avec C_s maximisent la valeur de l-diversité parmi celles telles que leur fusion avec C_s minimisent le coût de généralisation pour μ. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{l_{div}}(C_s, \Delta_{\overline{\mu}}(C_s, \mathcal{C}(T))).$$

Stratégie 3 C est choisie parmi les classes telles que leur fusion avec C_s minimisent le coût de généralisation pour μ parmi celles telles que leur fusion avec C_s maximisent la valeur de l-diversité. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{\overline{\mu}}(C_s, \Delta_{l_{div}}(C_s, \mathcal{C}(T))).$$

Stratégie 4 C est choisie parmi les classes telles que leur fusion avec C_s minimisent le coût de généralisation pour μ divisé par la valeur de l-diversité. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{\overline{\mu}_{l_{div}}}(C_s, \mathcal{C}(T)).$$

Stratégie 5 C est choisie parmi les classes telles que leur fusion avec C_s minisent la valeur de t-proximité parmi celles telles que leur fusion avec C_s minimisent le coût de généralisation pour μ. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{t_{prox}}\left(C_s, \Delta_{\overline{\mu}}\left(C_s, \mathcal{C}(T)\right)\right).$$

Stratégie 6 C est choisie parmi les classes telles que leur fusion avec C_s minimisent le coût de généralisation pour μ parmi celles telles que leur fusion avec C_s minimisent la valeur de t-proximité. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{\overline{\mu}}\left(C_s, \Delta_{t_{prox}}\left(C_s, \mathcal{C}(T)\right)\right).$$

Stratégie 7 C est choisie parmi les classes telles que leur fusion avec C_s minimisent le coût de généralisation pour μ multiplié par la valeur de t-proximité. La condition de sélection de C associée à cette stratégie est

$$C \in \Delta_{\overline{\mu}_{t_{prox}}}\left(C_s, \mathcal{C}(T)\right).$$

La Stratégie 1, dans laquelle seul le coût de généralisation est pris en compte, nous servira de référence pour les trois critères à optimiser. En effet, pour les valeurs de l-diversité et de t-proximité, les choix de fusions effectués par cette stratégie peuvent être considérés comme aléatoires. Pour le coût de généralisation pour μ, nous pouvons espérer que les tables k-anonymes produites en utilisant la Stratégie 1 aient de bons résultats. Les Stratégies 2 à 4 optimisent à la fois la l-diversité et le coût de généralisation pour μ. Les Stratégies 5 à 7 optimisent à la fois la t-proximité et le coût de généralisation pour μ.

4.4 Expérimentations

Dans cette section, nous allons étudier les performances des sept stratégies d'optimisation présentées en section 4.3 page 62. Pour cela, nous allons revenir sur le protocole expérimental mis en place en section 4.4.1 et nous analyserons les résultats obtenus en section 4.4.2 page 68.

4.4.1 Protocole expérimental

Nous souhaitons étudier les performances des stratégies d'optimisation lorsqu'elles sont utilisées lors d'un processus de k-anonymisation pour guider le choix des fusions à effectuer à chaque tour. Pour cela, nous proposons le protocole expérimental suivant.

Nous allons mener des expérimentations sur deux tables de 30 162 enregistrements : *Adult data set* et *florida_30162* (cf. section 2.4 page 22).

Soit $\mathcal{S} = \{\text{Stratégie 1}, \text{Stratégie 2}, \text{Stratégie 3}, \text{Stratégie 4}, \text{Stratégie 5}, \text{Stratégie 6}, \text{Stratégie 7}\}$ l'ensemble des stratégies d'optimisation que nous allons étudier (cf. section 4.3 page 62).

Certaines des stratégies de $\mathcal{S}$ se basent sur une métrique de perte d'information. Nous avons choisi la métrique *NLLM* (cf. section 4.3 page 62) pour notre étude.

Pour mener nos expérimentations, nous utilisons l'algorithme d'anonymisation *GAA* (cf. algorithme 2 page 61). Notre objectif étant la k-anonymisation de tables, le modèle d'anonymisation Φ passé en paramètre de *GAA* est la k-anonymité.

Dans ce chapitre, nous nous intéressons à la l-diversité et à la t-proximité. Ces deux modèles d'anonymisation influant sur les attributs sensibles d'une table, il est indispensable de spécifier l'attribut sensible étudié avant d'appliquer l'algorithme *GAA* sur une table. Pour une table T sur un ensemble d'attributs $\mathcal{A} = \{A_1, \ldots, A_m\}$, nous noterons T_{A_j} la table dans laquelle l'attribut A_j est considéré comme un attribut sensible et $\mathcal{A} - A_j$ sont des attributs quasi-identifiants pour $j \in [\![1, m]\!]$. Nous dirons que T_{A_j} est une *configuration de table*.

Nous allons mener deux types d'expérimentations selon le choix de l'attribut sensible étudié dans la table :

— l'attribut sensible étudié est un attribut présent dans la table d'origine. Il contient des données « réelles ».

— l'attribut sensible étudié est une nouvelle colonne ajoutée à la table d'origine. Il contient des données simulées suivant une répartition prédéterminée.

Premier type d'expérimentations Pour le premier type d'expérimentations, nous avons choisi comme attribut sensible dans *Adult data set* les attributs *Age* (72 valeurs possibles, cf hiérarchie en figure A.1 page 137) et *Statut marital* (7 valeurs possibles, cf hiérarchie en figure A.4 page 137) et dans *florida_30162* l'attribut *Parti politique* (9 valeurs possibles, cf hiérarchie en figure A.14 page 139).

Notons $K_1 = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000, 10\,000, 15\,000]$ l'ensemble des valeurs de k choisies pour ce premier type d'expérimentations. Pour chacune des trois configurations de table, pour chaque stratégie de $\mathcal{S}$, nous n'avons pas appliqué GAA pour les 14 valeurs de $k \in K_1$. Cela aurait représenté $14 \times 7 \times 3 = 294$ exécutions de GAA. A partir des remarques 3.3.1 page 37 et 3.3.2 page 38 de la section 3.3.2 page 37, nous avons procédé comme suit : pour chaque configuration de table, pour chaque stratégie de $\mathcal{S}$, nous avons exécuté GAA avec $k = \max(K_1)$ c'est-à-dire $k = 15\,000$. Lors de cette exécution, nous avons sauvegardé la première table k'-anonyme obtenue pour tout $k' \in K_1$.

Ainsi, nous n'avons à exécuter qu'une fois GAA pour chaque couple (configuration de table, stratégie) pour obtenir les 14 tables k-anonymes pour $k \in K_1$. Cela représente $7 \times 3 = 21$ exécutions de GAA au total.

Les résultats obtenus pour ces premières expérimentations seront analysés dans la section 4.4.2.1 page 68.

Second type d'expérimentations Avec des données « réelles », nous sommes limités dans le nombre de valeurs possibles de l'attribut sensible et nous ne pouvons pas contrôler la répartition de ses valeurs dans la table. C'est pourquoi, pour le second type d'expérimentations, afin d'analyser le comportement des stratégies pour différents types d'attribut sensible, nous avons généré aléatoirement des ensembles de données à utiliser comme attribut sensible dans les tables *Adult data set* et *florida_30162*. Ces ensembles de données sont ajoutés comme des colonnes sensibles dans les tables, les attributs originaux des tables sont des quasi-identifiants.

Afin de couvrir plusieurs possibilités de répartition des données, nous avons créé des attributs sensibles avec 5, 10, 20, 50, 100, 200 et 500 valeurs possibles. Pour chaque nouvel attribut sensible, nous générons des ensembles de données de taille 30 162 dans lesquels la répartition des valeurs suivent l'une des trois lois de probabilité suivantes : la loi Équivalente, la loi Géométrique ou la loi Normale centrée réduite. Les ensembles de données sont de taille 30 162 pour correspondre aux 30 162 enregistrements des tables *Adult data set* et *florida_30162*.

La loi Équivalente correspond à une répartition homogène des valeurs : chaque valeur de l'attribut sensible apparaît environ $\frac{\text{taille de l'ensemble de données}}{\text{nombre de valeurs possibles de l'attribut}}$ fois dans l'ensemble de données généré.

La loi Géométrique de paramètre p représente le nombre d'échecs à une épreuve de Bernoulli avant d'obtenir un premier succès, sachant que la probabilité d'un succès est p. Nous avons choisi $p = 0{,}5$.

La loi Normale est une loi de probabilité à densité fréquemment utilisée pour modéliser des phénomènes naturels. Elle est définie par deux paramètres : sa moyenne μ et sa variance σ^2. La loi Normale centrée réduite a pour paramètres $\mu = 0$ et $\sigma = 1$

Les diagrammes bâtons de la figure 4.2 page ci-contre représentent les répartitions des données selon la loi de probabilité utilisée pour les attributs sensibles créés. Chaque bâton d'un diagramme bâton correspond au nombre d'occurrences d'une valeur de l'attribut sensible dans l'ensemble de données généré. Par exemple, sur le diagramme bâton 4.2a page suivante, le premier bâton bleu correspond au nombre d'occurrences de la première valeur de l'attribut sensible à cinq valeurs possibles dans l'ensemble de données généré dans lequel la répartition des valeurs suit la loi Géométrique. Le quatrième bâton orange correspond au nombre d'occurrences de la quatrième valeur de l'attribut sensible à cinq valeurs possibles dans l'ensemble de données généré dans lequel la répartition des valeurs suit la loi Normale centrée réduite.

Nos expérimentations portent sur $2 \times 7 \times 3 = 42$ configurations de table (2 tables, 7 attributs sensibles, 3 lois de probabilité).

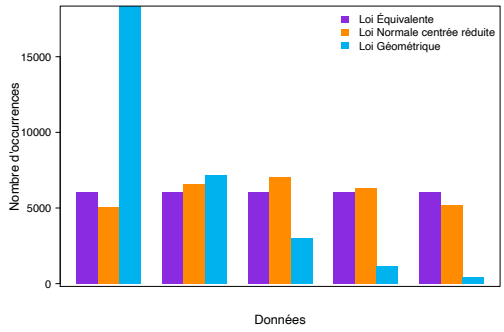
Notons $K_2 = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000]$ l'ensemble des valeurs de k choisies pour ce second type d'expérimentations. Comme pour le premier type d'expérimentation, pour chaque configuration de table, pour chaque stratégie de $\mathcal{S}$, nous n'avons pas appliqué GAA pour les 12 valeurs de k mais seulement pour $k = \max(K_2)$. Au lieu des $12 \times 42 \times 7 = 3528$ exécutions de GAA à réaliser (12 valeurs de k, 42 configurations de table, 7 stratégies), nous avons exécuté $42 \times 7 = 294$ fois GAA au total.

Les résultats obtenus pour ce second type d'expérimentations seront analysés dans la section 4.4.2.2 page 75.

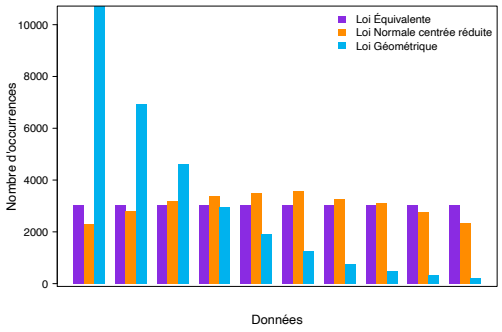
Mesures de qualité Pour juger de la qualité des tables k-anonymes produites avec GAA, nous allons utiliser trois mesures : l'altération pour $NLLM$ (cf. définition 3.1.6 page 31), la valeur de l-diversité (cf. définition 4.1.7 page 59) et la valeur de t-proximité (cf. définition 4.1.11 page 61).

L'altération pour $NLLM$, que nous appellerons également « altération », est une mesure à minimiser. Elle s'exprime en pourcentage. Une version anonyme d'une table ayant une altération de 0% n'a subi aucune modification de ses valeurs par rapport à la table d'origine. Inversement, une version anonyme d'une table ayant une altération de 100% ne contient plus aucune des informations présentes dans la table d'origine.

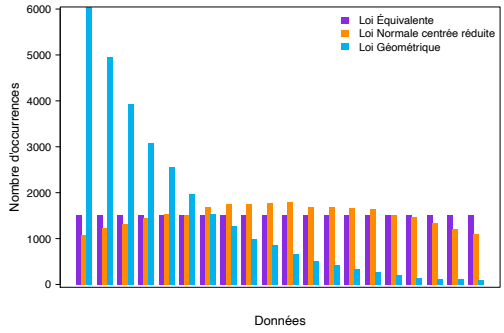
Comme l'altération pour $NLLM$, la valeur de t-proximité est une mesure à minimiser. Elle est comprise entre 0 et 1. Une version anonyme d'une table ayant une valeur de t-proximité de 0 a une répartition des valeurs de l'attribut sensible identique à celle dans la table d'origine. Une version anonyme d'une table ayant une valeur de t-proximité de 1 a une répartition des valeurs de l'attribut sensible très éloignée de celle dans la table d'origine.



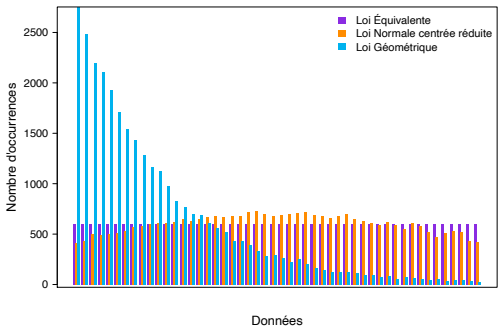
(a) Répartition des données pour 5 valeurs possibles



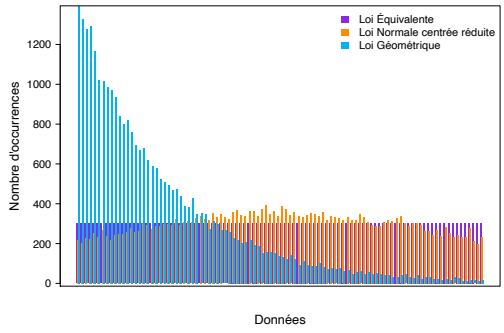
(b) Répartition des données pour 10 valeurs possibles



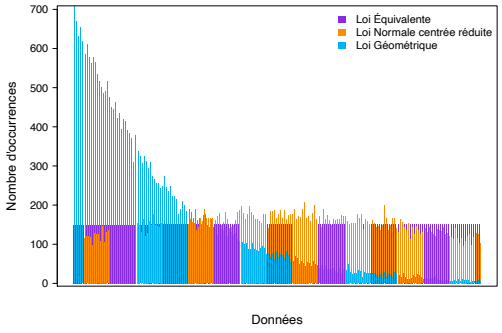
(c) Répartition des données pour 20 valeurs possibles



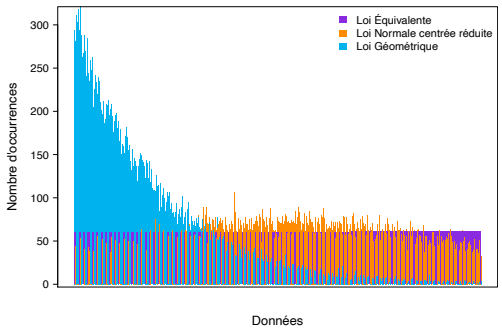
(d) Répartition des données pour 50 valeurs possibles



(e) Répartition des données pour 100 valeurs possibles



(f) Répartition des données pour 200 valeurs possibles



(g) Répartition des données pour 500 valeurs possibles

FIGURE 4.2 – Répartition des données pour les trois lois de probabilité et tous les nombres de valeurs possibles

Contrairement aux deux autres mesures considérées, la valeur de l-diversité est une mesure à maximiser. La valeur de l-diversité maximale dépend de la configuration de table considérée. Soient $\mathcal{A}$ un ensemble d'attributs contenant un attribut sensible S, $\mathcal{H}$ un vecteur de hiérarchies des attributs quasi-identifiants de $\mathcal{A}$ et T une table sur $(\mathcal{A}, \mathcal{H})$. Soit T_S une configuration de table de T. Soit T_S^* la table généralisée sur $(T_S, \mathcal{H})$ dans laquelle toutes les valeurs quasi-identifiantes sont généralisées au plus haut niveau (cf. définition 2.3.5 page 20). Pour toute table généralisée T_S^{gen} sur $(T_S, \mathcal{H})$, la valeur de l-diversité maximale que peut atteindre T_S^{gen} est $l_{div}(T_S^*)$. Une version anonyme d'une table ayant une valeur de l-diversité maximale a une diversité optimale des valeurs de l'attribut sensible dans ses classes d'équivalence. Une version anonyme d'une table ayant une valeur de l-diversité de 1 a une très faible diversité des valeurs de l'attribut sensible dans ses classes d'équivalence.

4.4.2 Analyse des résultats

Dans cette section, nous allons analyser les résultats obtenus pour les deux types d'expérimentations.

4.4.2.1 Expérimentations en considérant un attribut de la table comme attribut sensible

Dans cette section, nous allons analyser les résultats des expérimentations menées sur des configurations de table dans lesquelles l'attribut sensible étudié est un attribut présent dans la table d'origine. Pour *Adult data set*, nous avons choisi les attributs *Age* et *Statut marital*. Pour *florida_30162*, nous avons choisi l'attribut *Parti politique*. Nous obtenons donc les trois configurations de table suivantes : $Adult_{Age}$, $Adult_{Statut\ marital}$ et $florida\_30162_{Parti\ politique}$.

Étude des Stratégies l_{div} et t_{prox} Dans un premier temps, nous allons étudier deux stratégies ne faisant pas partie des sept stratégies que nous proposons : une stratégie dans laquelle la valeur de l-diversité seule est à optimiser et une stratégie dans laquelle la valeur de t-proximité seule est à optimiser. Cela signifie que pour ces deux stratégies, les fusions de classes d'équivalence dans *GAA* ne sont guidées que par la valeur de l-diversité ou la valeur de t-proximité (le coût de généralisation n'est pas pris en compte). L'objectif est de justifier la présentation de stratégies mêlant à la fois les modèles de l-diversité et de t-proximité et le coût de généralisation.

Appelons Stratégie l_{div} (Stratégie t_{prox}) la stratégie optimisant seulement sur la valeur de l-diversité (la valeur de t-proximité).

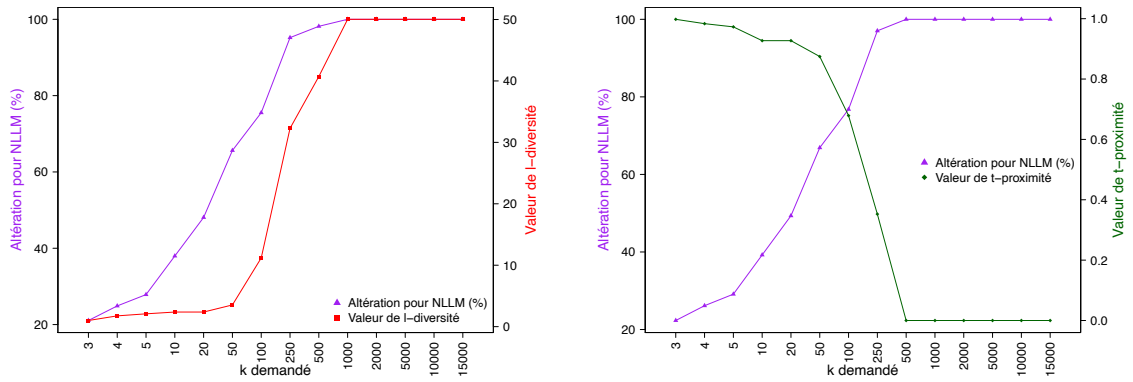
Pour chacune de ces deux stratégies, nous produisons avec l'algorithme *GAA* les versions k-anonymes des configurations de table $Adult_{Age}$, $Adult_{Statut\ marital}$ et $florida\_30162_{Parti\ politique}$ pour $k \in K_1 = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000, 10\,000, 15\,000]$.

Pour la Stratégie l_{div}, nous calculons l'altération pour *NLLM* et la valeur de l-diversité de chaque table k-anonyme produite. Les graphiques 4.3a page suivante, 4.4a page 70 et 4.5a page 70 présentent les résultats obtenus sur les trois configurations de table. Chaque graphique comporte deux courbes : la courbe violette représente l'altération pour *NLLM* de la table k-anonyme en fonction de k et la courbe rouge représente la valeur de l-diversité de la table k-anonyme en fonction de k. Rappelons que la valeur de l-diversité est une mesure à maximiser.

Afin d'analyser ces graphiques, nous calculons la valeur de l-diversité maximale pour chaque attribut sensible étudié. Cette valeur correspond au cas où tous les enregistrements de la configuration de table sont dans la même classe d'équivalence. Comme expliqué dans le protocole expérimental en section 4.4.1 page 65, pour chaque configuration de table T_S, nous calculons $l_{div}(T_S^*)$ avec la définition 4.1.7 page 59. Nous obtenons $l_{div}\left(Adult_{Age}^*\right) \simeq 50{,}03$, $l_{div}\left(Adult_{Statut\ marital}^*\right) \simeq 3{,}53$ et $l_{div}\left(florida\_30162_{Parti\ politique}^*\right) \simeq 3{,}16$.

Pour les configurations de table $Adult_{Statut\ marital}$ et $florida\_30162_{Parti\ politique}$, nous constatons sur les graphiques 4.4a page 70 et 4.5a page 70 que la valeur de l-diversité des tables k-anonymes est rapidement égale à la valeur maximale de l-diversité : pour $Adult_{Statut\ marital}$, la valeur maximale de l-diversité est atteinte à partir de la table 50-anonyme et pour $florida\_30162_{Parti\ politique}$, la valeur maximale de l-diversité est atteinte à partir de la table 10-anonyme. Pour la configuration de table $Adult_{Age}$, la courbe croît moins vite mais la valeur maximale de l-diversité est atteinte à partir de la table 500-anonyme.

Ces bons résultats de l-diversité sont contre-balancés par une altération des tables k-anonymes élevée dès les premières valeurs de k. Pour la configuration de table $Adult_{Statut\ marital}$, l'altération de la table 3-anonyme est de près de 60% et pour la configuration de table $florida\_30162_{Parti\ politique}$, l'altération de la table 3-anonyme est déjà de plus de 97%. A titre de comparaison, si nous utilisons dans *GAA* la Stratégie 1 qui guide les fusions seulement avec le coût de généralisation, la version 3-anonyme de $Adult_{Statut\ marital}$ produite avec *GAA* a une altération d'environ 2,77% et la version 3-anonyme de $florida\_30162_{Parti\ politique}$ produite avec *GAA* a une altération d'environ 1,23%.



(a) Optimisation de la l-diversité seulement

(b) Optimisation de la t-proximité seulement

FIGURE 4.3 – Optimisation sur l'attribut sensible *Age* de *Adult data set*

Étudions maintenant les résultats obtenus avec la Stratégie t_{prox}. Pour la Stratégie t_{prox}, nous calculons l'altération pour *NLLM* et la valeur de t-proximité de chaque table k-anonyme produite. Les graphiques 4.3b, 4.4b page suivante et 4.5b page suivante présentent les résultats obtenus sur les trois configurations de table. Chaque graphique comporte deux courbes : la courbe violette représente l'altération pour *NLLM* de la table k-anonyme en fonction de k et la courbe verte représente la valeur de t-proximité de la table k-anonyme en fonction de k. Rappelons que la valeur de t-proximité est comprise entre 0 et 1 et qu'elle est à minimiser dans une table.

Pour les configurations de table $Adult_{Statut\ marital}$ et $florida\_30162_{Parti\ politique}$, nous constatons sur les graphiques 4.4b page suivante et 4.5b page suivante que la valeur de t-proximité des tables k-anonymes est rapidement proche de 0 : pour $Adult_{Statut\ marital}$, la valeur de t-proximité est proche de 0 à partir de la table 50-anonyme et pour $florida\_30162_{Parti\ politique}$, la valeur de t-proximité est égale à 0 à partir de la table 10-anonyme. Pour la configuration de table $Adult_{Age}$, la courbe décroît moins vite mais la valeur de t-proximité est égale à 0 à partir de la table 500-anonyme.

En revanche, comme pour la Stratégie l_{div}, les bons résultats de t-proximité sont contre-balancés par une altération des tables k-anonymes élevée dès les premières valeurs de k. Pour la configuration de table $Adult_{Statut\ marital}$, l'altération de la table 3-anonyme est de près de 60% et pour la configuration de table $florida\_30162_{Parti\ politique}$, l'altération de la table 3-anonyme est de près de 97%.

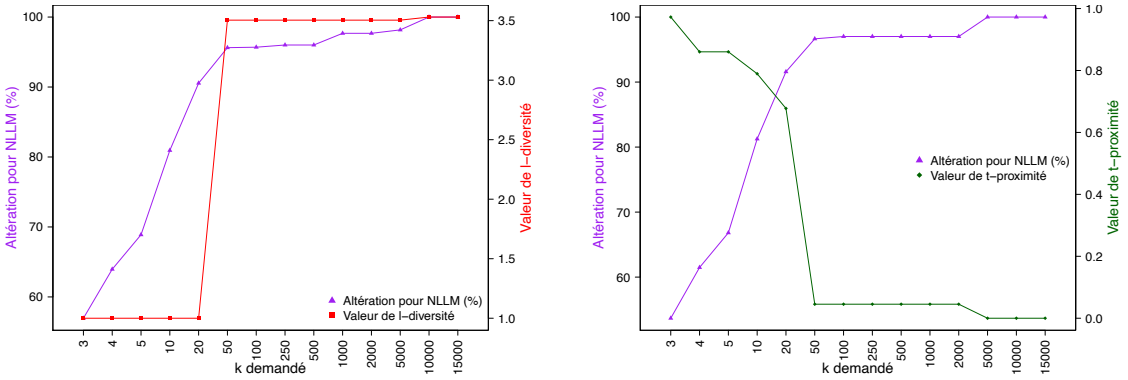
Pour conclure sur cette première expérimentation, bien que les valeurs de l-diversité et de t-proximité soient rapidement optimisées dans les tables k-anonymes quand nous utilisons les Stratégies l_{div} et t_{prox}, cela entraîne une altération des tables k-anonymes élevée dès les premières valeurs de k. Elle est de 100% lorsque les valeurs de l-diversité et de t-proximité sont optimales dans la table k-anonyme.

Ainsi, les résultats obtenus pour les Stratégies l_{div} et t_{prox} suggèrent qu'une optimisation ne tenant comptant que de la répartition des valeurs sensibles et pas du coût de généralisation ne permet pas de maintenir une altération raisonnable dans les tables k-anonymes produites. Dans le cas général, il est donc nécessaire de guider les fusions de classes d'équivalence dans *GAA* en considérant à la fois la valeur de l-diversité ou la valeur de t-proximité et le coût de généralisation.

Étude des sept stratégies proposées Étudions maintenant les performances des sept stratégies définies en section 4.3 page 62 lors d'expérimentations dans lesquelles l'attribut sensible considéré est un attribut de la table. Considérons à nouveau les trois configurations de table $Adult_{Age}$, $Adult_{Statut\ marital}$ et $florida\_30162_{Parti\ politique}$.

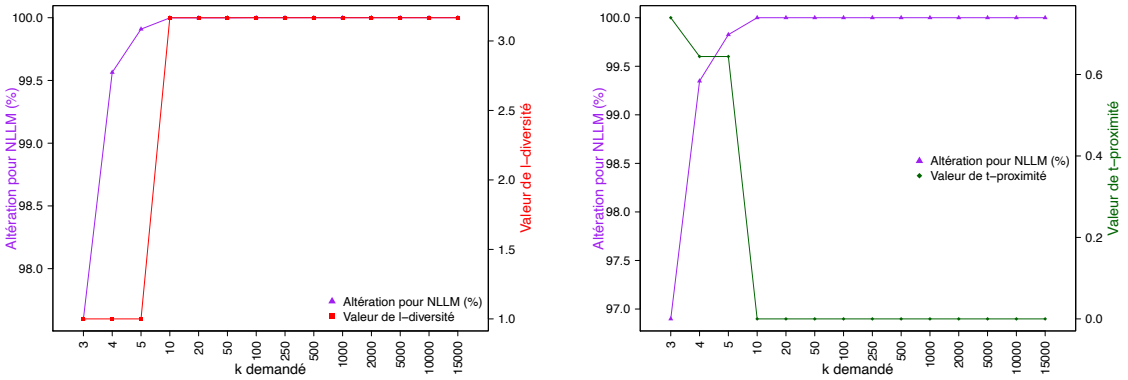
Comme expliqué dans le protocole expérimental en section 4.4.1 page 65, pour chaque configuration de table, pour chaque stratégie, nous avons appliqué l'algorithme *GAA* en utilisant la stratégie sur la configuration de table pour $k = \max(K_1)$ avec $K_1 = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000, 10\,000, 15\,000]$. Nous avons obtenu les versions k-anonymes de la configuration de table pour tous les k de K_1. Pour chaque table k-anonyme produite, nous avons calculé son altération pour *NLLM*, sa valeur de l-diversité et sa valeur de t-proximité.

Les graphiques des figures 4.6 page 71, 4.7 page 71 et 4.8 page 72 présentent les résultats obtenus pour les trois mesures sur les trois configurations de table. Chaque graphique comporte sept courbes correspondant aux sept stratégies étudiées. Un point d'une courbe représente la valeur pour la mesure correspondant au graphique de la table k-anonyme produite en utilisant la stratégie correspondant à la courbe dans l'algorithme *GAA* sur la configuration de table correspondant à la figure. Par exemple, sur le graphique 4.6b page 71, le septième point de la courbe verte est la valeur de l-diversité de la version 100-anonyme de $Adult_{Age}$ produite en utilisant la



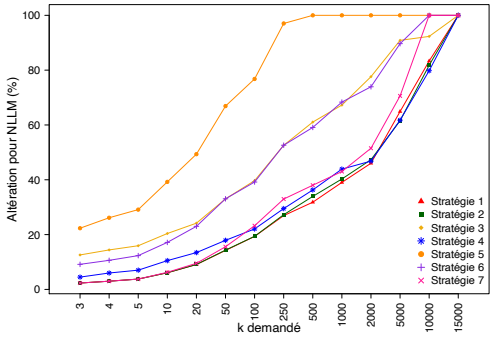
(a) Optimisation de la l-diversité seulement (b) Optimisation de la t-proximité seulement

FIGURE 4.4 – Optimisation sur l'attribut sensible *Statut marital* de *Adult data set*

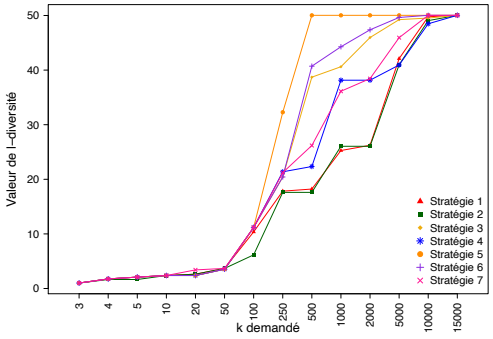


(a) Optimisation de la l-diversité seulement (b) Optimisation de la t-proximité seulement

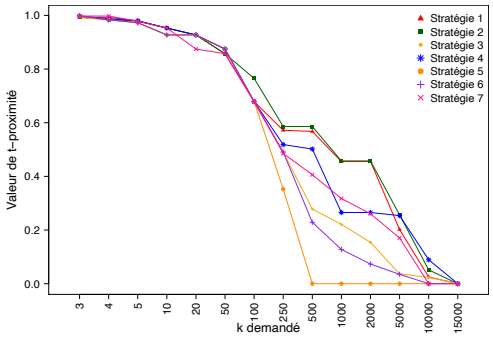
FIGURE 4.5 – Optimisation sur l'attribut sensible *Parti politique* de *florida_30162*



(a) Altération pour *NLLM*

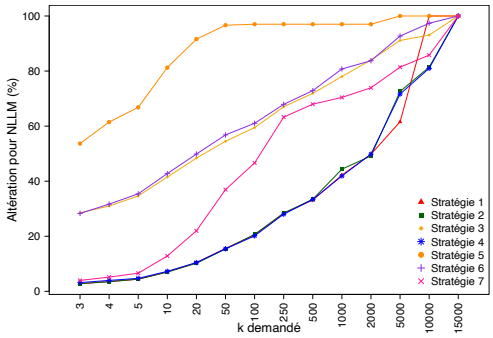


(b) Valeur de l-diversité (l_{div})

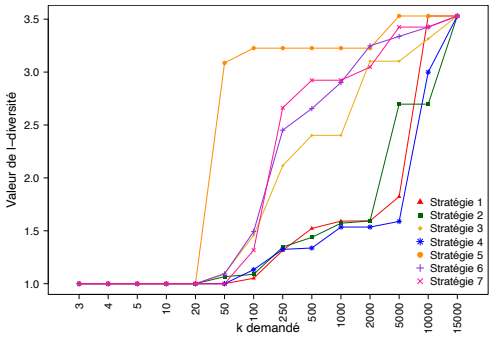


(c) Valeur de t-proximité (t_{prox})

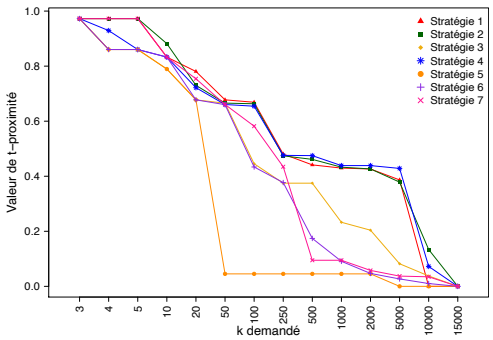
FIGURE 4.6 – Altération pour *NLLM*, valeur de l-diversité et valeur de t-proximité des tables k-anonymes produites en utilisant *Age* comme attribut sensible dans *Adult data set*



(a) Altération pour *NLLM*

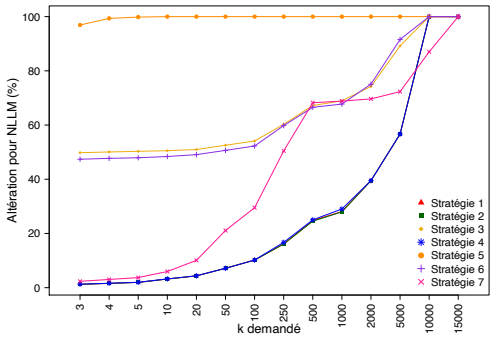


(b) Valeur de l-diversité (l_{div})

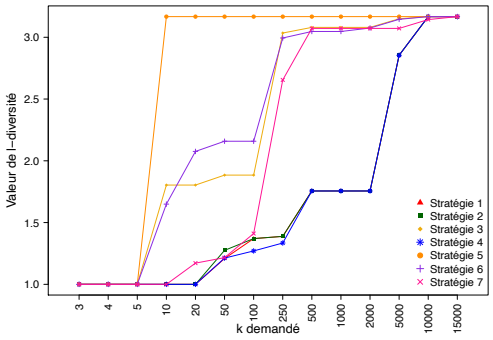


(c) Valeur de t-proximité (t_{prox})

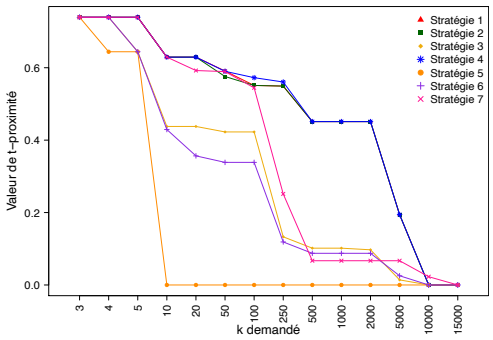
FIGURE 4.7 – Altération pour *NLLM*, valeur de l-diversité et valeur de t-proximité des tables k-anonymes produites en utilisant *Statut marital* comme attribut sensible dans *Adult data set*



(a) Altération pour *NLLM*



(b) Valeur de l-diversité (l_{div})



(c) Valeur de t-proximité (t_{prox})

FIGURE 4.8 – Altération pour *NLLM*, valeur de l-diversité et valeur de t-proximité des tables k-anonymes produites en utilisant *Parti politique* comme attribut sensible dans *florida_30162*

Stratégie 2 dans l'algorithme *GAA*. De même, sur le graphique 4.8c, le deuxième point de la courbe orange est la valeur de t-proximité de la version 4-anonyme de $florida\_30162_{Parti\ politique}$ produite en utilisant la Stratégie 5 dans l'algorithme *GAA*.

Comme dans la section 3.4 page 40 du chapitre 3 page 25, pour chaque configuration de table, pour chaque mesure, pour chaque stratégie, nous aimerions synthétiser en une unique valeur les résultats pour cette mesure obtenus par les versions k-anonymes de cette configuration de table produites avec *GAA* en utilisant cette stratégie.

Nous utilisons la notion d'aire sous la courbe qui permet de calculer la valeur moyenne d'une courbe sur un intervalle donné. En effet, pour une fonction f continue sur un intervalle $[a, b] \subset \mathbb{R}$, la valeur moyenne de f sur $[a, b]$ est $m \in \mathbb{R}$ tel que

$$m = \frac{1}{b-a} \int_a^b f(x)dx.$$

Nous utilisons la méthode des trapèzes (cf. définition 3.4.4 page 42) pour approcher l'aire sous la courbe. Dans ce chapitre, nous supposons que la méthode des trapèzes fournit une bonne approximation de l'aire sous la courbe, quelque soit la mesure considérée.

Pour chaque configuration de table, pour chaque stratégie de $\mathcal{S}$, pour chaque mesure, nous calculons donc une valeur moyenne des résultats pour cette mesure obtenus par les versions k-anonymes de cette configuration de table produites avec *GAA* en utilisant cette stratégie. Par exemple, nous avons produit 14 versions k-anonymes de $Adult_{Age}$ avec *GAA* en utilisant la Stratégie 1 pour $k \in K_1$. Pour chaque table k-anonyme, nous calculons son altération pour *NLLM*. Nous obtenons donc 14 valeurs d'altération (cf. courbe rouge du graphique 4.6a page précédente). Nous calculons ensuite la valeur moyenne de cette courbe sur $[3, 15\,000]$ grâce à la méthode précédente : les versions k-anonymes de $Adult_{Age}$ produites avec *GAA* en utilisant la Stratégie 1 ont une altération de 71,1894% en moyenne. Ce résultat est à retrouver dans le tableau 4.1a page ci-contre.

Cependant, en calculant une valeur moyenne des résultats pour une mesure, il n'est pas simple de comparer les résultats obtenus par les stratégies sur deux configurations de table différentes. Par exemple, sur $Adult_{Age}$, les tables k-anonymes produites avec *GAA* en utilisant la Stratégie 1 ont une valeur de l-diversité d'environ 41,7 en moyenne (cf. courbe rouge du graphique 4.6b page précédente). Sur $Adult_{Statut\ marital}$, les tables k-anonymes

Stratégie	*VMN*
Stratégie 4	69,7056
Stratégie 2	70,0732
Stratégie 1	71,1894
Stratégie 7	79,4616
Stratégie 3	88,0224
Stratégie 6	89,7723
Stratégie 5	99,5977

(a) Avec *Age* comme attribut sensible dans *Adult data set*

Stratégie	*VMN*
Stratégie 4	72,9193
Stratégie 2	73,3716
Stratégie 1	76,5846
Stratégie 7	83,3547
Stratégie 3	90,4344
Stratégie 6	92,4509
Stratégie 5	99,2764

(b) Avec *Statut marital* comme attribut sensible dans *Adult data set*

Stratégie	*VMN*
Stratégie 2	72,712
Stratégie 1	72,725
Stratégie 4	72,7857
Stratégie 7	80,3444
Stratégie 3	90,238
Stratégie 6	90,8862
Stratégie 5	99,9998

(c) Avec *Parti politique* comme attribut sensible dans *florida_30162*

TABLEAU 4.1 – Valeurs moyennes normalisées des sept stratégies pour l'altération pour *NLLM* calculées sur l'intervalle [3,15 000] pour les expérimentations sur des attributs sensibles des tables

Stratégie	*VMN*
Stratégie 5	98,5568
Stratégie 6	96,2862
Stratégie 3	94,8953
Stratégie 7	90,4144
Stratégie 4	86,6467
Stratégie 1	83,3517
Stratégie 2	82,2158

(a) Avec *Age* comme attribut sensible dans *Adult data set*

Stratégie	*VMN*
Stratégie 5	97,8471
Stratégie 7	94,0126
Stratégie 6	93,8467
Stratégie 3	89,4582
Stratégie 1	73,9677
Stratégie 2	72,6381
Stratégie 4	66,773

(b) Avec *Statut marital* comme attribut sensible dans *Adult data set*

Stratégie	*VMN*
Stratégie 5	99,9791
Stratégie 3	98,7433
Stratégie 6	98,6807
Stratégie 7	97,426
Stratégie 2	86,6461
Stratégie 1	86,6405
Stratégie 4	86,5971

(c) Avec *Parti politique* comme attribut sensible dans *florida_30162*

TABLEAU 4.2 – Valeurs moyennes normalisées des sept stratégies pour la valeur de l-diversité calculées sur l'intervalle [3,15 000] pour les expérimentations sur des attributs sensibles des tables

produites avec *GAA* en utilisant la Stratégie 1 ont une valeur de l-diversité d'environ 2,611 en moyenne (cf. courbe rouge du graphique 4.7b page 71). Cet écart dans les résultats ne signifie pas que la Stratégie 1 produit des versions k-anonymes de $Adult_{Age}$ de bien meilleure qualité en termes de valeur de l-diversité que les versions k-anonymes de $Adult_{Statut\ marital}$. Il est nécessaire de prendre en compte la valeur de l-diversité maximale de chacune des configurations de table.

Au lieu d'une valeur moyenne, nous souhaitons donc associer à chaque stratégie un pourcentage pour chaque mesure et pour chaque configuration de table. Pour chaque configuration de table, pour chaque stratégie, pour chacune des trois mesures, le pourcentage obtenu est appelé *VMN*, pour *Valeur Moyenne Normalisée*, de la stratégie pour la mesure sur la configuration de table.

L'altération pour *NLLM* s'exprime déjà en pourcentage. Ainsi, pour chaque configuration de table, pour chaque stratégie de $\mathcal{S}$, la *VMN* de la stratégie pour l'altération est la valeur moyenne calculée avec la méthode des trapèzes des altérations des versions k-anonymes de la configuration de table produites avec *GAA* en utilisant la stratégie.

Pour la valeur de t-proximité, nous multiplions par 100 la valeur moyenne pour obtenir un pourcentage. Par exemple, les versions k-anonymes de $florida\_30162_{Parti\ politique}$ produites avec *GAA* en utilisant la Stratégie 2 ont une valeur de t-proximité d'environ 0,159 786 en moyenne (cf. courbe verte du graphique 4.8c page précédente). La *VMN* de la Stratégie 2 pour la valeur de t-proximité sur $florida\_30162_{Parti\ politique}$ est donc de $0{,}159\,786 \times 100 = 15{,}9786\%$.

Pour la valeur de l-diversité, pour exprimer la valeur moyenne obtenue sous la forme d'un pourcentage, nous prenons le pourcentage de la valeur moyenne sur la valeur de l-diversité maximale. Par exemple, les versions k-anonymes de $Adult_{Statut\ marital}$ produites avec *GAA* en utilisant la Stratégie 1 ont une valeur de l-diversité d'environ 2,611 en moyenne (cf. courbe rouge du graphique 4.7b page 71). Sachant que la valeur de l-diversité maximale pour $Adult_{Statut\ marital}$ est de 3,53, la *VMN* de la Stratégie 1 pour la valeur de l-diversité sur $Adult_{Statut\ marital}$ est de $\frac{\text{valeur moyenne}}{\text{valeur maximale}} \times 100 = \frac{2{,}611}{3{,}53} \times 100 \simeq 73{,}97\%$.

Les tableaux 4.1 à 4.3 page suivante présentent les *VMN* calculées sur $[3, 15\,000]$ des sept stratégies pour les trois mesures et les trois configurations de table.

Étudions maintenant les résultats obtenus pour chaque mesure.

Pour l'altération pour *NLLM* (cf. tableau 4.1), nous observons des résultats similaires pour les trois confi-

Stratégie	*VMN*
Stratégie 5	1,3554
Stratégie 6	4,6532
Stratégie 3	7,1544
Stratégie 7	12,1441
Stratégie 4	17,4389
Stratégie 1	17,6274
Stratégie 2	20,1329

(a) Avec *Age* comme attribut sensible dans *Adult data set*

Stratégie	*VMN*
Stratégie 5	1,2027
Stratégie 6	3,6995
Stratégie 7	4,9465
Stratégie 3	9,3964
Stratégie 1	20,6913
Stratégie 4	24,4826
Stratégie 2	24,9918

(b) Avec *Statut marital* comme attribut sensible dans *Adult data set*

Stratégie	*VMN*
Stratégie 5	0,0196
Stratégie 6	3,0649
Stratégie 3	3,1047
Stratégie 7	4,9289
Stratégie 2	15,9786
Stratégie 1	15,9826
Stratégie 4	16,0118

(c) Avec *Parti politique* comme attribut sensible dans *florida_30162*

TABLEAU 4.3 – Valeurs moyennes normalisées des sept stratégies pour la valeur de t-proximité calculées sur l'intervalle [3,15 000] pour les expérimentations sur des attributs sensibles des tables

gurations de table. Les Stratégies 1 (optimisation sur le coût de généralisation), 2 (optimisation sur le coût de généralisation puis sur la valeur de l-diversité) et 4 (optimisation sur le coût de généralisation divisé par la valeur de l-diversité) ont les meilleures *VMN* pour l'altération ; les Stratégies 3 (optimisation sur la valeur de l-diversité puis sur le coût de généralisation), 5 (optimisation sur le coût de généralisation puis sur la valeur de t-proximité) et 6 (optimisation sur la valeur de t-proximité puis sur le coût de généralisation) ont les moins bonnes *VMN* pour l'altération ; la Stratégie 7 (optimisation sur le coût de généralisation multiplié par la valeur de t-proximité) a une *VMN* pour l'altération intermédiaire. Par exemple, sur la configuartion de table $\textit{florida\_30162}_{\textit{Parti politique}}$, nous observons dans le tableau 4.1c page précédente que les Stratégies 1, 2 et 4 ont une *VMN* pour l'altération d'environ 72%, que les Stratégies 3, 5 et 6 ont une *VMN* pour l'altération supérieure à 90% et que la Stratégie 7 a une *VMN* pour l'altération d'environ 80%.

Nous concluons de ces observations que les tables k-anonymes produites en utilisant les Stratégies 3, 5 et 6 sont plus altérées en moyenne que les tables k-anonymes produites en utilisant les Stratégies 1, 2 et 4. Ce constat est confirmé par les allures des courbes sur les graphiques 4.6a page 71, 4.7a page 71 et 4.8a page 72.

Il est à noter que la Stratégie 5 a une *VMN* pour l'altération de plus de 99% pour les trois configurations de table. Cette stratégie ne semble pas adaptée pour maintenir une altération raisonnable dans les tables k-anonymes.

Pour la valeur de l-diversité (cf. tableau 4.2 page précédente) et la valeur de t-proximité (cf. tableau 4.3), nous constatons également des résultats similaires sur les trois configurations de table. Rappelons que la valeur de l-diversité est à maximiser et que la valeur de t-proximité est à minimiser.

La Stratégie 5 a de bien meilleures *VMN* pour les valeurs de l-diversité et de t-proximité que les autres stratégies. Par exemple, sur la configuration de table $\textit{florida\_30162}_{\textit{Parti politique}}$, la *VMN* de la Stratégie 5 pour la valeur de l-diversité est de 99,98% et sa *VMN* pour la valeur de t-proximité est d'environ 0,02%. Cela signifie que les tables k-anonymes produites en utilisant la Stratégie 5 ont presque toutes les valeurs de l-diversité et de t-proximité optimales. Ce constat est confirmé par l'allure des courbes des graphiques 4.8b page 72 et 4.8c page 72 : nous observons que les valeurs optimales de l-diversité et de t-proximité sont atteintes à partir de la table 10-anonyme.

Les Stratégies 1, 2 et 4 ont les moins bonnes *VMN* pour les valeurs de l-diversité et de t-proximité sur les trois configurations de table. Toutefois, leur *VMN* pour la valeur de l-diversité sont supérieures à 66% et leur *VMN* pour la valeur de t-proximité sont inférieures à 25%.

Nous avons ensuite comparé les performances des stratégies en les regroupant selon des caractéristiques de leur définition. Les caractéristiques sont les suivantes :

- la stratégie tient compte de la valeur de l-diversité : Stratégies 2, 3 et 4
- la stratégie tient compte de la valeur de t-proximité : Stratégies 5, 6 et 7
- la stratégie optimise selon le coût de généralisation puis la valeur de l-diversité ou la valeur de t-proximité : Stratégies 2 et 5
- la stratégie optimise selon la valeur de l-diversité ou la valeur de t-proximité puis le coût de généralisation : Stratégies 3 et 5
- la stratégie optimise selon deux mesures à la fois : Stratégies 4 et 7

Cependant, excepté pour les Stratégies 3 et 6 qui ont des résultats proches pour les trois mesures et pour les trois configurations de table, nous n'avons pas observé de corrélations entre les résultats des stratégies regroupées selon les caractéristiques précédentes.

Conclusion sur les premières expérimentations Dans cette section, nous avons mené des expérimentations sur des configurations de table dans lesquelles l'attribut sensible étudié est présent dans la table d'origine.

Dans un premier temps, nous avons étudié deux stratégies ne tenant compte que de la valeur de l-diversité (Stratégie l_{div}) ou de la valeur de t-proximité (Stratégie t_{prox}). Les résultats obtenus sur trois configurations de table ont montré que les tables k-anonymes produites en utilisant ces deux stratégies dans *GAA* ont des valeurs de l-diversité et de t-proximité optimales pour une grande partie des valeurs de k. En revanche, l'altération des tables k-anonymes est élevée même pour les petites valeurs de k. Si les données étaient publiées ainsi, elles n'auraient aucune utilité. L'introduction de stratégies mêlant valeur de l-diversité ou valeur de t-proximité et coût de généralisation est donc justifiée.

Dans un second temps, nous avons évalué les sept stratégies présentées en section 4.3 page 62 sur trois configurations de table. Les résultats obtenus, similaires pour les trois configurations de table, montrent que les Stratégies 1 (optimisation sur le coût de généralisation), 2 (optimisation sur le coût de généralisation puis sur la valeur de l-diversité) et 4 (optimisation sur le coût de généralisation divisé par la valeur de l-diversité) sont les meilleures stratégies pour limiter l'altération dans les tables k-anonymes produites. En revanche, ces stratégies ont les moins bons résultats pour les valeurs de l-diversité et de t-proximité. La Stratégie 5 (optimisation sur le coût de généralisation puis sur la valeur de t-proximité) est équivalente aux Stratégies l_{div} et t_{prox} : les valeurs de l-diversité et de t-proximité sont rapidement optimisées, au détriment de l'altération très élevée dès les premières valeurs de k. Les Stratégies 3 (optimisation sur la valeur de l-diversité puis sur le coût de généralisation) et 6 (optimisation sur la valeur de t-proximité puis sur le coût de généralisation) ne parviennent pas à limiter l'altération dans les tables k-anonymes mais maintiennent de bons niveaux de l-diversité et de t-proximité dans les tables k-anonymes. La Stratégie 7 (optimisation sur le coût de généralisation multiplié par la valeur de t-proximité) ne présente pas davantage notable par rapport aux autres stratégies : elle obtient des résultats intermédiaires pour les trois mesures.

4.4.2.2 Expérimentations en considérant un attribut généré ajouté à la table comme attribut sensible

Dans cette section, nous allons analyser les résultats des expérimentations menées sur des configurations de table dans lesquelles l'attribut sensible étudié a été ajouté à la table d'origine. L'objectif de ces expérimentations sur des données simulées est de couvrir différents types d'attributs sensibles. Nous pouvons contrôler le nombre de valeurs possibles de l'attribut sensible ainsi que la répartition des données dans l'ensemble de données généré.

Nous avons créé sept attributs contenant 5, 10, 20, 50, 100, 200 ou 500 valeurs possibles. Pour chaque attribut créé, nous avons ensuite généré un ensemble de 30 162 valeurs dans lesquels la répartition des valeurs suit la loi Équivalente, la loi Géométrique ou la loi Normale centrée réduite. Ces ensembles de valeurs sont finalement ajoutés comme nouvelle colonne dans les tables *Adult data set* et *florida_30162*.

Notons $A_{n,\mathrm{L}}$ la colonne générée à partir de l'attribut créé à n valeurs possibles et dans laquelle la répartition des valeurs suit la loi L. Abrégeons les noms des lois de probabilité en Équiv pour Équivalente, Géom pour Géométrique et Norm pour Normale centrée réduite.

Pour chaque table $T \in \{$*Adult data set*, *florida_30162*$\}$, pour chaque colonne générée $A_{n,\mathrm{L}}$ avec $n \in \{5, 10, 20, 50, 100, 200, 500\}$ et L$\in$ {Équiv, Géom, Norm}, nous considérons la configuration de table $T_{A_{n,\mathrm{L}}}$ dans laquelle les attributs de la table d'origine sont des quasi-identifiants et l'attribut sensible étudié est $A_{n,\mathrm{L}}$, les autres colonnes ajoutées à la table T n'étant pas prises en compte.

Par exemple, la configuration de table $Adult_{A_{20,\mathrm{Géom}}}$ est constituée de neuf attributs quasi-identifiants correspondant aux attributs de la table *Adult data set* d'origine (à savoir *Age*, *Genre*, *Race*, *Statut marital*, *Éducation*, *Pays de naissance*, *Catégorie professionnelle*, *Occupation*, *Salaire*) et d'un attribut sensible $A_{20,\mathrm{Géom}}$ correspondant à l'ensemble de valeurs généré à partir de l'attribut créé à 20 valeurs possibles et dont la répartition des valeurs suit une loi Géométrique. De même, la configuration de table $florida\_30162_{A_{5,\mathrm{Norm}}}$ est constituée de cinq attributs quasi-identifiants correspondant aux attributs de la table *florida_30162* d'origine (à savoir *Code postal*, *Genre*, *Race*, *Année de naissance* et *Parti politique*) et d'un attribut sensible $A_{5,\mathrm{Norm}}$ correspondant à l'ensemble de valeurs généré à partir de l'attribut créé à 5 valeurs possibles et dont la répartition des valeurs suit la loi Normale centrée réduite.

Nos expérimentations portent sur :

$$\begin{aligned}\text{nombre de tables} \times \text{nombre d'attributs créés} \times \text{nombre de loi de probabilité} &= 2 \times 7 \times 3 \\ &= 42\end{aligned}$$

configurations de table.

Étudions maintenant les performances des sept stratégies présentées en section 4.3 page 62 lorsqu'elles sont

utilisées dans l'algorithme *GAA* sur ces 42 configurations de table.

Comme expliqué dans le protocole expérimental en section 4.4.1 page 65, pour chaque configuration de table, pour chaque stratégie, nous avons appliqué l'algorithme *GAA* en utilisant la stratégie sur la configuration de table pour $k = \max(K_2)$ avec $K_2 = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000]$. Nous avons obtenu les versions k-anonymes de la configuration de table pour tous les k de K_2. Pour chaque table k-anonyme produite, nous avons calculé son altération pour *NLLM*, sa valeur de l-diversité et sa valeur de t-proximité.

Comme le nombre de configurations de table est grand, nous ne présentons pas de graphique pour chaque mesure comme cela a été fait dans la section 4.4.2.1 page 68 (le nombre de graphiques serait de nombre de configurations de table × nombre de mesures = $42 \times 3 = 126$).

Pour chaque configuration de table, pour chaque mesure, pour chaque stratégie, nous avons calculé la *VMN* sur $[3, 5000]$ de la stratégie pour la mesure sur la configuration de table (la méthode de calcul des *VMN* est à lire en section 4.4.2.1 page 68). Nous obtenons : nombre de configurations de table × nombre de mesures × nombre de stratégies = $42 \times 3 \times 7 = 882$ valeurs de *VMN*.

Analyse des classements des stratégies Afin d'étudier les valeurs de *VMN* obtenues, pour chaque configuration de table, pour chaque mesure, nous avons réalisé un classement des stratégies selon leur *VMN* pour la mesure sur la configuration de table. Rappelons que la *VMN* pour la valeur de l-diversité est à maximiser et que les *VMN* pour l'altération et la valeur de t-proximité sont à minimiser. La stratégie ayant obtenu la meilleure *VMN* pour la mesure pour la configuration de table est au rang 1 et la couleur qui lui est associée est le vert. La stratégie ayant obtenu la moins bonne *VMN* pour la mesure sur la configuration de table est au rang 7 et la couleur qui lui est associée est le rouge.

Les tableaux de la figure 4.9 page suivante résument les classements obtenus pour chaque mesure sur chaque configuration de table pour chaque table : tableau 4.9a page ci-contre pour *Adult data set* et tableau 4.9b page suivante pour *florida_30162*. Les lignes de chaque tableau correspondent aux trois mesures pour les sept stratégies et les colonnes de chaque tableau correspondent aux nombres de valeurs possibles des sept attributs créés pour les trois lois de probabilité.

Par exemple, dans le tableau 4.9a page ci-contre en colonne (Loi Équivalente, 5), nous lisons que la Stratégie 1 est au rang 1 du classement pour l'altération pour *NLLM* sur $Adult_{A_{5,\text{Équiv}}}$ (case en ligne (Stratégie 1, Altération pour *NLLM*)). Le reste du classement est le suivant : Stratégie 4 ; Stratégie 2 ; Stratégie 6 ; Stratégie 3 ; Stratégie 7 ; Stratégie 5.

Étudions dans un premier temps les résultats obtenus sur *Adult data set* et *florida_30162* séparemment.

Pour *Adult data set*, dans le tableau 4.9a page suivante, nous observons que les Stratégies 1 (optimisation sur le coût de généralisation), 2 (optimisation sur le le coût de généralisation puis sur la valeur de l-diversité) et 4 (optimisation sur le coût de généralisation divisé par la valeur de l-diversité) sont classées aux premiers rangs pour l'altération sur une majorité des configurations de table de *Adult data set* (lignes (Stratégie i, Altération pour *NLLM*) pour $i \in \{1, 2, 4\}$). En revanche, elles sont moins bien classées que les autres stratégies pour les valeurs de l-diversité et de t-proximité sur la majorité des configurations de table : nous voyons qu'elles sont souvent classées aux rangs 5, 6 et 7 pour ces deux mesures.

Les Stratégies 3 (optimisation sur la valeur de l-diversité puis sur le coût de généralisation) et 6 (optimisation sur la valeur de t-proximité puis sur le coût de généralisation) sont classées dans les dernières dans les classements d'altération sur les configurations de table de *Adult data set*. En revanche, elles sont bien classées pour les valeurs de l-diversité et de t-proximité (lignes (Stratégie 3, l-diversité), (Stratégie 3, t-proximité), (Stratégie 6, l-diversité) et (Stratégie 3, t-proximité) du tableau 4.9a page ci-contre).

La Stratégie 5 (optimisation sur le coût de généralisation puis sur la valeur de t-proximité) est classée dernière pour les classements d'altération sur toutes les configurations de table de *Adult data set* (ligne (Stratégie 5, Altération pour *NLLM*)). En revanche, elle est classée première pour les classements de valeurs de l-diversité et de t-proximité sur toutes les configurations de table de *Adult data set* (lignes (Stratégie 5, l-diversité) et (Stratégie 5, t-proximité)).

Toujours dans le tableau 4.9a page suivante, nous observons que la Stratégie 7 (optimisation sur le coût de généralisation multiplié par la valeur de t-proximité) a des résultats intermédiaires mais hétérogènes pour les trois mesures sur les configurations de table de *Adult data set*. Elle est souvent classée au 4$^{\text{e}}$ rang (dans 30 classements sur 63) mais, pour le reste des classements, elle est autant classée dans les premières stratégies que dans les dernières (17 rangs 2 ou 3 et 16 rangs 5, 6 ou 7).

Pour *florida_30162*, dans le tableau 4.9b page ci-contre, nous observons que les Stratégies 2, 4 et 7 sont les stratégies les mieux classées pour l'altération sur une majorité des configurations de table de *florida_30162* (lignes (Stratégie i, Altération pour *NLLM*) pour $i \in \{2, 4, 7\}$). Les Stratégies 2 et 4 sont les moins bien classées pour les valeurs de l-diversité et de t-proximité sur une majorité des configurations de table de *florida_30162*.

		Loi Équivalente							Loi Géométrique							Loi Normale centrée réduite							Moyennes
		5	10	20	50	100	200	500	5	10	20	50	100	200	500	5	10	20	50	100	200	500	
Stratégie 1	Altération pour *NLLM*	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	1	1	1	1	1	1	51,974
	l-diversité	6	6	6	7	4	7	6	6	6	4	5	3	4	6	6	7	7	7	5	4	6	93,829
	Valeur de *t*-proximité	6	7	6	6	4	7	4	7	6	7	4	7	7	6	7	7	7	7	6	6	6	9,628
Stratégie 2	Altération pour *NLLM*	3	2	2	3	3	2	3	2	3	3	4	3	3	1	1	2	3	4	3	3	2	55,282
	l-diversité	7	7	5	6	7	5	5	5	5	5	7	7	6	5	5	6	6	6	7	6	7	93,775
	Valeur de *t*-proximité	7	6	5	7	6	5	6	6	5	6	6	5	5	5	6	6	5	5	7	7	7	9,619
Stratégie 3	Altération pour *NLLM*	5	5	5	5	5	6	5	6	6	6	6	6	6	6	6	5	6	6	5	5	5	77,286
	l-diversité	4	2	2	2	3	2	4	4	2	2	2	2	2	3	3	2	2	2	6	2	4	94,86
	Valeur de *t*-proximité	3	2	2	2	3	2	5	3	3	2	2	2	2	2	3	3	3	2	3	2	4	8,031
Stratégie 4	Altération pour *NLLM*	2	3	3	2	2	3	6	3	2	2	2	4	4	4	3	3	2	2	2	2	4	56,619
	l-diversité	5	5	4	5	5	4	2	7	7	7	6	4	3	4	7	5	5	5	4	3	2	94,189
	Valeur de *t*-proximité	5	5	7	5	7	4	2	5	7	5	7	4	3	7	5	5	6	6	4	3	3	9,191
Stratégie 5	Altération pour *NLLM*	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	99,938
	l-diversité	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	99,528
	Valeur de *t*-proximité	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0,455
Stratégie 6	Altération pour *NLLM*	4	4	6	6	6	5	4	5	5	5	5	5	5	5	5	6	5	5	6	6	6	75,462
	l-diversité	3	3	7	3	2	3	3	2	3	6	3	6	7	7	2	3	3	3	2	7	3	94,186
	Valeur de *t*-proximité	4	3	4	3	2	3	3	2	2	3	3	3	6	4	2	2	2	3	5	4	2	8,29
Stratégie 7	Altération pour *NLLM*	6	6	4	4	4	4	2	4	4	4	3	2	2	3	4	4	4	3	4	4	3	63,345
	l-diversité	2	4	3	4	6	6	7	3	4	3	4	5	5	2	4	4	4	4	3	5	5	94,168
	Valeur de *t*-proximité	2	4	3	4	5	6	7	4	4	4	5	6	4	3	4	4	4	4	2	5	5	8,887

(a) Pour *Adult data set*

		Loi Équivalente							Loi Géométrique							Loi Normale centrée réduite							Moyennes
		5	10	20	50	100	200	500	5	10	20	50	100	200	500	5	10	20	50	100	200	500	
Stratégie 1	Altération pour *NLLM*	3	3	4	4	4	5	6	4	4	4	4	4	5	5	4	4	4	4	4	4	6	61,146
	l-diversité	6	6	6	4	5	3	4	6	5	3	4	5	4	2	6	5	6	3	4	3	2	94,317
	Valeur de *t*-proximité	6	4	4	4	2	3	2	4	4	4	3	4	3	2	6	4	4	3	2	2	2	8,001
Stratégie 2	Altération pour *NLLM*	2	1	1	3	2	1	2	2	3	2	2	1	1	1	2	2	1	1	2	2	1	50,17
	l-diversité	7	7	7	7	6	7	7	5	6	6	6	4	6	6	5	6	7	7	7	7	7	93,7
	Valeur de *t*-proximité	7	7	7	7	6	7	6	7	6	5	7	5	5	7	4	6	7	7	7	7	7	9,635
Stratégie 3	Altération pour *NLLM*	6	4	5	5	5	4	5	5	6	6	6	6	6	4	6	6	6	6	5	6	3	73,326
	l-diversité	2	2	2	2	3	5	2	2	2	2	2	3	2	3	2	2	2	2	2	2	4	94,822
	Valeur de *t*-proximité	3	2	2	2	5	5	3	5	2	3	2	2	2	4	3	2	2	2	3	4	5	8,058
Stratégie 4	Altération pour *NLLM*	1	2	2	2	1	2	3	1	2	3	3	2	2	3	1	3	2	3	1	1	4	51,005
	l-diversité	5	5	5	5	7	4	5	7	4	5	5	6	5	5	7	7	5	6	5	6	6	93,897
	Valeur de *t*-proximité	5	6	6	6	7	4	5	6	5	6	5	7	6	5	7	7	6	6	5	6	6	9,349
Stratégie 5	Altération pour *NLLM*	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	99,999
	l-diversité	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	99,927
	Valeur de *t*-proximité	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0,073
Stratégie 6	Altération pour *NLLM*	5	5	6	6	6	6	4	6	5	5	5	5	4	6	5	5	5	5	6	5	5	71,669
	l-diversité	4	4	3	3	2	2	3	3	7	4	7	7	3	4	3	3	3	4	3	4	3	94,224
	Valeur de *t*-proximité	4	5	3	3	3	2	4	2	7	2	4	6	4	3	2	3	5	4	4	5	3	8,419
Stratégie 7	Altération pour *NLLM*	4	6	3	1	3	3	1	3	1	1	1	3	3	2	3	1	3	2	3	3	2	55,592
	l-diversité	3	3	4	6	4	6	6	4	3	7	3	2	7	7	4	4	4	5	6	5	5	93,976
	Valeur de *t*-proximité	2	3	5	5	4	6	7	3	3	7	6	3	7	6	5	5	3	5	6	3	4	9,031

(b) Pour *florida_30162*

FIGURE 4.9 – Classement des stratégies selon leur valeurs de *VMN* pour l'altération pour *NLLM*, la valeur de *l*-diversité et la valeur de *t*-proximité sur les configurations de table de *Adult data set* et *florida_30162*

La Stratégie 7 obtient des résultats hétérogènes pour ces deux mesures. Par exemple, elle est classée au 7^{e} rang pour les valeurs de l-diversité et de t-proximité sur la configuration de table $florida\_30162_{A_{200,\text{Géom}}}$ (cases en ligne (Stratégie 7, l-diversité) et (Stratégie 7, t-proximité) et en colonne (Loi Géométrique, 200) du tableau 4.9b page précédente). Cependant, elle est classée aux 2^{e} et 3^{e} rangs pour les valeurs de l-diversité et de t-proximité sur la configuration de table $florida\_30162_{A_{100,\text{Géom}}}$ (cases en ligne (Stratégie 7, l-diversité) et (Stratégie 7, t-proximité) et en colonne (Loi Géométrique, 100) du tableau 4.9b page précédente).

Comme sur *Adult data set*, les Stratégies 3 et 6 sont parmi les stratégies les mieux classées pour les valeurs de l-diversité et de t-proximité mais sont parmi les moins bien classées pour l'altération sur la majorité des configurations de table de *florida_30162*.

La Stratégie 5 est une nouvelle fois classée au premier rang pour les valeurs de l-diversité et de t-proximité mais est classée au 7^{e} rang pour l'altération sur toutes les configurations de table de *florida_30162*.

Comme la Stratégie 7 sur *Adult data set*, la Stratégie 1 obtient des résultats intermédiaires pour les trois mesures : elle est classée au 4^{e} rang pour 28 classements sur 63 sur les configurations de table de *florida_30162*. Il est à noter que la Stratégie 1 n'est pas au premier rang des classements d'altération sur les configurations de table de *florida_30162*. Or, avec la Stratégie 1, seul le coût de généralisation est à optimiser lors des choix des fusions de classes d'équivalence dans *GAA*. Cela peut être dû au fait que la méthode utilisée dans *GAA* est heuristique et ne garantit pas l'optimalité globale de la table k-anonyme produite. Nous verrons dans le chapitre 5 page 87 que d'autres méthodes permettent de produire de meilleures versions k-anonymes des tables.

Pour conclure sur ces premières observations, nous constatons que la Stratégie 5 a un comportement extrême : elle optimise les valeurs de l-diversité et de t-proximité au détriment de l'altération sur toutes les configurations de table de *Adult data set* et *florida_30162*. Sur les deux tables, les Stratégies 2 et 4 parviennent à limiter l'altération pour une majorité des configurations de table mais elles ne sont pas bien classées dans les classements de valeurs de l-diversité et de t-proximité. Au contraire, les Stratégies 3 et 6 favorisent les valeurs de l-diversité et de t-proximité mais les tables k-anonymes produites sont en moyenne plus altérées pour une majorité des configurations de table. Les Stratégies 1 et 7 obtiennent des résultats différents et hétérogènes sur les deux tables.

Dans un second temps, pour chaque table, pour chaque stratégie, pour chaque mesure, nous avons calculé la moyenne des *VMN* obtenues par la stratégie pour la mesure sur les configurations de table de la table. Ainsi, nous avons une valeur représentative de la mesure pour la stratégie sur l'ensemble des configurations de table.

Nous avons donc ajouté la colonne Moyennes aux tableaux 4.9a page précédente et 4.9b page précédente. Les valeurs obtenues ont été tronquées à 10^{-3}. Par exemple, dans le tableau 4.9a page précédente, nous lisons que la *VMN* moyenne de la Stratégie 1 pour l'altération pour *NLLM* est de 51,974% sur les configurations de table de *Adult data set* (case en ligne (Stratégie 1, Altération pour *NLLM*) et en colonne Moyennes). De même, dans le tableau 4.9b page précédente, nous lisons que la *VMN* moyenne de la Stratégie 5 pour la valeur de t-proximité est de 0,073% sur les configurations de table de *florida_30162* (case en ligne (Stratégie 5, t-proximité) et en colonne Moyennes).

Rappelons que, pour l'altération pour *NLLM* et la valeur de t-proximité, la *VMN* est à minimiser : plus une stratégie a une *VMN* pour une de ces deux mesures proche de 0% plus ses performances sont bonnes pour la mesure. En revanche, pour la valeur de l-diversité, la *VMN* est à maximiser : plus une stratégie a une *VMN* pour la valeur de l-diversité proche de 100% plus ses performances sont bonnes pour la valeur de l-diversité.

Nous constatons que, pour les deux tables, les *VMN* moyennes pour les valeurs de l-diversité et de t-proximité sont bonnes pour toutes les stratégies. Les *VMN* moyennes pour la valeur de l-diversité sont supérieures à 93% pour les sept stratégies : la *VMN* moyenne la plus basse pour la valeur de l-diversité est de 93,775% sur *Adult data set* et de 93,7% sur *florida_30162*, toutes deux obtenues par la Stratégie 2 (case en ligne (Stratégie 2, l-diversité) et en colonne Moyennes du tableau 4.9a page précédente et case en ligne (Stratégie 2, l-diversité) et en colonne Moyennes du tableau 4.9b page précédente). Les *VMN* moyennes pour la valeur de t-proximité sont inférieures à 10% pour les sept stratégies : la *VMN* moyenne la plus élevée pour la valeur de t-proximité est de 9,628% sur *Adult data set* et de 9,635% sur *florida_30162*, obtenues respectivement par les Stratégies 1 et 2. D'un autre côté, les *VMN* moyennes pour l'altération sont supérieures à 50% pour toutes les stratégies et pour les deux tables.

Ces observations suggèrent qu'une augmentation de la *VMN* moyenne d'une stratégie pour l'altération entraîne une optimisation des *VMN* moyennes de la stratégie pour les valeurs de l-diversité et de t-proximité. En d'autres termes, plus les tables k-anonymes produites en utilisant une stratégie dans l'algorithme *GAA* sont altérées, plus les valeurs de l-diversité et de t-proximité de ces tables k-anonymes sont optimisées. En effet, une table k-anonyme ayant une altération élevée a généralement peu de classes d'équivalence mais ces dernières sont de grandes tailles. Comme les modèles de l-diversité et de t-proximité s'intéressent aux répartitions des valeurs de l'attribut sensible dans chaque classe d'équivalence de la table, il est plus facile de respecter ces deux modèles quand les valeurs de l'attribut sensible sont nombreuses dans chaque classe d'équivalence.

Analyse des écarts à la moyenne Dans le paragraphe précédent, nous avons observé grâce aux *VMN* moyennes des stratégies sur les configurations de table que les performances des stratégies pour les valeurs de l-diversité et de t-proximité sont très proches. Ainsi, pour choisir la meilleure stratégie à utiliser sur une table, nous pourrions simplement regarder la stratégie qui a la meilleure *VMN* moyenne pour l'altération sur cette table. Par exemple, nous pouvons dire en observant le tableau 4.9a page 77 que la Stratégie 1 avec sa *VMN* moyenne de 51,974 pour l'altération est la stratégie à privilégier quand nous travaillons sur la table *Adult data set*. Dans le tableau 4.9b page 77, nous observons que la Stratégie 2 avec sa *VMN* moyenne de 50,17% est la meilleure stratégie à utiliser sur *florida_30162*.

Cependant, nous aimerions pousser l'analyse plus loin et essayer de déterminer la meilleure stratégie à utiliser selon l'attribut sensible de la configuration de table étudiée. L'objectif est de déterminer, pour un type d'attribut sensible, si une stratégie permet d'optimiser l'altération et les valeurs de l-diversité et de t-proximité mieux que les autres stratégies. Pour cela, dans un premier temps, nous allons nous intéresser aux écarts à la moyenne des *VMN* des stratégies sur une configuration de table. Dans un second temps, nous présenterons une méthode permettant de choisir la meilleure stratégie à utiliser sur une configuration de table en fonction des écarts à la moyenne calculés pour l'altération, la valeur de l-diversité et la valeur de t-proximité.

Les classements des stratégies pour une mesure sur une configuration de table ne permettent pas de connaître l'ampleur des écarts entre les *VMN* obtenues par les stratégies. Il est possible que les *VMN* des stratégies classées au premier rang et au dernier rang soient proches. Afin de représenter visuellement les écarts entre les *VMN* des stratégies, nous avons utilisé un gradient de 256 couleurs du vert au rouge. Pour chaque table, pour chaque configuration de table, pour chaque mesure, nous avons calculé la moyenne des *VMN* obtenues par les sept stratégies pour la mesure sur la configuration de table de la table. Nous associons ensuite une couleur à chaque stratégie en fonction de l'écart de sa *VMN* pour la mesure à la moyenne. Plus la *VMN* de la stratégie est éloignée positivement de la moyenne, plus la couleur associée est proche du vert. A contrario, plus la *VMN* de la stratégie est éloignée négativement de la moyenne, plus la couleur associée est proche du rouge.

Les tableaux de la figure 4.10 page suivante résument les résultats d'écart à la moyenne obtenus pour *Adult data set* (tableau 4.10a page suivante) et *florida_30162* (tableau 4.10b page suivante). Par exemple, intéressons nous aux *VMN* des sept stratégies pour l'altération pour *NLLM* sur la configuration de table $Adult_{A_{500,\ \text{Équiv}}}$ de *Adult data set* (colonne (Loi Équivalente, 500) du tableau 4.10a page suivante). Les *VMN* à 10^{-2} près obtenues par les stratégies pour l'altération sont les suivantes :

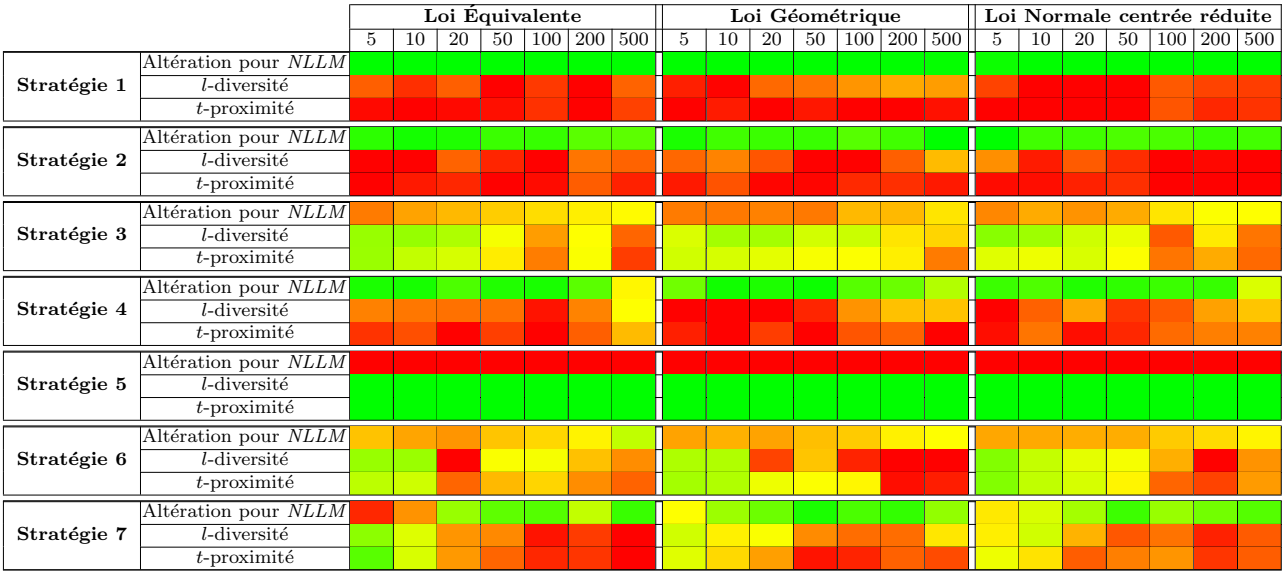
Stratégie 1 : 51,97%
Stratégie 2 : 56,88%
Stratégie 3 : 66,59%
Stratégie 4 : 67,25%
Stratégie 5 : 99,95%
Stratégie 6 : 62,19%
Stratégie 7 : 54,9%

La moyenne de ces sept valeurs est 65,68% à 10^{-2} près. Dans le tableau 4.10a page suivante, nous observons que la couleur verte a été associée aux Stratégies 1, 2 et 7 car leur *VMN* pour l'altération sont bien meilleures que la moyenne de 65,68% (case en ligne (Stratégie i, Altération pour *NLLM*) et en colonne (Loi Équivalente, 500) pour $i \in \{1, 2, 7\}$). Une couleur vert clair a été associée à la Stratégie 6 car sa *VMN* pour l'altération de 62,19% est inférieure mais proche de la moyenne. Une couleur jaune a été associée aux Stratégies 3 et 4 car leur *VMN* pour l'altération de 66,59% et 67,25% respectivement sont supérieures mais proches de la moyenne de 65,68%. Finalement, une couleur rouge a été associée à la Stratégie 5 car sa *VMN* pour l'altération de 99,95% est très supérieure à la moyenne.

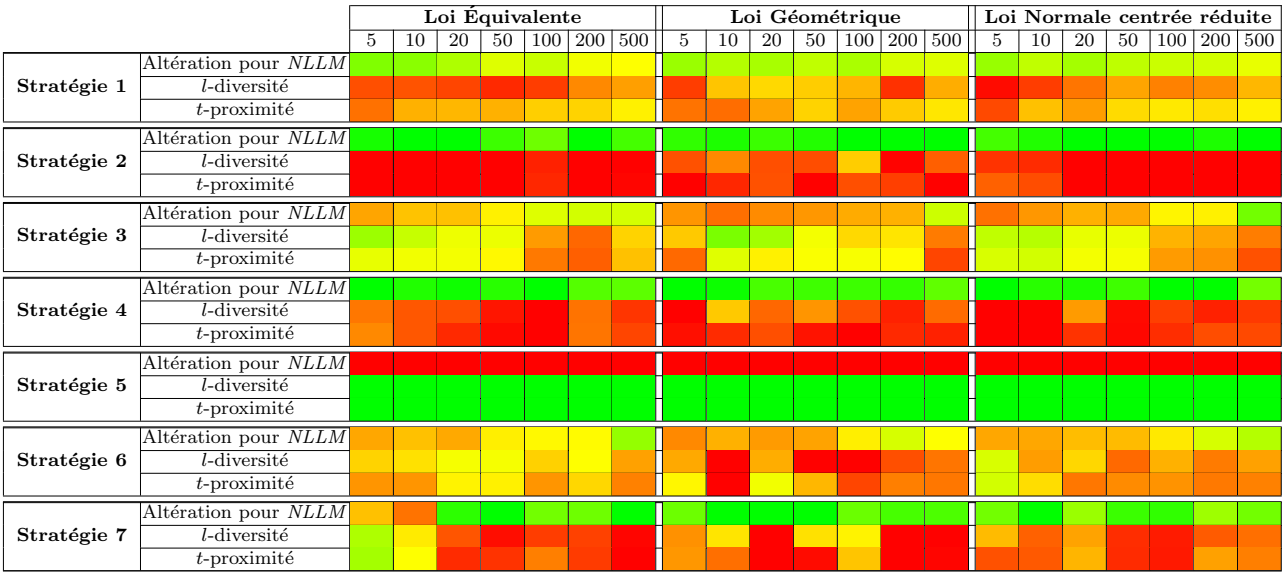
Étudions maintenant les résultats obtenus sur *Adult data set* et *florida_30162* séparemment.

Pour *Adult data set*, dans le tableau 4.10a page suivante, nous observons que les *VMN* des Stratégies 1 (optimisation sur le coût de généralisation), 2 (optimisation sur le coût de généralisation puis sur la valeur de l-diversité) et 4 (optimisation sur le coût de généralisation divisé par la valeur de l-diversité) pour l'altération sont bien meilleures que la moyenne sur une majorité des configurations de table de *Adult data set*. La couleur verte domine dans les cases des lignes (Stratégie i, Altération pour *NLLM*) pour $i \in \{1, 2, 4\}$. En revanche, les *VMN* de ces trois stratégies pour les valeurs de l-diversité et de t-proximité sont bien moins bonnes que la moyenne sur une majorité des configurations de table de *Adult data set*. La couleur rouge domine dans les cases des lignes (Stratégie i, l-diversité) et (Stratégie i, t-proximité) pour $i \in \{1, 2, 4\}$.

Les Stratégies 3 (optimisation sur la valeur de l-diversité puis sur le coût de généralisation) et 6 (optimisation sur la valeur de t-proximité puis sur le coût de généralisation) ont des *VMN* pour l'altération moins bonnes que la moyenne mais relativement proches de celle-ci sur une majorité des configurations de table de *Adult data set*. Nous observons en effet des teintes de jaune et d'orange clair dans les cases des lignes (Stratégie i, Altération pour *NLLM*) pour $i \in \{3, 6\}$ du tableau 4.10a page suivante.



(a) Pour *Adult data set*



(b) Pour *florida_30162*

FIGURE 4.10 – Écart à la moyenne des *VMN* des stratégies pour l'altération pour *NLLM*, la valeur de l-diversité et la valeur de t-proximité sur les configurations de table de *Adult data set* et *florida_30162*

La Stratégie 3 a des *VMN* pour les valeurs de l-diversité et de t-proximité meilleures que la moyenne mais proches de celle-ci sur les configurations de table dans lesquelles l'attribut sensible a jusqu'à 50 valeurs possibles quelque soit la loi de probabilité. Sur les configurations de table dans lesquelles l'attribut sensible a plus de 50 valeurs possibles, les *VMN* de la Stratégie 3 pour les valeurs de l-diversité et de t-proximité sont moins bonnes que la moyenne et plus éloignées de cette dernière.

La Stratégie 6 a des *VMN* pour les valeurs de l-diversité et de t-proximité meilleures que la moyenne mais proches de celle-ci sur les configurations de table dans lesquelles l'attribut sensible a 5 ou 10 valeurs possibles quelque soit la loi de probabilité. Pour une grande majorité des configurations de table dans lesquelles l'attribut sensible a plus de 10 valeurs possibles, les *VMN* de la Stratégie 6 pour les valeurs de l-diversité et de t-proximité sont moins bonnes et éloignées de la moyenne.

La Stratégie 5 (optimisation sur le coût de généralisation puis sur la valeur de t-proximité) a des *VMN* pour l'altération bien moins bonnes que la moyenne sur toutes les configurations de table de *Adult data set* : toutes les cases de la ligne (Stratégie 5, Altération pour *NLLM*) sont rouges. En revanche, les *VMN* de la Stratégie 5 pour les valeurs de l-diversité et de t-proximité sont bien meilleures que la moyenne pour toutes les configurations de table : les cases des lignes (Stratégie 5, l-diversité) et (Stratégie 5, t-proximité) sont vertes.

Pour la Stratégie 7 (optimisation sur le coût de généralisation multiplié par la valeur de t-proximité), nous observons dans le tableau 4.10a page précédente que cette stratégie obtient les mêmes résultats pour les trois mesures que les Stratégies 1, 2 et 4 sur les configurations de table de *Adult data set* dans lesquelles l'attribut sensible a au moins 20 valeurs possibles quelque soit la loi de probabilité. Pour les configurations de table dans lesquelles l'attribut sensible a 5 ou 10 valeurs possibles et la loi utilisée est la Loi Équivalente, la Stratégie 7 obtient les mêmes résulats pour les trois mesures que les Stratégies 3 et 6. Pour les lois Géométrique et Normale centrée réduite, les *VMN* de la Stratégie 7 pour les trois mesures sont proches de la moyenne pour les configurations de table dans lesquelles l'attribut sensible a 5 ou 10 valeurs possibles.

Pour *florida_30162*, dans le tableau 4.10b page ci-contre, nous observons que les Stratégies 2 et 4 ont des *VMN* pour l'altération bien meilleures que la moyenne et des *VMN* pour les valeurs de l-diversité et de t-proximité bien moins bonnes que la moyenne sur une grande majorité des configurations de table de *florida_30162*.

La Stratégie 6 a des *VMN* pour les trois mesures plutôt moins bonnnes que la moyenne sur toutes les configurations de table : la couleur orange domine dans les lignes correspondant à la Stratégie 6. Pour la Stratégie 3, le constat est le même bien qu'il y ait plus de cases dont la couleur tire sur le vert que de cases dont la couleur tire sur le rouge.

Pour la Stratégie 5, les résultats sont les mêmes que sur *Adult data set* : les *VMN* pour l'altération sont bien meilleures que la moyenne et les *VMN* pour les valeurs de l-diversité et de t-proximité sont bien moins bonnes que la moyenne sur toutes les configurations de table de *florida_30162*.

Pour la Stratégie 1, bien qu'elle n'ait pas été classée dans les premiers rangs des classements d'altération sur les configurations de table de *florida_30162*, nous remarquons que la couleur verte domine dans les cases de la ligne (Stratégie 1, Altération pour *NLLM*). Cela signifie que les *VMN* de la Stratégie 1 pour l'altération sont dans l'ensemble meilleures que la moyenne. En revanche, les *VMN* de la Stratégie 1 pour les valeurs de l-diversité et de t-proximité sont moins bonnes et plutôt éloignées de la moyenne : les teintes de orange et de orange foncé dominent dans les cases des lignes (Stratégie 1, l-diversité) et (Stratégie 1, t-proximité) du tableau 4.10b page précédente.

Pour la Stratégie 7, nous constatons que les *VMN* de cette stratégie pour l'altération sont bien meilleures que la moyenne pour toutes les configurations de table de *florida_30162* sauf pour les configurations de table $florida\_30162_{A_{5,\text{Équiv}}}$ et $florida\_30162_{A_{10,\text{Équiv}}}$ pour lesquelles les *VMN* de la Stratégie 7 pour l'altération sont moins bonnes que la moyenne. Les *VMN* de la Stratégie 7 pour les valeurs de l-diversité et de t-proximité sont bien moins bonnes que la moyenne pour une grande majorité des configurations de table de *florida_30162*.

Pour conclure sur ces premières observations, nous avons constaté le même comportement extrême de la Stratégie 5 que lors de l'analyse des classements des stratégies dans le paragraphe « Classements des stratégies » de la section 4.4.2.2 page 76 : les *VMN* de la Stratégie 5 pour l'altération sont bien moins bonnes que la moyenne et les *VMN* de la Stratégie 5 pour les valeurs de l-diversité et de t-proximité sont bien meilleures que la moyenne sur l'intégralité des configurations de table des deux tables. Cela est logique car les *VMN* de la Stratégie 5 sont les optima des *VMN* des sept stratégies pour chaque mesure. Les Stratégies 2 et 4 semblent une nouvelle fois favoriser l'altération au dépend des valeurs de l-diversité et de t-proximité : leurs *VMN* pour l'altération sont bien meilleures que la moyenne et leurs *VMN* pour les valeurs de l-diversité et de t-proximité sont bien moins bonnes que la moyenne sur toutes les configurations de table des deux tables. Pour les Stratégies 3 et 6, le constat global est que leurs *VMN* pour les trois mesures sont proches de la moyenne pour toutes les configurations de table des deux tables. La Stratégie 1 a de bons résultats pour l'altération sur l'ensemble des configurations de table. Elle a de meilleures résultats pour les valeurs de l-diversité et de t-proximité sur les configurations de table de *florida_30162* que sur les configurations de table de *Adult data set*. La Stratégie 7 a, dans l'ensemble,

le même comportement que les Stratégies 2 et 4 pour les trois mesures bien que ses *VMN* soient plus proches de la moyenne sur certaines configurations de table.

Dans un second temps, nous avons essayé de déterminer la meilleure stratégie à utiliser selon l'attribut sensible de la configuration de table étudiée. L'objectif est de déterminer, pour un type d'attribut sensible, si une stratégie permet d'optimiser l'altération et les valeurs de l-diversité et de t-proximité mieux que les autres stratégies.

Pour cela, nous nous sommes intéressés aux couleurs associées aux écarts à la moyenne pour les trois mesures par chaque stratégie sur chaque configuration de table dans les tableaux 4.10a page 80 et 4.10b page 80.

Nous avons d'abord réalisé une étude à l'œil nu. Pour chaque colonne des tableaux d'écart à la moyenne, nous avons comparé les groupes de trois couleurs obtenus par les stratégies et nous avons choisi la stratégie qui, selon nous, parvient à conserver le meilleur compromis entre altération pour *NLLM*, valeur de l-diversité et valeur de t-proximité. Les critères pour déterminer la meilleure stratégie selon les couleurs obtenues sur lesquels nous avons basés nos observations sont les suivants :

— les trois couleurs ont des teintes assez proches. Avec ce critère, nous écartons une stratégie ayant des couleurs trop éloignées les unes des autres. Par exemple, une stratégie ayant deux cases de couleurs vertes et une case de couleur rouge ne nous intéresse pas car elle est très mauvaise pour l'une des trois mesures.

— la couleur dominante doit être aussi proche du vert que possible. Si les couleurs d'une première stratégie ont des teintes proches et sont proches du vert et les couleurs d'une seconde stratégie ont des teintes proches et sont proches du rouge, nous préférerons la première stratégie à la seconde.

Nous appelons cette méthode *méthode à l'œil nu*.

Par exemple, dans le tableau 4.10a page 80, pour la configuration de table $Adult_{A_{5,\text{Équiv}}}$ de *Adult data set*, nous écartons les Stratégies 1, 2, 4, 5 et 7 car les teintes des couleurs sont trop disparates (elles sont soit vertes soit rouges). Reste les Stratégies 3 et 6. Les couleurs obtenues par les deux stratégies pour les valeurs de l-diversité et de t-proximité sont quasiment identiques. Nous nous intéressons donc aux couleurs obtenues pour l'altération. La couleur de la Stratégie 6 pour l'altération (case en ligne (Stratégie 6, Altération pour *NLLM*) et en colonne (Loi Équivalente, 5)) est plus claire et plus éloignée du rouge que la couleur de la Stratégie 3 pour l'altération (case en ligne (Stratégie 3, Altération pour *NLLM*) et en colonne (Loi Équivalente, 5)). La stratégie que nous choisissons pour la configuration de table $Adult_{A_{5,\text{Équiv}}}$ est la Stratégie 6.

Parfois, il n'est pas évident de choisir à l'œil nu la meilleure stratégie pour une configuration de table. Par exemple, dans le tableau 4.10b page 80, pour la configuration de table $florida\_30162_{A_{5,\text{Géom}}}$, nous écartons les Stratégies 1, 2, 4 et 5 car leurs teintes de couleurs sont trop disparates. Pour les Stratégies 3 et 6, les couleurs obtenues semblent plus proches que les couleurs obtenues pour la Stratégie 7. Mais la teinte moyenne des couleurs obtenues pour la Stratégie 7 semble meilleure que les teintes moyennes des couleurs obtenues pour les Stratégies 3 et 6.

Nous avons donc cherché une autre méthode pour choisir automatiquement les meilleures stratégies pour chaque configuration de table des deux tables. Précisons que, dans les tableaux d'écart à la moyenne de la figure 4.10 page 80, chaque couleur est associée à un indice compris entre 0 et 255 correspondant à des quantités de vert et de rouge dans le format *RGB*. Plus l'indice est proche de 0, plus la couleur est verte et plus l'indice est proche de 255, plus la couleur est rouge. Pour chaque configuration de table, pour chaque stratégie, nous avons donc trois indices correspondant aux couleurs obtenues par la stratégie pour chaque mesure sur la configuration de table.

Pour retranscrire les critères de sélection sur lesquels nous nous sommes basés dans la méthode à l'œil nu, nous utilisons les notions de moyenne et d'*écart-type* d'une série de données.

Définition 4.4.1 (Écart-type)
Soit $(x_1, \ldots, x_n)$ un vecteur de $n \in \mathbb{N}$ valeurs quantitatives. On note $\bar{x}$ la moyenne des $(x_i)_{1 \leqslant i \leqslant n}$: $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$.

L'*écart-type* σ de $(x_1, \ldots, x_n)$ est défini par :

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2}.$$

Pour chaque configuration de table, pour chaque stratégie, nous calculons l'écart-type et la moyenne des trois indices correspondant aux couleurs obtenues par la stratégie pour les trois mesures sur la configuration de table. L'écart-type donne une indication sur la dispersion des indices : plus les indices seront éloignés les uns des autres, plus l'écart-type sera grand. Autrement dit, plus les écarts entre les *VMN* de la stratégie pour les trois mesures sur la configuration de table sont importants, plus l'écart-type que nous calculons sur les indices des couleurs associées aux *VMN* sera grand. Cela rejoint le premier critère que nous avons utilisé dans la méthode à l'œil nu traitant de la disparité des teintes des couleurs. De plus, en faisant la moyenne des trois indices, nous

	Loi Équivalente							Loi Géométrique							Loi Normale centrée réduite						
	5	10	20	50	100	200	500	5	10	20	50	100	200	500	5	10	20	50	100	200	500
Méthode à l'œil nu	6	3 ou 6	3	3	6	3	4	7	6 ou 7	3 ou 7	3	3	3?	7	6	7	7	3	7	3	6
Méthode par le calcul : Écart-type + Moyenne des indices des couleurs	6	7	7	3	6	3	4	7	7	6	6	6	3	3	7	7	7	6	7	3	6

(a) Pour *Adult data set*

	Loi Équivalente							Loi Géométrique							Loi Normale centrée réduite						
	5	10	20	50	100	200	500	5	10	20	50	100	200	500	5	10	20	50	100	200	500
Méthode à l'œil nu	7	3	3	3	3 ou 6	6	3	7?	3 ou 7	1 ou 3	1	7	3	1	6	3	3	1	1 ou 3	1	1
Méthode par le calcul : Écart-type + Moyenne des indices des couleurs	4	1	1	6	1	6	1	7	1	1	1	7	1	1	6	1	1	1	1	1	1

(b) Pour *florida_30162*

FIGURE 4.11 – Choix de la meilleure stratégie en se basant sur les couleurs obtenues dans les calculs d'écart à la moyenne des *VMN* sur les configurations de table de *Adult data set* et *florida_30162* en utilisant différentes méthodes

obtenons une indication sur la teinte moyenne des couleurs obtenues. Cela rejoint le second critère que nous avons utilisé dans la méthode à l'œil nu.

Pour chaque configuration de table, la stratégie retenue comme étant la meilleure sera celle qui minimise la somme de la moyenne et de l'écart-type des indices des couleurs. Nous appelons cette méthode *méthode par le calcul*.

Par exemple, intéressons nous à la configuration de table $Adult_{A_{5,\text{Équiv}}}$ de *Adult data set* (colonne (Loi Équivalente, 5) du tableau 4.10a page 80). Pour chaque stratégie, nous déterminons la moyenne des indices des couleurs, l'écart-type des indices des couleurs et nous calculons la somme de la moyenne et de l'écart-type. Le tableau suivant répertorie les indices des couleurs, la moyenne et l'écart-type de ces indices et la somme de l'écart-type et de la moyenne pour chaque stratégie. Les indices des couleurs sont présentés sous la forme [Altération pour *NLLM*, l-diversité, t-proximité].

	Indices des couleurs	Moyenne	Écart-type	Écart-type + Moyenne
Stratégie 1	[0, 208, 251]	153	109	262
Stratégie 2	[22, 255, 255]	177	109	286
Stratégie 3	[194, 76, 77]	115	55	170
Stratégie 4	[13, 191, 230]	144	94	238
Stratégie 5	[255, 0, 0]	85	120	205
Stratégie 6	[158, 76, 94]	109	35	144
Stratégie 7	[236, 70, 44]	116	85	201

La méthode par le calcul a choisi la Stratégie 6 comme étant la meilleure sur la configuration de table $Adult_{A_{5,\text{Équiv}}}$: elle obtient en effet le plus petit score d'écart-type + moyenne. D'autre part, la Stratégie 5 a obtenu deux cases vertes (indices 0) et une case rouge (indice 255) correspondant aux très bons résultats de valeurs de l-diversité et de t-proximité et au mauvais résultat d'altération. Nous remarquons que la moyenne des indices des couleurs est de 85, ce qui est bien inférieur aux moyennes obtenues pour les autres stratégies. Or l'écart-type des indices des couleurs pour la Stratégie 5 est de 120, l'une des valeurs les plus élevées. Ainsi, bien qu'en moyenne la Stratégie 5 soit la meilleure, le fait qu'elle optimise deux mesures mais soit mauvaise pour la dernière fait qu'elle n'est pas sélectionnée comme la meilleure stratégie par la méthode par le calcul.

Les tableaux de la figure 4.11 résument les choix des meilleures stratégies obtenus pour les configurations de table de *Adult data set* (tableau 4.11a) et *florida_30162* (tableau 4.11b).

Dans la ligne Méthode à l'œil nu, nous retrouvons les choix de meilleures stratégies que nous avons déterminés à l'œil nu. Par exemple, dans le tableau 4.11a, nous lisons en ligne Méthode à l'œil nu et en colonne (Loi Équivalente, 500) que la stratégie que nous avons jugé être la meilleure sur la configuration de table $Adult_{A_{500,\text{Équiv}}}$ de *Adult data set* est la Stratégie 4. Quand deux numéros de stratégies figurent dans une même case, cela signifie que nous n'avons pas réussi à discrimer les deux stratégies en question. Par exemple, dans le tableau 4.11b, nous lisons en ligne Méthode à l'œil nu et en colonne (Loi Géométrique, 20) que nous n'avons pas pu départager les Stratégies 1 et 3 sur la configuration de table $florida\_30162_{A_{20,\text{Géom}}}$ de *florida_30162* (les couleurs associées aux *VMN* de ces deux stratégies pour les trois mesures sont à retrouver dans le tableau 4.10b page 80 en colonne (Loi Géométrique, 20)). Quand un numéro de stratégie est suivi d'un point d'interrogation dans une case, cela

signifie que nous n'avons pas réussi à départager plus de deux stratégies et que nous avons choisi cette stratégie par défaut.

Dans la ligne Méthode par le calcul des tableaux, nous retrouvons les choix de meilleures stratégies déterminés par la méthode par le calcul.

Tout d'abord, nous constatons que les résultats obtenus avec la méthode par le calcul sont assez fidèles aux résultats obtenus avec la méthode à l'œil nu. Sur 42 configurations de table étudiées, la même stratégie a été choisie comme la meilleure par les deux méthodes pour 35 configurations de table. Pour 5 des 7 configurations de table pour lesquelles les résultats des deux méthodes ne sont pas identiques, nous constatons que le choix de la méthode par le calcul est acceptable a posteriori ; il s'agit des configurations de table $Adult_{A_{50,\text{Géom}}}$, $Adult_{A_{200,\text{Géom}}}$, $Adult_{A_{5,\text{Norm}}}$ et $Adult_{A_{500,\text{Norm}}}$ de *Adult data set* et $florida\_30162_{A_{50,\text{Norm}}}$ de *florida_30162*.

En revanche, pour les deux configurations de table restantes la méthode par le calcul donne un résultat que nous jugeons discutable. Sur $Adult_{A_{100,\text{Norm}}}$, la méthode par le calcul a choisi la Stratégie 6 comme étant la meilleure stratégie (case en ligne Méthode par le calcul et en colonne (Loi Normale centrée réduite, 100) du tableau 4.11a page précédente). Or nous avons sélectionné la Stratégie 7 à l'œil nu. Les couleurs de la Stratégie 6 sont plus proches que les couleurs de la Stratégie 7 : l'écart-type des indices des couleurs de la Stratégie 6 est de 21 et l'écart-type des indices des couleurs de la Stratégie 7 est de 53. Or la couleur moyenne des couleurs de la Stratégie 7 est plus proche du vert que la couleur moyenne des couleurs de la Stratégie 6 : la moyenne des indices des couleurs de la Stratégie 6 est de 151 et la moyenne des indices des couleurs de la Stratégie 7 est de 175. Finalement, quand nous faisons la somme de l'écart-type et de la moyenne des indices des couleurs des deux stratégies, nous obtenons 196 pour la Stratégie 6 et 204 pour la Stratégie 7. Les valeurs obtenues sont donc assez proches. Sur la configuration de table $florida\_30162_{A_{5,\text{Géom}}}$, les remarques sont les mêmes. Dans les deux cas, l'écart-type des indices des couleurs a pris le dessus sur la moyenne des indices des couleurs dans la méthode par le calcul. La méthode par le calcul a favorisé la proximité des couleurs à la teinte moyenne des couleurs.

Étudions maintenant les résultats obtenus par la méthode par le calcul sur les tables *Adult data set* et *florida_30162*.

Globalement, ce sont les Stratégies 3 (optimisation sur la valeur de l-diversité puis sur le coût de généralisation), 6 (optimisation sur la valeur de t-proximité puis sur le coût de généralisation) et 7 (optimisation sur le coût de généralisation multiplié par la valeur de t-proximité) qui ont été le plus souvent choisies comme étant les meilleures sur la totalité des configurations de table. La Stratégie 3 a été sélectionnée pour environ 45% des configurations de table et les Stratégies 6 et 7 ont toutes les deux été sélectionnées pour environ 19% des configurations de table.

Pour les configurations de table de *Adult data set* et *florida_30162* dans lesquelles la répartition des valeurs de l'attribut sensible suit une loi Équivalente, la Stratégie 3 est choisie comme étant la meilleure pour la majorité de ces configurations de table (ligne Méthode par le calcul et colonne Loi Équivalente des tableaux 4.11a page précédente et 4.11b page précédente).

Pour les configurations de table de *Adult data set* dans lesquelles la répartition des valeurs de l'attribut sensible suit une loi Géométrique ou Normale centrée réduite, la Stratégie 7 est choisie comme étant la meilleure quand le nombre de valeurs possibles de l'attribut sensible est compris entre 3 et 20. Lorsque le nombre de valeurs possibles de l'attribut sensible est supérieur à 20, la méthode par le calcul choisit la Stratégie 3 pour 75% des configurations de table de *Adult data set* dans lesquelles la répartition des valeurs de l'attribut sensible suit une Loi Géométrique mais les résultats sont plus mitigés pour les configurations de table de *Adult data set* dans lesquelles la répartition des valeurs de l'attribut sensible suit une loi Normale centrée réduite.

Pour les configurations de table de *florida_30162* dans lesquelles la répartition des valeurs de l'attribut sensible suit une loi Géométrique ou Normale centrée réduite, il est à noter que la Stratégie 1 (optimisation sur le coût de généralisation) est choisie par la méthode par le calcul pour environ 36% de ces configurations de table. Pour les configurations de table de *florida_30162* dans lesquelles la répartition des valeurs de l'attribut sensible suit une loi Normale centrée réduite, la Stratégie 3 est sélectionnée comme étant la meilleure par la méthode par le calcul pour plus de la moitié de ces configurations. Pour les configurations de table de *florida_30162* dans lesquelles la répartition des valeurs de l'attribut sensible suit une loi Géométrique, aucune stratégie ne se détache.

Pour conclure sur cette étude, les résultats obtenus sur les configurations de table de *Adult data set* et *florida_30162* suggèrent que les Stratégies 2, 4 et 5 sont à écarter. En effet, elles ne sont pas ou peu sélectionnées comme étant les stratégies conservant le meilleur compromis entre altération, valeur de l-diversité et valeur de t-proximité par la méthode par le calcul. Cela est dû au fait que ces stratégies favorisent une ou deux mesures au détriment de la ou des autres (les écarts-type des indices des couleurs obtenues pour ces stratégies sont plus élevés que ceux des autres stratégies). Sur près de la moitié des configurations de tables étudiées, la Stratégie 3 peut-être considérée comme la stratégie permettant de conserver le meilleur compromis entre altération, valeur de l-diversité et valeur de t-proximité dans les tables k-anonymes produites. Les Stratégies 6 et 7 sont également

bien représentées. Cependant, aucun résultat significatif n'est à noter pour une loi de probabilité particulière ou un nombre particulier de valeurs possibles de l'attribut sensible.

4.5 Conclusion du chapitre

Dans ce chapitre, notre objectif a été de proposer des stratégies permettant de produire des tables k-anonymes présentant des valeurs de l-diversité et de t-proximité intéressantes. Nous avions en effet noté dans le chapitre 1 page 5 que les tables k-anonymes souffrent parfois d'un manque de diversité des valeurs de l'attribut sensible dans leurs classes d'équivalence. Nous avons donc, dans ce chapitre, tenu compte de la répartition des valeurs de l'attribut sensible dans les classes d'équivalence lors d'un processus de k-anonymisation.

Dans la section 4.1 page 56, nous sommes d'abord revenus sur la l-diversité et la t-proximité, deux modèles d'anonymisation dont les exigences portent sur la répartition des valeurs de l'attribut sensible dans les classes d'équivalence de la table. Après avoir proposé des définitions se basant sur les articles [34] et [29] et sur les notations du chapitre 2 page 13, nous avons introduit deux mesures permettant d'évaluer la qualité d'une table en termes de l-diversité et de t-proximité : la valeur de l-diversité (cf. définition 4.1.7 page 59) et la valeur de t-proximité (cf. définition 4.1.11 page 61).

Dans la section 4.2 page 61, nous avons présenté l'algorithme *GAA* permettant de produire une version d'une table respectant un certain modèle d'anonymisation. Dans cet algorithme, une succession de fusions de classes d'équivalence est effectuée dans la table jusqu'à atteindre la contrainte imposée par le modèle d'anonymisation. De plus, chaque fusion de classes d'équivalence est choisie de telle sorte à optimiser une stratégie passée en paramètres de l'algorithme *GAA*.

Dans la section 4.3 page 62, nous avons présenté sept stratégies d'optimisation à utiliser dans *GAA* pour guider les fusions de classes d'équivalence à effectuer. Les stratégies proposées permettent d'optimiser, à chaque tour de l'algorithme, l'altération ou les valeurs de l-diversité ou de t-proximité de la table. La Stratégie 1 n'optimise que le coût de généralisation ; elle nous a servi de référence pour comparer les performances de stratégies mêlant coût de généralisation et valeurs de l-diversité ou de t-proximité. Les Stratégies 2, 3 et 4 ont pour objectif d'optimiser le coût de généralisation et la valeur de l-diversité. Les Stratégies 5, 6 et 7 quant à elles optimisent à la fois le coût de généralisation et la valeur de t-proximité.

Dans la section 4.4 page 65, nous avons mené des expérimentations pour comparer les performances des sept stratégies sur la production de tables k-anonymes. Nous avons considéré deux tables, *Adult data set* et *florida_30162* (cf. section 2.4 page 22), et trois mesures pour évaluer la qualité des tables k-anonymes produites, l'altération (cf. définition 3.1.6 page 31), la valeur de l-diversité (cf. définition 4.1.7 page 59) et la valeur de t-proximité (cf. définition 4.1.11 page 61). Nous avons mené deux types d'expérimentations selon le choix de l'attribut sensible dans la table : soit l'attribut sensible est présent dans la table (il contient des données réelles) soit l'attribut est une nouvelle colonne générée (il contient des données simulées suivant une répartition prédéterminée).

Pour le premier type d'expérimentations (cf. section 4.4.2.1 page 68), nous avons considéré deux configurations de table de *Adult data set* et une configuration de table de *florida_30162*. Nous avons tout d'abord justifié l'introduction de stratégies mêlant coût de généralisation et valeur de l-diversité ou de t-proximité. Nous avons observé que l'utilisation dans *GAA* de stratégies n'optimisant que la valeur de l-diversité ou que la valeur de t-proximité ne permet pas de limiter l'altération des tables k-anonymes produites. Puis nous avons confronté les sept stratégies proposées sur les trois configurations de table. En étudiant les *VMN* (cf. section 4.4.1 page 65) de chaque stratégie pour l'altération, la valeur de l-diversité et la valeur de t-proximité, nos observations ont été les suivantes. La Stratégie 5 est équivalente aux stratégies n'optimisant que la l-diversité ou que la t-proximité : ses *VMN* pour les valeurs de l-diversité et de t-proximité sont presque optimales alors que sa *VMN* pour l'altération est proche de la perte totale d'information dans toutes les tables k-anonymes produites. Ce comportement est à noter pour les trois configurations de table étudiées. Les Stratégies 1, 2 et 4 ont de bons résultats pour l'altération mais peinent à maintenir de bonnes valeurs de l-diversité et de t-proximité dans les tables k-anonymes. Les Stratégies 3 et 6 ont le comportement inverse : elles sont parmi les meilleures stratégies pour les valeurs de l-diversité et de t-proximité mais ne limitent pas efficacement l'altération dans les tables k-anonymes. La Stratégie 7 obtient des résultats intermédiaires pour les trois mesures. D'autre part, les écarts entre les meilleures *VMN* et les moins bonnes *VMN* pour les trois mesures sont assez significatifs. Ainsi, nous avons remarqué une nette balance entre optimisation de l'altération et optimisation des valeurs de l-diversité et de t-proximité.

Pour le second type d'expérimentations (cf. section 4.4.2.2 page 75), nous avons généré 21 nouvelles colonnes de 30 162 valeurs ayant de 5 à 500 valeurs possibles et dont la répartition des valeurs suit une loi Équivalente, une loi Géométrique ou une loi Normale centrée réduite. Ces nouvelles colonnes ont été ajoutées aux tables *Adult data set* et *florida_30162*, fournissant ainsi 42 configurations de tables à étudier dans nos expérimentations. Dans un premier temps, nous avons classé les stratégies selon leur *VMN* pour chaque mesure sur chaque configuration de

table. Nos observations sur les performances des stratégies ont été globalement similaires à celles des premières expérimentations. Nous avons ensuite calculé la *VMN* moyenne de chaque stratégie pour chaque mesure sur toutes les configurations de table de *Adult data set* et de *florida_30162*. Nous avons constaté que les *VMN* moyennes des stratégies pour les valeurs de l-diversité et de t-proximité sont très proches et très bonnes pour les deux tables (supérieures à 93% pour la valeur de l-diversité et inférieures à 10% pour la valeur de t-proximité). En revanche, les écarts sont plus prononcées entre les *VMN* moyennes des stratégies pour l'altération. De plus, elles sont toutes supérieures à 50%. Ainsi une première conclusion que nous pouvons tirer est que, sur des configurations de table dans lesquelles l'attribut sensible est une colonne générée dont la répartition des valeurs suivent une certaine loi de probabilité, les stratégies ont sensiblement les mêmes performances en termes de valeur de l-diversité et de valeur de t-proximité. Pour choisir la meilleure stratégie à utiliser sur une table, il suffit donc de sélectionner celle ayant la *VMN* moyenne pour l'altération la plus basse, soit la Stratégie 1 pour *Adult data set* et la Stratégie 2 pour *florida_30162*.

Pour aller plus loin dans l'analyse des résultats des expérimentations dans lesquelles l'attribut sensible contient des valeurs simulées, nous avons proposé une méthode se basant sur les écarts à la moyenne des *VMN* des stratégies. Cette méthode a pour objectif de choisir la meilleure stratégie à utiliser selon les caractéristiques de l'attribut sensible considéré dans la table. Bien que nous n'ayons pas dégagé de résultats significatifs pour un nombre de valeurs possibles ou une loi de probabilité particulière, nous avons observé que les Stratégies 3, 6 et 7 sont les stratégies sélectionnées comme étant les meilleures sur une grande majorité des configurations de table de *Adult data set* et *florida_30162*.

Pour conclure, les expérimentations sur des données réelles ont montré que les Stratégies 1, 2 et 4 sont meilleures pour limiter l'altération que pour optimiser les valeurs de l-diversité et de t-proximité. Au contraire, les Stratégies 3, 6 et 7 sont meilleures pour favoriser les valeurs de l-diversité et de t-proximité que pour limiter l'altération. La Stratégie 5 se comporte comme une stratégie n'optimisant que la valeur de l-diversité ou la valeur de t-proximité. D'un autre côté, les expérimentations sur des données simulées suggèrent que les performances des stratégies sont équivalentes en termes d'optimisation des valeurs de l-diversité et de t-proximité et que seuls les résultats d'altération sont à considérer pour déterminer la stratégie permettant de produire les meilleures versions k-anonymes d'une table.

Perspective. Pour continuer ce travail, nous pourrions adopter une approche opposée à celle proposée dans ce chapitre. Au lieu de chercher à construire une table k-anonyme en optimisant les valeurs de l-diversité et de t-proximité, nous pourrions construire une table respectant les modèles de l-diversité ou de t-proximité en optimisant la valeur de k à chaque fusion de classes d'équivalence dans *GAA*. L'expérimentation à mener serait la suivante. nous utilisons l'algorithme *GAA* avec comme condition d'arrêt le respect de la l-diversité ou de la t-proximité pour des valeurs l et t données. La stratégie à passer en paramètres de *GAA* aurait pour condition

$$C \in \{C' \in \mathcal{C}(T) \,:\, |C' \cup C_s| = \min_{C'' \in \mathcal{C}(T)} |C'' \cup C_s|\},$$

avec T une table à anonymiser et C_s une classe d'équivalence ne respectant pas les contraintes d'anonymité et de plus petite taille. Pour garder un contrôle sur l'altération de la table anonyme produite, nous pourrions également ajouter une condition de minimisation du coût de généralisation à notre stratégie. Ainsi, l'objectif serait de trouver le k optimal pour lequel la table k-anonyme respecte les valeurs de l-diversité et de t-proximité demandées.

Chapitre 5

Procédure d'amélioration de tables k-anonymes

Sommaire du présent chapitre

L'algorithme *GkAA* présenté dans le chapitre 3 page 25 ne donne aucune garantie d'optimalité en termes d'altération des tables k-anonymes produites. Il n'est pas non plus possible de connaître la différence entre l'altération de la table k-anonyme produite et l'altération minimale possible. Autrement dit, nous ne pouvons pas savoir si la solution obtenue est éloignée ou non d'une solution optimale. En étudiant des exécutions de *GkAA*, nous observons que certains des choix effectués par l'algorithme ont entraîné des généralisations trop élevées de certains enregistrements de la table. L'exemple 5.0.1 illustre ce point.

Exemple 5.0.1
Soient $\mathcal{Q} = \{Q_1, Q_2\}$ un ensemble d'attributs quasi-identifiants avec $Q_1 = \{a_1, a_2, a_3\}$ et $Q_2 = \{b_1, b_2\}$.

Considérons $\mathcal{H} = (H_1, H_2)$ un couple de hiérarchies associées aux attributs de $\mathcal{Q}$ et décrites dans la figure 5.3. La hiérarchie H_1 est composée de trois feuilles (a_1, a_2 et a_3), d'un nœud de niveau 1 ($a_{1,2}$ qui est une généralisation de a_1 et a_2) et d'une racine ($a_{1,2,3}$). La hiérarchie H_2 est composée de deux feuilles (b_1 et b_2) et d'une racine ($b_{1,2}$).

Considérons la table sur $\mathcal{Q}$ décrite par la figure 5.1 et notée T. T a quatre enregistrements $E^1 = (a_3, b_1)$, $E^2 = (a_1, b_1)$, $E^3 = (a_2, b_1)$ et $E^4 = (a_3, b_2)$. T a quatre classes d'équivalence contenant un enregistrement car les enregistrements de T sont tous distincts.

En utilisant la formule donnée en section 3.2 page 32, les poids sur les arêtes des hiérachies H_1 et H_2 pour *NLLM* sont les suivants : $\omega(a_1, a_{1,2}) = 2/3$, $\omega(a_2, a_{1,2}) = 2/3$, $\omega(a_{1,2}, a_{1,2,3}) = 1/3$, $\omega(a_3, a_{1,2,3}) = 1$, $\omega(b_1, b_{1,2}) = 3/2$ et $\omega(b_2, b_{1,2}) = 3/2$.

Nous allons appliquer l'algorithme *GkAA* sur la table T en utilisant la métrique *NLLM* pour obtenir une version 2-anonyme de T. Décrivons l'exécution de *GkAA*.

Au premier tour de l'algorithme, nous cherchons à fusionner la classe $\{E^1\}$ avec une autre classe telle que le coût de généralisation pour *NLLM* soit minimisé. Nous calculons $\overline{NLLM}(E^1, E^2) = 2$, $\overline{NLLM}(E^1, E^3) = 2$ et

T	Q_1	Q_2
E^1	a_3	b_1
E^2	a_1	b_1
E^3	a_2	b_1
E^4	a_3	b_2

FIGURE 5.1 – Représentation d'une table sur $\mathcal{Q}$.

T_1^{gen}	Q_1	Q_2
F_1^1	$a_{1,2,3}$	$b_{1,2}$
F_1^2	$a_{1,2,3}$	$b_{1,2}$
F_1^3	$a_{1,2,3}$	$b_{1,2}$
F_1^4	$a_{1,2,3}$	$b_{1,2}$

(a) Une version 2-anonyme de T ayant une altération pour $NLLM$ de 100%.

T_2^{gen}	Q_1	Q_2
F_2^1	a_3	$b_{1,2}$
F_2^2	$a_{1,2}$	b_1
F_2^3	$a_{1,2}$	b_1
F_2^4	a_3	$b_{1,2}$

(b) Une version 2-anonyme de T ayant une altération pour $NLLM$ d'environ 43,3%.

FIGURE 5.2 – Deux versions 2-anonymes d'une table T.

$\overline{NLLM}(E^1, E^4) = 3$. Les classes $\{E^1\}$ et $\{E^2\}$ sont donc fusionnées. Nous obtenons une table à trois classes d'équivalence : $C_1 = \{E^1, E^2\}$, $C_2 = \{E^3\}$ et $C_3 = \{E^4\}$. Au deuxième tour de l'algorithme, nous cherchons à fusionner la classe $C_2 = \{E^3\}$ avec une autre classe. Nous calculons $\overline{NLLM}(C_2, C_1) = 1$ et $\overline{NLLM}(C_2, C_3) = 5$. Les classes C_2 et C_1 sont donc fusionnées. Nous obtenons une table à deux classes d'équivalence : $C_1 = \{E^1, E^2, E^3\}$ et $C_2 = \{E^4\}$. Au troisième tour de l'algorithme, il reste à traiter C_2 qui n'est pas de taille supérieure à 2. Il n'y a qu'une possibilité : les classes C_2 et C_1 sont fusionnées. Nous obtenons une table T_1^{gen} à une classe d'équivalence et dont les enregistrements sont égaux à $(a_{1,2,3}, b_{1,2})$, soit la généralisation maximale. Le coût de généralisation de cette version 2-anonyme pour $NLLM$ est de $NLLM_T(T_1^{gen}) = 10$. Une représentation de la table T_1^{gen} est à retrouver en figure 5.2a.

Cependant, nous remarquons que nous pouvons fusionner les enregistrements E^1 et E^4 d'une part et E^2 et E^3 d'autre part pour obtenir une seconde version 2-anonyme de T notée T_2^{gen}. Une représentation de la table T_2^{gen} est à retrouver en figure 5.2b. Le coût de généralisation de cette seconde version 2-anonyme de T pour $NLLM$ est $NLLM_T(T_2^{gen}) = 13/3$. T_2^{gen} est donc de meilleure qualité en termes de coût de généralisation pour $NLLM$ que la table T_1^{gen} produite avec $GkAA$.

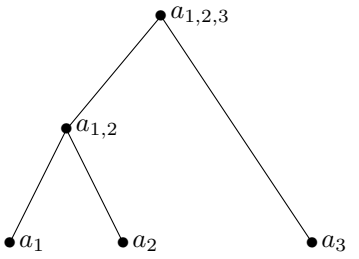
De plus, $\Lambda_{NLLM,T}(T_1^{gen}) = 100\%$ et $\Lambda_{NLLM,T}(T_2^{gen}) \simeq 43,3\%$.

Bien que les fusions effectuées à chaque tour aient été choisies de telle sorte à optimiser le coût de généralisation pour $NLLM$, la table 2-anonyme produite par $GkAA$ n'est clairement pas optimale en termes d'altération pour $NLLM$.

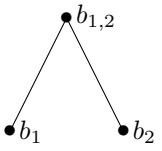
Il semble donc possible de réduire les généralisations appliquées aux enregistrements tout en conservant la k-anonymité de la table.

Dans ce chapitre, nous nous proposons de construire une table k-anonyme d'altération moins élevée que la table k-anonyme produite avec l'algorithme $GkAA$. Pour cela, nous partirons d'une table k'-anonyme avec $k' \geqslant k$ et, en cherchant d'autres généralisations de ses enregistrements, nous tâcherons de produire une table k-anonyme d'altération moins élevée que celle produite avec $GkAA$. Rappelons qu'une table dont le paramètre de k-anonymité est supérieur à k est en particulier k-anonyme.

Dans une table k-anonyme, les classes d'équivalence ne sont pas toujours exactement de taille k : certaines



(a) Représentation d'une hiérarchie de Q_1



(b) Représentation d'une hiérarchie de Q_2

FIGURE 5.3 – Représentation d'un vecteur de hiérarchies de $\mathcal{Q} = \{Q_1, Q_2\}$.

classes d'équivalence peuvent contenir bien de plus de k enregistrements. Ces classes de grande taille posent problème : elles ont parfois un coût de généralisation très élevé, ce qui entraîne une forte altération de la table k-anonyme, alors qu'un regroupement différent de leurs enregistrements aurait limité le coût de généralisation tout en maintenant la k-anonymité. Notre objectif est donc de réduire le coût de généralisation de ces classes de grande taille en partitionnant leurs enregistrements en sous-ensembles de tailles plus proches de k. Nous espérons que de telles modifications des classes d'équivalence entraîneront une baisse de l'altération de la table k-anonyme. Pour déterminer de nouveaux partitionnements, nous rassemblerons les enregistrements selon leurs valeurs quasi-identifiantes et les généralisations possibles de ces valeurs : nous définirons les *groupes de généralisation*.

Dans la suite de ce chapitre, nous présenterons en section 5.1 les groupes de généralisation qui nous permettront de rassembler les enregistrements en fonction de leurs valeurs quasi-identifiantes et des généralisations possibles de ces valeurs. Nous construirons également l'ensemble minimal des groupes de généralisation. Cet ensemble ne contient que les groupes de généralisation qui apportent une information pertinente sur la topologie de la table. À partir de l'ensemble minimal des groupes de généralisation, nous proposerons en section 5.2 page 100 une nouvelle formulation du problème de k-anonymité sur une table. Pour cela, nous reviendrons sur la notion de partitionnements d'une table sur son ensemble minimal de groupes de généralisation (cf. section 5.2.1 page 100) et nous utiliserons un hypergraphe pour représenter le lien entre la table et ses groupes de généralisation (cf. section 5.2.2 page 102). En nous appuyant sur la formulation du problème de k-anonymité sur une table, nous proposerons une procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ en section 5.3 page 105. Nous présenterons également cinq algorithmes se basant sur cette procédure et produisant des tables k-anonymes. En section 5.4 page 113, nous reviendrons sur le protocole expérimental nous permettant de comparer les performances de nos algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ à celles de *GkAA*. Nous analyserons les résultats obtenus dans la section 5.4.2 page 115. Nous conclurons ce chapitre en section 5.5 page 126.

Dans ce chapitre, pour ne pas alourdir les définitions, nous considérerons que l'ensemble d'attributs n'est composé que de quasi-identifiants ; il sera noté $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ avec $m \in \mathbb{N}^*$.

5.1 Groupes de généralisation

Dans cette section, nous allons définir les groupes de généralisation d'une table sur un uplet de hiérarchies de généralisation. Ces ensembles permettent de regrouper les enregistrements dont les valeurs quasi-identifiantes sont les mêmes ou peuvent se généraliser en la même valeur (par exemple, *Chat* et *Lion* sont des valeurs pouvant se généraliser en *Félin*). Ils seront au cœur de notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$.

5.1.1 Ensemble de groupes de généralisation

Dans cette section, nous introduisons l'ensemble de groupes de généralisation d'une table sur un uplet de hiérarchies de généralisation. L'exemple 5.1.1 présente un cas de figure qui a motivé l'introduction des groupes de généralisation.

Exemple 5.1.1
Soient $\mathcal{Q} = \{Q_1, Q_2\}$ un ensemble d'attributs quasi-identifiants avec $Q_1 = \{a_1, a_2, a_3\}$ et $Q_2 = \{b_1, b_2\}$.

Considérons $\mathcal{H} = (H_1, H_2)$ un couple de hiérarchies associées aux attributs de $\mathcal{Q}$ et décrites dans la figure 5.3. La hiérarchie H_1 est composée de trois feuilles (a_1, a_2 et a_3), d'un nœud de niveau 1 ($a_{1,2}$ qui est une généralisation de a_1 et a_2) et d'une racine ($a_{1,2,3}$). La hiérarchie H_2 est composée de deux feuilles (b_1 et b_2) et d'une racine ($b_{1,2}$).

Considérons la table sur $\mathcal{Q}$ décrite par la figure 5.1 et notée T. T a quatre enregistrements $E^1 = (a_3, b_1)$, $E^2 = (a_1, b_1)$, $E^3 = (a_2, b_1)$ et $E^4 = (a_3, b_2)$.

Dans l'exemple 5.0.1 page 87, nous avons montré que l'algorithme *GkAA* ne donnait pas une version 2-anonyme de T optimale en termes d'altération. En étudiant la table et les hiérarchies H_1 et H_2, nous avons réussi à construire une version 2-anonyme d'altération beaucoup moins élevée que la version 2-anonyme produite avec *GkAA*. L'objectif est maintenant de trouver une méthode pour construire une telle version 2-anonyme sans que cela soit trop coûteux en temps de calcul.

Dans cet exemple, nous constatons que les enregistrements E^2 et E^3 peuvent se généraliser en $(a_{1,2}, b_1)$. De plus, les enregistrements E^1 et E^4 peuvent se généraliser en $(a_3, b_{1,2})$. Ces deux groupes d'enregistrements, en les fusionnant, permettent d'obtenir la version 2-anonyme T_2^{gen} de T de l'exemple 5.0.1 page 87.

L'exemple 5.1.1 page précédente suggère que regrouper les enregistrements d'une table en fonction de leurs valeurs quasi-identifiantes et des généralisations possibles de ces valeurs peut permettre de construire de bonnes versions k-anonymes de la table en termes d'altération.

Pour regrouper les enregistrements d'une table, nous allons définir les *groupes de généralisation* (cf. définition 5.1.1). Un groupe de généralisation est un ensemble d'enregistrements de la table pouvant se généraliser en un enregistrement donné.

Rappelons qu'un nœud v' d'une hiérarchie est une généralisation d'un nœud v si v' est sur le chemin de v à la racine de la hiérarchie (cf. définition 2.3.1 page 19). Rappelons également qu'un enregistrement F' est une généralisation d'un enregistrement F si les nœuds de F' sont des généralisations des nœuds de F (cf. définition 2.3.2 page 19).

Définition 5.1.1 (Groupe de généralisation sur un ensemble d'attributs)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$.

Soit $(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})$ un enregistrement sur $(\mathcal{Q}, \mathcal{H})$ avec $v^j_{l_j,p_j}$ le $p_j{}^{\text{e}}$ nœud du niveau l_j de la hiérarchie H_j pour $l_j \in [\![0, h_{H_j} - 1]\!]$, $p_j \in [\![1, \text{nn}(l_j)]\!]$ pour tout $j \in [\![1, m]\!]$ (cf. définition 2.2.3 page 16).

On définit le *groupe de généralisation* $G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})}$ de T sur $\mathcal{H}$ comme suit :

$$G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})} = \{E = (e_1, \ldots, e_m) \in T \,:\, v^j_{l_j,p_j} \text{ soit une généralisation de } e_j \text{ pour tout } j \in [\![1, m]\!]\}.$$

On dit que l'enregistrement $(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})$ est l'*enregistrement de référence* du groupe de généralisation $G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})}$.

On note $\mathcal{G}_{\mathcal{H}}(T) = \{G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})} \neq \emptyset \,:\, l_j \in [\![0, h_{H_j} - 1]\!] \text{ et } p_j \in [\![1, \text{nn}(l_j)]\!] \text{ pour tout } j \in [\![1, m]\!]\}$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Par abus de notations, l'ensemble des groupes de généralisation de T sur une hiérachie H_j, pour $j \in [\![1, m]\!]$, est noté $\mathcal{G}_{H_j}(T)$.

Remarque 5.1.1
En reprenant les notations de la définition 5.1.1, le groupe de généralisation $G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})}$ peut aussi s'écrire

$$G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})} = \{E \in T \,:\, (v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m}) \text{ soit une généralisation de } E\}.$$

Exemple 5.1.2
Soient $\mathcal{Q} = \{Q_1, Q_2\}$ un ensemble d'attributs quasi-identifiants avec $Q_1 = \{a_1, a_2, a_3\}$ et $Q_2 = \{b_1, b_2\}$.

Considérons $\mathcal{H} = (H_1, H_2)$ un couple de hiérarchies associées aux attributs de $\mathcal{Q}$ et décrites dans la figure 5.3. La hiérarchie H_1 est composée de trois feuilles (a_1, a_2 et a_3), d'un nœud de niveau 1 ($a_{1,2}$ qui est une généralisation de a_1 et a_2) et d'une racine ($a_{1,2,3}$). La hiérarchie H_2 est composée de deux feuilles (b_1 et b_2) et d'une racine ($b_{1,2}$).

Considérons la table sur $\mathcal{Q}$ décrite par la figure 5.5 et notée T. T a deux enregistrements $E^1 = (a_1, b_1)$ et $E^2 = (a_2, b_2)$.

Calculons les groupes de généralisation de T sur $\mathcal{H}$. Pour cela, considérons tous les couples (α, β) avec α un nœud de H_1 et β un nœud de H_2 et appliquons la définition 5.1.1.

Pour le couple (a_1, b_1), nous cherchons les enregistrements de T tels que (a_1, b_1) soit une de leurs généralisations. E^1 vérifie la condition précédente car $E^1 = (a_1, b_1)$. En revanche, E^2 ne vérifie pas la condition car b_1 n'est pas une généralisation de $b_{1,2}$. Nous obtenons $G_{(a_1,b_1)} = \{E^1\}$.

Pour le couple (a_1, b_2), nous cherchons les enregistrements de T tels que (a_1, b_2) soit une de leurs généralisations. E^1 ne vérifie pas cette condition car b_2 n'est pas une généralisation de b_1. De même, E^2 ne vérifie pas la condition car a_1 n'est pas une généralisation de a_2. Le groupe de généralisation $G_{(a_1,b_2)}$ est vide.

Pour le couple $(a_{1,2}, b_{1,2})$, nous cherchons les enregistrements de T tels que $(a_{1,2}, b_{1,2})$ soit une de leurs généralisations. E^1 vérifie cette condition car $a_{1,2}$ est une généralisation de a_1 et $b_{1,2}$ est une généralisation de b_1. De même, E^2 remplit la condition car $a_{1,2}$ est une généralisation de a_2 et $b_{1,2}$ est une généralisation de b_2. Nous obtenons $G_{(a_{1,2},b_{1,2})} = \{E^1, E^2\} = T$.

Nous appliquons le même méthode pour les autres couples de nœuds (α, β).

L'ensemble $\mathcal{G}_{\mathcal{H}}(T)$ contient $G_{(a_1,b_1)} = \{E^1\}$, $G_{(a_1,b_{1,2})} = \{E^1\}$, $G_{(a_2,b_2)} = \{E^2\}$, $G_{(a_2,b_{1,2})} = \{E^2\}$, $G_{(a_{1,2},b_1)} = \{E^1\}$, $G_{(a_{1,2},b_2)} = \{E^2\}$, $G_{(a_{1,2},b_{1,2})} = T$, $G_{(a_{1,2,3},b_1)} = \{E^1\}$, $G_{(a_{1,2,3},b_2)} = \{E^2\}$ et $G_{(a_{1,2,3},b_{1,2})} = T$.

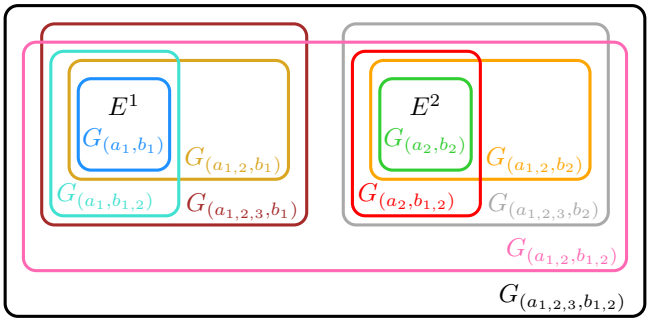


FIGURE 5.4 – Représentation de la composition des groupes de généralisation d'une table à deux enregistrements

Nous obtenons

$$\mathcal{G}_{\mathcal{H}}(T) = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_1)}, G_{(a_{1,2},b_2)}, G_{(a_{1,2},b_{1,2})}, G_{(a_{1,2,3},b_1)}, G_{(a_{1,2,3},b_2)}, G_{(a_{1,2,3},b_{1,2})}\}.$$

Pour représenter visuellement la composition des groupes de généralisation de T sur $\mathcal{H}$, nous proposons la représentation de la figure 5.4. On y retrouve les deux enregistrements de T, E^1 et E^2. Les groupes de généralisation sont représentés par des rectangles de différentes couleurs. Grâce à cette représentation, nous observons par exemple que trois groupes de généralisation ne contiennent que l'enregistrement E^1 : il s'agit de $G_{(a_1,b_1)}$, $G_{(a_{1,2,b_1})}$ et $G_{(a_1,b_{1,2})}$ de couleur respective bleue, jaune et turquoise.

Afin de rendre l'exemple précédent plus concret, associons à chaque nœud de H_1 et H_2 une valeur plus parlante. Supposons que l'atttribut Q_1 contienne des races d'animaux : a_1 correspond à *Chat*, a_2 correspond à *Lion* et a_3 correspond à *Chien*. Dans la hiérarchie H_1, nous associons donc à $a_{1,2}$ la valeur *Félin* et à $a_{1,2,3}$ la valeur *Mammifère*. Supposons que l'attribut Q_2 soit l'attribut *Genre* : b_1 correspond à *Mâle* et b_2 correspond à *Femelle*. Dans la hiérarchie H_2, nous associons donc à $b_{1,2}$ la valeur *Inconnu*.

Dans la table T, nous avons donc un (*Chat*, *Mâle*) (l'enregistrement $E^1 = (a_1, b_1)$) et un (*Lion*, *Femelle*) (l'enregistrement $E^2 = (a_2, b_2)$).

Pour le premier groupe de généralisation que nous avons calculé d'enregistrement de référence (a_1, b_1), nous cherchons les enregistrements de T pouvant se généraliser en (*Chat*, *Mâle*). Comme nous l'avons vu précédemment, seul E^1 correspond à cette description car il s'agit d'un (*Chat*, *Mâle*). *Lion* n'étant pas une généralisation de *Chat*, l'enregistrement E^2 n'appartient pas à ce groupe de généralisation.

Pour le troisième groupe de généralisation que nous avons calculé d'enregistrement de référence $(a_{1,2}, b_{1,2})$, nous cherchons les enregistrements de T pouvant se généraliser en (*Félin*, *Inconnu*). Il s'agit donc de chercher dans T des félins quelque soit leur genre. Les enregistrements E^1 et E^2 appartiennent à ce groupe de généralisation car *Chat* et *Lion* peuvent se généraliser en *Félin*.

La remarque 5.1.2 montre que l'ensemble des groupes de généralisation d'une table sur un uplet de hiérarchie de généralisation contient toujours au moins un élément. Ce groupe de généralisation a pour enregistrement de référence l'enregistrement constitué des racines des hiérarchies des attributs quasi-identifiants.

Remarque 5.1.2
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Pour tout $j \in [\![1, m]\!]$, notons r_j la racine de la hiérarchie H_j. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$.

Le groupe de généralisation $G_{(r_1,\ldots,r_m)}$ de T sur $\mathcal{H}$ contient les enregistrements de T pouvant se généraliser en $(r_1, \ldots, r_m)$, l'enregistrement constitué des racines des hiérarchies $H_1, \ldots, H_m$. Ce groupe de généralisation est égal à T et n'est donc jamais vide.

En effet, soit $E = (e_1, \ldots, e_m) \in T$. Si $E \in G_{(r_1,\ldots,r_m)}$ alors $E \in T$ par définition d'un groupe de généralisation. Si $E \in T$ alors on veut montrer que $E \in G_{(r_1,\ldots,r_m)}$ c'est-à-dire $\forall j \in [\![1, m]\!]$, r_j est une généralisation de e_j c'est-à-dire $\forall j \in [\![1, m]\!]$, r_j est sur le chemin de e_j à r_j. Soit $j \in [\![1, m]\!]$, r_j est sur le chemin de e_j à r_j par définition d'un chemin dans un arbre.

Dans l'exemple 5.1.2 page ci-contre, le groupe de généralisation $G_{(a_{1,2,3},b_{1,2})}$ a pour enregistrement de référence l'enregistrement $(a_{1,2,3}, b_{1,2})$ constitué des racines des hiérarchies H_1 et H_2. Il correspond à l'ensemble d'enregistrement $\{E^1, E^2\} = T$. En reprenant les valeurs plus concrètes, nous avons $E^1 = (Chat, Mâle)$ et

$E^2 = (Lion, Femelle)$. Il est clair que ces deux enregistrements correspondent à des mammifères de genre quelconque.

La propriété 5.1.1 montre qu'un groupe de généralisation dont l'enregistrement de référence est $(v_1, \dots, v_m)$ pour $m \in \mathbb{N}^*$ peut s'écrire comme l'intersection des groupes de généralisation dont les enregistrements de référence sont les coordonnées de $(v_1, \dots, v_m)$. En d'autres termes, pour calculer $G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})}$, nous pouvons calculer les groupes de généralisation $G_{(v_1)}, \dots, G_{(v_m)}$ et en prendre l'intersection. Cette propriété sera utilisée pour démontrer une inclusion dans la remarque 5.1.7 page 97.

Propriété 5.1.1
Soient $\mathcal{Q} = \{Q_1, \dots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \dots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$.

Soit $(v^1_{l_1,p_1}, \dots, v^m_{l_m,p_m})$ un enregistrement sur $(\mathcal{Q}, \mathcal{H})$ avec $v^j_{l_j,p_j}$ le $p_j{}^{\text{e}}$ nœud du niveau l_j de la hiérarchie H_j pour $l_j \in [\![0, h_{H_j} - 1]\!]$, $p_j \in [\![1, \text{nn}(l_j)]\!]$ pour tout $j \in [\![1, m]\!]$.

Soient $G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})} \in \mathcal{G}_{\mathcal{H}}(T)$ et $G_{(v^j_{l_j,p_j})} \in \mathcal{G}_{H_j}(T)$ pour tout $j \in [\![1, m]\!]$.

On a

$$G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})} = \bigcap_{j=1}^{m} G_{(v^j_{l_j,p_j})}.$$

Ainsi,

$$\mathcal{G}_{\mathcal{H}}(T) = \{G_1 \cap \dots \cap G_m \neq \emptyset \text{ avec } G_j \in \mathcal{G}_{H_j}(T)\ \forall\, j \in [\![1, m]\!]\}.$$

Démonstration : Montrons la première égalité de la propriété par double inclusion. Soit $E = (e_1, \dots, e_m)$ un enregistrement de T.

$\subseteq$. Montrons que $G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})} \subseteq \bigcap_{j=1}^m G_{(v^j_{l_j,p_j})}$. Si $E \in G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})}$, alors $\forall j \in [\![1, m]\!]$, $v^j_{l_j,p_j}$ est une généralisation de e_j. Donc, $\forall j \in [\![1, m]\!]$, $E \in G_{(v^j_{l_j,p_j})}$ c'est-à-dire $E \in \bigcap_{j=1}^m G_{(v^j_{l_j,p_j})}$.

$\supseteq$. Montrons que $\bigcap_{j=1}^m G_{(v^j_{l_j,p_j})} \subseteq G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})}$. Si $E \in \bigcap_{j=1}^m G_{(v^j_{l_j,p_j})}$ alors $\forall j \in [\![1, m]\!]$, $E \in G_{(v^j_{l_j,p_j})}$. Donc, $\forall j \in [\![1, m]\!]$, $v^j_{l_j,p_j}$ est une généralisation de e_j. Par définition de $G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})}$, on a $E \in G_{(v^1_{l_1,p_1},\dots,v^m_{l_m,p_m})}$.
■

Dans l'exemple 5.1.2 page 90, nous avons déterminé que le groupe de généralisation $G_{(a_1,b_{1,2})}$ correspond à l'ensemble d'enregistrements $\{E^1\}$. Rappelons que la table T est composée des enregistrements $E^1 = (a_1, b_1)$ et $E^2 = (a_2, b_2)$. Le groupe de généralisation $G_{(a_1)}$ contient les enregistrements de T dont la valeur pour l'attribut Q_1 peut se généraliser en a_1. Seul E^1 remplit cette condition donc $G_{(a_1)} = \{E^1\}$. Le groupe de généralisation $G_{(b_{1,2})}$ contient les enregistrements de T dont la valeur pour l'attribut Q_2 peut se généraliser en $b_{1,2}$. E^1 et E^2 remplissent cette condition donc $G_{(b_{1,2})} = \{E^1, E^2\}$. Nous obtenons donc que $G_{(a_1)} \cap G_{(b_{1,2})} = \{E^1\} \cap \{E^1, E^2\} = \{E^1\} = G_{(a_1,b_{1,2})}$. En reprenant les valeurs plus concrètes, déterminer les enregistrements dans le groupe de généralisation $G_{(Chat,Inconnu)}$ revient à chercher les enregistrements de la table qui sont des *Chats* et de genre quelconque donc les enregistrements dans $G_{(Chat)} \cap G_{(Inconnu)}$.

La remarque 5.1.3 donne une borne supérieure au cardinal de l'ensemble des groupes de généralisation d'une table sur un uplet de hiérarchies de généralisation. Le nombre maximal de groupes de généralisation non vides est le produit des nombres de nœuds des hiérarchies. Si $(H_1, \dots, H_m)$ est un m-uplet de hiérarchies avec $m \in \mathbb{N}^*$, il s'agit du nombre d'enregistrements $(v_1, \dots, v_m)$ possibles avec $v_j \in H_j$ pour tout $j \in [\![1, m]\!]$.

Remarque 5.1.3 (Cardinal de $\mathcal{G}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \dots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \dots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Le cardinal de $\mathcal{G}_{\mathcal{H}}(T)$ vérifie l'inégalité suivante :

$$1 \leqslant |\mathcal{G}_{\mathcal{H}}(T)| \leqslant \prod_{j=1}^{m} |\mathcal{N}_{H_j}|,$$

avec $\mathcal{N}_{H_j}$ l'ensemble des nœuds de H_j.

L'inégalité $1 \leqslant |\mathcal{G}_{\mathcal{H}}(T)|$ vient de la remarque 5.1.2 page précédente.

Dans l'exemple 5.1.2 page 90, la hiérarchie H_1 est constituée des nœuds a_1, a_2, a_3, $a_{1,2}$ et $a_{1,2,3}$. Nous avons donc $|\mathcal{N}_{H_1}| = 5$. De plus, la hiérarchie H_2 est constituée des nœuds b_1, b_2 et $b_{1,2}$. Nous avons donc $|\mathcal{N}_{H_2}| = 3$. Le cardinal de $\mathcal{G}_{\mathcal{H}}(T)$ est donc majoré par $5 \times 3 = 15$. Dans les faits, seuls 10 groupes de généralisation ne sont pas vides.

5.1.2 Ensemble minimal de groupes de généralisation

Dans l'ensemble des groupes de généralisation d'une table, tous les groupes de généralisation ne sont pas bons à garder.

Dans l'exemple 5.1.2 page 90, le groupe de généralisation $G_{(a_{1,2,3},b_{1,2})}$ n'est pas utile à considérer car le groupe de généralisation $G_{(a_{1,2},b_{1,2})}$ correspond exactement au même ensemble d'enregistrements que $G_{(a_{1,2,3},b_{1,2})}$, $\{E^1, E^2\}$, et les généralisations de $(a_{1,2}, b_{1,2})$ sont moins élevées que les généralisations de $(a_{1,2,3}, b_{1,2})$. Il ne sert à rien de savoir que les enregistrements E^1 et E^2 peuvent se généraliser en $(a_{1,2,3}, b_{1,2})$ alors que ce sont les seuls à pouvoir se généraliser en $(a_{1,2}, b_{1,2})$. En reprenant les valeurs plus concrètes, il ne sert à rien de savoir que les deux enregistrements sont des mammifères de genre inconnu puisque ce sont plus précisément des félins de genre inconnu.

Il est donc nécessaire de faire un tri dans les groupes de généralisation pour ne garder que ceux qui contiennent une information pertinente. Nous appellerons le sous-ensemble de groupes de généralisation obtenu l'*ensemble minimal des groupes de généralisation*.

Dans cette section, nous allons définir l'ensemble minimal des groupes de généralisation. Dans la section 5.1.2.1, grâce à une relation d'équivalence sur l'ensemble de groupes de généralisation (cf. définition 5.1.2) et à une relation d'ordre sur chaque classe de la précédente relation d'équivalence (cf. définition 5.1.4 page suivante), nous donnerons une méthode de construction de l'ensemble minimal des groupes de généralisation en définition 5.1.6 page 96. Afin de rendre l'utilisation de l'ensemble minimal des groupes de généralisation plus simple, nous donnerons dans la section 5.1.2.2 page 96 une caractérisation de ses éléments en définition 5.1.8 page 96. Nous conclurons cette section par un exemple (cf. exemple 5.1.3 page 97).

5.1.2.1 Construction de l'ensemble minimal des groupes de généralisation

Pour construire l'ensemble minimal des groupes de généralisation, commençons par définir une relation sur l'ensemble des groupes de généralisation qui réunira les groupes de généralisation correspondant au même sous-ensemble d'enregistrements de la table.

Définition 5.1.2 (Relation $\equiv$ sur $\mathcal{G}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$.

Soit $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Soient Γ et Γ' deux groupes de généralisation de $\mathcal{G}_{\mathcal{H}}(T)$.

On dit que Γ et Γ' sont *congruents*, noté $\Gamma \equiv \Gamma'$, si les deux ensembles d'enregistrements correspondants sont égaux.

Par exemple, les groupes de généralisation $G_{(a_{1,2},b_{1,2})}$ et $G_{(a_{1,2,3},b_{1,2})}$ de l'exemple 5.1.2 page 90 sont conguents car ils correspondent tous deux à l'ensemble d'enregistrements $\{E^1, E^2\}$.

Propriété 5.1.2 ($\equiv$ relation d'équivalence sur $\mathcal{G}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

La relation $\equiv$ est une relation d'équivalence sur $\mathcal{G}_{\mathcal{H}}(T)$. On note $\mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$ l'ensemble des classes d'équivalence de la relation $\equiv$ sur $\mathcal{G}_{\mathcal{H}}(T)$.

Démonstration : Il faut montrer que la relation $\equiv$ est réflexive, symétrique et transitive sur $\mathcal{G}_{\mathcal{H}}(T)$.

Réflexive. Soit $\Gamma \in \mathcal{G}_{\mathcal{H}}(T)$. On a $\Gamma \equiv \Gamma$ car l'application $id : \Gamma \to \Gamma$ qui à tout enregistrement $E \in \Gamma$ associe E est bijective.

Symétrique. Soient $\Gamma, \Gamma' \in \mathcal{G}_{\mathcal{H}}(T)$. Si $\Gamma \equiv \Gamma'$ alors Γ et Γ' correspondent au même ensemble d'enregistrements. Ainsi, l'application bijective $id : \Gamma \to \Gamma'$ qui à tout enregistrement $E \in \Gamma$ associe $E \in \Gamma'$ donne $\Gamma' \equiv \Gamma$.

Transitive. Soient $\Gamma, \Gamma', \Gamma'' \in \mathcal{G}_{\mathcal{H}}(T)$. Si $\Gamma \equiv \Gamma'$ et $\Gamma' \equiv \Gamma''$ alors on peut construire deux bijections $\varphi : \Gamma \to \Gamma'$ et $\varphi' : \Gamma' \to \Gamma''$. Comme la composée de deux bijections est une bijection, $\varphi' \circ \varphi : \Gamma \to \Gamma''$ est une bijection. On a donc $\Gamma \equiv \Gamma''$. ■

La définition de l'égalité sur l'ensemble des groupes de généralisation doit être clarifiée. En effet, dire que deux groupes de généralisation sont congruents ne signifie pas qu'ils sont égaux. L'hypothèse de congruence est nécessaire mais pas suffisante. La proposition 5.1.1 page suivante va nous permettre de définir la notion d'égalité de deux groupes de généralisation (cf. définition 5.1.3 page suivante).

Proposition 5.1.1
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Soient $(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})$ et $(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})$ deux enregistrements sur $(\mathcal{Q}, \mathcal{H})$ avec $v^j_{l_j,p_j}$ ($v^j_{l'_j,p'_j}$ le $p_j{}^{\text{e}}$ ($p'_j{}^{\text{e}}$) nœud du niveau l_j (l'_j) de la hiérarchie H_j pour $l_j, l'_j \in [\![0, h_{H_j} - 1]\!]$, $p_j \in [\![1, \text{nn}(l_j)]\!]$ et $p'_j \in [\![1, \text{nn}(l'_j)]\!]$ pour tout $j \in [\![1, m]\!]$.

Soient $G_{(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})}$ et $G_{(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})}$ deux groupes de généralisation de $\mathcal{G}_{\mathcal{H}}(T)$.

Si $G_{(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})} \equiv G_{(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})}$ et $l_j = l'_j$ pour tout $j \in [\![1, m]\!]$ alors $v^j_{l_j,p_j} = v^j_{l'_j,p'_j}$ pour tout $j \in [\![1, m]\!]$.

Démonstration : Soient $(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})$ et $(v^1_{l'_1,p'_1}, \ldots, v^m_{l_m,p_m})$ deux enregistrements sur $(\mathcal{Q}, \mathcal{H})$ avec $l_j, l'_j \in [\![0, h_{H_j} - 1]\!]$, $p_j \in [\![1, \text{nn}(l_j)]\!]$ et $p'_j \in [\![1, \text{nn}(l'_j)]\!]$ pour tout $j \in [\![1, m]\!]$.

Soient $\Gamma := G_{(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})}$ et $\Gamma' := G_{(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})}$ deux groupes de généralisation de $\mathcal{G}_{\mathcal{H}}(T)$ tels que $\Gamma \equiv \Gamma'$ et $l_j = l'_j$ pour tout $j \in [\![1, m]\!]$. Posons $E = \{E^{s_1}, \ldots, E^{s_q}\}$ l'ensemble d'enregistrements correspondant à Γ et Γ' avec $q \in [\![1, n]\!]$, $s_i \in [\![1, n]\!]$ et $E^{s_i} = (e^{s_i}_1, \ldots, e^{s_i}_m)$ pour tout $i \in [\![1, q]\!]$.

Pour tout $i \in [\![1, q]\!]$, pour tout $j \in [\![1, m]\!]$, $e^{s_i}_j$ peut se généraliser en $v^j_{l_j,p_j}$, d'après la définition de Γ, et $e^{s_i}_j$ peut se généraliser en $v^j_{l'_j,p'_j}$, d'après la définition de Γ'. Or, on a supposé que $l_j = l'_j$ pour tout $j \in [\![1, m]\!]$ donc $\forall i \in [\![1, q]\!]$, $\forall j \in [\![1, m]\!]$, $e^{s_i}_j$ peut se généraliser en $v^j_{l_j,p_j}$ et en $v^j_{l_j,p'_j}$ c'est-à-dire $v^j_{l_j,p_j}$ et $v^j_{l_j,p'_j}$ sont sur le chemin de $e^{s_i}_j$ à la racine r_j de H_j. Par définition d'une arborescence, le chemin d'un nœud à la racine est unique donc un nœud n'a au maximum qu'un ancêtre par niveau de la hiérachie. Ainsi, $v^j_{l_j,p_j} = v^j_{l'_j,p'_j}$ pour tout $j \in [\![1, m]\!]$. ■

Définition 5.1.3 (Égalité sur $\mathcal{G}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Pour tout $j \in [\![1, m]\!]$, soient $l_j, l'_j \in [\![0, h_{H_j} - 1]\!]$, $p_j \in [\![1, \text{nn}(l_j)]\!]$ et $p'_j \in [\![1, \text{nn}(l'_j)]\!]$. Soient $\Gamma := G_{(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})}$ et $\Gamma' := G_{(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})}$ deux groupes de généralisation de $\mathcal{G}_{\mathcal{H}}(T)$.

On dit que Γ et Γ' sont *égaux*, noté $\Gamma = \Gamma'$, si $\Gamma \equiv \Gamma'$ et $l_j = l'_j$ pour tout $j \in [\![1, m]\!]$.

La proposition 5.1.1 nous donne que l'égalité de groupes de généralisation sur $\mathcal{G}_{\mathcal{H}}(T)$ est bien définie.

Nous avons réuni les groupes de généralisation en fonction de l'ensemble d'enregistrements auxquels ils correspondent grâce à la relation d'équivalence $\equiv$. Nous voulons maintenant ordonner les groupes de généralisation d'une même classe d'équivalence de la relation $\equiv$. Au sein d'une classe d'équivalence de $\equiv$, nous allons nous intéresser aux niveaux de généralisation des enregistrements de référence des groupes de généralisation. En comparant les niveaux de généralisation de deux groupes de généralisation, nous allons déterminer si l'un des groupes est plus petit que l'autre ou s'ils sont incomparables. La définition 5.1.4 explicite une relation à mettre sur chaque classe de la relation $\equiv$.

Définition 5.1.4 (Relation $\leqslant$ sur chaque classe de $\mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$.

Soient $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$ et $\mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$ l'ensemble des classes d'équivalence de la relation $\equiv$ sur $\mathcal{G}_{\mathcal{H}}(T)$.

Sur chaque classe $C^{\equiv}$ de $\mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$, on définit la relation $\leqslant$ comme suit :

$$G_{(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})} \leqslant G_{(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})} \Leftrightarrow l_j \leqslant l'_j \text{ pour tout } j \in [\![1, m]\!],$$

avec $G_{(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})}$ et $G_{(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})}$ dans $C^{\equiv}$, $(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})$ et $(v^1_{l'_1,p'_1}, \ldots, v^m_{l'_m,p'_m})$ deux enregistrements sur $(\mathcal{Q}, \mathcal{H})$ avec $v^j_{l_j,p_j}$ ($v^j_{l'_j,p'_j}$ le $p_j{}^{\text{e}}$ ($p'_j{}^{\text{e}}$) nœud du niveau l_j (l'_j) de la hiérarchie H_j pour $l_j, l'_j \in [\![0, h_{H_j} - 1]\!]$, $p_j \in [\![1, \text{nn}(l_j)]\!]$ et $p'_j \in [\![1, \text{nn}(l'_j)]\!]$ pour tout $j \in [\![1, m]\!]$.

Remarque 5.1.4 ($\Gamma \not\leqslant \Gamma'$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soient $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Soit $C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$. Soient $\Gamma, \Gamma' \in C^{\equiv}$. Dire que $\Gamma \not\leqslant \Gamma'$ signifie qu'il existe $j \in [\![1, m]\!]$ tel que $l_j > l'_j$, c'est-à-dire $\Gamma' \leqslant \Gamma$ ou Γ et Γ' sont incomparables.

De plus, $\neg(\Gamma \not\leqslant \Gamma')$ signifie que $\forall\, j \in [\![1, m]\!]$, $l_j \leqslant l'_j$, c'est-à-dire $\Gamma \leqslant \Gamma'$.

Dans l'exemple 5.1.2 page 90, les groupes de généralisation $G_{(a_1,b_1)}$, $G_{(a_1,b_{1,2})}$ et $G_{(a_{1,2},b_1)}$ sont dans la même classe d'équivalence de la relation $\equiv$. En effet, ils correspondent à l'ensemble d'enregistrements $\{E^1\}$. A partir de la définition 5.1.4 page précédente, nous pouvons dire par exemple que $G_{(a_1,b_1)} \leqslant G_{(a_1,b_{1,2})}$ car le niveau de b_1 dans la hiérarchie H_2 est inférieur au niveau de $b_{1,2}$ dans H_2. Les groupes $G_{(a_1,b_{1,2})}$ et $G_{(a_{1,2},b_1)}$ sont incomparables car le niveau de a_1 dans H_1 est inférieur au niveau de $a_{1,2}$ dans H_1 mais le niveau de $b_{1,2}$ dans H_2 est supérieur au niveau de b_1.

Propriété 5.1.3 ($\leqslant$ relation d'ordre)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soient $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

La relation $\leqslant$ est une relation d'ordre sur $C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$.

Démonstration : Il faut montrer que la relation $\leqslant$ est réflexive, antisymétrique et transitive sur $C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$.

Réflexive. Soit $\Gamma = G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})} \in C^{\equiv}$. On a $l_j = l_j$ pour tout $j \in [\![1, m]\!]$ donc $\Gamma \leqslant \Gamma$.

Antisymétrique. Soient $\Gamma = G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})}$ et $\Gamma' = G_{(v^1_{l'_1,p'_1},\ldots,v^m_{l'_m,p'_m})}$ dans $C^{\equiv}$. Si $\Gamma \leqslant \Gamma'$ et $\Gamma' \leqslant \Gamma$ alors $l_j \leqslant l'_j$ et $l'_j \leqslant l_j$ pour tout $j \in [\![1, m]\!]$ donc $l_j = l'_j$. De plus, $\Gamma \equiv \Gamma'$ (car ils appartiennent à $C^{\equiv}$) donc $\Gamma = \Gamma'$ d'après la définition 5.1.3.

Transitive. Soient $\Gamma = G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})}, \Gamma' = G_{(v^1_{l'_1,p'_1},\ldots,v^m_{l'_m,p'_m})}$ et $\Gamma'' = G_{(v^1_{l''_1,p''_1},\ldots,v^m_{l''_m,p''_m})}$ dans $C^{\equiv}$. Si $\Gamma \leqslant \Gamma'$ et $\Gamma' \leqslant \Gamma''$ alors $l_j \leqslant l'_j$ et $l'_j \leqslant l''_j$ pour tout $j \in [\![1, m]\!]$. Cela implique que $l_j \leqslant l''_j$ pour tout $j \in [\![1, m]\!]$ donc $\Gamma \leqslant \Gamma''$. ■

Nous définissons maintenant les classes et représentants de classe de la relation d'ordre $\leqslant$ (cf. définition 5.1.5).

Définition 5.1.5 (Classes et représentants de classe de $\leqslant$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soient $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

Soit $C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$.

Pour tout $\Gamma \in C^{\equiv}$, on définit sa classe pour la relation $\leqslant$ par

$$C^{\leqslant}(\Gamma) = \begin{cases} \emptyset & \text{s'il existe } \Gamma' \in C^{\equiv} - \Gamma \text{ tel que } \Gamma' \leqslant \Gamma, \\ \{\Gamma' \in C^{\equiv} \,:\, \Gamma \leqslant \Gamma'\} & \text{sinon.} \end{cases}$$

On note $\mathcal{C}^{\leqslant}(C^{\equiv}) = \{C^{\leqslant}(\Gamma) \neq \emptyset \,:\, \Gamma \in C^{\equiv}\}$ l'ensemble des classes non vides de la relation $\leqslant$ sur $C^{\equiv}$.

Pour $C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})$, le *représentant de* $C^{\leqslant}$, noté $\text{repr}(C^{\leqslant})$, est le groupe de généralisation Γ de $C^{\leqslant}$ tel que, pour tout $\Gamma' \in C^{\leqslant}$, $\Gamma \leqslant \Gamma'$.

Remarque 5.1.5
Reprenons les notations de la définition 5.1.5.

Pour $\Gamma \in C^{\equiv}$ et $C^{\leqslant}(\Gamma) \in \mathcal{C}^{\leqslant}(C^{\equiv})$ la classe de Γ pour la relation $\leqslant$ dans $C^{\equiv}$, on a $\text{repr}(C^{\leqslant}(\Gamma)) = \Gamma$.

Réciproquement, si $\Gamma \in C^{\equiv}$ et s'il existe $C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})$ telle que $\Gamma = \text{repr}(C^{\leqslant})$ alors $C^{\leqslant} = C^{\leqslant}(\Gamma)$.

De plus, si $\Gamma \in C^{\equiv}$ et s'il existe $C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})$ telle que $\Gamma \in C^{\leqslant}$, alors $\text{repr}(C^{\leqslant}) \leqslant \Gamma$.

Dans l'exemple 5.1.2 page 90, $G_{(a_{1,2},b_{1,2})}$ et $G_{(a_{1,2,3},b_{1,2})}$ forment une classe d'équivalence de la relation $\equiv$. Nous observons que $G_{(a_{1,2},b_{1,2})} \leqslant G_{(a_{1,2,3},b_{1,2})}$. Avec la définition 5.1.5, nous déterminons les classes pour la relation $\leqslant$ de $G_{(a_{1,2},b_{1,2})}$ et $G_{(a_{1,2,3},b_{1,2})}$. Comme $G_{(a_{1,2},b_{1,2})} \leqslant G_{(a_{1,2,3},b_{1,2})}$, la classe de $G_{(a_{1,2},b_{1,2})}$ pour la relation $\leqslant$ est $C^{\leqslant}(G_{(a_{1,2},b_{1,2})}) = \{G_{(a_{1,2},b_{1,2})}, G_{(a_{1,2,3},b_{1,2})}\}$ et celle de $G_{(a_{1,2,3},b_{1,2})}$ est $C^{\leqslant}(G_{(a_{1,2,3},b_{1,2})}) = \emptyset$. Le représentant de $C^{\leqslant}(G_{(a_{1,2},b_{1,2})})$ est $G_{(a_{1,2},b_{1,2})}$.

Grâce à la relation d'équivalence $\equiv$ sur l'ensemble des groupes de généralisation et à la relation d'ordre $\leqslant$ sur chaque classe d'équivalence de $\equiv$, nous pouvons définir l'ensemble minimal des groupes de généralisation (cf. définition 5.1.6 page suivante).

Définition 5.1.6 (Ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$.

On définit $\overline{\mathcal{G}}_{\mathcal{H}}(T)$, l'ensemble *minimal* des groupes de généralisation de T sur $\mathcal{Q}$, comme suit :

$$\overline{\mathcal{G}}_{\mathcal{H}}(T) = \bigcup_{C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))} \{\text{repr}(C^{\leqslant}) \,:\, C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})\}.$$

Parmi les groupes de généralisation dans $\mathcal{G}_{\mathcal{H}}(T)$ correspondant au même ensemble d'enregistrements, ne sont présents dans $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ que les groupes de généralisation correspondant aux généralisations les moins importantes.

Remarque 5.1.6
Reprenons les notations de la définition 5.1.6.

D'après la définition 5.1.6 de l'ensemble minimal des groupes de généralisation, si $\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T)$ alors il existe $C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$ telle qu'il existe $C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})$ telle que Γ soit le représentant de $C^{\leqslant}$ pour la relation $\leqslant$.

D'après la remarque 5.1.5 sur les classes de la relation $\leqslant$, Γ est le représentant de $C^{\leqslant}(\Gamma)$ donc $C^{\leqslant}(\Gamma) = C^{\leqslant}$. De plus, d'après la définition 5.1.5 page précédente de $\mathcal{C}^{\leqslant}(C^{\equiv})$, si $C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})$ alors $C^{\leqslant} \neq \emptyset$.

Nous en déduisons donc que

$$\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T) \Rightarrow C^{\leqslant}(\Gamma) \neq \emptyset,$$

avec $C^{\leqslant}(\Gamma) \in \mathcal{C}^{\leqslant}(C^{\equiv})$ et $C^{\equiv}$ la classe d'équivalence de la relation $\equiv$ de Γ.

À chaque groupe de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$, nous pouvons associer un unique enregistrement généralisé sur $(\mathcal{Q}, \mathcal{H})$ correspondant au résultat de la fusion de tous les enregistrements du groupe de généralisation. Cet enregistrement est appelé *représentant* du groupe de généralisation.

Définition 5.1.7 (Représentant d'un groupe de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$.

On définit l'application repr : $\overline{\mathcal{G}}_{\mathcal{H}}(T) \to \mathcal{F}_{(\mathcal{Q},\mathcal{H})}$ qui à tout groupe de généralisation $\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T)$ associe son *représentant* repr(Γ) défini par :

$$\text{repr}(\Gamma) = \overline{gen}(\Gamma)\,.$$

Dans l'exemple 5.1.2 page 90, le groupe de généralisation $G_{(a_{1,2},b_{1,2})}$ correspondant à l'ensemble d'enregistrements $\{E^1, E^2\}$. Ce groupe appartient à l'ensemble minimal des groupes de généralisation (une justification sera apportée dans l'exemple 5.1.3 page suivante). Sachant que $E^1 = (a_1, b_1)$ et $E^2 = (a_2, b_2)$, le représentant de $G_{(a_{1,2},b_{1,2})}$ est donc $\overline{gen}\big(G_{(a_{1,2},b_{1,2})}\big) = \overline{gen}\big(\{E^1, E^2\}\big) = (\text{LCA}(a_1, a_2), \text{LCA}(b_1, b_2)) = (a_{1,2}, b_{1,2})$ d'après la définition 2.3.8 page 21 de $\overline{gen}$.

5.1.2.2 Caractérisation des éléments de l'ensemble minimal des groupes de généralisation et exemple

La définition 5.1.6 par construction de l'ensemble minimal des groupes de généralisation n'est pas évidente à utiliser en pratique. Nous allons donc proposer une caractérisation des éléments de l'ensemble minimal des groupes de généralisation en définition 5.1.8 et nous démontrerons l'équivalence des deux définitions.

Définition 5.1.8 (Caractérisation des éléments de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soient $\mathcal{G}_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$ et $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$.

On a l'équivalence suivante

$$\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T) \Leftrightarrow \forall\, \Gamma' \in \mathcal{G}_{\mathcal{H}}(T), \Gamma' \not\equiv \Gamma \text{ ou } \Gamma' \not\leqslant \Gamma.$$

Démonstration : On montre l'équivalence des deux définitions.

$\Rightarrow$. Supposons $\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T)$. Soit $C^{\equiv}$ la classe d'équivalence de Γ pour la relation $\equiv$. D'après la remarque 5.1.6, on a $C^{\leqslant}(\Gamma) \neq \emptyset$ avec $C^{\leqslant}(\Gamma) \in \mathcal{C}^{\leqslant}(C^{\equiv})$.

Montrons que $\forall\, \Gamma' \in \mathcal{G}_{\mathcal{H}}(T), \Gamma' \not\equiv \Gamma$ ou $\Gamma' \not\leqslant \Gamma$. Soit $\Gamma' \in \mathcal{G}_{\mathcal{H}}(T)$.

Dans un premier temps, supposons que $\Gamma' \equiv \Gamma$ et montrons que $\Gamma' \not\leqslant \Gamma$. Par l'absurde, supposons $\neg(\Gamma' \not\leqslant \Gamma)$ c'est-à-dire $\Gamma' \leqslant \Gamma$ d'après la remarque 5.1.4. Comme $\Gamma' \equiv \Gamma$, $\Gamma' \in C^{\equiv}$. Comme $\Gamma' \leqslant \Gamma$, $C^{\leqslant}(\Gamma) = \emptyset$ d'après la définition 5.1.5 des classes de la relation $\leqslant$ dans $C^{\equiv}$. Cela contredit notre hypothèse $C^{\leqslant}(\Gamma) \neq \emptyset$.

Dans un second temps, supposons que $\Gamma' \leqslant \Gamma$ et montrons que $\Gamma' \not\equiv \Gamma$. Par l'absurde, supposons $\Gamma' \equiv \Gamma$. On a $\Gamma' \in C^{\equiv}$. Comme $\Gamma' \leqslant \Gamma$, $C^{\leqslant}(\Gamma) = \emptyset$ d'après la définition 5.1.5 des classes de la relation $\leqslant$ dans $C^{\equiv}$. Cela contredit notre hypothèse $C^{\leqslant}(\Gamma) \neq \emptyset$.

$\Leftarrow$. Montrons la seconde implication par contraposée. Soit $\Gamma \in \mathcal{G}_{\mathcal{H}}(T)$. Supposons que $\forall\, C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$, $\forall\, C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})$, Γ ne soit pas le représentant de $C^{\leqslant}$ pour la relation $\leqslant$. Montrons qu'il existe $\Gamma' \in \mathcal{G}_{\mathcal{H}}(T)$ tel que $\Gamma' \equiv \Gamma$ et $\Gamma' \leqslant \Gamma$.

Soit $C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$ la classe d'équivalence de Γ pour la relation $\equiv$. Dire que Γ n'est le représentant d'aucune classe de la relation $\leqslant$ dans $C^{\equiv}$ signifie que $C^{\leqslant}(\Gamma) \notin \mathcal{C}^{\leqslant}(C^{\equiv})$ d'après la remarque 5.1.5 et donc $C^{\leqslant}(\Gamma) = \emptyset$. Si $C^{\leqslant}(\Gamma) = \emptyset$ alors il existe $\Gamma' \in C^{\equiv} - \Gamma$, c'est-à-dire $\Gamma' \equiv \Gamma$, tel que $\Gamma' \leqslant \Gamma$ d'après la définition 5.1.5 page 95 des classes de la relation $\leqslant$. ■

Remarque 5.1.7
En reprenant les notations de la définition 5.1.8 page ci-contre, on a

$$\overline{\mathcal{G}}_{\mathcal{H}}(T) \subset \{G_1 \cap \cdots \cap G_m \neq \emptyset \text{ avec } G_j \in \overline{\mathcal{G}}_{H_j}(T) \text{ pour tout } j \in [\![1, m]\!]\},$$

mais l'inclusion inverse est fausse en général (cf. exemple 5.1.3).

En posant $X := \{G_1 \cap \cdots \cap G_m \neq \emptyset \text{ avec } G_j \in \overline{\mathcal{G}}_{H_j}(T) \text{ pour tout } j \in [\![1, m]\!]\}$, nous pourrions montrer que :

$$\overline{\mathcal{G}}_{\mathcal{H}}(T) = \overline{X}.$$

Démonstration : Montrons la première inclusion de la remarque 5.1.7.

Posons $X = \{G_1 \cap \cdots \cap G_m \neq \emptyset \text{ avec } G_j \in \overline{\mathcal{G}}_{H_j}(T) \text{ pour tout } j \in [\![1, m]\!]\}$. Montrons que $\overline{\mathcal{G}}_{\mathcal{H}}(T) \subset X$.

Soit $\Gamma = G_{(v^1_{l_1,p_1},\ldots,v^m_{l_m,p_m})} \in \overline{\mathcal{G}}_{\mathcal{H}}(T)$ avec $(v^1_{l_1,p_1}, \ldots, v^m_{l_m,p_m})$ un enregistrement sur $(\mathcal{Q}, \mathcal{H})$ avec $v^j_{l_j,p_j}$ le $p_j{}^{\text{e}}$ nœud du niveau l_j de la hiérarchie H_j pour $l_j \in [\![0, h_{H_j} - 1]\!]$ et $p_j \in [\![1, \text{nn}(l_j)]\!]$ pour tout $j \in [\![1, m]\!]$.

D'après la caractérisation des éléments de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ de la définition 5.1.8 page précédente, on a $\forall\, \Gamma' \in \mathcal{G}_{\mathcal{H}}(T)$, $\Gamma' \not\equiv \Gamma$ ou $\Gamma' \not\leqslant \Gamma$.

D'après la propriété 5.1.1 et $\overline{\mathcal{G}}_{\mathcal{H}}(T) \subseteq \mathcal{G}_{\mathcal{H}}(T)$, on a

$$\Gamma = \cap_{j=1}^{m} G_{(v^j_{l_j,p_j})},$$

pour $G_{(v^j_{l_j,p_j})} \in \mathcal{G}_{H_j}(T)$.

Il faut montrer que $G_{(v^j_{l_j,p_j})} \in \overline{\mathcal{G}}_{H_j}(T)$ pour tout $j \in [\![1, m]\!]$, c'est-à-dire $G_{(v^j_{l_j,p_j})}$ vérifie : $\forall\, G_{(v^j_{l'_j,p'_j})} \in \mathcal{G}_{H_j}(T)$, $G_{(v^j_{l'_j,p'_j})} \not\equiv G_{(v^j_{l_j,p_j})}$ ou $G_{(v^j_{l'_j,p'_j})} \not\leqslant G_{(v^j_{l_j,p_j})}$ pour tout $j \in [\![1, m]\!]$.

Par l'absurde, supposons qu'il existe $s \in [\![1, m]\!]$ tel que $G_{(v^s_{l_s,p_s})} \notin \overline{\mathcal{G}}_{H_s}(T)$, c'est-à-dire il existe $G_{(v^s_{l'_s,p'_s})} \in \mathcal{G}_{H_s}(T)$ tel que $G_{(v^s_{l'_s,p'_s})} \equiv G_{(v^s_{l_s,p_s})}$ et $G_{(v^s_{l'_s,p'_s})} \leqslant G_{(v^s_{l_s,p_s})}$.

On construit le groupe de généralisation

$$V = G_{(v^1_{l'_1,p'_1},\ldots,v^m_{l'_m,p'_m})} = \cap_{j=1}^{m} G_{(v^j_{l'_j,p'_j})},$$

avec $G_{(v^j_{l'_j,p'_j})} = G_{(v^j_{l_j,p_j})}$ pour tout $j \in [\![1, m]\!]$ et $j \neq s$.

Pour tout $j \in [\![1, m]\!]$, on a $G_{(v^j_{l'_j,p'_j})} \equiv G_{(v^j_{l_j,p_j})}$ donc $\cap_{j=1}^{m} G_{(v^j_{l'_j,p'_j})} \equiv \cap_{j=1}^{m} G_{(v^j_{l_j,p_j})}$. Ainsi, $V \equiv \Gamma \neq \emptyset$ et donc $V \in \mathcal{G}_{\mathcal{H}}(T)$.

De plus, pour tout $j \in [\![1, m]\!]$, $j \neq s$, $l'_j = l_j$ par construction de V et $G_{(v^s_{l'_s,p'_s})} \leqslant G_{(v^s_{l_s,p_s})}$ c'est-à-dire $l'_s \leqslant l_s$ par l'hypothèse de l'absurde, donc pour tout $j \in [\![1, m]\!]$, $l'_j \leqslant l_j$. Ainsi, $V \leqslant \Gamma$ d'après la définition 5.1.4 page 94.

On a construit un élément V de $\mathcal{G}_{\mathcal{H}}(T)$ tel que $V \equiv \Gamma$ et $V \leqslant \Gamma$. C'est en contradiction avec l'hypothèse $\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T)$ donc pour tout $j \in [\![1, m]\!]$, $G_{(v^j_{l_j,p_j})} \in \overline{\mathcal{G}}_{H_j}(T)$. Ainsi, $\Gamma \in X$. ■

Exemple 5.1.3
Soient $\mathcal{Q} = \{Q_1, Q_2\}$ un ensemble d'attributs quasi-identifiants avec $Q_1 = \{a_1, a_2, a_3\}$ et $Q_2 = \{b_1, b_2\}$.

T	Q_1	Q_2
E^1	a_1	b_1
E^2	a_2	b_2

FIGURE 5.5 – Représentation d'une table sur $\mathcal{Q}$.

Considérons $\mathcal{H} = (H_1, H_2)$ un couple de hiérarchies associées aux attributs de $\mathcal{Q}$ et décrites dans la figure 5.3. La hiérarchie H_1 est composée de trois feuilles (a_1, a_2 et a_3), d'un nœud de niveau 1 ($a_{1,2}$ qui est une généralisation de a_1 et a_2) et d'une racine ($a_{1,2,3}$). La hiérarchie H_2 est composée de deux feuilles (b_1 et b_2) et d'une racine ($b_{1,2}$).

Considérons la table sur $\mathcal{Q}$ décrite par la figure 5.5 et notée T. T a deux enregistrements $E^1 = (a_1, b_1)$ et $E^2 = (a_2, b_2)$.

Dans l'exemple 5.1.2, nous avons déterminé les groupes de généralisation de T sur $\mathcal{H}$: $G_{(a_1,b_1)} = \{E^1\}$, $G_{(a_1,b_{1,2})} = \{E^1\}$, $G_{(a_2,b_2)} = \{E^2\}$, $G_{(a_2,b_{1,2})} = \{E^2\}$, $G_{(a_{1,2},b_1)} = \{E^1\}$, $G_{(a_{1,2},b_2)} = \{E^2\}$, $G_{(a_{1,2},b_{1,2})} = T$, $G_{(a_{1,2,3},b_1)} = \{E^1\}$, $G_{(a_{1,2,3},b_2)} = \{E^2\}$ et $G_{(a_{1,2,3},b_{1,2})} = T$.

L'ensemble des groupes de généralisation de T sur $\mathcal{H}$ est donc

$$\mathcal{G}_{\mathcal{H}}(T) = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_1)}, G_{(a_{1,2},b_2)}, G_{(a_{1,2},b_{1,2})}, G_{(a_{1,2,3},b_1)}, G_{(a_{1,2,3},b_2)}, G_{(a_{1,2,3},b_{1,2})}\}.$$

Nous cherchons à présent à déterminer l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$, noté $\overline{\mathcal{G}}_{\mathcal{H}}(T)$.

Tout d'abord, nous quotientons $\mathcal{G}_{\mathcal{H}}(T)$ par la relation $\equiv$. Nous trouvons les trois classes d'équivalence suivantes :

— $C_1^{\equiv} = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_{1,2},b_1)}, G_{(a_{1,2,3},b_1)}\}$ correspondant à $\{E^1\}$,
— $C_2^{\equiv} = \{G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_2)}\}$ correspondant à $\{E^2\}$,
— $C_3^{\equiv} = \{G_{(a_{1,2},b_{1,2})}, G_{(a_{1,2,3},b_{1,2})}\}$ correspondant à T.

Ensuite, dans les classes de $\mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))$ contenant plus d'un élément, nous ordonnons les éléments par la relation $\leqslant$ et nous calculons la classe de chaque élément pour la relation $\leqslant$. Nous ne préciserons que les classes non vides.

Dans $C_1^{\equiv}$, nous avons $G_{(a_1,b_1)} \leqslant G_{(a_1,b_{1,2})}$, $G_{(a_1,b_1)} \leqslant G_{(a_{1,2},b_1)}$ et $G_{(a_1,b_1)} \leqslant G_{(a_{1,2,3},b_1)}$. De plus, $G_{(a_{1,2},b_1)} \leqslant G_{(a_{1,2,3},b_1)}$. $G_{(a_1,b_{1,2})}$ et $G_{(a_{1,2},b_1)}$ sont incomparables ainsi que $G_{(a_1,b_{1,2})}$ et $G_{(a_{1,2,3},b_1)}$. Nous obtenons donc une seule classe non vide :

$$C^{\leqslant}(G_{(a_1,b_1)}) = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_{1,2},b_1)}, G_{(a_{1,2,3},b_1)}\},$$

de représentant $G_{(a_1,b_1)}$.

Dans $C_2^{\equiv}$, nous avons $G_{(a_2,b_2)} \leqslant G_{(a_2,b_{1,2})}$ et $G_{(a_2,b_2)} \leqslant G_{(a_{1,2},b_2)}$. $G_{(a_2,b_{1,2})}$ et $G_{(a_{1,2},b_2)}$ sont incomparables. Nous obtenons donc une seule classe non vide :

$$C^{\leqslant}(G_{(a_2,b_2)}) = \{G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_2)}\},$$

de représentant $G_{(a_2,b_2)}$.

Dans $C_3^{\equiv}$, nous avons $G_{(a_{1,2},b_{1,2})} \leqslant G_{(a_{1,2,3},b_{1,2})}$. Nous obtenons une classe non vide :

$$C^{\leqslant}(G_{(a_{1,2},b_{1,2})}) = \{G_{(a_{1,2},b_{1,2})}, G_{(a_{1,2,3},b_{1,2})}\},$$

de représentant $G_{(a_{1,2},b_{1,2})}$.

Finalement, en appliquant la définition 5.1.6 page 96, nous obtenons l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$:

$$\begin{aligned}\overline{\mathcal{G}}_{\mathcal{H}}(T) &= \bigcup_{C^{\equiv} \in \mathcal{C}^{\equiv}(\mathcal{G}_{\mathcal{H}}(T))} \{\text{repr}(C^{\leqslant}) \,:\, C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})\} \\ &= \{\text{repr}(C^{\leqslant}(G_{(a_1,b_1)}))\} \cup \{\text{repr}(C^{\leqslant}(G_{(a_2,b_2)}))\} \cup \{\text{repr}(C^{\leqslant}(G_{(a_{1,2},b_{1,2})}))\} \\ &= \{G_{(a_1,b_1)}, G_{(a_2,b_2)}, G_{(a_{1,2},b_{1,2})}\}\end{aligned}$$

Une représentation de la composition des éléments de l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$ est en figure 5.6 page ci-contre. On y retrouve les deux enregistrements de T E^1 et E^2. Les groupes de généralisation sont représentés par des rectangles de couleurs différentes : $G_{(a_1,b_1)}$ est en bleu, $G_{(a_2,b_2)}$ est en

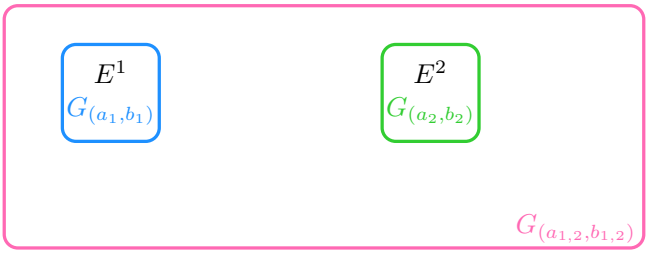


FIGURE 5.6 – Représentation de la composition des éléments de l'ensemble minimal des groupes de généralisation d'une table à deux enregistrements

vert et $G_{(a_{1,2},b_{1,2})}$ est en rose.

Pour terminer cet exemple, nous allons illustrer la remarque 5.1.7 page 97. Calculons $X := \{G_1 \cap G_2 \neq \emptyset$ avec $G_j \in \overline{\mathcal{G}}_{H_j}(T)$ pour tout $j \in [\![1,2]\!]\}$.

Les éléments de X sont $G_{(a_1)} \cap G_{(b_1)}$, $G_{(a_1)} \cap G_{(b_{1,2})}$, $G_{(a_2)} \cap G_{(b_2)}$, $G_{(a_2)} \cap G_{(b_{1,2})}$, $G_{(a_{1,2})} \cap G_{(b_1)}$, $G_{(a_{1,2})} \cap G_{(b_2)}$ et $G_{(a_{1,2})} \cap G_{(b_{1,2})}$.

D'après la propriété 5.1.1 et comme $\overline{\mathcal{G}}_{H_j}(T) \subseteq \mathcal{G}_{H_j}(T)$ pour $j \in [\![1,2]\!]$, nous avons

$$X = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_1)}, G_{(a_{1,2},b_2)}, G_{(a_{1,2},b_{1,2})}\}.$$

Nous avons donc $\overline{\mathcal{G}}_{\mathcal{H}}(T) \subset X$ mais $X \notin \overline{\mathcal{G}}_{\mathcal{H}}(T)$.

En revanche, si nous calculons $\overline{X}$ comme étant $\bigcup_{C^{\equiv} \in \mathcal{C}^{\equiv}(X)} \{\text{repr}(C^{\leqslant}) : C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})\}$, nous obtenons

$$\begin{aligned}\mathcal{C}^{\equiv}(X) = \{&\{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_{1,2},b_1)}\},\\ &\{G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_2)}\},\\ &\{G_{(a_{1,2},b_{1,2})}\}\},\end{aligned}$$

puis, en posant $C_1^{\equiv} = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_{1,2},b_1)}\}$, $C_2^{\equiv} = \{G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_2)}\}$ et $C_3^{\equiv} = \{G_{(a_{1,2},b_{1,2})}\}$

$$\begin{aligned}\mathcal{C}^{\leqslant}(C_1^{\equiv}) &= \{\{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_{1,2},b_1)}\}\}\\ \mathcal{C}^{\leqslant}(C_2^{\equiv}) &= \{\{G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_2)}\}\}\\ \mathcal{C}^{\leqslant}(C_3^{\equiv}) &= \{\{G_{(a_{1,2},b_1)}\}\},\end{aligned}$$

et enfin, en posant $C_1^{\leqslant} = \{G_{(a_1,b_1)}, G_{(a_1,b_{1,2})}, G_{(a_{1,2},b_1)}\}$, $C_2^{\leqslant} = \{G_{(a_2,b_2)}, G_{(a_2,b_{1,2})}, G_{(a_{1,2},b_2)}\}$ et $C_3^{\leqslant} = \{G_{(a_{1,2},b_{1,2})}\}$

$$\begin{aligned}\overline{X} &= \bigcup_{C^{\equiv} \in \mathcal{C}^{\equiv}(X)} \{\text{repr}(C^{\leqslant}) : C^{\leqslant} \in \mathcal{C}^{\leqslant}(C^{\equiv})\}\\ &= \{\text{repr}(C_1^{\leqslant})\} \cup \{\text{repr}(C_2^{\leqslant})\} \cup \{\text{repr}(C_3^{\leqslant})\}\\ &= \{G_{(a_1,b_1)}, G_{(a_2,b_2)}, G_{(a_{1,2},b_{1,2})}\}\\ &= \overline{\mathcal{G}}_{\mathcal{H}}(T).\end{aligned}$$

Dans ce chapitre, notre objectif est toujours de produire des tables k-anonymes. Nous allons donc considérer le sous-ensemble de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ contenant les groupes de généralisation de taille supérieure à k (cf. définition 5.1.9).

Définition 5.1.9 (Ensemble de groupes de généralisation de taille supérieure à k)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$. Soit $k \in \mathbb{N}^*$.

On note $\overline{\mathcal{G}}^k_{\mathcal{H}}(T)$ l'ensemble des groupes de généralisation de T sur $\mathcal{H}$ de taille supérieure à k.

$$\overline{\mathcal{G}}^k_{\mathcal{H}}(T) = \{\Gamma \in \overline{\mathcal{G}}_{\mathcal{H}}(T) : |\Gamma| \geqslant k\}.$$

Dans cette section, nous avons introduit la notion de groupes de généralisation d'une table sur un uplet de hiérarchies (cf. section 5.1.1 page 89). Ces groupes de généralisation permettent d'identifier des groupes d'enregistrements de la table partageant les mêmes valeurs quasi-identifiantes ou des valeurs pouvant se généraliser

en la même valeur. En étudiant l'ensemble des groupes de généralisation d'une table, nous nous sommes aperçus que certains groupes de généralisation contiennent des informations redondantes. Nous avons donc présenté une méthode de construction permettant de ne conserver que les groupes de généralisation contenant une information pertinente (cf. section 5.1.2.1 page 93). Nous avons alors obtenu l'ensemble minimal des groupes de généralisation d'une table (cf. définition 5.1.6 page 96).

Dans la suite de la section 5.1 page 89, nous allons montrer que l'ensemble minimal des groupes de généralisation d'une table permet de construire des versions k-anonymes de la table (cf. proposition 5.2.1 page suivante). La notion de partitionnement d'une table sur son ensemble minimal de groupes de généralisation sera introduite (cf. définition 5.2.1). D'autre part, nous fournirons une représentation sous forme d'hypergraphe de la table et de son ensemble minimal de groupes de généralisation. Cette représentation nous permettra d'énoncer une formulation du problème de k-anonymisation d'une table en section 5.2.3 page 104.

5.2 Formulation du problème de k-anonymisation d'une table

Dans cette section, nous allons exprimer le problème de k-anonymisation d'une table en fonction de l'ensemble minimal des groupes de généralisation. Pour ce faire, nous allons commencer par définir un partitionnement d'une table sur son ensemble minimal de groupes de généralisation en section 5.2.1. Nous verrons ensuite que l'on peut définir un hypergraphe dont les sommets sont les enregistrements de la table et les hyperarêtes sont les groupes de généralisation de l'ensemble minimal des groupes de généralisation (cf. section 5.2.2 page 102). A partir de la matrice d'incidence de cet hypergraphe, nous pourrons reformuler le problème de k-anonymisation sur un ensemble d'enregistrements en section 5.2.3 page 104. Nous aurons alors les outils nécessaires pour présenter notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ en section 5.3 page 105.

5.2.1 Partitionnement et k-partitionnement

Dans cette section, nous allons montrer que l'ensemble minimal de groupes de généralisation permet de construire des versions k-anonymes d'une table. Pour cela, nous allons définir les partitionnements d'une table sur son ensemble minimal de groupes de généralisation (cf. définition 5.2.1). Un tel partitionnement consiste en le choix d'un sous-ensemble d'enregistrements de chaque groupe de généralisation de telle sorte que ces sous-ensembles soient disjoints et que leur union soit égale à la table.

Définition 5.2.1 (Partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T) = \{\Gamma_1, \ldots, \Gamma_p\}$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$ avec $p \in \mathbb{N}^*$.

On dit que $P = \{P_i \ : \ P_i \subseteq \Gamma_i \text{ et } \Gamma_i \in \overline{\mathcal{G}}_{\mathcal{H}}(T) \ \forall \ i \in [\![1, p]\!]\}$ est un *partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$* si P vérifie les conditions suivantes :

— $\forall r, t \in [\![1, p]\!]$, $P_r \cap P_t = \emptyset$,
— $\bigcup_{i=1}^{p} P_i = T$.

Les P_i sont appelés *éléments* de P.

On note $\mathcal{P}$ l'ensemble des partionnements de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$.

On note $\mathcal{P}_\alpha$ l'ensemble des partitionnements de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ à exactement α éléments non vides.

Exemple 5.2.1
Soient $\mathcal{Q} = \{Q_1, Q_2\}$ un ensemble d'attributs quasi-identifiants avec $Q_1 = \{a_1, a_2, a_3\}$ et $Q_2 = \{b_1, b_2\}$.

Considérons $\mathcal{H} = (H_1, H_2)$ un couple de hiérarchies associées aux attributs de $\mathcal{Q}$ et décrites dans la figure 5.3. La hiérarchie H_1 est composée de trois feuilles (a_1, a_2 et a_3), d'un nœud de niveau 1 ($a_{1,2}$ qui est une généralisation de a_1 et a_2) et d'une racine ($a_{1,2,3}$). La hiérarchie H_2 est composée de deux feuilles (b_1 et b_2) et d'une racine ($b_{1,2}$).

Considérons la table sur $\mathcal{Q}$ décrite par la figure 5.5 et notée T. T a deux enregistrements $E^1 = (a_1, b_1)$ et $E^2 = (a_2, b_2)$.

Dans l'exemple 5.1.3 page 97, nous avons calculé l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$:

$$\overline{\mathcal{G}}_{\mathcal{H}}(T) = \{G_{(a_1,b_1)}, G_{(a_2,b_2)}, G_{(a_{1,2},b_{1,2})}\}.$$

avec $G_{(a_1,b_1)} = \{E^1\}$, $G_{(a_2,b_2)} = \{E^2\}$ et $G_{(a_{1,2},b_{1,2})} = T$.

Pour déterminer un partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$, il faut choisir un sous-ensemble de chaque groupe de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ de telle sorte que ces sous-ensembles soient disjoints et que leur union soit égale à T.

Par exemple, si nous choisissons $P_1 = \emptyset \subset G_{(a_1,b_1)}$, $P_2 = \{E^2\} \subset G_{(a_2,b_2)}$ et $P_3 = \{E^1\} \subset G_{(a_{1,2},b_{1,2})}$, nous obtenons le partitionnement $P = \{P_1, P_2, P_3\}$ de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$. En effet, $P_1 \cap P_2 = P_1 \cap P_3 = P_2 \cap P_3 = \emptyset$ et $P_1 \cup P_2 \cup P_3 = T$. De plus, P a exactement deux éléments non vides, P_2 et P_3, donc il appartient à l'ensemble $\mathcal{P}_2$.

Dans l'optique de construire des versions k-anonymes d'une table, nous définissons les k-partitionnements de la table sur son ensemble de groupes de généralisation comme des partitionnements tels que leurs éléments soient de cardinal supérieur à k (cf. définition 5.2.2).

Définition 5.2.2 (k-partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$. Soit $\mathcal{P}$ l'ensemble des partitionnements de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$.

Soit $k \in \mathbb{N}^*$.

On dit que $P \in \mathcal{P}$ est un *k-partionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$* si pour tout $i \in [\![1, p]\!]$, $|P_i| = 0$ ou $|P_i| \geqslant k$.

On note $\mathcal{P}^k$ l'ensemble des k-partitionnements de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$. On a $\mathcal{P}^k \subseteq \mathcal{P}$.

Quand le nombre d'enregistrements de la table et le nombre de groupes de généralisation sont raisonnables, la représentation de la composition des groupes de généralisation proposée dans la figure 5.6 page 99 peut permettre d'identifier visuellement des k-partitionnements de la table sur son ensemble de groupes de généralisation. Dans l'exemple 5.2.1 page ci-contre, à partir de la représentation de la figure 5.6 page 99, nous observons que le seul 2-partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ est composé d'un seul élément $\{E^1, E^2\}$ correspondant au groupe de généralisation $G_{(a_{1,2},b_{1,2})}$.

Voyons à présent comment construire une version k-anonyme d'une table à partir de son ensemble minimal de groupes de généralisation. La proposition 5.2.1 montre qu'un k-partitionnement d'une table sur l'ensemble minimal des groupes de généralisation permet de construire une version k-anonyme de la table.

Proposition 5.2.1
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T) = \{\Gamma_1, \ldots, \Gamma_p\}$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$ avec $p \in \mathbb{N}^*$.

Tout k-partitionnement $P = \{P_1, \ldots, P_p\}$ de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ donne une table k-anonyme sur $(T, \mathcal{H})$:

$$T^{gen} = gen_T(P_1, \ldots, P_p)\,.$$

Démonstration : Justifions que $T^{gen} = gen_T(P_1, \ldots, P_p)$ est bien défini. D'après la définition 2.3.9 de gen_T, il faut vérifier que pour tout $i \in [\![1, p]\!]$, P_i est un sous-ensemble de T et que pour $r, t \in [\![1, p]\!]$, $P_r \cap P_t = \emptyset$. Ces deux conditions sont remplies par définition d'un partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$.

Vérifions maintenant que $T^{gen} = gen_T(P_1, \ldots, P_p)$ est une table k-anonyme sur $(T, \mathcal{H})$. D'après la définition 3.3.1 de table k-anonyme, il faut montrer que T^{gen} est une table généralisée sur $(T, \mathcal{H})$ et que chaque classe d'équivalence de T^{gen} est de taille supérieure à k.

Par la définition 2.3.9 de gen_T, T^{gen} est une table généralisée sur $(T, \mathcal{H})$.

Soit $F^i \in T^{gen}$ avec $i \in [\![1, n]\!]$. Comme P est un partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$, il existe $s \in [\![1, p]\!]$ tel que l'enregistrement E^i de T appartienne à P_s. Donc, par la définition 2.3.9, on a $F^i = \overline{gen}(\text{Repr}(P_s))$. Or $|P_s| \geqslant k$ car P est un k-partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$. Donc il existe $|P_s| - 1$ autres enregistrements de T^{gen} égaux à $\overline{gen}(\text{Repr}(P_s))$. Ainsi, les enregistrements de T^{gen} correspondant aux enregistrements de P_s sont dans la même classe d'équivalence de représentant $\overline{gen}(\text{Repr}(P_s))$ et de taille supérieure à k. Comme les classes d'équivalence de T^{gen} forment une partition de T^{gen}, on a $\forall C \in \mathcal{C}(T^{gen})$, $|C| \geqslant k$. ■

Exemple 5.2.2
Reprenons les notations de l'exemple 5.2.1 page précédente.

Nous voulons construire une version 2-anonyme de T.

Le seul 2-partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ est $P = \{P_1, P_2, P_3\}$ avec $P_1 = \emptyset \subset G_{(a_1,b_1)}$, $P_2 = \emptyset \subset G_{(a_2,b_2)}$ et $P_3 = \{E^1, E^2\} \subset G_{(a_{1,2},b_{1,2})}$. En effet, P est bien un partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ et $|P_1| = 0$, $|P_2| = 0$ et $|P_3| = 2$.

Comme l'affirme la proposition 5.2.1, le 2-partitionnement P fournit une version 2-anonyme T^{gen} de la table T définie par :

$$T^{gen} = gen_T(P_1, P_2, P_3)\,.$$

En suivant la définition 2.3.9 page 21 de gen_T, nous généralisons les enregistrements de P_3 (P_1 et P_2 sont vides). Nous obtenons $T^{gen} = \{F^1, F^2\}$ avec $F^1 = (a_{1,2}, b_{1,2})$ et $F^2 = (a_{1,2}, b_{1,2})$ (cf. figure 5.7 page suivante).

T^{gen}	Q_1	Q_2
F^1	$a_{1,2}$	$b_{1,2}$
F^2	$a_{1,2}$	$b_{1,2}$

FIGURE 5.7 – Représentation d'une version 2-anonyme d'une table.

Afin de pouvoir classifier les partitionnements d'une table selon leur qualité en termes de coût de généralisation, nous définissons le coût d'un partitionnement pour une métrique de perte d'information (cf. définition 5.2.3). Cela nous sera utile dans notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ pour choisir le meilleur partitionnement à effectuer dans une classe d'équivalence (cf. section 5.3 page 105).

Définition 5.2.3 (Coût d'un partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T) = \{\Gamma_1, \ldots, \Gamma_p\}$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$ avec $p \in \mathbb{N}^*$. Soit μ une métrique sur $\mathcal{H}$. Soit $\mathcal{P}$ l'ensemble des partitionnements de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$.

On définit l'application $\mathrm{CP}_\mu : \mathcal{P} \to \mathbb{R}$ qui à tout partitionnement $P = \{P_i : P_i \subseteq \Gamma_i \text{ et } \Gamma_i \in \overline{\mathcal{G}}_{\mathcal{H}}(T) \ \forall\, i \in [\![1, p]\!]\} \in \mathcal{P}$ associe son coût pour la métrique μ :

$$\mathrm{CP}_\mu(P) = \sum_{i=1}^{p} \sum_{E \in P_i} \overline{\mu}(E, \mathrm{repr}(\Gamma_i)).$$

5.2.2 Hypergraphe reliant table et ensemble minimal de groupes de généralisation

Lors de la présentation des groupes de généralisation de la section 5.1 page 89, nous avons proposé une représentation permettant de visualiser la composition des groupes de généralisation d'une table (cf. figures 5.4 page 91 et 5.6 page 99). Dans cette section, nous souhaitons associer à cette représentation visuelle une représentation plus formelle. Originellement définie par Claude Berge dans les années 1960, la notion d'*hypergraphe* nous fournit une représentation du lien entre table et ensemble minimal de groupes de généralisation. On en trouve une définition dans le livre [6] de Alain Bretto.

Définition 5.2.4 (Hypergraphe et matrice d'incidence)
Un hypergraphe Ψ, noté $\Psi = (V, E)$, sur un ensemble fini V est une famille $E = (e_j)_{1 \leqslant j \leqslant m}$ de sous-ensembles de V appelés hyperarêtes.

Si $\Psi = (V = \{v_1, \ldots, v_n\}, E = (e_1, \ldots, e_m))$ est tel que $\bigcup_{j=1}^{m} e_j = V$ alors Ψ a une matrice d'incidence $A = (a_{ij})$ telle que

$$a_{ij} = \begin{cases} 1 & \text{si } v_i \in e_j \\ 0 & \text{sinon} \end{cases}$$

Exemple 5.2.3
Considérons l'hypergraphe $\Psi = (V, E)$ composé de quatre sommets v_1, v_2, v_3 et v_4 et de trois hyperarêtes $e_1 = \{v_1, v_2, v_3\}$, $e_2 = \{v_3, v_4\}$ et $e_3 = \{v_1\}$. Une représentation de Ψ est en figure 5.8 page ci-contre.

Comme Ψ vérifie $\cup_{j=1}^{3} e_j = V$, la matrice d'incidence de Ψ est A :

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{ccc} e_1 & e_2 & e_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \end{array}$$

La matrice d'incidence de l'hypergraphe permet de visualiser la répartition des sommets dans les hyperarêtes.

À une table T et à son ensemble minimal de groupes de généralisation, nous associons l'hypergraphe $\Psi = (T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ (cf. définition 5.2.5 page suivante). Nous définissons ensuite la matrice d'incidence ainsi que la matrice des coûts pour une métrique de perte d'information de cet hypergraphe (cf. définition 5.2.6 page ci-contre). Dans notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ de la section 5.3 page 105, ces deux matrices nous permettront de déterminer

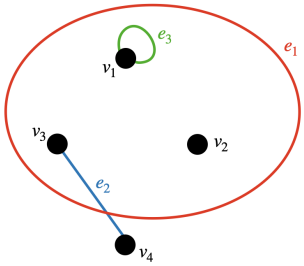


FIGURE 5.8 – Représentation d'un hypergraphe composé de quatre sommets v_1, v_2, v_3 et v_4 et de trois hyperarêtes $e_1 = \{v_1, v_2, v_3\}$, $e_2 = \{v_3, v_4\}$ et $e_3 = \{v_1\}$.

des k-partitionnements des enregistrements d'une classe d'équivalence qui minimisent le coût de généralisation pour une métrique.

Définition 5.2.5 (Hypergraphe $(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$.

On peut associer à T et à $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ un *hypergraphe* Ψ dont les sommets sont les enregistrements de T et les hyperarêtes sont les groupes de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$.

Définition 5.2.6 (Matrice d'incidence et matrice des coûts de l'hypergraphe $(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T) = \{\Gamma_1, \ldots, \Gamma_p\}$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$ avec $p \in \mathbb{N}^*$. Soit μ une métrique sur $\mathcal{H}$.

La matrice d'incidence de $\Psi = (T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$, notée $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T)) = (m_{ij})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant p}$, est définie comme suit :

$$m_{ij} = \begin{cases} 1 & \text{si } E^i \in \Gamma_j \\ 0 & \text{sinon} \end{cases}$$

avec E^i le i^{e} enregistrement de T et Γ_j le j^{e} groupe de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$, pour tout $i \in [\![1, n]\!]$ et tout $j \in [\![1, p]\!]$.

De plus, on définit la *matrice des coûts pour* μ de l'hypergraphe $\Psi = (T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$, notée $M^{\text{c}}_{\mu}(T, \overline{\mathcal{G}}_{\mathcal{H}}(T)) = (c_{ij})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant p}$, par :

$$c_{ij} = \begin{cases} \overline{\mu}\left(E^i, \text{repr}(\Gamma_j)\right) & \text{si } E^i \in \Gamma_j \\ 0 & \text{sinon} \end{cases}$$

avec E^i le i^{e} enregistrement de T et Γ_j le j^{e} groupe de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$, pour tout $i \in [\![1, n]\!]$ et tout $j \in [\![1, p]\!]$.

En substance, la matrice d'incidence de l'hypergraphe permet de visualiser la composition des groupes de généralisation. La matrice des coûts, quant à elle, contient dans chaque coefficient le coût de généralisation de l'enregistrement labellisant la ligne en le représentant du groupe de généralisation labellisant la colonne (cf. définition 5.1.7 page 96 pour la définition du représentant d'un groupe de généralisation).

Exemple 5.2.4
Soient $\mathcal{Q} = \{Q_1, Q_2\}$ un ensemble d'attributs quasi-identifiants avec $Q_1 = \{a_1, a_2, a_3\}$ et $Q_2 = \{b_1, b_2\}$.

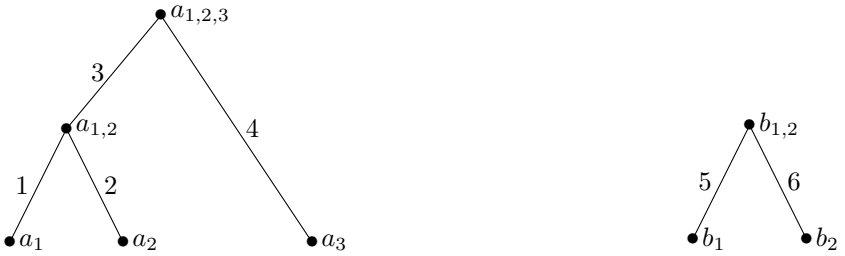
Considérons $\mathcal{H} = (H_1, H_2)$ un couple de hiérarchies associées aux attributs de $\mathcal{Q}$ et décrites dans la figure 5.3. La hiérarchie H_1 est composée de trois feuilles (a_1, a_2 et a_3), d'un nœud de niveau 1 ($a_{1,2}$ qui est une généralisation de a_1 et a_2) et d'une racine ($a_{1,2,3}$). La hiérarchie H_2 est composée de deux feuilles (b_1 et b_2) et d'une racine ($b_{1,2}$).

Considérons la table sur $\mathcal{Q}$ décrite par la figure 5.5 et notée T. T a deux enregistrements $E^1 = (a_1, b_1)$ et $E^2 = (a_2, b_2)$.

Dans l'exemple 5.1.3 page 97, nous avons calculé l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$:

$$\overline{\mathcal{G}}_{\mathcal{H}}(T) = \{G_{(a_1,b_1)}, G_{(a_2,b_2)}, G_{(a_{1,2},b_{1,2})}\}.$$

avec $G_{(a_1,b_1)} = \{E^1\}$, $G_{(a_2,b_2)} = \{E^2\}$ et $G_{(a_{1,2},b_{1,2})} = T$.



(a) Représentation d'une hiérarchie de Q_1 avec des poids sur les arêtes.

(b) Représentation d'une hiérarchie de Q_2 avec des poids sur les arêtes.

FIGURE 5.9 – Représentation de hiérarchies de $\mathcal{Q}$ avec des poids sur les arêtes.

Posons $\Gamma_1 = G_{(a_1,b_1)}$, $\Gamma_2 = G_{(a_2,b_2)}$ et $\Gamma_3 = G_{(a_{1,2},b_{1,2})}$.

Considérons Ψ l'hypergraphe dont les sommets sont les enregistrements de T et les hyperarêtes sont les groupes de généralisation de $\overline{\mathcal{G}}_{\mathcal{H}}(T)$. La matrice d'incidence de Ψ est

$$M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T)) = \begin{array}{c} \\ E^1 \\ E^2 \end{array} \begin{array}{c} \begin{array}{ccc} \Gamma_1 & \Gamma_2 & \Gamma_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \end{array}.$$

Considérons une métrique μ sur $\mathcal{H}$ définie par les poids sur les arêtes des hiérarchies H_1 et H_2 en figure 5.9. Nous avons $\omega(a_1, a_{1,2}) = 1$, $\omega(a_2, a_{1,2}) = 2$, $\omega(a_{1,2}, a_{1,2,3}) = 3$, $\omega(a_3, a_{1,2,3}) = 4$ pour la hiérarchie H_1 et $\omega(b_1, b_{1,2}) = 5$ et $\omega(b_2, b_{1,2}) = 6$ pour la hiérarchie H_2.

Pour compléter la matrice des coûts pour μ de Ψ, il faut déterminer le représentant de chaque groupe de généralisation. Pour ce faire, nous utilisons la définition 5.1.7 page 96 :

$$\begin{aligned} \text{repr}\big(G_{(a_1,b_1)}\big) &= \overline{gen}\big(\{E^1\}\big) = (a_1, b_1) \\ \text{repr}\big(G_{(a_2,b_2)}\big) &= \overline{gen}\big(\{E^2\}\big) = (a_2, b_2) \\ \text{repr}\big(G_{(a_{1,2},b_{1,2})}\big) &= \overline{gen}\big(\{E^1, E^2\}\big) = (a_{1,2}, b_{1,2}) \end{aligned}$$

Ensuite, pour chaque coefficient (E^i, Γ_j) de la matrice des coûts $M^c_\mu(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ avec $i \in \{1, 2\}$ et $j \in \{1, 2, 3\}$, nous calculons $\overline{\mu}\left(E^i, \text{repr}(\Gamma_j)\right)$ si $E^i \in \Gamma_j$:

$$\begin{aligned} &(E^1, \Gamma_1) = \overline{\mu}\left(E^1, \text{repr}(\Gamma_1)\right) = \overline{\mu}\left(E^1, (a_1, b_1)\right) = 0 + 0 = 0 \\ &(E^1, \Gamma_2) = 0 \text{ car } E^1 \notin \Gamma_2 \\ &(E^1, \Gamma_3) = \overline{\mu}\left(E^1, \text{repr}(\Gamma_3)\right) = \overline{\mu}\left(E^1, (a_{1,2}, b_{1,2})\right) = 1 + 5 = 6 \\ &(E^2, \Gamma_1) = 0 \text{ car } E^2 \notin \Gamma_1 \\ &(E^2, \Gamma_2) = \overline{\mu}\left(E^2, \text{repr}(\Gamma_2)\right) = \overline{\mu}\left(E^2, (a_2, b_2)\right) = 0 + 0 = 0 \\ &(E^2, \Gamma_3) = \overline{\mu}\left(E^2, \text{repr}(\Gamma_3)\right) = \overline{\mu}\left(E^2, (a_{1,2}, b_{1,2})\right) = 2 + 6 = 8 \end{aligned}$$

La matrice des coûts pour μ de Ψ est donc

$$M^c_\mu(T, \overline{\mathcal{G}}_{\mathcal{H}}(T)) = \begin{array}{c} \\ E^1 \\ E^2 \end{array} \begin{array}{c} \begin{array}{ccc} \Gamma_1 & \Gamma_2 & \Gamma_3 \end{array} \\ \begin{pmatrix} 0 & 0 & 6 \\ 0 & 0 & 8 \end{pmatrix} \end{array}.$$

5.2.3 Formulation du problème de k-anonymisation d'une table

Grâce à la représentation sous forme d'hypergraphe établie en définition 5.2.5 page précédente, nous pouvons formuler le problème de k-anonymisation d'une table à partir de la matrice d'incidence de l'hypergraphe dont les sommets sont les enregistrements de la table et les hyperarêtes sont les groupes de généralisation de la table.

Définition 5.2.7 (Formulation du problème de k-anonymisation d'une table)
Soient $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants. Soit $\mathcal{H} = (H_1, \ldots, H_m)$ un

m-uplet de hiérarchies des attributs de $\mathcal{Q}$. Soit T une table sur $(\mathcal{Q}, \mathcal{H})$. Soit $\overline{\mathcal{G}}_{\mathcal{H}}(T)$ l'ensemble minimal des groupes de généralisation de T sur $\mathcal{H}$.

Soit $\Psi = (T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ un hypergraphe. Soit $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ la matrice d'incidence de Ψ. Soit $k \in \mathbb{N}^*$.

Le problème de k-anonymité sur T peut s'énoncer comme suit :

Une k-anonymisation de T revient à choisir un unique 1 par ligne de $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ de telle sorte que la somme des coefficients de chaque colonne de $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ soit égale à 0 ou supérieure à k.

Démonstration : Montrons que la formulation de la définition 5.2.7 page précédente correspond bien à la construction d'une version k-anonyme d'une table.

Soit $S = (s_{ij})_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant p}$ la matrice respectant les conditions de la formulation de la définition 5.2.7 page ci-contre. S est telle que pour tout $i \in [\![1, n]\!]$, il existe un unique $j \in [\![1, p]\!]$ tel que $s_{ij} = 1$ et pour tout $j' \neq j$, $s_{ij'} = 0$. Pour tout $i \in [\![1, n]\!]$ et tout $j \in [\![1, p]\!]$, si $s_{ij} = 1$ alors $m_{ij} = 1$ (les coefficients sont choisis à partir de la matrice $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$). De plus, pour tout $j \in [\![1, p]\!]$, on a $\sum_{i=1}^{n} s_{ij} \geqslant k$ ou $\sum_{i=1}^{n} s_{ij} = 0$.

On va montrer qu'on peut associer à S un k-partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$. La proposition 5.2.1 page 101 nous permettra de conclure. Soit $P = \{P_1, \ldots, P_p\}$ un ensemble de sous-ensembles de T. On pose $P_j := \{E^i \in T : i \in [\![1, n]\!] \text{ et } s_{ij} = 1\}$ pour tout $j \in [\![1, p]\!]$.

Soit $j \in [\![1, p]\!]$. Soit $E^i \in P_j$ pour $i \in [\![1, n]\!]$. On a $s_{ij} = 1$ donc, par construction de S, le coefficient m_{ij} de $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ est égal à 1. Or $M(T, \overline{\mathcal{G}}_{\mathcal{H}}(T))$ est la matrice d'incidence de Ψ donc $m_{ij} = 1$ implique que $E^i \in \Gamma_j$. Donc $P_j \subseteq \Gamma_j$.

Par construction de S, pour tout $i \in [\![1, n]\!]$, il existe un $j \in [\![1, p]\!]$ tel que $s_{ij} = 1$ donc, pour tout $i \in [\![1, n]\!]$, l'enregistrement E^i appartient à un P_j de P. Donc $\cup_{j=1}^{p} P_j = T$.

De plus, pour tout $i \in [\![1, n]\!]$, $s_{ij} = 1$ pour un unique $j \in [\![1, p]\!]$. Donc, pour tout $i \in [\![1, n]\!]$, l'enregistrement E^i appartient à un unique P_j de P. On en déduit que $P_r \cap P_t = \emptyset$ pour tout $r, t \in [\![1, p]\!]$.

Finalement, l'hypothèse pour tout $j \in [\![1, p]\!]$, $\sum_{i=1}^{n} s_{ij} \geqslant k$ ou $\sum_{i=1}^{n} s_{ij} = 0$ implique que soit $|P_j| \geqslant k$ soit $|P_j| = 0$ pour tout $j \in [\![1, p]\!]$.

P remplit toutes les conditions de la définition 5.2.1 page 100 : il s'agit donc d'un k-partitionnement de T sur $\overline{\mathcal{G}}_{\mathcal{H}}(T)$. Ainsi, par la proposition 5.2.1 page 101, on peut construire une version k-anonyme de T à partir de P. ■

5.3 Procédure et algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$

Dans cette section, nous allons présenter notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ (cf. section 5.3.1). Rappelons que, dans ce chapitre, nous cherchons à construire une table k-anonyme de meilleure qualité en termes d'altération que celle produite par l'algorithme *GkAA* (cf. algorithme 1 page 38). La procédure s'appuie sur la formulation du problème de k-anonymisation d'une table de la définition 5.2.7 page ci-contre. La formulation en question permet de construire une version k-anonyme d'une table à partir de la matrice d'incidence de l'hypergraphe de sommets les enregistrements de la table et d'hyperarêtes les groupes de généralisation de la table.

Pour chaque classe d'équivalence de la table k'-anonyme, l'idée est de récupérer les enregistrements correspondants dans la table d'origine et de trouver un k-partitionnement de ces enregistrements dont le coût de généralisation est moins élevé que le coût de généralisation de la classe d'équivalence. La recherche d'un k-partitionnement qui minimise le coût de généralisation se fera à l'aide d'un solveur appliqué à la matrice d'incidence de l'hypergraphe de la définition 5.2.5 page 103 et d'un ensemble de contraintes traduisant le problème de k-anonymisation.

Une fois la procédure décrite et expliquée, nous proposerons en section 5.3.2 page 108 cinq algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ se basant sur notre procédure. Pour chaque algorithme, nous préciserons son mode de fonctionnement et nous évoquerons certains de ses avantages et inconvénients.

5.3.1 Procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$

Dans cette section, nous allons détailler notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$. Le pseudo-code de la procédure est à lire dans l'algorithme 3 page 108. Par la suite, nous abrégerons le nom de la procédure en *PCkPCk'*.

La $PCkPCk'$ prend en paramètres un ensemble d'attributs quasi-identifiants, un m-uplet de hiérarchies de cet ensemble d'attributs, une table T, un entier k représentant la valeur de k-anonymité recherchée, une table généralisée sur la table de paramètre de k-anonymité supérieur à k et une métrique de perte d'information sur les hiérarchies. La sortie de la procédure est une table k-anonyme d'altération inférieure à l'altération de la table généralisée passée en paramètres.

Pour décrire les étapes de $PCkPCk'$, considérons $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ un entier, T^{gen} une table généralisée sur $(T, \mathcal{H})$ telle que $\kappa(T^{gen}) \geqslant k$ et μ une métrique de perte d'information sur $\mathcal{H}$.

Nous allons traiter toutes les classes d'équivalence de T^{gen} pour obtenir des k-partitionnements de leurs enregistrements. Comme les classes d'équivalence sont disjointes et couvrent l'ensemble des enregistrements de T^{gen}, les nouveaux partitionnements trouvés pour les classes d'équivalence seront directement appliqués à la table d'origine T. Ainsi, T sera modifiée de telle sorte qu'elle soit k-anonyme à la sortie de la $PCkPCk'$.

Solveur, problème d'optimisation et contraintes. Pour déterminer des k-partitionnements optimaux en termes de coût de généralisation pour μ, nous pourrons faire appel au solveur *CP-SAT* proposé par OR-Tools.

OR-Tools est un ensemble de logiciels développé par Google pour résoudre des problèmes d'optimisation [19]. De nombreux solveurs sont implémentés tels que *CPLEX* et *CP-SAT*. Les solveurs sont utilisables dans plusieurs langages de programmation. Dans notre étude, nous avons choisi le solveur *CP-SAT* et nous avons modélisé notre problème en Python. Le solveur *CP-SAT* utilise un solveur de génération de clauses paresseuses en plus d'un solveur SAT [5].

Notre problème d'optimisation et les contraintes utilisées sont décrits ci-après.

Soit C un sous-ensemble d'enregistrements de T. Posons $C = \{E^1, \ldots, E^n\}$ avec $n \in \mathbb{N}^*$. Soit $\overline{\mathcal{G}}^k_{\mathcal{H}}(C) = \{\Gamma_1, \ldots, \Gamma_p\}$ l'ensemble minimal des groupes de généralisation de taille supérieure à k de C sur $\mathcal{H}$ pour $p \in \mathbb{N}^*$.

Soit Ψ l'hypergraphe de sommets C et d'hyperarêtes $\overline{\mathcal{G}}^k_{\mathcal{H}}(C)$. Soient $M(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C)) = (m_{ij})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant p}$ la matrice d'incidence de Ψ et $M^c_\mu(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C)) = (c_{ij})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant p}$ la matrice des coûts pour μ de Ψ.

Nous cherchons à construire une matrice $S = (s_{ij})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant p}$ satisfaisant les contraintes suivantes :

C1 Pour tout $j \in [\![1, p]\!]$, $\sum_{i=1}^{n} s_{ij} = 0$ ou $\sum_{i=1}^{n} s_{ij} \geqslant k$

C2 Pour tout $i \in [\![1, n]\!]$, $\sum_{j=1}^{p} s_{ij} = 1$

C3 Pour tout $i \in [\![1, n]\!]$ et tout $j \in [\![1, p]\!]$, $s_{ij} \leqslant m_{ij}$

C4 Le minimum de l'application

$$\begin{array}{rccl} cost : & \mathcal{M}_{n \times p} & \longrightarrow & \mathbb{R} \\ & \Lambda & \longmapsto & \sum_{i=1}^{n} \sum_{j=1}^{p} \lambda_{ij} \times c_{ij} \text{ pour } \Lambda = (\lambda_{ij})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant p} \end{array}$$

est atteint en S.

Dans notre contexte, la contrainte **C1** traduit le fait que les éléments du k-partitionnement retourné doivent être soit vides soit de taille supérieure à k. La contrainte **C2** signifie que chaque enregistrement doit être associé à un unique groupe de généralisation et donc n'appartenir qu'à un unique élément du k-partitionnement. La contrainte **C3** signifie qu'un enregistrement ne peut pas être associé à un groupe de généralisation s'il ne lui appartient pas. La contrainte **C4** est une contrainte de minimisation du coût du k-partitionnement pour μ.

Les variables traitées par le solveur *CP-SAT* sont les coefficients de la matrice S.

La sortie du solveur *CP-SAT* sera une matrice S correspondant à un k-partitionnement de C sur $\overline{\mathcal{G}}^k_{\mathcal{H}}(C)$. Le k-partitionnement obtenu est $P_{\text{solveur}} = \{P_j : P_j \subseteq \Gamma_j \text{ et } \Gamma_j \in \overline{\mathcal{G}}^k_{\mathcal{H}}(C)\ \forall\ j \in [\![1, p]\!]\}$ avec $P_j = \{E^i \in C : i \in [\![1, n]\!] \text{ et } s_{ij} = 1\}$ pour tout $j \in [\![1, p]\!]$. La proposition 5.2.1 page 101 nous garantit qu'il s'agit bien d'une k-anonymisation des enregistrements de C.

Description de la $PCkPCk'$. Décrivons maintenant les étapes de la $PCkPCk'$.

Dans un premier temps, la procédure traite les classes d'équivalence de T^{gen} contenant strictement moins de $2k$ enregistrements. Pour chaque classe C^{gen} de T^{gen} de taille strictement inférieure à $2k$, nous ne pouvons pas obtenir un k-partitionnement des enregistrements de la classe en deux éléments distincts. La $PCkPCk'$ généralise donc les enregistrements correspondant aux enregistrements de C^{gen} dans T (lignes 3 à 7).

Ensuite, la $PCkPCk'$ traite les classes d'équivalence de T^{gen} contenant au moins $2k$ enregistrements. Soit C^{gen} une classe de T^{gen} de taille supérieure à $2k$. La $PCkPCk'$ récupère dans C les enregistrements de T correspondant aux enregistrements de C^{gen} (cf. définition 2.3.4). Puis la procédure construit l'ensemble minimal des groupes de généralisation de taille supérieure à k de C sur $\mathcal{H}$ (cf. définition 5.1.6 page 96).

Si au moins un groupe de généralisation a été calculé (ligne 12), deux traitements sont possibles selon la taille de C.

Si C contient strictement moins de $3k$ enregistrements, nous ne pourrons trouver qu'un k-partitionnement des enregistrements contenant au maximum deux éléments distincts. La $PCkPCk'$ calcule donc un k-partitionnement de C en deux éléments de coût de généralisation minimal pour μ (ligne 14). Les enregistrements de C sont ensuite généralisés dans T selon les éléments du k-partitionnement obtenu (ligne 15).

Dans le cas où C contient au moins $3k$ enregistrements (ligne 16), la procédure sépare le cas où seuls deux groupes de généralisation ont été calculés. Dans ce cas, nous ne pouvons calculer que des k-partitionnements contenant au maximun deux éléments distincts. La $PCkPCk'$ calcule donc un k-partitionnement de C en deux éléments de coût de généralisation minimal pour μ (ligne 18). Les enregistrements de C sont ensuite généralisés dans T selon les éléments du k-partitionnement obtenu (ligne 19).

Dans le cas où plus de deux groupes de généralisation ont été calculés, la $PCkPCk'$ fait appel au solveur *CP-SAT* de OR-Tools pour résoudre le problème d'optimisation sous contraintes présenté dans le paragraphe précédent. Dans un premier temps, la procédure construit les matrices $M(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ et $M^c_\mu(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ associées à l'hypergraphe $(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ (cf. définitions 5.2.5 et 5.2.6 page 103) (ligne 21). La procédure appelle ensuite le solveur *CP-SAT* avec les contraintes voulues sur $M(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ et $M^c_\mu(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$. Le temps maximal d'exécution du solveur est fixé à 120 secondes. Notons P_{solveur} le k-partitionnement retourné par le solveur (ligne 23). Dans le cas où le solveur ne retournerait pas de solution valide, un k-partitionnement de C en deux éléments de coût de généralisation minimal pour μ est calculé (ligne 24). La recherche d'une k-partitionnement en deux éléments de coût de généralisation minimal reste raisonnable en temps de calcul. Les enregistrements de C sont ensuite généralisés dans T selon les éléments du k-partitionnement le moins coûteux entre P_{solveur} et $P_2^{k,\min}$ (lignes 25 à 29).

Si aucun groupe de généralisation de C sur $\mathcal{H}$ n'est calculé, les enregistrements de C sont généralisés dans T (lignes 32 à 34).

Finalement, la table k-anonymisée T est retournée (ligne 36).

L'avantage de la procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ est que nous pouvons choisir la table k'-anonyme de départ. Par exemple, nous pouvons choisir une table $2k$-anonyme de départ et chercher à construire une table k-anonyme. Ainsi, toutes les classes d'équivalence de la table $2k$-anonyme seront traitées lors de la $PCkPCk'$ et des k-partitionnements seront cherchés pour toutes les classes. En section 5.3.2 page suivante, nous présentons cinq algorithmes se basant sur la $PCkPCk'$ et qui diffèrent notamment par le choix de la table k'-anonyme de départ.

Remarque 5.3.1
Pour simplifier la description de son fonctionnement, la $PCkPCk'$ a été présentée de manière séquentielle : les classes d'équivalence de T sont traitées les unes après les autres dans l'algorithme 3 page suivante. Or, en procédant ainsi, le temps de calcul d'une exécution de la procédure peut vite devenir trop important. Nous avons effectivement fixé le temps limite d'exécution du solveur *CP-SAT* à 120 secondes, ce qui peut être long quand le nombre de classes d'équivalence à traiter est grand.

En pratique, nous avons codé la $PCkPCk'$ pour qu'elle traite les classes d'équivalence en parallèle.

Algorithme 3 Procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ ($PCkPCk'$)

Entrées: $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$ de cardinal $n \in \mathbb{N}^*$ telle que $T = \{E^1, \ldots, E^n\}$, $k \in \mathbb{N}^*$ la valeur de k-anonymité recherchée, T^{gen} une table généralisée sur $(T, \mathcal{H})$ telle que $\kappa(T^{gen}) \geqslant k$, μ une métrique de perte d'information sur $\mathcal{H}$
Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

procédure $PCkPCk'(\mathcal{Q}, \mathcal{H}, T, k, T^{gen}, \mu)$
  Soit $\mathcal{C}(T^{gen})$ l'ensemble des classes d'équivalence de T^{gen}
  pour $C^{gen} \in \mathcal{C}(T^{gen})$ telle que $|C^{gen}| < 2k$ **faire**
    On pose $C^{gen} = \{F^{s_1}, \ldots, F^{s_q}\}$ *avec* $F^i \in T^{gen}$ *pour tout* $i \in \{s_1, \ldots, s_q\}$, $s_j \in [\![1, n]\!]$ *pour tout* $j \in [\![1, n]\!]$
    On détermine C le sous-ensemble de T tel que $C = \{E^i \in T : i \in [\![1, n]\!] \text{ et } F^i = \text{repr}(C^{gen})\}$
    $T \leftarrow gen_T(C)$
  fin pour
  pour $C^{gen} \in \mathcal{C}(T^{gen})$ telle que $|C^{gen}| \geqslant 2k$ **faire**
    On pose $C^{gen} = \{F^{s_1}, \ldots, F^{s_q}\}$ *avec* $F^i \in T^{gen}$ *pour tout* $i \in \{s_1, \ldots, s_q\}$, $s_j \in [\![1, n]\!]$ *pour tout* $j \in [\![1, n]\!]$
    On détermine C le sous-ensemble de T tel que $C = \{E^i \in T : i \in [\![1, n]\!] \text{ et } F^i = \text{repr}(C^{gen})\}$
    On construit $\overline{\mathcal{G}}^k_{\mathcal{H}}(C)$ l'ensemble minimal des groupes de généralisations de C sur $\mathcal{H}$ de cardinaux supérieurs à k
    si $|\overline{\mathcal{G}}^k_{\mathcal{H}}(C)| \neq \emptyset$ **alors**
      si $|C| < 3k$ **alors**
        On cherche $P_2^{k,\min} = \{P, P'\}$ un k-partitionnement de C en deux éléments tel que $\text{CP}_\mu\left(P_2^{k,\min}\right) = \min_{\phi \in \mathcal{P}_2^k} \text{CP}_\mu(\phi)$
        $T \leftarrow gen_T(P, P')$
      sinon
        si $|\overline{\mathcal{G}}^k_{\mathcal{H}}(C)| = 2$ **alors**
          On cherche $P_2^{k,\min} = \{P, P'\}$ un k-partitionnement de C en deux éléments tel que $\text{CP}_\mu\left(P_2^{k,\min}\right) = \min_{\phi \in \mathcal{P}_2^k} \text{CP}_\mu(\phi)$
          $T \leftarrow gen_T(P, P')$
        sinon
          On construit les matrices $M(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ et $M^c_\mu(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ associées à l'hypergraphe $\Psi = (C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$
          On appelle le solveur *CP-SAT* sur les matrices $M(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ et $M^c_\mu(C, \overline{\mathcal{G}}^k_{\mathcal{H}}(C))$ avec un temps d'exécution limite de 120 secondes
          On note $P_{\text{solveur}} = \{P_{r_1}, \ldots, P_{r_t}\}$ le k-partitionnement obtenu avec $r_j \in [\![1, |\overline{\mathcal{G}}^k_{\mathcal{H}}(C)|]\!]$ pour tout $j \in [\![1, t]\!]$, $t \in [\![1, |\overline{\mathcal{G}}^k_{\mathcal{Q}}(C)|]\!]$
          On cherche $P_2^{k,\min} = \{P, P'\}$ un k-partitionnement de C en deux éléments tel que $\text{CP}_\mu\left(P_2^{k,\min}\right) = \min_{\phi \in \mathcal{P}_2^k} \text{CP}_\mu(\phi)$
          si $\text{CP}_\mu(P_{\text{solveur}}) \leqslant \text{CP}_\mu\left(P_2^{k,\min}\right)$ **alors**
            $T \leftarrow gen_T(P_{r_1}, \ldots, P_{r_t})$
          sinon
            $T \leftarrow gen_T(P, P')$
          fin si
        fin si
      fin si
    sinon
      $T \leftarrow gen_T(C)$
    fin si
  fin pour
  retourne T
fin procédure

5.3.2 Algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$

Dans ce chapitre, nous cherchons à construire une table k-anonyme de meilleure qualité en termes d'altération que la table k-anonyme produite avec *GkAA*. Dans cette section, nous proposons donc cinq algorithmes produisant des tables k-anonymes et se basant sur la procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$. La principale différence entre ces cinq algorithmes

réside dans le choix de la table k'-anonyme de départ.

Les noms des algorithmes suivent la nomenclature suivante : *Gk* signifie que l'algorithme *GkAA* est appliqué en demandant une k-anonymité, *Pk* signifie que la *PCkPCk'* est appliquée en demandant des k-partitionnements et *conv* signifie que le traitement est répété jusqu'à atteindre un certain critère de convergence.

Les algorithmes *GkPk* (cf. section 5.3.2.1) et *G3kPk* (cf. section 5.3.2.2) ont pour objectif de construire une table k-anonyme en k-partitionnant les classes d'équivalence de tables k-anonyme et $3k$-anonyme respectivement. Dans les trois autres algorithmes, une succession de tables k-anonymes sont construites jusqu'à atteindre un critère d'arrêt ou la limite du temps d'exécution. Dans l'algorithme *G2kPkconv* (cf. section 5.3.2.3 page suivante), nous k-partitionnons les classes d'équivalence d'une succession de tables $2k$-anonymes. Dans *G2kP2kPkconv* (cf. section 5.3.2.4 page 111), nous ajoutons une étape pour $2k$-partitionner les classes d'équivalence de la succession de tables $2k$-anonymes. Dans *G4kP2kPkconv* (cf. section 5.3.2.5 page 112), nous effectuons une première étape de $2k$-partitionnement des classes d'équivalence de tables $4k$-anonymes puis nous k-partitionnons les classes d'équivalence des tables $2k$-anonymes obtenues.

Pour décrire le fonctionnement des cinq algorithmes de construction, considérons $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ un entier, T^{gen} une table généralisée sur $(T, \mathcal{H})$ telle que $\kappa(T^{gen}) \geqslant k$ et μ une métrique de perte d'information sur $\mathcal{H}$. Notons $T^{GkAA,k}$ la table k-anonyme sur $(T, \mathcal{H})$ produite à la fin de l'exécution $GkAA(\mathcal{Q}, \mathcal{H}, T, \mu, k)$.

Pour chaque algorithme de construction, nous proposerons un schéma récapitulant les étapes effectuées lors de leur exécution. Pour les algorithmes de construction dits avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, le schéma représentera les étapes effectuées lors du i^e tour de l'exécution de l'algorithme.

Dans la suite de cette section, nous dirons *altération* à la place d'*altération pour μ* pour alléger les explications (cf. définition 3.1.6 page 31 pour la définition de l'altération pour μ).

5.3.2.1 L'algorithme *GkPk*

Dans l'algorithme *GkPk* (cf. algorithme 4), l'objectif est de construire une table k-anonyme sur $(T, \mathcal{H})$ d'altération moins élevée que celle de $T^{GkAA,k}$ en k-partitionnant les classes d'équivalence de $T^{GkAA,k}$.

Dans *GkPk*, nous commençons par appliquer *GkAA* sur T pour obtenir une table k-anonyme. Puis nous k-partitionnons les classes d'équivalence de $T^{GkAA,k}$ en appliquant *PCkPCk'*. Le schéma représentant les étapes effectuées lors de l'exécution de *GkPk* est en figure 5.10.

En partant directement de $T^{GkAA,k}$, nous sommes sûrs de construire une table k-anonyme d'altération inférieure, ou au pire égale, à l'altération de $T^{GkAA,k}$.

Cependant, il est possible qu'aucune classe d'équivalence de $T^{GkAA,k}$ ne soit de taille supérieure à $2k$. Dans ce cas, aucun k-partitionnement non trivial ne sera effectué et l'altération de $T^{GkAA,k}$ ne sera donc pas diminuée lors de la *PCkPCk'*.

Algorithme 4 *GkPk*

Entrées: $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ la valeur de k-anonymité recherchée, μ une métrique sur $\mathcal{H}$

Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

1: **procédure** *GkPk*($\mathcal{Q}$, $\mathcal{H}$, T, μ, k)
2: $T^{GkAA,k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T, \mu, k)$
3: Retourne $PCkPCk'\left(\mathcal{Q}, T, \mathcal{H}, k, T^{GkAA,k}, \mu\right)$
4: **fin procédure**

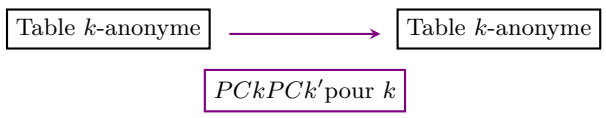


FIGURE 5.10 – Schéma représentant les étapes effectuées lors de l'exécution de *GkPk* pour un k demandé

5.3.2.2 L'algorithme *G3kPk*

Dans l'algorithme *G3kPk* (cf. algorithme 5 page suivante), l'objectif est de construire une table k-anonyme sur $(T, \mathcal{H})$ d'altération moins élevée que celle de $T^{GkAA,k}$ en k-partitionnant les classes d'équivalence de $T^{GkAA,3k}$,

la table $3k$-anonyme produite avec *GkAA*.

Dans *G3kPk*, nous commençons par appliquer *GkAA* sur T pour obtenir une table $3k$-anonyme. Puis nous k-partitionnons les classes d'équivalence de $T^{GkAA,3k}$ en appliquant la $PCkPCk'$. Le schéma représentant les étapes effectuées lors de l'exécution de *G3kPk* est en figure 5.11.

En partant de $T^{GkAA,3k}$, nous sommes sûrs que toutes les classes d'équivalence seront traitées par la $PCkPCk'$. En effet, $T^{GkAA,3k}$ étant $3k$-anonyme, ses classes d'équivalence sont de taille supérieure à $3k$ et *a fortiori* à $2k$. Des classes d'équivalence de tailles bien supérieures à k peuvent permettre de trouver des k-partitionnements plus fins et donc moins coûteux en termes de généralisation.

Cependant, des classes d'équivalence de tailles trop importantes par rapport à k peuvent être longues à traiter dans la $PCkPCk'$. En effet, un grand nombre d'enregistrements signifie potentiellement un grand nombre de groupes de généralisation et peut notamment entraîner un long temps de calcul de la matrice d'incidence de l'hypergraphe définie en 5.2.6 page 103. De plus, le solveur ne trouve pas toujours un bon k-partitionnement compte tenu du grand nombre de variables à traiter et du temps limite d'exécution fixé à 120 secondes. Avec cet algorithme, rien ne garantit que la table k-anonyme construite sera d'altération inférieure à l'altération de $T^{GkAA,k}$.

Algorithme 5 *G3kPk*

Entrées: $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ la valeur de k-anonymité recherchée, μ une métrique sur $\mathcal{H}$

Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

1: **procédure** *G3kPk*($\mathcal{Q}$, $\mathcal{H}$, T, μ, k)
2: $T^{GkAA,3k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T, \mu, 3k)$
3: Retourne $PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T^{GkAA,3k}, \mu\right)$
4: **fin procédure**

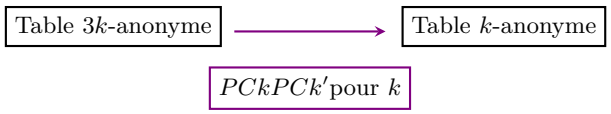


FIGURE 5.11 – Schéma représentant les étapes effectuées lors de l'exécution de *G3kPk* pour un k demandé

5.3.2.3 L'algorithme *G2kPkconv*

Dans l'algorithme *G2kPkconv*(cf. algorithme 6 page ci-contre), l'objectif est de construire une table k-anonyme sur $(T, \mathcal{H})$ d'altération moins élevée que celle de $T^{GkAA,k}$ en répétant des appels à *GkAA* pour obtenir des tables $2k$-anonymes et des appels à la $PCkPCk'$ en cherchant des k-partionnements de classes d'équivalence.

Dans *G2kPkconv*, nous initialisons le processus en $2k$-anonymisant T avec *GkAA*. Puis nous k-partitionnons les classes d'équivalence de $T^{GkAA,2k}$ en appliquant la $PCkPCk'$. Nous réitérons les deux traitements précédents : à chaque tour, nous $2k$-anonymisons la table k-anonyme T_k obtenue à la fin du tour précédent avec *GkAA* puis nous k-partitionnons les classes d'équivalence de $T_k^{GkAA,2k}$ en appliquant la $PCkPCk'$. L'algorithme *G2kPkconv* s'arrête quand les coûts de généralisation de la table k-anonyme du dernier tour et de l'avant-dernier tour sont proches ou que le temps d'exécution dépasse 24 heures. Le schéma représentant les étapes effectuées lors du i^{e} tour de l'exécution de *G2kPkconv* est en figure 5.12 page suivante.

En partant de $T^{GkAA,2k}$, nous sommes sûrs que toutes les classes d'équivalence seront traitées par la $PCkPCk'$. En effet, $T^{GkAA,2k}$ étant $2k$-anonyme, ses classes d'équivalence sont de taille supérieure à $2k$. De plus, la répétition d'étapes de *GkAA* et de $PCkPCk'$ permet de k-partitionner les enregistrements plusieurs fois et de, peut-être, tendre vers une table k-anonyme sur $(T, \mathcal{H})$ optimale en termes d'altération.

Cependant, le fait que l'algorithme s'arrête avant la fin des 24 heures d'exécution ne signifie pas qu'une table k-anonyme optimale en termes d'altération ait été trouvée. En effet, la courbe des altérations des tables k-anonymes successives n'est pas strictement décroissante. La table k-anonyme du dernier tour n'est pas forcément la meilleure de celles construites aux tours précédents en termes d'altération. De plus, même si la table k-anonyme construite au dernier tour de l'algorithme est d'altération minimale dans l'ensemble des tables k-anonymes construites, nous ne savons pas s'il s'agit de la table k-anonyme optimale en termes d'altération. D'autre part, le temps d'exécution sera de 24 heures si aucune convergence n'est trouvée.

Algorithme 6 *G2kPkconv*

Entrées: $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ la valeur de k-anonymité recherchée, μ une métrique sur $\mathcal{H}$, *prec* une valeur de précision, t_{out} temps limite d'exécution

Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

procédure *G2kPkconv*($\mathcal{Q}$, $\mathcal{H}$, T, μ, k, *prec*)
  $t_{exe} \leftarrow$ temps d'exécution total de l'algorithme en seconde
  $T^{GkAA,2k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T, \mu, 2k)$
  $T_k \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T^{GkAA,2k}, \mu\right)$
  $T_{k,bef} \leftarrow T^*$
  tant que $|\mu_T(T_k) - \mu_T(T_{k,bef})| > prec$ et $t_{exe} < t_{out}$ **faire**
    $T_{k,bef} \leftarrow T_k$
    $T_k^{GkAA,2k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T_k, \mu, 2k)$
    $T_k \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T_k^{GkAA,2k}, \mu\right)$
  fin tant que
  Retourne T_k
fin procédure

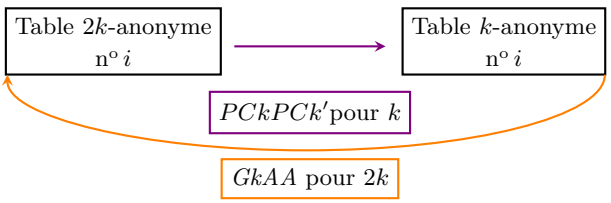


FIGURE 5.12 – Schéma représentant les étapes effectuées lors du i^e tour de l'exécution de *G2kPkconv* pour un k demandé

5.3.2.4 L'algorithme *G2kP2kPkconv*

Dans l'algorithme *G2kP2kPkconv*(cf. algorithme 7 page suivante), l'objectif est de construire une table k-anonyme sur $(T, \mathcal{H})$ d'altération moins élevée que celle de $T^{GkAA,k}$ en répétant des appels à *GkAA* pour obtenir des tables $2k$-anonymes, des appels à la $PCkPCk'$ en cherchant des $2k$-partionnements de classes d'équivalence et des appels à la $PCkPCk'$ en cherchant des k-partitionnements de classes d'équivalence.

Dans *G2kP2kPkconv*, nous initialisons le processus en $2k$-anonymisant T avec *GkAA*. Puis nous $2k$-partitionnons les classes d'équivalence de $T^{GkAA,2k}$ en appliquant la $PCkPCk'$. Nous obtenons une table $2k$-anonyme T_{2k}. Enfin, nous k-partitionnons les classes d'équivalence de T_{2k} en appliquant la $PCkPCk'$. Nous réitérons les trois traitements précédents : à chaque tour, nous $2k$-anonymisons la table k-anonyme T_k produite à la fin du tour précédent avec *GkAA* puis nous $2k$-partitionnons les classes d'équivalence de $T_k^{GkAA,2k}$ en appliquant la $PCkPCk'$, nous obtenons une table $2k$-anonyme T_{2k}. Enfin, nous k-partitionnons les classes d'équivalence de T_{2k} en appliquant la $PCkPCk'$. L'algorithme *G2kP2kPkconv* s'arrête quand les coûts de généralisation de la table k-anonyme du dernier tour et de l'avant-dernier tour sont proches ou que le temps d'exécution dépasse 24 heures. Le schéma représentant les étapes effectuées lors du i^e tour de l'exécution de *G2kP2kPkconv* est en figure 5.13 page suivante.

La première étape de $PCkPCk'$ permet d'obtenir une table $2k$-anonyme d'altération inférieure à la table $2k$-anonyme produite avec *GkAA*. Ainsi, nous appliquerons la seconde étape de $PCkPCk'$ sur une table de meilleure qualité en termes d'altération ; la table k-anonyme obtenue sera potentiellement de meilleure qualité.

Les limites de cet algorithme sont les mêmes que celles de l'algorithme *G2kPkconv*. L'optimalité en termes d'altération de la table k-anonyme construite n'est pas garantie et le temps d'exécution peut être long.

Algorithme 7 *G2kP2kPkconv*

Entrées: $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ la valeur de k-anonymité recherchée, μ une métrique sur $\mathcal{H}$, *prec* une valeur de précision, t_{out} temps limite d'exécution

Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

procédure *G2kP2kPkconv*($\mathcal{Q}$, $\mathcal{H}$, T,μ, k, *prec*)
  $t_{exe} \leftarrow$ temps d'exécution total de l'algorithme en seconde
  $T^{GkAA,2k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T, \mu, 2k)$
  $T_{2k} \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, 2k, T^{GkAA,2k}, \mu\right)$
  $T_k \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T_{2k}, \mu\right)$
  $T_{k,bef} \leftarrow T^*$
  tant que $|\mu_T(T_k) - \mu_T(T_{k,bef})| > prec$ et $t_{exe} < t_{out}$ **faire**
    $T_{k,bef} \leftarrow T_k$
    $T_k^{GkAA,2k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T_k, \mu, 2k)$
    $T_{2k} \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, 2k, T_k^{GkAA,2k}, \mu\right)$
    $T_k \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T_{2k}, \mu\right)$
  fin tant que
  Retourne T_k
fin procédure

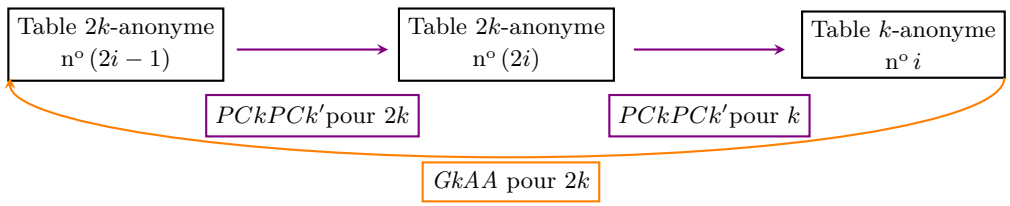


FIGURE 5.13 – Schéma représentant les étapes effectuées lors du i^e tour de l'exécution de *G2kP2kPkconv* pour un k demandé

5.3.2.5 L'algorithme *G4kP2kPkconv*

Dans l'algorithme *G4kP2kPkconv* (cf. algorithme 8 page ci-contre), l'objectif est de construire une table k-anonyme sur $(T, \mathcal{H})$ d'altération moins élevée que celle de $T^{GkAA,k}$ en répétant des appels à *GkAA* pour obtenir des tables $4k$-anonymes, des appels à la $PCkPCk'$ en cherchant des $2k$-partionnements de classes d'équivalence et des appels à la $PCkPCk'$ en cherchant des k-partitionnements de classes d'équivalence.

Dans *G4kP2kPkconv*, nous initialisons le processus en $4k$-anonymisant T avec *GkAA*. Puis nous $2k$-partitionnons les classes d'équivalence de $T^{GkAA,4k}$ en appliquant la $PCkPCk'$. Nous obtenons une table $2k$-anonyme T_{2k}. Enfin, nous k-partitionnons les classes d'équivalence de T_{2k} en appliquant la $PCkPCk'$. Nous réitérons les trois traitements précédents : à chaque tour, nous $4k$-anonymisons la table k-anonyme T_k produite à la fin du tour précédent avec *GkAA* puis nous $2k$-partitionnons les classes d'équivalence de $T_k^{GkAA,4k}$ en appliquant la $PCkPCk'$, nous obtenons une table $2k$-anonyme T_{2k}. Enfin, nous k-partitionnons les classes d'équivalence de T_{2k} en appliquant la $PCkPCk'$. L'algorithme *G4kP2kPkconv* s'arrête quand les coûts de généralisation de la table k-anonyme du dernier tour et de l'avant-dernier tour sont proches ou que le temps d'exécution dépasse 24 heures. Le schéma représentant les étapes effectuées lors du i^e tour de l'exécution de *G4kP2kPkconv* est en figure 5.14 page suivante.

Partir d'une table $4k$-anonyme peut permettre d'obtenir des k-partitionnements des classes d'équivalence plus fins car les classes sont de grandes tailles. En revanche, il n'est pas envisageable d'appliquer la $PCkPCk'$ sur une table $4k$-anonyme en cherchant des k-partitionnements : comme mentionné dans les limites de l'algorithme *G3kPk*, le temps de calcul de certaines variables peut être long quand la taille de la classe est très supérieure à k. Nous passons donc par une table $2k$-anonyme intermédiaire. Des puissances de 2 sont choisies pour les valeurs de k-anonymité des tables car il s'agit de la taille minimale des classes d'équivalence permettant d'obtenir des k-partitionnements de toutes les classes d'équivalence.

Les limites de cet algorithme sont les mêmes que celles des algorithmes *G2kPkconv* et *G2kP2kPkconv*. L'optimalité en termes d'altération de la table k-anonyme construite n'est pas garantie et le temps d'exécution peut être long.

Algorithme 8 *G4kP2kPkconv*

Entrées: $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ un ensemble de $m \in \mathbb{N}^*$ attributs quasi-identifiants, $\mathcal{H} = (H_1, \ldots, H_m)$ un m-uplet de hiérarchies pour les attributs quasi-identifiants de $\mathcal{Q}$, T une table sur $(\mathcal{Q}, \mathcal{H})$, $k \in \mathbb{N}^*$ la valeur de k-anonymité recherchée, μ une métrique sur $\mathcal{H}$, *prec* une valeur de précision, t_{out} temps limite d'exécution

Sortie: Une table k-anonyme sur $(T, \mathcal{H})$

procédure *G4kP2kPkconv*($\mathcal{Q}$, $\mathcal{H}$, T, μ, k, *prec*)
  $t_{exe} \leftarrow$ temps d'exécution total de l'algorithme en seconde
  $T^{GkAA,4k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T, \mu, 4k)$
  $T_{2k} \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, 2k, T^{GkAA,4k}, \mu\right)$
  $T_k \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T_{2k}, \mu\right)$
  $T_{k,bef} \leftarrow T^*$
  tant que $|\mu_T(T_k) - \mu_T(T_{k,bef})| > prec$ et $t_{exe} < t_{out}$ **faire**
    $T_{k,bef} \leftarrow T_k$
    $T_k^{GkAA,4k} \leftarrow GkAA(\mathcal{Q}, \mathcal{H}, T_k, \mu, 4k)$
    $T_{2k} \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, 2k, T_k^{GkAA,4k}, \mu\right)$
    $T_k \leftarrow PCkPCk'\left(\mathcal{Q}, \mathcal{H}, T, k, T_{2k}, \mu\right)$
  fin tant que
  Retourne T_k
fin procédure

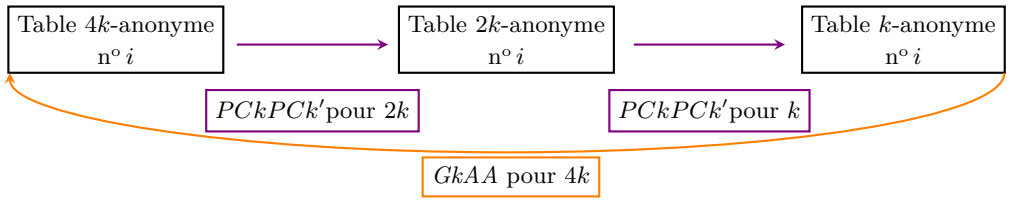


FIGURE 5.14 – Schéma représentant les étapes effectuées lors du i^e tour de l'exécution de *G4kP2kPkconv* pour un k demandé

5.4 Expérimentations

Dans cette section, nous allons comparer les performances des algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$, présentés en section 5.3.2 page 108, avec les résultats de l'algorithme *GkAA*, présenté en section 3.3.2 page 37. Pour cela, nous allons revenir sur le protocole expérimental mis en place en section 5.4.1. Puis nous analyserons les résultats obtenus en section 5.4.2 page 115 en illustrant certaines faiblesses de chaque algorithme évoquées en section 5.3.2 page 108.

5.4.1 Protocole expérimental

Nous souhaitons comparer les tables k-anonymes produites avec les algorithmes de construction présentés en section 5.3.2 page 108 et les tables k-anonymes produites avec l'algorithme *GkAA* en termes d'altération pour une métrique de perte d'information. Pour cela, nous présentons le protocole expérimental suivant.

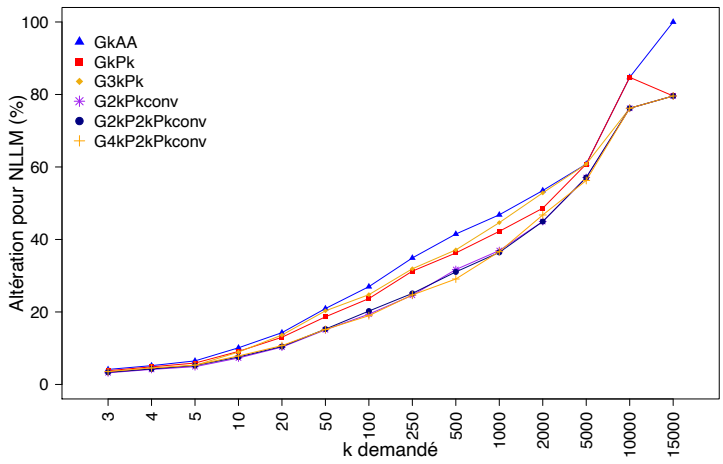
Nous allons mener des expérimentations sur deux tables de 30 162 enregistrements : *Adult data set* et *florida_30162* (cf. section 2.4 page 22). Tous les attributs des tables seront considérés quasi-identifiants.

Pour utiliser l'algorithme *GkAA*, il faut spécifier une métrique de perte d'information. Nous avons choisi *NLLM* (cf. section 3.2 page 32) pour notre étude.

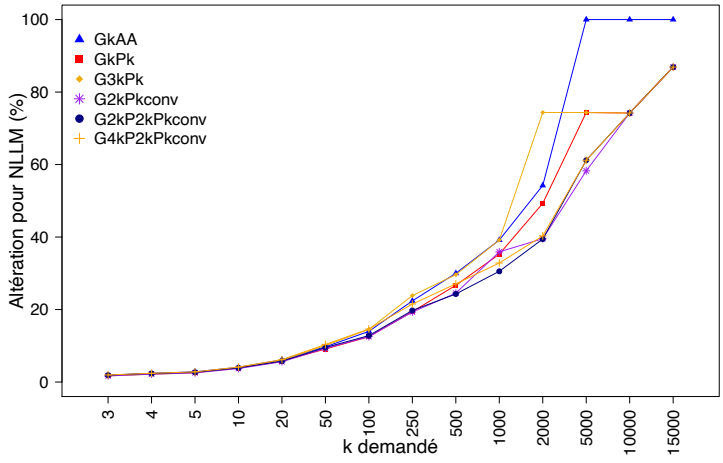
Nous allons évaluer *GkAA* et les cinq algorithmes de construction pour 14 valeurs de k pour $k \in K = [3, 4, 5, 10, 20, 100, 250, 500, 1000, 2000, 5000, 10\,000, 15\,000]$.

Dans un premier temps, nous construisons les 14 versions k-anonymes de *Adult data set* et *florida_30162* avec *GkAA*. Grâce aux caractéristiques de *GkAA* évoquées dans les remarques 3.3.1 page 37 et 3.3.2 page 38, nous n'avons exécuté qu'une fois *GkAA* par table pour $k = \max(K)$ pour obtenir les 28 tables k-anonymes souhaitées.

Pour les algorithmes de construction, nous sommes contraints de lancer une exécution pour chaque valeur de k dans K car les étapes effectuées dans ces algorithmes dépendent de la valeur de k passée en paramètres. Pour



(a) Pour *Adult data set*



(b) Pour *florida_30162*

FIGURE 5.15 – Altération pour *NLLM* des versions k-anonymes de *Adult data set* et *florida_30162* produites avec *GkAA*et les cinq algorithmes de construction

chaque table, pour chaque algorithme de construction, pour chaque $k \in K$, nous produisons donc une version k-anonyme de la table avec l'algorithme de construction.

Pour comparer la qualité des tables k-anonymes produites avec *GkAA* et les cinq algorithmes de construction, nous utilisons l'altération pour *NLLM*(cf. définition 3.1.6 page 31 du chapitre 3 page 25). Dans la suite de ce chapitre, nous dirons *altération* au lieu de *altération pour* NLLM pour alléger les explications. L'altération s'exprime en pourcentage. Si une table k-anonyme obtient une altération de 0%, cela signifie qu'aucune information n'a été perdue par rapport à la table d'origine (aucune généralisation n'a été effectuée). Au contraire, si une table k-anonyme obtient une altération de 100%, cela signifie que toute l'information a été perdue par rapport à la table d'origine (toutes les valeurs quasi-identifiantes de la table ont été généralisées à la racine, la table k-anonyme est donc la table T^* de la définition 2.3.5 page 20).

Pour les algorithmes de construction dits avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, nous fixons la valeur de précision *prec* à 10^{-6}. Dès que l'altération de la table k-anonyme obtenue à la fin d'un tour sera égale à 10^{-6} près à l'altération de la table k-anonyme du tour précedent, l'algorithme s'arrêtera. Si aucune convergence n'est trouvée, l'algorithme s'arrête après 24 heures d'exécution.

Pour chaque table, nous avons réalisé un graphique représentant l'altération des tables k-anonymes produites avec les algorithmes en fonction du k demandé : figure 5.15a pour *Adult data set* et figure 5.15b pour *florida_30162*. L'échelle de l'axe des abscisses n'est volontairement pas respectée pour que les résultats pour les petites valeurs

Algorithmes	*VMN*
G4kP2kPkconv	62,9938
G2kP2kPkconv	63,0187
G2kPkconv	63,0472
G3kPk	65,7643
GkPk	67,8853
GkAA	72,3762

(a) Pour *Adult data set*

Algorithmes	*VMN*
G2kPkconv	62,8301
G2kP2kPkconv	63,3326
G4kP2kPkconv	63,6718
GkPk	68,4481
G3kPk	72,1331
GkAA	87,0531

(b) Pour *florida_30162*

TABLEAU 5.1 – Valeurs moyennes normalisées calculées sur l'intervalle [3,15 000] de *GkAA* et des cinq algorithmes de construction sur *Adult data set* et *florida_30162*

de k soient lisibles. Sur chaque graphique, il y a six courbes : la courbe bleue correspond aux résultats obtenus pour *GkAA* et les autres courbes correspondent aux résultats obtenus pour les cinq algorithmes de construction. Un point d'une courbe représente l'altération de la version k-anonyme de la table produite avec l'algorithme correspondant à la courbe. Par exemple, sur le graphique 5.15a page précédente, le quatrième point de la courbe jaune représente l'altération de la version 20-anonyme de *Adult data set* produite avec l'algorithme *G3kPk*.

Pour chaque algorithme de construction avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, un point de la courbe correspond à l'altération minimale observée lors de tous les tours de l'exécution de l'algorithme. Ce n'est pas forcément l'altération de la table k-anonyme obtenue au dernier tour comme nous le verrons dans la section 5.4.2.

Comme dans la section 3.4 page 40 du chapitre 3 page 25, pour chaque table, pour chaque algorithme, nous aimerions synthétiser en une unique valeur les résultats d'altération obtenus par les versions k-anonymes de la table produites avec l'algorithme.

Nous utilisons la notion d'aire sous la courbe qui permet de calculer la valeur moyenne d'une courbe sur un intervalle donné. En effet, pour une fonction f continue sur un intervalle $[a, b] \subset \mathbb{R}$, la valeur moyenne de f sur $[a, b]$ est $m \in \mathbb{R}$ tel que

$$m = \frac{1}{b-a} \int_a^b f(x)dx.$$

Nous utilisons la méthode des trapèzes (cf. définition 3.4.4 page 42) pour approcher l'aire sous la courbe. Dans ce chapitre, nous supposons que la méthode des trapèzes fournit une bonne approximation de l'aire sous la courbe.

Pour chaque table, pour chaque algorithme, nous calculons donc une valeur moyenne des résultats d'altération obtenus par les versions k-anonymes de la table produites avec l'algorithme. Nous appelons cette valeur *VMN*, pour *Valeur Moyenne Normalisée*, de l'algorithme pour l'altération sur la table. Ainsi, pour chaque table, nous pouvons dresser les tableaux de *VMN* calculées sur l'intervalle [3, 15 000] des algorithmes : tableau 5.1a pour *Adult data set* et tableau 5.1b pour *florida_30162*. La *VMN* s'exprime en pourcentage, 0% étant le meilleur des cas et 100% le pire.

Par exemple, nous avons produit 14 versions k-anonymes de *Adult data set* avec *GkPk* pour $k \in K$. Pour chaque table k-anonyme, nous calculons son altération. Nous obtenons donc 14 valeurs d'altération (cf. courbe rouge du graphique 5.15a page ci-contre). Nous calculons ensuite la valeur moyenne de cette courbe sur [3, 15 000] grâce à la méthode précédente : les versions k-anonymes de *Adult data set* produites avec *GkPk* ont une altération de 67,8853% en moyenne. On peut aussi dire que la *VMN* calculée sur [3, 15 000] de *GkPk* est de 67,8853%. Ce résultat est à retrouver dans le tableau 5.1a.

5.4.2 Analyse des résultats

Dans cette section, nous allons analyser les résultats obtenus par les algorithmes de construction.

Pour des *VMN* calculées sur [3, 15 000], les algorithmes de construction ont sensiblement les mêmes performances sur les deux tables (tableaux 5.1a et 5.1b). Les trois algorithmes de construction avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, obtiennent les meilleurs résultats avec une *VMN* calculée sur [3, 15 000] d'environ 63%. Les algorithmes de construction *GkPk* et *G3kPk* ont des *VMN* plus élevées que les algorithmes de construction avec convergence mais elles restent inférieures aux *VMN* de *GkAA*. En effet, sur *Adult data set* (tableau 5.1a), *GkPk* a une *VMN* sur [3, 15 000] d'environ 68% et *G3kPk* a une *VMN* sur [3, 15 000] d'environ 66% contre environ 72% pour *GkAA*. De même, sur *florida_30162* (tableau 5.1b), *GkPk* a une *VMN* sur [3, 15 000] d'environ 68% et *G3kPk* a une *VMN* sur [3, 15 000] d'environ 72% contre environ 87% pour *GkAA*.

Algorithmes	*VMN*
G2kP2kPkconv	34,744
G4kP2kPkconv	34,804
G2kPkconv	34,9582
GkPk	39,7153
G3kPk	41,9919
GkAA	44,2246

(a) Pour *Adult data set*

Algorithmes	*VMN*
G2kP2kPkconv	28,7996
G4kP2kPkconv	30,6909
G2kPkconv	30,813
GkPk	33,438
GkAA	37,1434
G3kPk	42,2887

(b) Pour *florida_30162*

TABLEAU 5.2 – Valeurs moyennes normalisées calculées sur l'intervalle [3,2000] de *GkAA* et des cinq algorithmes de construction sur *Adult data set* et *florida_30162*

Nous constatons que *GkAA* a des *VMN* sur $[3, 15\,000]$ nettement plus élevées que celles des cinq algorithmes de construction. Cela peut s'expliquer par le fait que les versions k-anonymes de *Adult data set* pour $k \in \{15\,000\}$ et les versions k-anonymes de *florida_30162* pour $k \in \{5000, 10\,000, 15\,000\}$ produites avec *GkAA* ont une altération de 100% (cf. graphiques 5.15a page 114 et 5.15b page 114). Ainsi, comme les valeurs de k concernées représentent une grande majorité de l'intervalle $[3, 15\,000]$, les *VMN* sur $[3, 15\,000]$ de *GkAA* sont élevées.

Pour comparer les résultats obtenus par les algorithmes sur des valeurs de k plus raisonnables, nous calculons la *VMN* sur l'intervalle $[3, 2000]$ de chaque algorithme. Ainsi, les grandes valeurs de k, pas forcément pertinentes sur des tables de 30 162 enregistrements, ne sont plus considérées. Nous calculons donc l'altération moyenne des versions k-anonymes de chaque table pour k entre 3 et 2000 pour chaque algorithme. Les tableaux 5.2a et 5.2b présentent les résultats de *VMN* calculées sur $[3, 2000]$ de chaque algorithme pour *Adult data set* et *florida_30162* respectivement.

La seule différence notable avec les résultats de *VMN* calculées sur $[3, 15\,000]$ est observée pour *florida_30162*. Dans le tableau 5.2b, nous observons que la *VMN* calculée sur $[3, 2000]$ de l'algorithme de construction *G3kPk* est supérieure à celle des autres algorithmes et notamment à celle de *GkAA*. Sa *VMN* est d'environ 42% alors que celle de *GkAA* est d'environ 37%. Sur le graphique 5.15b page 114, nous observons que le point correspondant à $k = 2000$ pour *G3kPk* (courbe jaune) est bien au-dessus du point correspondant à $k = 2000$ pour *GkAA* (courbe bleue). Nous constatons à nouveau qu'un mauvais comportement d'un algorithme pour une valeur de k peut entraîner une *VMN* élévée.

5.4.2.1 Illustration de limites des algorithmes de construction

Nous allons maintenant étudier les algorithmes de construction individuellement pour illustrer certaines faiblesses mentionnées en section 5.3.2 page 108.

Dans l'algorithme de construction *GkPk*, nous cherchons à construire une table k-anonyme en k-partitionnant les classes d'équivalence de la table k-anonyme produite avec *GkAA*.

Un inconvénient de cet algorithme est que, si aucune classe d'équivalence de la table k-anonyme produite avec *GkAA* n'est de taille supérieure à $2k$, aucun k-partitionnement non trivial ne pourra être calculé. Ainsi, l'altération de la table k-anonyme produite avec *GkPk* sera égale à l'altération de la table k-anonyme produite avec *GkAA*. Par exemple, ce comportement est constaté pour $k \in \{5000, 10\,000\}$ sur *Adult data set*. Nous observons sur le graphique 5.15a page 114 que les points pour $k \in \{5000, 10\,000\}$ de *GkPk* (courbe rouge) et *GkAA* (courbe bleue) sont confondus. La version 5000-anonyme de *Adult data set* produite avec *GkAA* a cinq classes d'équivalence. Sa classe de taille maximale contient 7565 enregistrements, ce qui est inférieur à $2 \times 5000 = 10\,000$. Par conséquent, aucun 5000-partitionnement en au moins deux éléments distincts ne pourra être calculé sur ces classes d'équivalence avec la *PCkPCk′*. Donc la version 5000-anonyme de *Adult data set* produite avec *GkPk* est la même que celle produite avec *GkAA*. De même, la classe de taille maximale de la version 10 000-anonyme de *Adult data set* produite avec *GkAA* contient 18 830 enregistrements, ce qui est inférieur à $2 \times 10000 = 20000$. La *PCkPCk′* ne pourra donc pas trouver de k-partitionnements non triviaux des enregistrements de ces classes d'équivalence. Donc la version 10 000-anonyme de *Adult data set* produite avec *GkPk* est la même que celle produite avec *GkAA*.

Dans l'algorithme de construction *G3kPk*, nous cherchons à construire une table k-anonyme en k-partitionnant les classes d'équivalence de la table $3k$-anonyme produite avec *GkAA*.

Un inconvénient de cet algorithme est que, si une classe d'équivalence considérée dans la *PCkPCk′* est de trop grande taille par rapport au k demandé, aucun bon k-partitionnement ne peut être trouvé en un temps raisonnable. Illustrons cet inconvénient en étudiant le cas $k = 2000$ sur *florida_30162*.

Comme mentionné dans la section 5.4.2 page précédente, la version 2000-anonyme de *florida_30162* produite

avec *G3kPk* a une altération plus élevée que celle de la version 2000-anonyme de *florida_30162* produite avec *GkAA* : l'altération de la table est d'environ 74% pour *G3kPk* et d'environ 54% pour *GkAA*. Pour $k = 2000$, la table $3k$-anonyme de départ en entrée de l'algorithme *G3kPk* est la version 6000-anonyme de *florida_30162* produite avec *GkAA*. Nous cherchons ensuite à 2000-partitionner les classes d'équivalence de cette table 6000-anonyme avec la *PCkPCk′*. Or, l'altération de la table 6000-anonyme est de 100% : elle ne contient qu'une seule classe d'équivalence de taille 30 162 dans laquelle tous les enregistrements sont généralisés aux niveaux les plus élevés. L'ensemble minimal des groupes de généralisation de taille supérieure à 2000 de cette classe est de cardinal 170. Avec nos configurations, le solveur *CP-SAT* ne parvient pas à résoudre notre problème d'optimisation et le 2000-partitionnement effectué est un 2000-partitionnement en deux éléments distincts de coût de généralisation minimal. Les éléments de ce 2000-partitionnement sont de taille 13 962 et 16 200 ce qui est très éloigné du $k = 2000$ initialement demandé. Ceci illustre bien une faiblesse de ce type d'algorithme : si la classe d'équivalence considérée est de trop grande taille par rapport au k-demandé, un bon k-partitionnement ne peut pas être trouvé en un temps raisonnable.

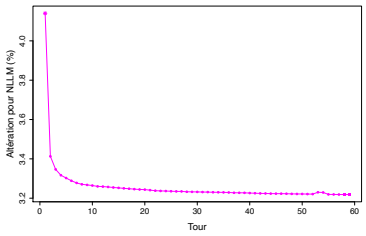
Intéressons nous maintenant aux algorithmes de construction avec convergence : *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*. Dans ces algorithmes, des étapes d'anonymisation avec *GkAA* et de partitionnement de classes d'équivalence avec la *PCkPCk′* sont répétées jusqu'à atteindre une convergence de l'altération des tables k-anonymes produites ou que le temps d'exécution total excède 24 heures. À chaque tour, l'algorithme *G2kPkconv* consiste à $2k$-anonymiser avec *GkAA* la table k-anonyme obtenue à la fin du tour précédent puis à k-partitionner avec la *PCkPCk′* les classes d'équivalence de la table $2k$-anonyme. L'algorithme *G2kP2kPkconv* ajoute une étape de $2k$-partitionnement des classes d'équivalence de la table $2k$-anonyme avant de k-partitionner les classes d'équivalence de la table résultante. Dans l'algorithme *G4kP2kPkconv*, à chaque début de tour, nous $4k$-anonymisons la table k-anonyme obtenue à la fin du tour précédent puis nous $2k$-partitionnons les classes d'équivalence de la table $4k$-anonyme et nous terminons le tour en k-partitionnant les classes d'équivalence de la table $2k$-anonyme résultant de l'étape précédente.

Les comportements de ces algorithmes de construction avec convergence sont parfois chaotiques. Pour visualiser ces comportements, nous avons réalisé des graphiques pour chacun des algorithmes avec convergence pour chaque table et pour des valeurs de k entre 3 et 2000 : figures 5.16 page suivante et 5.19 page 121 pour *G2kPkconv*, figures 5.17 page 119 et 5.20 page 122 pour *G2kP2kPkconv* et figures 5.18 page 120 et 5.21 page 123 pour *G4kP2kPkconv*. Un graphique représente les altérations des tables k-anonymes produites lors des tours de l'exécution de l'algorithme de construction avec convergence sur la table et pour le k correspondant au graphique. Le premier point de la courbe correspond à l'altération de la table k-anonyme produite avec *GkAA* et est symbolisé par une étoile. Afin de faciliter la lecture du graphique, nous avons symbolisé par un carré le point correspondant à l'altération minimale observée. Quand l'altération minimale est observée pour *GkAA*, nous laisserons le symbole correspondant à *GkAA*. Par exemple, le graphique 5.16a page suivante présente l'altération de la version 3-anonyme de *Adult data set* produite avec *GkAA* (premier point de la courbe) et les altérations des tables 3-anonymes produites lors des 58 tours de l'exécution de *G2kPkconv* sur *Adult data set* pour $k = 3$. Le minimum d'altération est atteint aux 57^{e} et 58^{e} tours de cette exécution de *G2kPkconv*.

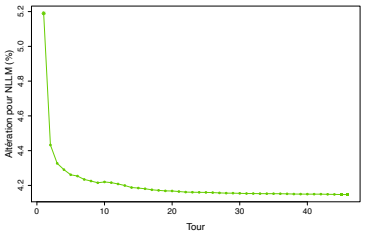
Lors de certaines exécutions des algorithmes de construction avec convergence, l'altération minimale n'est pas atteinte au dernier tour de l'exécution. Cependant, l'algorithme s'arrête par convergence des altérations des deux dernières tables k-anonymes. Par exemple, lors de l'exécution de *G4kP2kPkconv* sur *florida_30162* pour $k = 50$ (cf. graphique 5.21f page 123), les tables 50-anonymes des deux derniers tours de l'exécution ont la même altération à 10^{-6} près. L'algorithme s'arrête donc après quatre heures d'exécution. En revanche, l'altération minimale est observée pour la table 50-anonyme obtenue à la fin du premier tour de l'exécution. Pour continuer l'exécution de l'algorithme et chercher de meilleures tables k-anonymes, nous pourrions effectuer des fusions de classes d'équivalence aléatoirement après l'application de *GkAA* : partir d'une table plus généralisée pourrait relancer l'exécution et permettre de se rapprocher d'une table k-anonyme optimale en termes d'altération.

Une convergence de l'altération des tables k-anonymes n'est pas toujours observée. Dans ce cas, l'algorithme tournera pendant 24 heures avant de s'arrêter. Par exemple, l'exécution de l'algorithme *G2kPkconv* sur *Adult data set* pour $k = 20$ dure 24 heures et effectue 338 tours ce qui représente le calcul de 338 tables 20-anonymes (cf. graphique 5.16e page suivante). L'altération de la table 20-anonyme du tour 337 est d'environ 10,352 032% et l'altération de la table 20-anonyme du tour 338 est d'environ 10,352 816%.

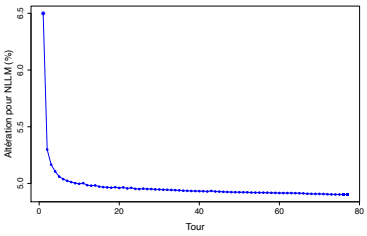
Il peut arriver, lors de certaines exécutions des algorithmes de construction avec convergence, que les altérations des tables k-anonymes obtenues à la fin des tours ne soient pas monotones. Les graphiques 5.21d page 123 à 5.21h page 123 sont de bonnes illustrations de ce phénomène. Nous observons que les variations des courbes sont très irrégulières et qu'elles ne sont pas toujours décroissantes. Nous aurions préféré observer des décroissances des altérations signifiant qu'à chaque tour, la table k-anonyme obtenue est de meilleure qualité en termes d'altération que celle du tour précédent. Nous observons ce comportement idéal pour *G2kPkconv* sur *Adult data set* pour $k \in \{3, 4, 5, 10, 20, 50\}$ par exemple (cf. graphiques de la figure 5.16 page suivante). Sur ces



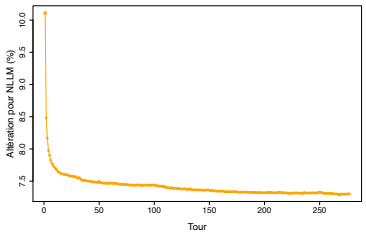
(a) Pour $k = 3$



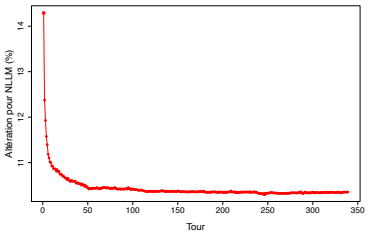
(b) Pour $k = 4$



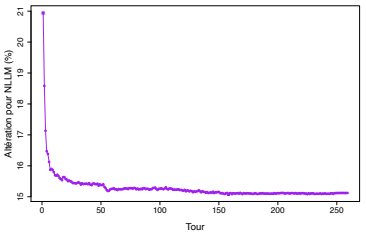
(c) Pour $k = 5$



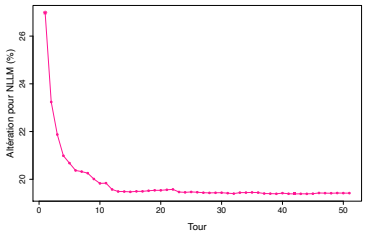
(d) Pour $k = 10$



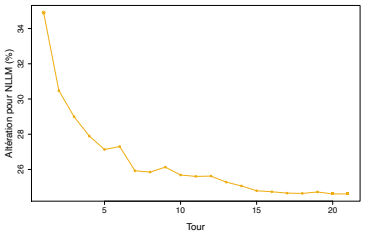
(e) Pour $k = 20$



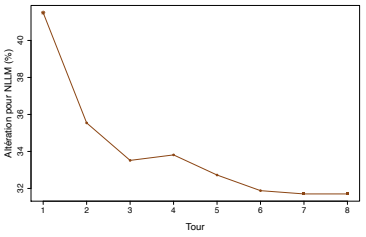
(f) Pour $k = 50$



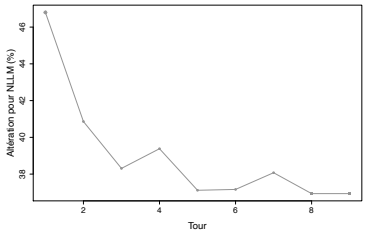
(g) Pour $k = 100$



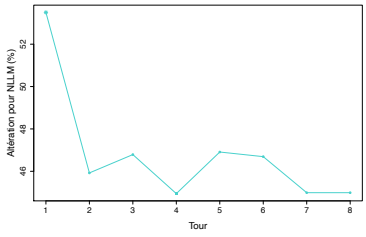
(h) Pour $k = 250$



(i) Pour $k = 500$

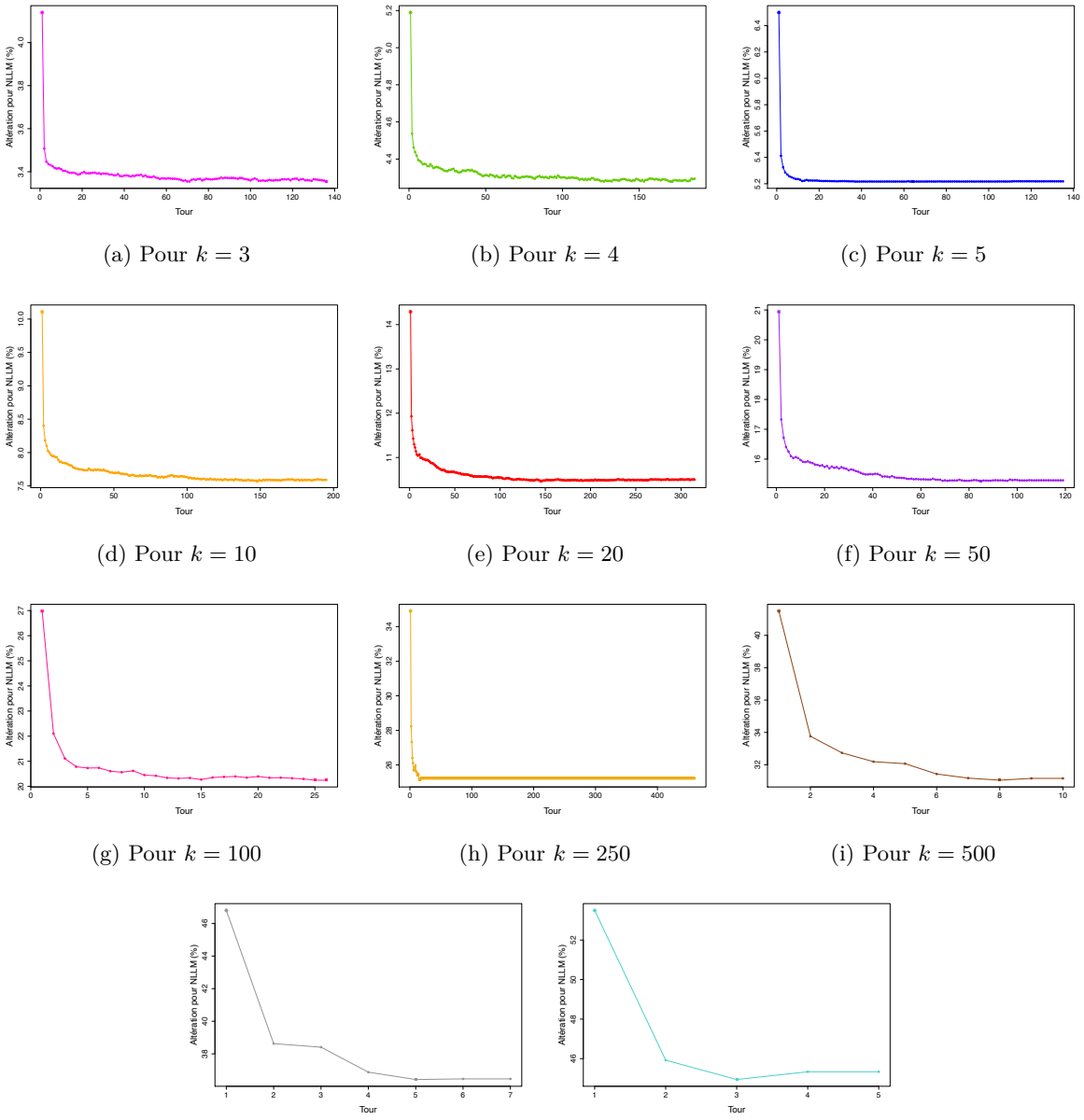


(j) Pour $k = 1000$



(k) Pour $k = 2000$

FIGURE 5.16 – Altération pour *NLLM* des versions k-anonymes successives de *Adult data set* produites lors de l'exécution de *G2kPkconv* pour k entre 3 et 2000



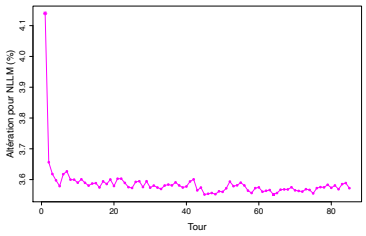
(a) Pour $k = 3$ (b) Pour $k = 4$ (c) Pour $k = 5$

(d) Pour $k = 10$ (e) Pour $k = 20$ (f) Pour $k = 50$

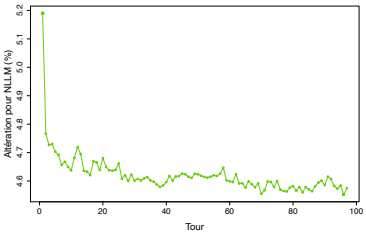
(g) Pour $k = 100$ (h) Pour $k = 250$ (i) Pour $k = 500$

(j) Pour $k = 1000$ (k) Pour $k = 2000$

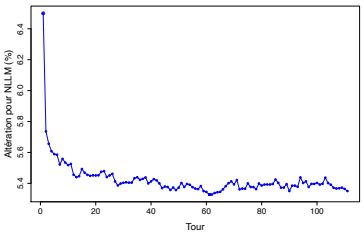
FIGURE 5.17 – Altération pour *NLLM* des versions k-anonymes successives de *Adult data set* produites lors de l'exécution de *G2kP2kPkconv* pour k entre 3 et 2000



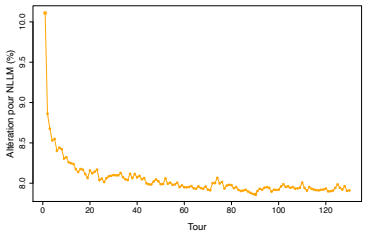
(a) Pour $k = 3$



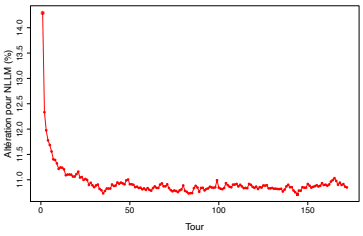
(b) Pour $k = 4$



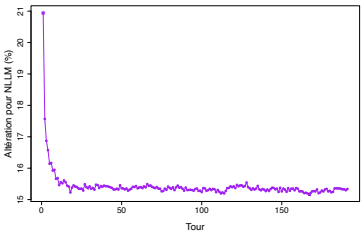
(c) Pour $k = 5$



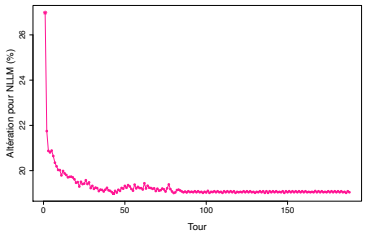
(d) Pour $k = 10$



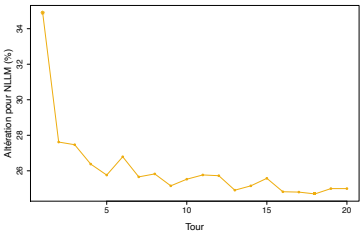
(e) Pour $k = 20$



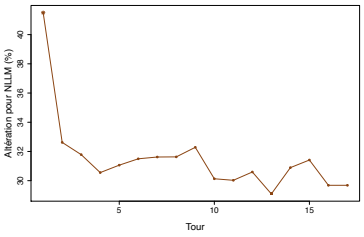
(f) Pour $k = 50$



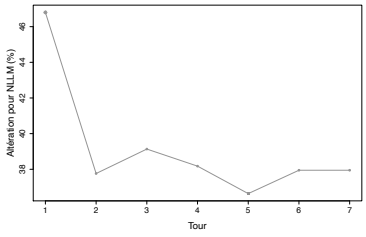
(g) Pour $k = 100$



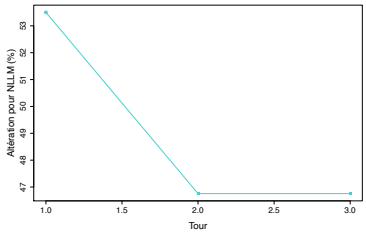
(h) Pour $k = 250$



(i) Pour $k = 500$

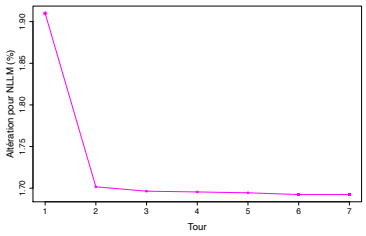


(j) Pour $k = 1000$

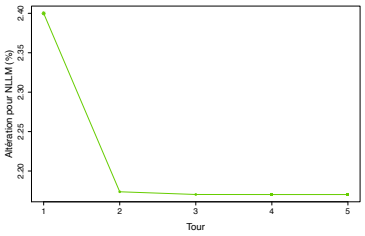


(k) Pour $k = 2000$

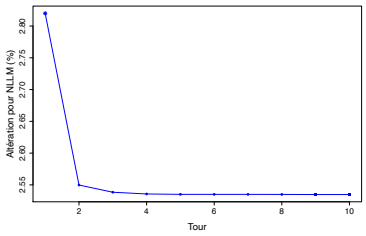
FIGURE 5.18 – Altération pour *NLLM* des versions k-anonymes successives de *Adult data set* produites lors de l'exécution de *G4kP2kPkconv* pour k entre 3 et 2000



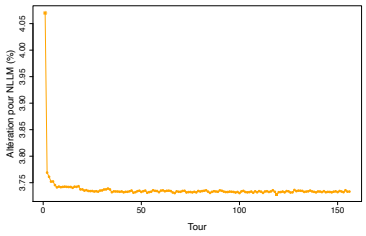
(a) Pour $k = 3$



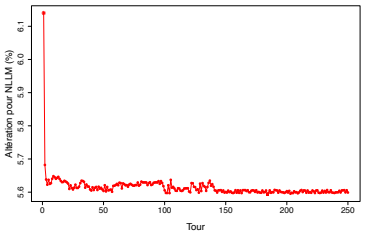
(b) Pour $k = 4$



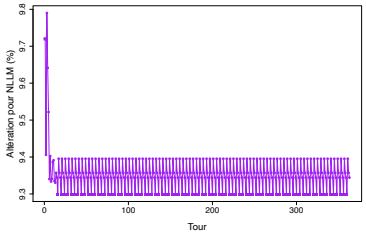
(c) Pour $k = 5$



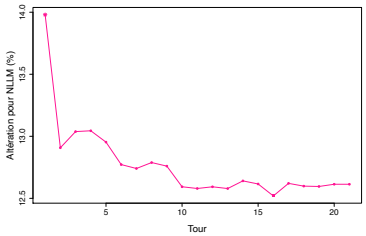
(d) Pour $k = 10$



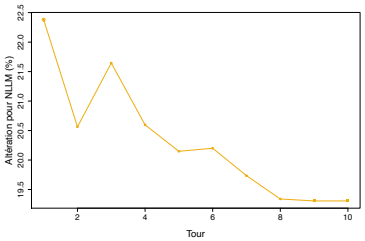
(e) Pour $k = 20$



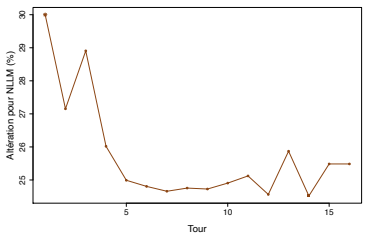
(f) Pour $k = 50$



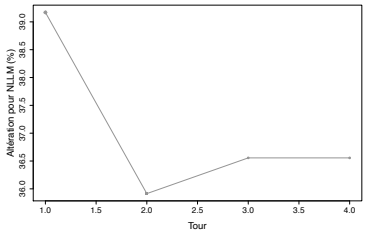
(g) Pour $k = 100$



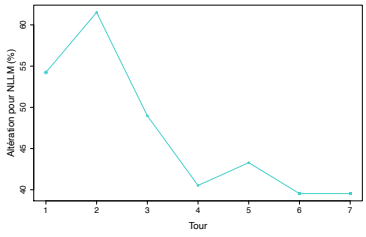
(h) Pour $k = 250$



(i) Pour $k = 500$

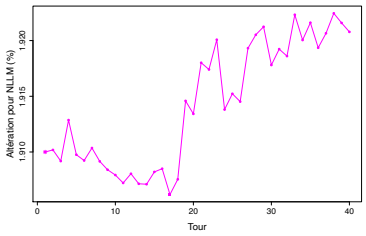


(j) Pour $k = 1000$

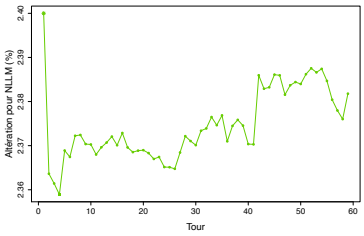


(k) Pour $k = 2000$

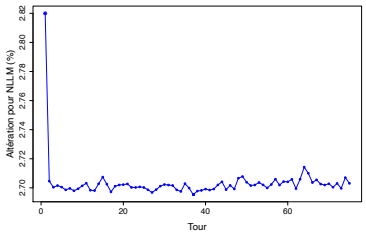
FIGURE 5.19 – Altération pour *NLLM* des versions k-anonymes successives de *florida_30162* produites lors de l'exécution de *G2kPkconv* pour k entre 3 et 2000



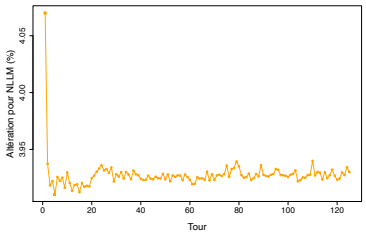
(a) Pour $k = 3$



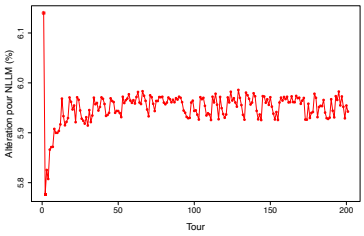
(b) Pour $k = 4$



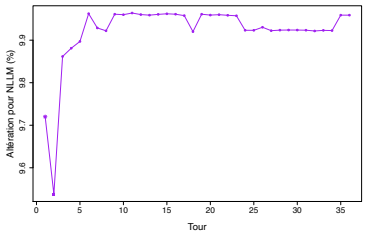
(c) Pour $k = 5$



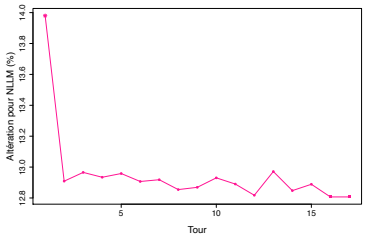
(d) Pour $k = 10$



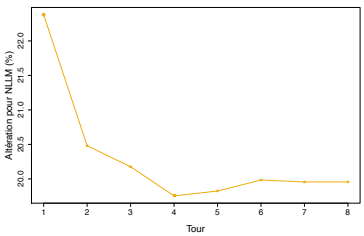
(e) Pour $k = 20$



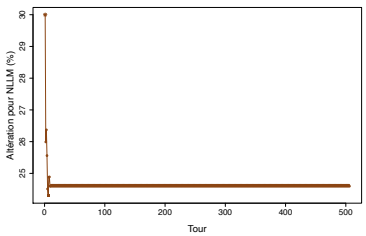
(f) Pour $k = 50$



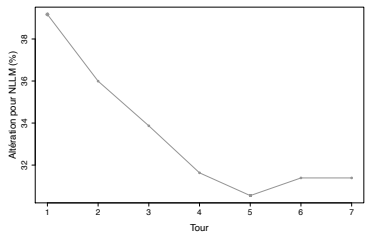
(g) Pour $k = 100$



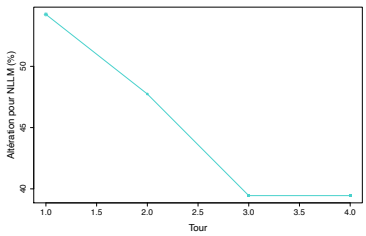
(h) Pour $k = 250$



(i) Pour $k = 500$

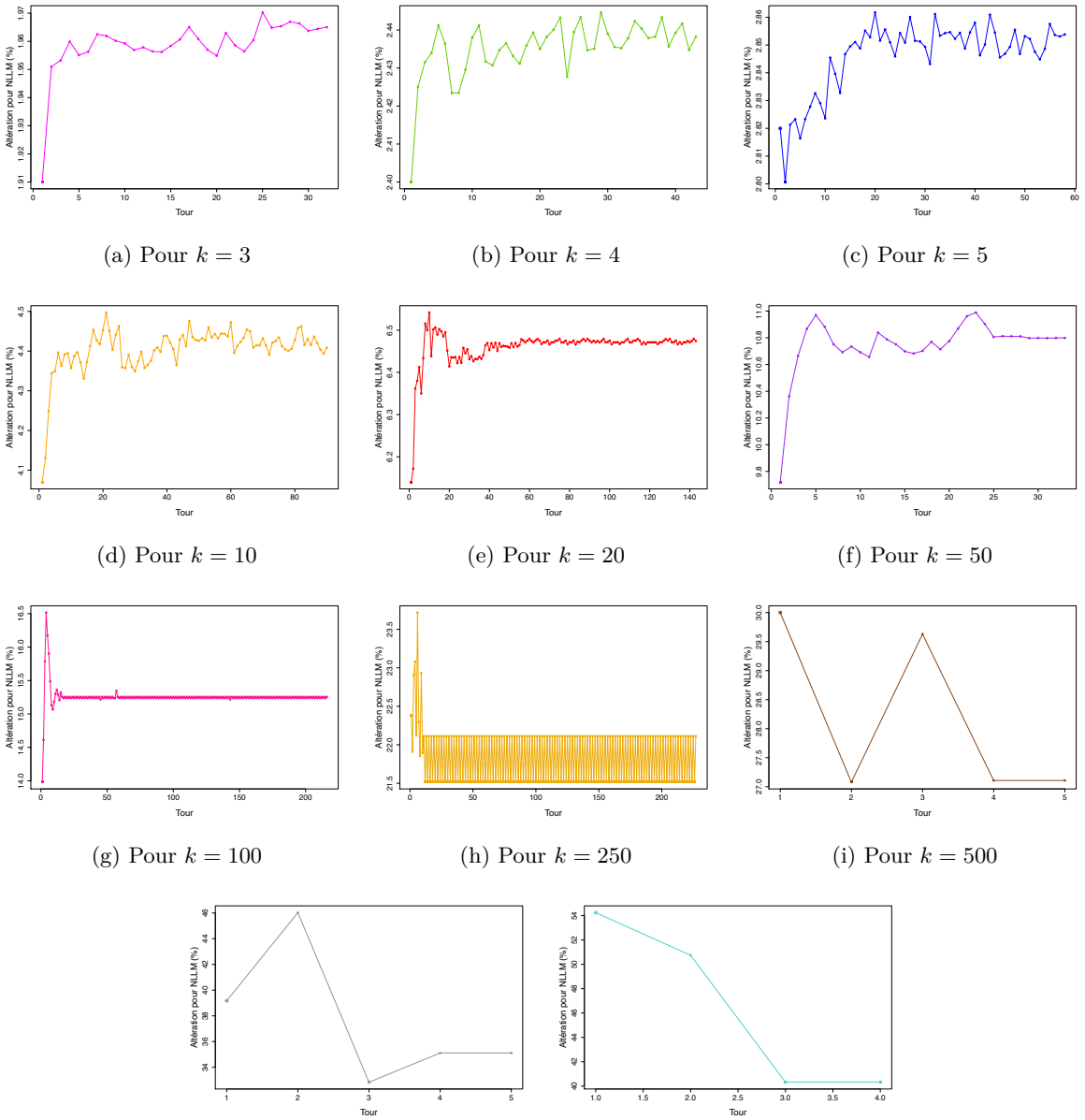


(j) Pour $k = 1000$



(k) Pour $k = 2000$

FIGURE 5.20 – Altération pour *NLLM* des versions k-anonymes successives de *florida_30162* produites lors de l'exécution de *G2kP2kPkconv* pour k entre 3 et 2000



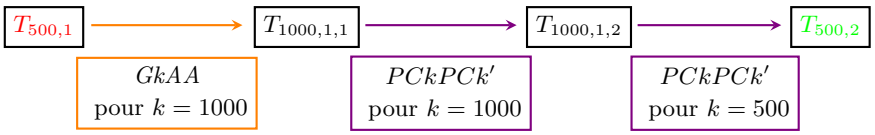
(a) Pour $k = 3$ (b) Pour $k = 4$ (c) Pour $k = 5$

(d) Pour $k = 10$ (e) Pour $k = 20$ (f) Pour $k = 50$

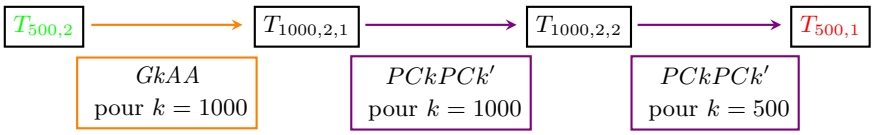
(g) Pour $k = 100$ (h) Pour $k = 250$ (i) Pour $k = 500$

(j) Pour $k = 1000$ (k) Pour $k = 2000$

FIGURE 5.21 – Altération pour *NLLM* des versions k-anonymes successives de *florida_30162* produites lors de l'exécution de *G4kP2kPkconv* pour k entre 3 et 2000



(a) Au 10e tour



(b) Au 11e tour

FIGURE 5.22 – Schémas récapitulant les étapes effectuées lors de deux tours de l'exécution de *G2kP2kPkconv* pour $k = 500$ sur *florida_30162*

graphiques, nous notons une diminution de l'altération de la table k-anonyme pour une grande majorité des tours.

De plus, l'altération minimale est parfois observée pour la table k-anonyme produite avec *GkAA*. Cela signifie qu'aucune des tables k-anonymes calculées lors de l'exécution de l'algorithme de construction par convergence n'est de meilleure qualité en termes d'altération que celle produite avec *GkAA*. Par exemple, l'algorithme *G4kP2kPkconv* appliqué à *florida_30162* pour $k \in \{3, 4, 10, 20, 50, 100\}$ ne donne pas de meilleure version k-anonyme de *florida_30162* que celle produite avec *GkAA* en termes d'altération (cf. graphiques de la figure 5.21 page précédente). Le problème réside peut-être dans le fait que les tables $4k$-anonymes considérées au début de chaque tour sont déjà beaucoup altérées et que les traitements avec la $PCkPCk'$ ne sont pas suffisants pour obtenir une bonne table k-anonyme.

Nous observons parfois des motifs réguliers dans les altérations successives. Par exemple, quand nous appliquons *G2kP2kPkconv* sur *florida_30162* pour $k = 500$ (cf. graphique 5.20i page 122), nous observons une alternance entre deux valeurs à partir du 10e tour et ce jusqu'à l'arrêt de l'algorithme après 24 heures d'exécution et 505 tours. Les deux tables 500-anonymes correspondantes ont des altérations d'environ 24,627 487% et 24,595 182%. Notons les $T_{500,1}$ et $T_{500,2}$ respectivement. Décrivons les étapes effectuées lors des 10e et 11e tours de l'exécution de *G2kP2kPkconv* pour $k = 500$ sur *florida_30162*. Lors du 10e tour de l'exécution, quand nous 1000-anonymisons la table $T_{500,1}$, nous obtenons une table 1000-anonyme d'altération 29,962 314% notée $T_{1000,1,1}$. Quand nous 1000-partitionnons les classes d'équivalence de $T_{1000,1,1}$, nous obtenons une table 1000-anonyme d'altération 29,946 097% notée $T_{1000,1,2}$. Quand nous 500-partitionnons les classes d'équivalence de $T_{1000,1,2}$, nous obtenons la table 500-anonyme $T_{500,2}$. Lors du 11e tour de l'exécution, quand nous 1000-anonymisons la table $T_{500,2}$, nous obtenons une table 1000-anonyme d'altération 30,133 196% notée $T_{1000,2,1}$. Quand nous 1000-partitionnons les classes d'équivalence de $T_{1000,2,1}$, nous obtenons une table 1000-anonyme d'altération 30,116 98% notée $T_{1000,2,2}$. Quand nous 500-partitionnons les classes d'équivalence de $T_{1000,2,2}$, nous retrouvons la table 500-anonyme $T_{500,1}$. Comme tous les processus appliqués lors d'un tour sont déterministes, les étapes effectuées au 12e tour seront les mêmes qu'au 10e tour et donc les étapes effectuées au 13e tour seront les mêmes que celles du 11e tour etc...

Des schémas récapitulant les étapes effectuées lors des 10e et 11e tours de l'exécution de *G2kP2kPkconv* pour $k = 500$ sur *florida_30162* sont à retrouver en figure 5.22.

De même, pour *G2kPkconv* appliqué sur *florida_30162* pour $k = 50$ (cf. graphique 5.19f page 121), nous notons une alternance entre quatre tables 50-anonymes à partir du 13e tour. Pour sortir de tels schémas, nous pourrions décider d'appliquer de plus fortes généralisations que nécessaires lors de la l'étape de $2k$-anonymisation de début de tour. La table $2k$-anonyme obtenue serait alors différente et permettrait peut-être de partir dans une autre direction de recherche.

5.4.2.2 Analyse des résultats obtenus sur des sous-intervalles de $[3, 15\,000]$

Dans la section 5.4.2.1 page 116, nous avons réalisé des graphiques représentant l'altération des tables k-anonymes obtenues lors des exécutions des algorithmes de construction avec convergence sur *Adult data set* et *florida_30162* pour des valeurs de k entre 3 et 2000 : figures 5.16 page 118 et 5.19 page 121 pour *G2kPkconv*, figures 5.17 page 119 et 5.20 page 122 pour *G2kP2kPkconv* et figures 5.18 page 120 et 5.21 page précédente pour *G4kP2kPkconv*.

En étudiant ces graphiques, nous avons illustré certaines faiblesses des algorithmes de construction avec convergence. D'aute part, sur les graphiques de la figure 5.21 page précédente, nous constatons que *G4kP2kPkconv*

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
GkAA	7,4336	20,5383	40,2956	65,0456
GkPk	6,7471	18,2689	35,7667	63,7078
G3kPk	6,2979	19,4322	37,0219	62,4089
G2kPkconv	5,5296	14,7831	30,5551	58,5767
G2kP2kPkconv	5,7922	15,1595	30,3244	58,5483
G4kP2kPkconv	5,9956	14,8301	29,3682	58,6056

(a) Pour *Adult data set*

	[3, 10]	[10, 100]	[100, 1000]	[1000, 10000]
GkAA	3,1414	9,7939	29,5189	86,4517
GkPk	2,955	9,02	26,2814	66,5811
G3kPk	2,9071	10,0528	29,7339	72,3661
G2kPkconv	2,8488	9,0608	25,5256	57,2947
G2kP2kPkconv	3,0249	9,2975	24,063	58,2758
G4kP2kPkconv	3,161	10,2651	26,4036	58,5972

(b) Pour *florida_30162*

TABLEAU 5.3 – Valeurs moyennes normalisées des six algorithmes pour l'altération pour *NLLM* calculées sur plusieurs intervalles sur les deux tables

ne parvient pas à obtenir de versions k-anonymes de *florida_30162* d'altération moins élevée que celle de la version k-anonyme de *florida_30162* produite avec *GkAA* pour $k \in \{3, 4, 10, 20, 50, 100\}$. Nous nous sommes donc demandés si les conclusions de notre analyse sur les intervalles $[3, 15\,000]$ et $[3, 2000]$ dans la section 5.4.2 page 115 seraient différentes sur des intervalles dans lesquels les valeurs de k sont dans le même ordre de grandeur.

Pour chaque table, pour *GkAA* et les cinq algorithmes de construction, nous avons calculé leur *VMN* pour l'altération sur quatre sous-intervalles : $[3, 10]$, $[10, 100]$, $[100, 1000]$ et $[1000, 10\,000]$. Les tableaux 5.3a et 5.3b résument les résultats obtenus. Pour un même sous-intervalle, les *VMN* sont classées de la moins élevée à la plus élevée : le classement est représenté par des couleurs du vert au rouge. Par exemple, dans le tableau 5.3b, nous lisons que l'algorithme *G2kPkconv* a une *VMN* calculée sur $[3, 10]$ de 2,8488% pour l'altération sur *florida_30162*. Il s'agit du meilleur algorithme sur cet intervalle pour *florida_30162* car la case correspondante est vert foncé.

Sur *Adult data set*, nous observons dans le tableau 5.3a que les résultats de *VMN* obtenus sur les sous-intervalles sont similaires à ceux obtenus sur l'intervalle $[3, 2000]$. Les trois algorithmes de construction avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, ont les meilleurs résultats sur cette table. *GkAA* est le moins bon algorithme quelque soit le sous-intervalle considéré. Les écarts sont nets pour trois des quatre sous-intervalles étudiés. Par exemple, sur $[10, 100]$, les algorithmes de construction avec convergence ont une *VMN* d'environ 15% alors que les autres algorithmes ont une *VMN* entre 18 et 20%. Nous constatons par ailleurs que les résultats des algorithmes de construction avec convergence sont très proches quelque soit le sous-intervalle considéré. Par exemple, sur $[1000, 10\,000]$, la *VMN* de *G2kP2kPkconv* est de 58,5483%, celle de *G2kPkconv* est de 58,5767% et celle de *G4kP2kPkconv* est de 58,6056%.

Sur *florida_30162*, les résultats obtenus sont moins tranchés. Sur $[3, 10]$ et $[10, 100]$, nous observons dans le tableau 5.3b que les *VMN* des algorithmes sont très proches : l'écart entre la *VMN* la plus basse et la *VMN* la plus haute est d'environ 0,3 point sur $[10, 100]$ et d'environ 1,2 points sur $[10, 100]$. En revanche, l'algorithme par construction avec convergence *G4kP2kPkconv* obtient les moins bons résultats sur ces deux sous-intervalles. D'autre part, l'algorithme par construction avec convergence *G2kPkconv* est toujours parmi les deux meilleurs algorithmes sur les quatre sous-intervalles considérés : il est premier sur $[3, 10]$ et $[1000, 10\,000]$ et deuxième sur $[10, 100]$ et $[100, 1000]$. Cela rejoint les résultats obtenus sur *Adult data set*.

Sur $[1000, 10\,000]$, *GkAA* a une *VMN* nettement supérieure à celle des algorithmes de construction : sa *VMN* est de 86,4517% alors que celle du deuxième moins bon algorithme est de 72,3661%. De plus, sur ce sous-intervalle, les résultats des algorithmes de construction avec convergence sur *florida_30162* sont similaires à ceux obtenus sur *Adult data set* sur ce même sous-intervalle.

Pour conclure sur cette analyse, bien que les algorithmes de construction avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, obtiennent les meilleurs résultats sur une majorité des sous-intervalles sur les deux tables, il est à noter que, pour de petites valeurs de k, l'algorithme *G4kP2kPkconv* obtient les moins bonnes *VMN* sur *florida_30162*. Il existe donc des cas pour lesquels les algorithmes de construction peuvent être moins efficaces que *GkAA*. En revanche, nous constatons que, globalement, les tables k-anonymes produites avec nos algorithmes de construction ont une altération plus faible que les tables k-anonymes produites avec *GkAA*.

5.5 Conclusion du chapitre

Dans ce chapitre, notre objectif a été de construire des tables k-anonymes de meilleure qualité en termes d'altération pour une métrique de perte d'information que les tables k-anonymes produites avec l'algorithme *GkAA* présenté dans le chapitre 3 page 25.

En étudiant certaines exécutions de *GkAA*, nous avons remarqué que certaines tables k-anonymes produites avec l'algorithme ne sont pas optimales en termes d'altération. Nous avons notamment constaté que certaines tables contiennent des classes d'équivalence de grandes tailles par rapport au k initialement demandé et dans lesquelles de meilleurs partitionnements des enregistrements sont possibles. Nous avons donc cherché une procédure permettant de réduire le coût de généralisation des grandes classes d'équivalence des tables k-anonymes produites avec *GkAA* et ainsi de diminuer l'altération de ces tables.

Pour ce faire, nous avons tout d'abord défini les groupes de généralisation en section 5.1 page 89. Les groupes de généralisation permettent de rassembler les enregistrements d'une table en fonction de leurs valeurs quasi-identifiantes et des généralisations possibles de ces valeurs dans les hiérarchies de généralisation. Parmi les groupes de généralisation que nous avons calculés, nous avons constaté que certains contenaient une information redondante. Nous avons donc proposé une construction de l'ensemble minimal des groupes de généralisation en section 5.1.2 page 93.

À partir de l'ensemble minimal des groupes de généralisation, nous avons donné une nouvelle formulation du problème de k-anonymisation d'une table en section 5.2 page 100. Pour cela, nous avons défini un hypergraphe dont les sommets sont les enregistrements de la table et les hyperarêtes sont les groupes de généralisation de la table. En pratique, nous avons utilisé la matrice d'incidence de cet hypergraphe. Ainsi, trouver une k-anonymisation d'une table revient à choisir un unique 1 par ligne de la matrice d'incidence de l'hypergraphe de telle sorte que la somme des coefficients de chaque colonne de la matrice soit égale à 0 ou supérieure à k. Nous avons montré que cette formulation correspond à un k-partitionnement des enregistrements de la table, ce qui fournit une version k-anonyme de la table (cf. proposition 5.2.1 page 101).

Dans la section 5.3 page 105, nous avons proposé notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ ou $PCkPCk'$. La procédure traite toutes les classes d'équivalence de la table k'-anonyme passée en paramètre et produit une table k-anonyme en sortie. Pour les classes d'équivalence de taille supérieure à $2k$, la $PCkPCk'$ cherche un meilleur k-partitionnement des enregistrements en termes de coût de généralisation en s'appuyant sur la formulation du problème de k-anonymisation sur une table. Pour cela, elle fait appel au solveur *CP-SAT* de OR-Tools appliqué sur la matrice d'incidence de l'hypergraphe définie précédemment et avec les contraintes traduisant notre problème d'optimisation.

Le choix de la table k'-anonyme de départ dans $PCkPCk'$ étant libre, nous avons développé cinq algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ se basant sur notre procédure. Dans *GkPk*, nous k-partitionnons les classes d'équivalence de la table k-anonyme produite avec *GkAA*. Dans *G3kPk*, nous k-partitionnons les classes d'équivalence de la table $3k$-anonyme produite avec *GkAA*. Dans les trois autres algorithmes, dits avec convergence, une succession de tables k-anonymes sont construites jusqu'à atteindre une convergence des altérations des tables k-anonymes ou la limite du temps d'exécution. Dans l'algorithme *G2kPkconv*, à chaque tour, nous $2k$-anonymisons la table obtenue à la fin du tour précédent avec *GkAA* puis nous k-partitionnons les classes d'équivalence de la table $2k$-anonyme produite avec la $PCkPCk'$. Dans *G2kP2kPkconv*, nous commençons également par $2k$-anonymiser la table obtenue à la fin du tour précédent avec *GkAA*. Ensuite, nous $2k$-partitionnons les classes d'équivalence de table $2k$-anonyme avec la $PCkPCk'$. Enfin, nous appliquons la $PCkPCk'$ une seconde fois pour k-partitionner les classes d'équivalence de la table résultante de l'étape précédente. Dans *G4kP2kPkconv*, nous $4k$-anonymisons la table obtenue à la fin du tour précédent avec *GkAA*. Puis nous effectuons une première étape de $2k$-partitionnement des classes d'équivalence de la table $4k$-anonyme avec la $PCkPCk'$. Finalement, nous k-partitionnons les classes d'équivalence de la table $2k$-anonyme précédemment obtenue avec la $PCkPCk'$.

En section 5.4 page 113, nous avons comparé les performances des cinq algorithmes de construction avec celles de l'algorithme *GkAA*. Nous avons évalué les cinq algorithmes sur les tables *Adult data set* et *florida_30162* pour 14 valeurs de k entre 3 et 15 000. Nous avons comparé les altérations pour *NLLM* des tables k-anonymes produites avec chacun des algorithmes. Pour des valeurs de k raisonnables par rapport au nombre d'enregistrements de la table, nous avons constaté que les algorithmes de construction produisent, en général, des tables k-anonymes de meilleure qualité en termes d'altération que les tables k-anonymes produites par *GkAA*. Les algorithmes de construction avec convergence obtiennent les meilleurs résultats sur les deux tables.

Cependant, les inconvénients des algorithmes de construction que nous avons pointés en section 5.3.2 page 108 lors de la présentation des algorithmes ont été mis en évidence par l'étude expérimentale (cf. section 5.4.2.1 page 116). Pour chacun des algorithmes de construction, nous avons pu illustrer certains de ses points faibles en

étudiant les tables k-anonymes produites. Nous avons notamment montré que les exécutions des algorithmes de construction avec convergence, *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv*, sont parfois chaotiques. Cela suggère que des améliorations de ces algorithmes sont possibles. Par exemple, nous avons constaté que, lors de certaines exécutions de ces algorithmes, des motifs réguliers apparaissaient dans les altérations des tables k-anonymes successives. Une alternance entre plusieurs tables k-anonymes identiques se met en place et aucune nouvelle table k-anonyme n'est calculée avant la fin du temps limite d'exécution de 24 heures. Pour sortir de ce genre de situation, nous pourrions modifier l'une des tables k-anonymes du motif en effectuant des fusions de classes d'équivalence choisies aléatoirement. Ainsi, l'algorithme aurait une nouvelle table k-anonyme à traiter et pourrait partir dans une autre direction de recherche.

De plus, sur les graphiques représentant les altérations des tables k-anonymes successives obtenues lors d'une exécution d'un algorithme de construction avec convergence de la section 5.4.2.1 page 116, nous avons observé que les tables k-anonymes produites avec les algorithmes de construction avec convergence sont moins bonnes en termes d'altération que celles produites avec *GkAA* pour certaines valeurs de k. Nous nous sommes donc intéressés aux performances des algorithmes sur des sous-intervalles de $[3, 15\,000]$ regroupant des valeurs de k dans le même ordre de grandeur. Les résultats sur *Adult data set* sont les mêmes quelque soit l'intervalle de valeurs de k considéré : les algorithmes avec convergence *G2kPkconv*, *G2kP2kPkconv* et *G4kP2kPkconv* obtiennent les meilleurs résultats et surpassent nettement les performances de *GkAA*. En revanche, les conclusions sont légèrement différentes sur *florida_30162*. Nous avons par exemple constaté que l'algorithme *G4kP2kPkconv* a de mauvais résultats quand le k demandé est inférieur à 100. Néanmoins, l'algorithme *G2kPkconv* reste l'un des meilleurs sur les quatre sous-intervalles étudiés.

Perspectives Lors des expérimentations menées avec les algorithmes de construction, nous avons pointé certains comportements non souhaitables de ces algorithmes. Les algorithmes de construction avec convergence ont notamment des points faibles qui peuvent conduire à une exécution de 24 heures sans réel intérêt. Nous pourrions donc modifier les algorithmes de construction avec convergence pour essayer d'améliorer les tables k-anonymes produites et pour gagner en efficacité lors de l'exécution. La première piste d'amélioration consiste à sortir des motifs réguliers dans les tables k-anonymes produites lors de certaines exécutions des algorithmes de construction par convergence. Il s'agit dans un premier temps de détecter de tels motifs lors de l'exécution. Puis, pour que l'algorithme ait une nouvelle table k-anonyme à traiter et aille dans une autre direction de recherche, nous pourrions effectuer des fusions de classes d'équivalence aléatoirement dans la premier table k-anonyme apparaissant dans le motif régulier. La second piste à explorer est de relancer l'algorithme de construction avec convergence en cas d'arrêt prématuré de son exécution. En effet, lors de certaines exécutions, une convergence dans les altérations des tables k-anonymes est trouvée dans les dix premiers tours. Or l'altération de la table k-anonyme finale n'est pas l'altération minimale observée. Nous pourrions donc ajouter une condition d'arrêt dans les algorithmes de construction avec convergence sur le nombre de tours minimal à effectuer ou sur la minimalité de l'altération de la table k-anonyme du dernier tour. Ainsi, l'algorithme aurait une chance de trouver d'autres tables k-anonymes de meilleure qualité en termes d'altération.

Dans les algorithmes de construction proposés, nous avons utilisé l'algorithme *GkAA* pour k-anonymiser les tables. Bien qu'il ait l'avantage d'être simple à mettre en place et d'une exécution assez rapide, il est clair que les tables k-anonymes qu'il produit ne sont pas optimales en termes d'altération. Nous pourrions donc remplacer *GkAA* par un autre algorithme de k-anonymisation dans nos algorithmes de construction. Nous pouvons citer par exemple *k-member* [7] et *GCCG* [39] deux algorithmes de k-anonymisation par *clustering*. Ainsi nous pourrions observer si la procédure et les algorithmes de construction que nous avons proposés sont aussi efficaces quand nous utilisons un autre algorithme de k-anonymisation.

Dans les algorithmes de construction avec convergence, nous passons par des étapes intermédiaires durant lesquelles nous traitons des tables $4k$ ou $2k$-anonymes. Le choix de prendre des multiples de 2 en facteur de k vient du fait qu'il s'agit des plus petites valeurs pour lesquelles toutes les classes d'équivalence sont traitées par la procédure de construction. En effet, si nous demandons à la procédure de k-partitionner les classes d'équivalence d'une table, elle agira sur les classes dans lequelles un k-partitionnement est possible c'est-à-dire sur les classes de taille supérieure à $2k$. Nous pourrions adopter une autre approche pour déterminer les valeurs de k' des étapes intermédiaires. Supposons que nous souhaitons produire une table k-anonyme. Nous allons calculer une liste de valeurs de k' par lesquelles nous allons passer pour atteindre k. Deux méthodes de calcul des k' sont envisageables : la méthode incrémentale et la méthode décrémentale. Par exemple, supposons que nous voulions produire une version 500-anonyme de *Adult data set*. Rappelons que *Adult data set* contient $n = 30\,162$ enregistrements. Dans la méthode incrémentale, nous partons du $k = 500$ demandé et nous calculons les k' des étapes intermédiaires en multipliant par 2 :

$$\begin{array}{ccccccccccc} k = 500 & \rightarrow & 500 \times 2 & \rightarrow & 1000 \times 2 & \rightarrow & 2000 \times 2 & \rightarrow & 4000 \times 2 & \rightarrow & 8000 \times 2 \\ & & k' = 1000 & & k' = 2000 & & k' = 4000 & & k' = 8000 & & 16000 > \frac{n}{2} = 15081 \end{array}$$

Dans la méthode décrémentale, nous partons de $\frac{n}{2} = 15081$ et nous calculons les k' des étapes intermédiaires en divisant par 2 :

$$k' = 15081 \quad \rightarrow \quad \underset{k'=7540}{15081/2} \quad \rightarrow \quad \underset{k'=3770}{7540/2} \quad \rightarrow \quad \underset{k'=1885}{3770/2} \quad \rightarrow \quad \underset{k'=942}{1885/2} \quad \rightarrow \quad \underset{471<500}{942/2}$$

L'avantage d'une telle technique est que l'on part d'une table fortement anonymisée et avec peu de classes d'équivalence. Il est donc possible de regrouper les enregistrements de façon plus optimale. En revanche, le temps de calcul de la première étape du processus peut être très longue et donner de mauvais résultats dans la méthode incrémentale. En effet, le passage de $k' = 15081$ à $k' = 8000$ n'aboutit à rien puisque $8000 \times 2 > 15081$. Nous devons donc passer d'une table 15081-anonyme à une table 4000-anonyme.

Conclusion

Le volume de données personnelles collectées sur internet ne cesse de croître. Le cadre d'exploitation de ces données n'étant pas toujours clairement indiqué, les utilisateurs sont de plus en plus concernés par les problématiques de respect de leur vie privée. Pour rassurer les propriétaires des données et pouvoir continuer à exploiter les données en toute légalité, de nombreux modèles et algorithmes d'anonymisation ont été proposés dans le domaine Privacy-Preserving Data Publishing. Dans cette thèse, nous avons étudié le modèle de k-anonymité. Ce modèle garantit que tout enregistrement de la table anonymisée est indistinguable d'au moins $k-1$ autres enregistrements de la table par rapport à l'ensemble des attributs quasi-identifiants. Dans nos travaux, nous avons cherché des moyens d'optimiser l'utilité des données dans les tables k-anonymes.

Dans un premier temps, nous nous sommes intéressés dans le chapitre 3 page 25 aux métriques de perte d'information. Ces métriques permettent d'évaluer la quantité d'information perdue lors de l'anonymisation d'une table par la technique de généralisation. Comme de très nombreuses versions k-anonymes d'une table existent, il est nécessaire d'avoir un moyen de les comparer : les métriques de perte d'information remplissent ce rôle. Bien que très couramment utilisées pour tester les performances d'algorithmes d'anonymisation, les définitions des métriques et les notations utilisées sont variables d'un article à l'autre. De plus, plusieurs métriques ont été proposées mais peu de justifications sur leur pertinence ont été données.

Notre première contribution a donc consisté à proposer un modèle unifiant l'écriture des métriques de perte d'information et simplifiant leur utilisation. Une métrique est définie comme un ensemble de poids à mettre sur les arêtes des hiérarchies de généralisation des attributs quasi-identifiants. À partir de ces poids, nous avons construit une matrice des coûts pour chaque attribut quasi-identifiant contenant les coûts de généralisation des nœuds de la hiérarchie de cet attribut. Grâce à cette nouvelle définition, nous avons ensuite présenté trois métriques de perte d'information de la littérature *Distortion*, *NCP* et *Total* et quatre métriques issues de nos travaux *LLM*, *NLLM*, *WLLM* et *WNLLM*. Nous avons cherché à comparer les performances de ces sept métriques de perte d'information quand elles sont utilisées dans un algorithme de k-anonymisation pour guider les fusions de classes d'équivalence à effectuer. Pour cela, nous avons mené des expérimentations sur les tables *Adult data set* et *florida_30162* (cf. section 2.4 page 22). Pour chaque table, pour chaque métrique, nous avons produit des versions k-anonymes de la table pour 14 valeurs de k en utilisant la métrique dans l'algorithme *GkAA*. Pour évaluer la qualité des ces tables k-anonymes, nous avons utilisé trois critères : l'altération moyenne, le pourcentage de valeurs généralisées sur le nombre total de valeurs et le pourcentage de valeurs généralisées à la racine sur le nombre total de valeurs. Les résultats ont montré que les métriques ayant les meilleures performances ne sont pas les mêmes sur les deux tables. Sur *Adult data set*, la métrique *NLLM* permet de produire des tables k-anonymes de bonne qualité au regard des trois critères alors que sur *florida_30162*, il s'agit de la métrique *Total*. De plus, nous avons comparé les métriques selon plusieurs caractéristiques de leur définition. Ce qui est ressorti de cette analyse est que les métriques *LLM* et *WLLM* ont les moins bonnes perfomances quelque soit la table et quelque soit le critère considéré. Ce sont les deux seules métriques à ne pas avoir une étape de normalisation dans le calcul du coût de généralisation du nœud.

En conclusion de cette étude, nous pensons néanmoins que le choix de la métrique à utiliser pour k-anonymiser une table dépend notamment de la valeur de k demandé et des hiérarchies de généralisation choisies pour les attributs quasi-identifiants.

Dans un second temps, nous avons cherché à limiter l'une des faiblesses de la k-anonymité dans le chapitre 4 page 55. Dans certaines tables k-anonymes, il peut arriver qu'un manque de diversité apparaisse dans les valeurs de l'attribut sensible des classes d'équivalence. Pour remédier à ce problème, les modèles de l-diversité et de t-proximité ont été proposés dans [34] et [29] respectivement. Ces deux modèles d'anonymisation donnent des contraintes sur la répartition des valeurs de l'attribut sensible dans les classes d'équivalence de la table. En nous appuyant sur la l-diversité et la t-proximité, nous avons donc cherché à produire des tables k-anonymes conservant une bonne utilité des données tout en gardant un contrôle sur la répartition des valeurs de l'attribut sensible dans les classes d'équivalence. Pour cela, nous avons présenté l'algorithme *GAA* dans lequel les fusions de classes d'équivalence sont guidées par les contraintes d'une stratégie donnée en entrée de l'algorithme. Puis

nous avons proposé sept stratégies à utiliser dans *GAA* ayant pour objectif d'optimiser l'altération ou les valeurs de l-diversité et de t-proximité des tables k-anonymes produites. Pour comparer les performances de ces sept stratégies lors d'un processus de k-anonymisation, nous avons mené des expérimentations sur des données réelles et sur des données simulées. Nous avons utilisé trois critères pour évaluer la qualité des tables k-anonymes produites : l'altération, la valeur de l-diversité et la valeur de t-proximité.

En expérimentant sur des données réelles, nous avons remarqué que les stratégies peuvent se séparer en quatre groupes selon les résultats obtenus pour les trois critères. Les Stratégies 1 (optimisation sur le coût de généralisation), 2 (optimisation sur le coût de généralisation puis sur la valeur de l-diversité) et 4 (optimisation sur le coût de généralisation divisé par la valeur de l-diversité) permettent de produire des tables k-anonymes en limitant l'altération des données mais ne parviennent pas à maintenir de bonnes valeurs de l-diversité et de t-proximité dans ces tables. Les Stratégies 3 (optimisation sur la valeur de l-diversité puis sur le coût de généralisation) et 6 (optimisation sur la valeur de t-proximité puis sur le coût de généralisation) sont parmi les meilleures stratégies pour les valeurs de l-diversité et de t-proximité mais ne limitent pas efficacement l'altération des données dans les tables k-anonymes. La Stratégie 5 a un comportement extrême : elle optimise les valeurs de l-diversité et de t-proximité au détriment d'une altération proche des 100% pour toutes les tables k-anonymes produites en utilisant cette stratégie. La Stratégie 7 (optimisation sur le coût de généralisation multiplié par la valeur de t-proximité) a des résultats intermédiaires pour les trois critères.

En expérimentant sur des données simulées, nous avons globalement tiré les mêmes conclusions que pour les expérimentations sur les données réelles. Cependant, il est à noter que les performances des stratégies sont très proches pour les valeurs de l-diversité et de t-proximité. Nous pensons donc que, pour choisir la meilleure stratégie à utiliser sur une table dans ce cas, il suffit de se baser sur les performances des stratégies pour l'altération. Ainsi, sur *Adult data set*, la Stratégie 1 est un bon compromis pour optimiser l'altération et les valeurs de l-diversité et de t-proximité et sur *florida_30162*, nous choisirions la Stratégie 2.

Dans ces deux chapitres, nous avons utilisé des algorithmes gloutons pour k-anonymiser les tables : *GkAA* dans le chapitre 3 page 25 et *GAA* dans le chapitre 4 page 55. Or nous avons vu que ces algorithmes ne fournissent pas des tables k-anonymes optimales en termes d'altération. Nous avons notamment constaté que certaines classes d'équivalence de ces tables k-anonymes contiennent bien plus de k enregistrements. De plus, il est parfois possible de décomposer ces classes d'équivalence en plusieurs classses de plus petites tailles respectant la k-anonymité. L'objet du chapitre 5 page 87 a donc été de trouver une méthode pour améliorer la qualité des tables k-anonymes produites avec *GkAA* en cherchant de meilleures k-partitionnements des enregistrements des classes d'équivalence. Pour cela, nous avons donné une nouvelle formulation du problème de k-anonymisation d'une table se basant sur le concept de groupes de généralisation d'un ensemble d'enregistrements. Puis nous avons proposé notre procédure de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$ ou $PCkPCk'$. La procédure traite toutes les classes d'équivalence de la table k'-anonyme passée en paramètre et produit une table k-anonyme en sortie. Pour les classes d'équivalence de taille supérieure à $2k$, la $PCkPCk'$ cherche un meilleur k-partitionnement des enregistrements en termes de coût de généralisation. À partir de la $PCkPCk'$, nous avons développé cinq algorithmes de construction d'une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$. Dans *GkPk*, nous k-partitionnons les classes d'équivalence de la table k-anonyme produite avec *GkAA*. Dans *G3kPk*, nous k-partitionnons les classes d'équivalence de la table $3k$-anonyme produite avec *GkAA*. Dans les trois autres algorithmes, *G2kPkconv*,*G2kP2kPkconv* et *G4kP2kPkconv*, dits avec convergence une succession de tables k-anonymes sont construites jusqu'à atteindre une convergence des altérations des tables k-anonymes ou la limite du temps d'exécution.

Nous avons comparé les tables k-anonymes produites avec nos cinq algorithmes de construction et les tables k-anonymes produites avec *GkAA* en termes d'altération en expérimentant sur *Adult data set* et *florida_30162*. D'une manière générale, les algorithmes de construction et en particulier ceux avec convergence, produisent des tables k-anonymes moins altérées que *GkAA*. Néanmoins, nous avons illustré certains points faibles des algorithmes de construction ce qui suggère que des modifications pour améliorer leurs performances pourraient être apportées.

Collaboration et travaux connexes

Représentation des données anonymisées pour une tâche de *data mining* Dans le travaux présentés dans ce manuscrit, nous avons anonymisé les tables sans considérer les éventuelles exploitations qui pourraient être réalisées dessus. Afin de confronter les tables k-anonymes produites à des tâches de *data mining*, nous avons collaboré avec Fabien Viton, doctorant au MIS, et Jean-Luc Guérin, MCF au MIS dont les recherches portent sur des problématiques d'exploitation et d'analyse de données médicales. Cette collaboration a donné lieu à la publication d'un article à la conférence IEEE ISCC 2021 [47].

Dans cet article, nous avons proposé une nouvelle représentation des données, appelée *proportional*, permettant de conserver plus d'informations sur les données d'origine dans les classes d'équivalence de la table k-anonyme tout en étant directement utilisable en entrées d'un algorithme de *Machine Learning* pour des tâches de *data mining*. Pour les tâches de *data mining*, nous avons utilisé l'algorithme *Multi Layer Perceptron* (MLP) et pour k-anonymiser les tables nous avons utilisé l'algorithme *GkAA*.

Pour évaluer les performances de notre nouvelle représentation des données sur la qualité de prédiction des modèles générés par MLP, nous avons mené des expérimentations sur les tables *Adult data set* et *florida_30162*. Nous avons comparé la précision des modèles obtenus avec celle des modèles obtenus pour trois autres représentations des données présentées dans l'article [23].

Dans un premier temps, nous avons entraîné MLP sur les données brutes puis nous avons appliqué les modèles générés sur les données anonymisées sous les quatre représentations. Nous avons montré que les perfomances de prédiction des modèles sur notre représentation surpassent celles sur les trois autres représentations. Dans un second temps, nous avons entraîné MLP sur des tables 100-anonymes sous les quatre représentations puis nous avons appliqué les modèles générés sur les données anonymisées. Nous avons constaté que les meilleurs résultats de prédiction des modèles sont obtenus sur des données sous notre représentation et ce quelque soit la représentation choisie pour entraîner l'algorithme MLP. De plus, les meilleurs résultats de prédiction des modèles sont obtenus quand l'algorithme a été entraîné sur des données sous notre représentation et l'une des représentations de la littérature quelque soit la représentation des données utilisée pour évaluer les modèles.

Bien que la représentation des données que nous avons proposée introduise un peu plus de divulgation d'informations sur les attributs (la proportion de leurs valeurs dans les classes d'équivalence), la définition de la k-anonymité est strictement respectée dans les tables k-anonymes sous la représentation *propotional*.

Parallélisation de l'algorithme d'anonymisation glouton L'algorithme *GkAA* (cf. section 3.3.2 page 37) peut vite être coûteux en temps de calcul notamment quand le nombre de classes d'équivalence est grand dans la table à k-anonymiser. Par exemple, les tables *Adult data set* et *florida_30162* (cf. section 2.4 page 22) contiennent 30 162 enregistrements. Or *Adult data set* a 19 502 classes d'équivalence et *florida_30162* a 28 896 classes d'équivalence. L'exécution de *GkAA* sur *Adult data set* pour $k = 3$ en utilisant la métrique *Distortion* effectue 13 091 tours et dure environ 41 minutes alors que la même exécution sur *florida_30162* effectue 23 846 et dure environ 78 minutes sur un MacBook Pro 2,7GHz Intel Core i7 de 2018.

Arsème Vadèle Djeufack Nanfack, étudiant en seconde année de Master à l'Université de Dschang au Cameroun, a réalisé un stage pour paralléliser l'algorithme *GkAA*. La solution proposée est 100 fois plus rapide que *GkAA* sur 50 processeurs pour des valeurs de k entre 3 et 10. Pour parvenir à ce résultat, la base est triée en fonction des valeurs des attributs quasi-identifiants de telle sorte que les classes les plus susceptibles d'être fusionnées soient regroupées, puis elle est partitionnée en autant de sous-bases que de processeurs sur lesquels elles seront traitées indépendamment. Cette solution n'engendre pas beaucoup plus d'altération des données que *GkAA*. Elle est de plus assez robuste face à l'augmentation du nombre d'enregistrements dans les tables et n'a aucun problème pour traiter de grandes valeurs de k. Les résultats obtenus étant concluants, ils feront ultérieurement l'objet d'une publication académique.

Bibliographie

[1] Rakesh Agrawal et Ramakrishnan Srikant. « Privacy-preserving data mining ». In : *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 2000, p. 439-450.

[2] R. J. Bayardo et Rakesh Agrawal. « Data privacy through optimal k-anonymization ». In : *21st International Conference on Data Engineering (ICDE'05)*. Avr. 2005, p. 217-228.

[3] Michael A Bender et al. « Lowest common ancestors in trees and directed acyclic graphs ». In : *Journal of Algorithms* 57.2 (2005), p. 75-94.

[4] Claude Berge. *The theory of graphs*. Courier Corporation, 2001.

[5] Armin Biere, Marijn Heule et Hans van Maaren. *Handbook of satisfiability*. T. 185. IOS press, 2009.

[6] Alain Bretto. « Hypergraphs : Basic Concepts ». In : *Hypergraph Theory : An Introduction*. Heidelberg : Springer International Publishing, 2013, p. 1-21.

[7] Ji-Won Byun et al. « Efficient k-Anonymization Using Clustering Techniques ». In : *Advances in Databases : Concepts, Systems and Applications*. Sous la dir. de Ramamohanarao Kotagiri et al. Berlin, Heidelberg : Springer Berlin Heidelberg, 2007, p. 188-200.

[8] CNIL. *Règlement général sur la protection des données*. `https://www.cnil.fr/fr/reglement-europeen-protection-donnees`.

[9] Harold Connamacher et Julia Dobrosotskaya. « On the uniformity of the approximation for r-associated Stirling numbers of the second Kind ». In : *Contributions to Discrete Mathematics* 15.3 (2020), p. 25-42.

[10] Tore Dalenius. « Finding a needle in a haystack or identifying anonymous census records ». In : *Journal of official statistics* 2.3 (1986), p. 329.

[11] Josep Domingo-Ferrer et Jordi Soria-Comas. « From t-closeness to differential privacy and vice versa in data anonymization ». In : *Knowledge-Based Systems* 74 (2015), p. 151-158.

[12] Josep Domingo-Ferrer et Vicenç Torra. « Ordinal, Continuous and Heterogeneous k-Anonymity Through Microaggregation ». In : *Data Mining and Knowledge Discovery* 11.2 (sept. 2005), p. 195-212.

[13] Riccardo Dondi, Giancarlo Mauri et Italo Zoppis. « On the complexity of the l-diversity problem ». In : *International Symposium on Mathematical Foundations of Computer Science*. Springer. 2011, p. 266-277.

[14] Yang Du et al. « On multidimensional k-anonymity with local recoding generalization ». In : *2007 IEEE 23rd International Conference on Data Engineering*. IEEE. 2007, p. 1422-1424.

[15] Cynthia Dwork. « Differential Privacy : A Survey of Results ». In : *Theory and Applications of Models of Computation*. Sous la dir. de Manindra Agrawal et al. Berlin, Heidelberg : Springer Berlin Heidelberg, 2008, p. 1-19.

[16] Evolucare. *Projet SmartAngel*. `https://www.evolucare.com/fr/smart-angel-surveillance-medicale-individualisee/?region=fra`.

[17] Benjamin C.M. Fung et al. *Introduction to Privacy-Preserving Data Publishing : Concepts and Techniques*. 1st. Chapman & Hall/CRC, 2010.

[18] Benjamin CM Fung et al. « Privacy-preserving data publishing : A survey of recent developments ». In : *ACM Computing Surveys (Csur)* 42.4 (2010), p. 1-53.

[19] Google. *OR-Tools*. `https://developers.google.com/optimization`.

[20] US Government. *Registered voters in the State of Florida, U.S.A.* [Online ; accessed on May 2020] `http://flvoters.com/`.

[21] Naoise Holohan et al. *(k,ϵ)-Anonymity : k-Anonymity with ϵ-Differential Privacy*. 2017.

[22] Zhengli Huang, Wenliang Du et Biao Chen. « Deriving private information from randomized data ». In : *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. 2005, p. 37-48.

[23] Ali Inan, Murat Kantarcioglu et Elisa Bertino. « Using anonymized data for classification ». In : *2009 IEEE 25th International Conference on Data Engineering*. IEEE. 2009, p. 429-440.

[24] Vijay S. Iyengar. « Transforming Data to Satisfy Privacy Constraints ». In : *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. Edmonton, Alberta, Canada : ACM, 2002, p. 279-288.

[25] Krishnaram Kenthapadi, Ilya Mironov et Abhradeep Guha Thakurta. « Privacy-Preserving Data Mining in Industry ». In : *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. WSDM '19. Melbourne VIC, Australia : Association for Computing Machinery, 2019, p. 840-841.

[26] K. LeFevre, D. J. DeWitt et R. Ramakrishnan. « Mondrian Multidimensional K-Anonymity ». In : *22nd International Conference on Data Engineering (ICDE'06)*. Avr. 2006, p. 25-25.

[27] Kristen LeFevre, David J. DeWitt et Raghu Ramakrishnan. « Incognito : Efficient Full-domain K-anonymity ». In : *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*. SIGMOD '05. Baltimore, Maryland : ACM, 2005, p. 49-60.

[28] Jiuyong Li et al. « Achieving k-Anonymity by Clustering in Attribute Hierarchical Structures ». In : *Data Warehousing and Knowledge Discovery* (2006). Sous la dir. d'A. Min Tjoa et Juan Trujillo, p. 405-416.

[29] Ninghui Li, Tiancheng Li et Suresh Venkatasubramanian. « t-closeness : Privacy beyond k-anonymity and l-diversity ». In : *2007 IEEE 23rd International Conference on Data Engineering*. IEEE. 2007, p. 106-115.

[30] Ninghui Li, Wahbeh H Qardaji et Dong Su. « Provably private data anonymization : Or, k-anonymity meets differential privacy ». In : *CoRR, abs/1101.2604* 49 (2011), p. 55.

[31] Tiancheng Li et al. « Slicing : A new approach for privacy preserving data publishing ». In : *IEEE transactions on knowledge and data engineering* 24.3 (2010), p. 561-574.

[32] Hongyu Liang et Hao Yuan. « On the complexity of t-closeness anonymization and related problems ». In : *International Conference on Database Systems for Advanced Applications*. Springer. 2013, p. 331-345.

[33] Jun-Lin Lin et Meng-Cheng Wei. « An Efficient Clustering Method for K-anonymization ». In : *Proceedings of the 2008 International Workshop on Privacy and Anonymity in Information Society*. PAIS '08. Nantes, France : ACM, 2008, p. 46-50.

[34] Ashwin Machanavajjhala et al. « l-diversity : Privacy beyond k-anonymity ». In : *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), 3-es.

[35] Clémence Mauger, Gaël Le Mahec et Gilles Dequen. « Modeling and Evaluation of k-anonymization Metrics ». In : *Privacy Preserving Artificial Intelligence Workshop of AAAI 2020*. 2020.

[36] Clémence Mauger, Gaël Le Mahec et Gilles Dequen. « Multi-criteria Optimization Using l-diversity and t-closeness for k-anonymization ». In : *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2020, p. 73-88.

[37] Adam Meyerson et Ryan Williams. « On the Complexity of Optimal K-anonymity ». In : *Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. PODS '04. Paris, France : ACM, 2004, p. 223-228.

[38] Mehmet Ercan Nergiz et Muhammed Zahit Gök. « Hybrid k-anonymity ». In : *Computers & security* 44 (2014), p. 51-63.

[39] Sang Ni, Mengbo Xie et Quan Qian. « Clustering Based K-anonymity Algorithm for Privacy Preservation. » In : *Int. J. Netw. Secur.* 19.6 (2017), p. 1062-1071.

[40] M. I. Pramanik, R. Y. K. Lau et W. Zhang. « K-Anonymity through the Enhanced Clustering Method ». In : *2016 IEEE 13th International Conference on e-Business Engineering (ICEBE)*. Nov. 2016, p. 85-91.

[41] Pierangela Samarati. « Protecting respondents identities in microdata release ». In : *IEEE transactions on Knowledge and Data Engineering* 13.6 (2001), p. 1010-1027.

[42] Claude Elwood Shannon. « A mathematical theory of communication ». In : *ACM SIGMOBILE mobile computing and communications review* 5.1 (2001), p. 3-55.

[43] Latanya Sweeney. « Achieving k-anonymity privacy protection using generalization and suppression ». In : *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), p. 571-588.

[44] Latanya Sweeney. « k-anonymity : A model for protecting privacy ». In : *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), p. 557-570.

[45] Yufei Tao et al. « Angel : Enhancing the utility of generalization for privacy preserving publication ». In : *IEEE transactions on knowledge and data engineering* 21.7 (2009), p. 1073-1087.

[46] UCIrvine. *Machine Learning Repository*. [Online ; accessed on June 2019] `https://archive.ics.uci.edu/ml/index.php`. 1987.

[47] Fabien Viton et al. « Proportional representationto increase data utility in k-anonymous tables ». In : *Proceedings of the 26th IEEE Symposium on Computers and Communications (ISCC 2021)*. Athens, Greece, 2021.

[48] Xiaokui Xiao et Yufei Tao. « Anatomy : Simple and effective privacy preservation ». In : *Proceedings of the 32nd international conference on Very large data bases*. 2006, p. 139-150.

[49] Xiaokui Xiao, Ke Yi et Yufei Tao. « The hardness and approximation algorithms for l-diversity ». In : *Proceedings of the 13th International Conference on Extending Database Technology*. 2010, p. 135-146.

[50] Jian Xu et al. « Utility-based Anonymization Using Local Recoding ». In : *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '06. Philadelphia, PA, USA : ACM, 2006, p. 785-790.

Annexe A

Hiérarchies des attributs quasi-identifants de *Adult data set* et *florida_30162*

A.1 Hiérarchies des attributs quasi-identifiants de *Adult data set*

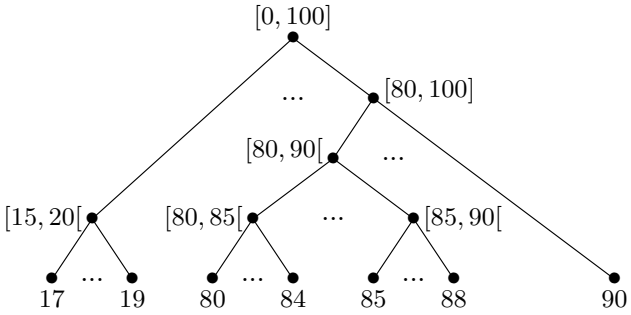


FIGURE A.1 – Hiérarchie de l'attribut *Age*

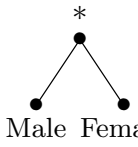


FIGURE A.2 – Hiérarchie de l'attribut *Genre*

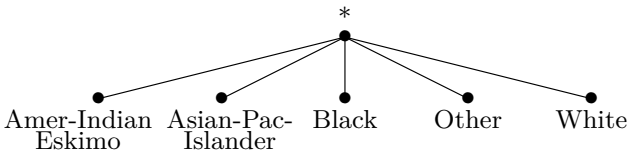


FIGURE A.3 – Hiérarchie de l'attribut *Race*

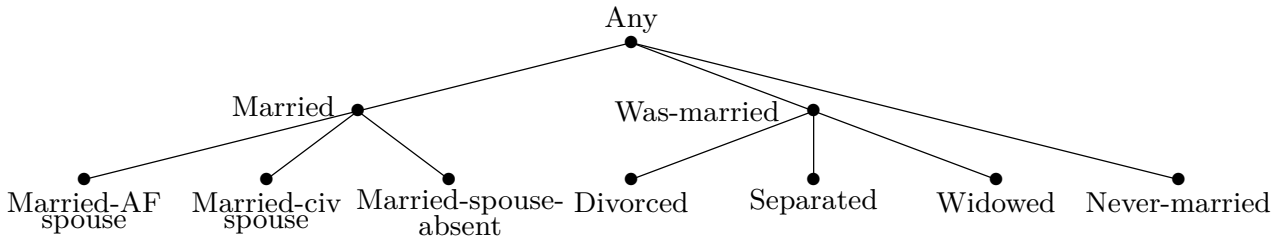


FIGURE A.4 – Hiérarchie de l'attribut *Statut marital*

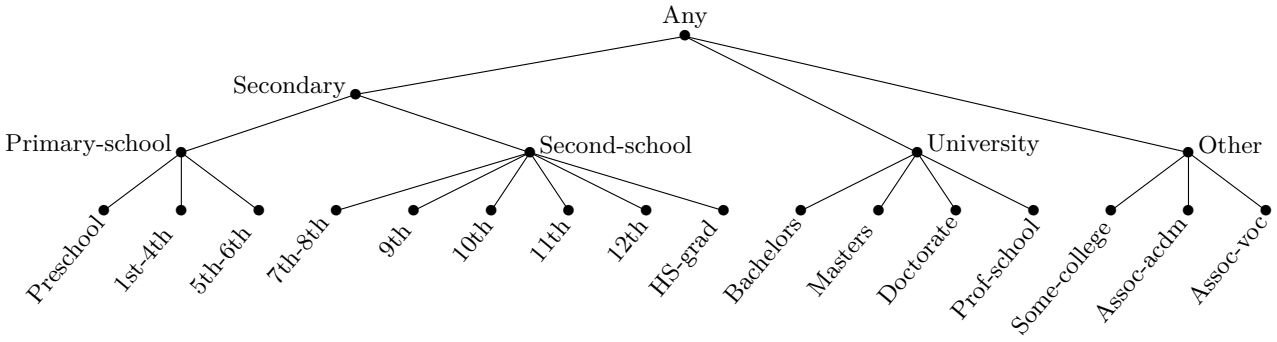


FIGURE A.5 – Hiérarchie de l'attribut *Éducation*

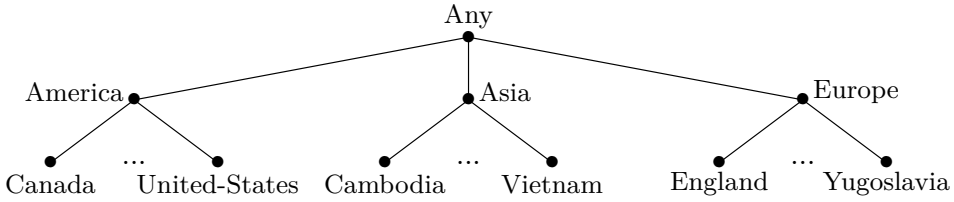


FIGURE A.6 – Hiérarchie de l'attribut *Pays de naissance*

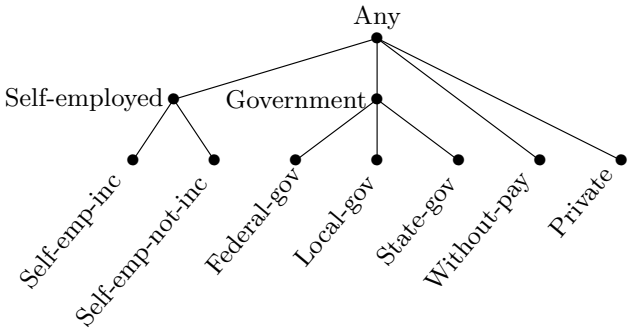


FIGURE A.7 – Hiérarchie de l'attribut *Catégorie professionnelle*

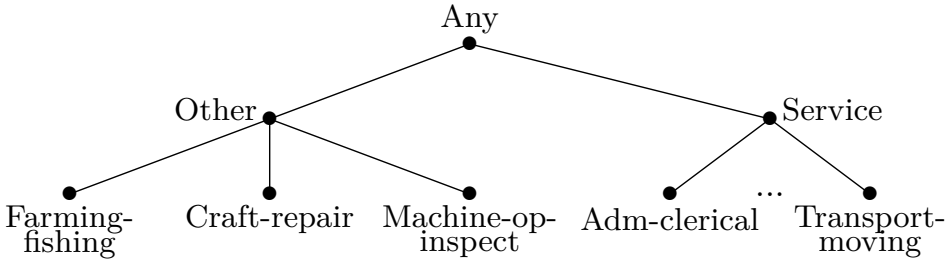


FIGURE A.8 – Hiérarchie de l'attribut *Occupation*

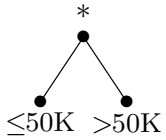


FIGURE A.9 – Hiérarchie de l'attribut *Salaire*

A.2 Hiérachies des attributs de *florida_30162*

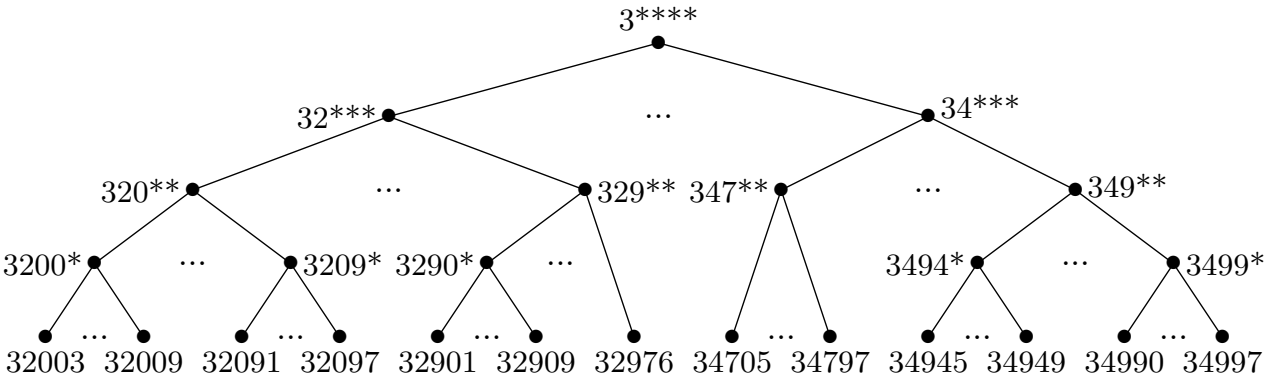


FIGURE A.10 – Hiérarchie de l'attribut *Code postal*

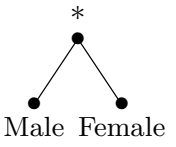


FIGURE A.11 – Hiérarchie de l'attribut *Genre*

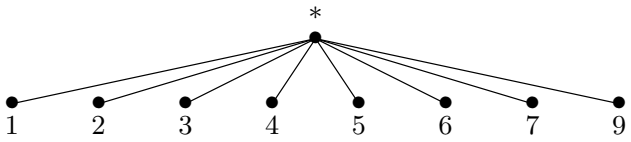


FIGURE A.12 – Hiérarchie de l'attribut *Race*

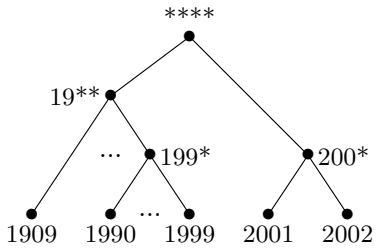


FIGURE A.13 – Hiérarchie de l'attribut *Année de naissance*

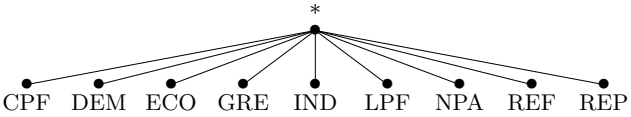


FIGURE A.14 – Hiérarchie de l'attribut *Parti politique*

Optimisation de l'utilité des données lors d'un processus de k-anonymisation

Résumé

Pour donner des garanties de protection de la vie privée aux bases de données anonymisées, des modèles d'anonymisation ont vu le jour ces dernières décennies. Parmi ceux-ci, on peut citer la k-anonymité, la l-diversité, la t-proximité ou encore la confidentialité différentielle. Dans cette thèse, je me suis intéressée au modèle de k-anonymité à travers une analyse approfondie des manières de produire des bases remplissant ces critères de confidentialité tout en optimisant l'utilité des données. Partant d'une base de données, on peut en effet construire plusieurs versions k-anonymes de cette base. Certaines de ces versions k-anonymes comportent moins de modifications des données que les autres et maintiennent ainsi une meilleure utilité des données lors de leur publication. Mes travaux proposent une étude de l'optimisation de l'utilité des données lors du processus de k-anonymisation d'une base.

Dans un premier temps, j'ai étudié des métriques de perte d'information permettant d'estimer la quantité d'information perdue dans une table lors d'un processus de k-anonymisation. Les métriques ont été utilisées dans un algorithme de k-anonymisation pour guider les fusions de classes d'équivalence menant à la production d'une table k-anonyme. J'ai tâché de dégager de cette étude des caractéristiques dans les définitions des métriques de perte d'information permettant de produire des tables k-anonymes de bonne qualité au regard de plusieurs critères.

Dans un second temps, je me suis intéressée à la répartition des données sensibles dans les tables k-anonymes grâce aux modèles de l-diversité et de t-proximité. Plus précisément, j'ai proposé des stratégies d'optimisation mêlant métrique de perte d'information, l-diversité et t-proximité à utiliser dans un algorithme de k-anonymisation. L'objectif a été de maintenir de bons niveaux de l-diversité et de t-proximité dans les tables k-anonymes produites sans sacrifier l'utilité des données.

Dans un troisième temps, je suis revenue sur la formulation du problème de k-anonymisation d'une table. Je me suis appuyée sur une nouvelle notion, les groupes de généralisation, pour énoncer le problème de k-anonymisation d'une table en fonction de la matrice d'incidence d'un hypergraphe. Grâce à cette nouvelle représentation, j'ai proposé une procédure ainsi que cinq algorithmes permettant de construire une table k-anonyme par partitionnement des classes d'équivalence d'une table k'-anonyme avec $k' \geqslant k$. Des expérimentations menées sur deux tables publiques ont montré que les algorithmes proposés surpassent les performances de l'algorithme de k-anonymisation utilisé précédemment en termes de préservation d'information.

Mots clés : k-anonymisation, utilité des données, optimisation

Abstract

So that providing privacy guarantees to anonymized databases, anonymization models have emerged few decades ago. Among them, you can find k-anonymity, l-diversity, t-proximity or differential confidentiality. In this thesis, we mainly focused on the k-anonymity model through an in-depth analysis of the ways to produce databases that meet these confidentiality criteria while optimizing data utility. From a table, you can consider the set of its k-anonymous versions, which can be of exponential cardinality according to k. In a vacuum, these k-anonymous versions can be scored thanks to the amount of data modification that is correlated to the data utility. Thus, this work proposes a study of how to optimize the data utility during the process of k-anonymizing a database.

First, we studied information loss metrics to estimate the amount of information lost in a table during a k-anonymization process. The metrics were used within a k-anonymization algorithm to guide equivalence class mergers leading to the production of a k-anonymous table. We tried to identify from this study characteristics in the definitions of information loss metrics allowing the production of good quality k-anonymous tables with regard to several criteria.

Second, we were interested in the distribution of sensitive data into k-anonymous tables by using l-diversity and t-proximity models. More specifically, we proposed optimization strategies combining information loss metrics, l-diversity and t-proximity to be used during a k-anonymization process. The aim was then to preserv good levels of l-diversity and t-proximity of the k-anonymous tables produced, and this without sacrificing the data utility.

Third, we tackled the question of the formulation of the problem of k-anonymization of a table. We relied on the original notion of generalization groups, to state the problem of k-anonymization of a table according to the incidence matrix of its associated hypergraph. Thanks to this new representation, we proposed an original procedure, declined to five algorithms, allowing to build a k-anonymous table by partitioning the equivalence classes of a k'-anonymous table with $k' \geqslant k$. Experiments carried out on two public tables have shown that the proposed algorithms outperfom the k-anonymization algorithm used previously in terms of information preservation.

Keywords: k-anonymization, data utility, optimization