REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ Dr. TAHAR MOULAY – SAIDA

FACULTÉ DE TECHNOLOGIE

DÉPARTEMENT INFORMATIQUE

MÉMOIRE

Présenté par

BENDIDA Abd El Hamid Ibn Badis
GUENDOUZ Baghdad

Pour l'obtention du diplôme de

MASTER en Informatique

Option : Sécurité Informatique et Cryptographie (SIC)

THÈME

La Vie Privée dans les Graphes de Connaissances

Nom et prénom	Qualité	Etablissement
Mr Ahmed ZAHAF	Encadreur	Université de Saïda

Promotion : Septembre 2020

Remerciements

Au terme de ce travail, je tiens à remercier Dieu le tout puissant de m'avoir donné le courage, la volonté et la patience pour achever ce travail.

Que nos chers parents et familles, trouvent ici l'expression de nos remerciements les plus sincères et les plus profonds en reconnaissance de leurs sacrifices, aides, soutien et encouragement.

J'ai l'honneur et le plaisir de présenter ma profonde gratitude et mes sincères remerciements à mon encadreur Dr. Ahmed ZAHAF, pour ses précieuses aides, ces orientations et le temps qu'il m'a accordé pour mon encadrement.

Je remercie profondément tous les enseignants qui m'ont encouragé et soutenu pendant mon cursus.

Je remercie aussi tous ceux qui ont contribué de prêt ou de loin à la réalisation de mon mémoire

Résumé

L'anonymisation est une technique de protection de la confidentialité qui a été appliquée avec succès dans les bases de données et les graphes. Cependant, les études sur l'anonymisation dans les graphes de connaissance sont très limitées. Ces études sont les travaux initiaux de protection de la vie privée, puisqu'ils montrent les approches pratiques d'anonymisation pour des scénarios simples comme l'utilisation d'opérations de généralisation et d'opérations de suppression basées sur des hiérarchies. Cependant, pour des scénarios complexes, où une diversité de données est présentée, les approches d'anonymisations existantes n'assurent pas une confidentialité suffisante. Dans ce mémoire, nous proposons une anonymisation par l-diversité, dans le contexte des graphes de connaissances. Cette approche est une continuité de travaux fournis pendant les années précédentes avec le même encadrement pour des fin amélioratrices et correctives. Nous conduisons une expérimentation pour évaluer l'effort fournit.

MOTS-CLES : graphes de connaissance, sécurité, la vie privée, anonymisation, De-anonymisation

Abstract

Anonymization is a privacy protection technique that has been successfully applied in databases and graphs. However, studies on anonymization in knowledge graphs are very limited. These studies are the initial privacy work, as they show practical anonymization approaches for simple scenarios such as the use of hierarchy-based generalization and deletion operations. However, for complex scenarios, where a variety of data is presented, the existing anonymization approaches do not ensure sufficient confidentiality. In this thesis, we propose an anonymization by diversity, in the context of knowledge graphs. This approach is a continuation of work provided during previous years with the same supervision for ameliorative and corrective purposes. We are conducting an experiment to assess the effort provided.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

L'avènement du numérique a entrainé la collecte massive de données. Pour exploiter pleinement l'utilité analytique de ces données, ces dernières ont besoin d'être rendues disponibles aux chercheurs et/ou aux professionnels. La dernière décennie a également vu la naissance du mouvement Open Data qui s'est traduit par la volonté de nombreux acteurs, publiques comme privés, de diffuser leurs données jugées d'intérêt général. Cependant, une part non négligeable de ces données concerne des individus et constitue des informations sensibles qui peuvent menacer la vie privée de ces personnes. Si l'on prend l'exemple du domaine médical, la distribution de certains jeux de données pose des problèmes pour la protection du secret médical. Beaucoup d'études ont été effectuées en vue de trouver une solution à ce problème d'anonymisation des données. Ce domaine consiste à modifier le contenu ou la structure de ces données afin de rendre très difficile voire impossible la « ré-identification » des personnes ou des entités concernées. De manière générale, nous appellerons "entités d'intérêt" (EI) les cibles d'une technique d'anonymisation **(Maxime th et al,2020).** Chaque EI est identifiée par un ensemble de valeurs que l'on appelle des attributs que l'on peut séparer en quatre catégories différentes :

— Les identifiants explicites (IDE) : ils permettent d'identifier explicitement une entité. Exemple : le numéro de sécurité sociale.

— Les quasi-identifiants (QID) : un ensemble d'attributs qui, lorsqu'ils sont utilisés ensemble, rendent possible la ré-identification. Exemple : l'âge, le code postal, etc..

Pour résumer, nos contributions de recherche sont de plusieurs ordres. La première contribution concerne est une étude détaillée sur les techniques d'anonymisation sur les graphes de connaissance. Cela nous permettons à évaluer avec une technique de dés-anonymisation basée sur la similarité entre chaque

nœud de graphe adverse et tous les nœuds de graphe cibles. La deuxième contribution est évaluée les travaux antérieur **(SHWAN,2019**), La troisième contribution comme perspectives

Pour faire face à cette situation, nous proposons dans ce mémoire une approche qui se base essentiellement sur la technique « l-diversité » pour améliorer les résultats d'anonymisation des graphes de connaissances. Nous avons organisé notre travail sur quatre chapitres : Le premier chapitre est consacré à la notion de graphe de connaissances. Le deuxième chapitre traite les notions de protection de la vie privée, on se focalisée sur la définition de modèles de protection de la vie privée. Le troisième chapitre est consacré à la présentation de notre contribution, dans nous mettons l'accent sur le processus anonymisation par l-diversité. Nous terminerons notre mémoire par une conclusion générale et quelques perspectives on se basant sur les mesures de similarité sémantique entre les ressources.

Chapitre I

Graphique des connaissances

Introduction

La définition d'un "graphe de connaissance" reste discutable, où un certain nombre de définitions (parfois contradictoires) ont émergé, Nous adoptons ici une définition inclusive, dans laquelle nous considérons un graphe de connaissances comme un graphe de données destiné à accumuler et à transmettre des connaissances du monde réel, dont les nœuds représentent des entités d'intérêt et Les liens dirigé relie une entité à une autre représente des relations entre ces entités.

Le graphe de données se conforme à un modèle de données basé sur un graphe. Par connaissance, nous entendons quelque chose qui déjà connu. Cette connaissance peut être accumulée à partir de sources externes ou extraite du graphique de connaissance lui-même. La connaissance peut être composée d'énoncés simples, comme "Paris est la capitale du France", ou d'énoncés quantifiés, comme "toutes les capitales sont des villes".

Des énoncés simples peuvent être accumulés sous forme d'arêtes dans le graphique de données. Si le graphe de connaissances vise à accumuler des énoncés quantifiés, une manière plus expressive de représenter les connaissances - comme des ontologies ou des règles - est nécessaire. Des méthodes déductives peuvent alors être utilisées pour impliquer et accumuler d'autres connaissances (par exemple, "Paris est une ville"). Des connaissances supplémentaires - basées sur des énoncés simples ou quantifiés - peuvent également être extraites du graphique des connaissances et accumulées par celui-ci en utilisant des méthodes inductives.

Les graphiques de connaissances sont souvent assemblés à partir de nombreuses sources et peuvent donc être très divers en termes de structure et de degré de complexité. Pour faire face à cette diversité, la représentation du schéma, de l'identité et du contexte joue souvent un rôle clé. Un schéma définit une structure de haut niveau pour le graphe de connaissances, l'identité indique quels nœuds du graphe (ou de sources externes) font référence à la même entité du monde réel, tandis que le contexte peut indiquer un cadre spécifique dans lequel une certaine unité de connaissances est considérée comme vraie. Comme indiqué précédemment, des méthodes efficaces d'extraction, d'enrichissement, de qualité

L'évaluation et l'affinement sont nécessaires pour qu'un graphique de connaissances puisse s'accroître et s'améliorer au fil du temps.

En pratique. Les graphiques de connaissances visent à fournir un support partagé de connaissances qui évolue toujours au sein d'une organisation ou d'une communauté. Dans la pratique, nous distinguons deux types de graphiques de connaissances : les graphiques de connaissances ouvertes et les graphiques de connaissances d'entreprise. Les graphiques de connaissances ouverts sont publiés en ligne, ce qui rend leur contenu accessible pour le public. Les exemples les plus connus sont **DBpedia** [1], **Freebase** [2], **Wikidata** [3], **YAGO** [4], etc. couvrent de nombreux domaines et sont soit extraits de **Wikipédia**[5], soit construits par des communautés de volontaires. Nous allons présenter quelques notions importantes relatives aux graphes de connaissances.

1. GRAPHE DE DONNEES

Le principe de la première application d'une abstraction de graphe aux données est à la base de tout graphe de connaissances, ce qui résulte un graphe de données initial. Dans la section courante, nous discutons d'une sélection de modèles de données structurées par graphes qui sont couramment utilisés dans la pratique pour représenter des graphes de données. Nous discutons ensuite des primitives qui forment la base des langages d'interrogation de graphes utilisés pour interroger ces graphes de données.

1.1. Les modèles

Pour démarrer, nous considérons l'exemple de la figure 1. Cet exemple de l'office du tourisme ainsi que tous ceux qui le suivant sont présentés initialement dans **(Hogan et al, 2020).**

[1] https://fr.wikipedia.org/wiki/DBpedia.
[2] https://fr.wikipedia.org/wiki/Free_base
[3] https://fr.wikipedia.org/wiki/Wikidata.
[4] https://fr.wikipedia.org/wiki/YAGO
[5] https://fr.wikipedia.org/wiki/Wikipédia

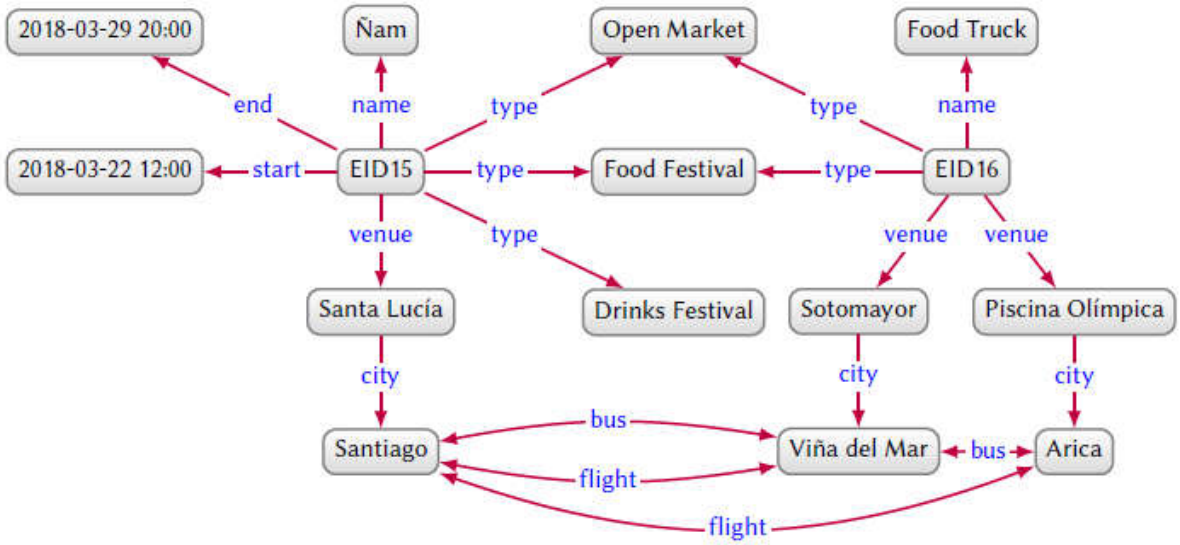


Figure 1 : Directed edge-labelled +graph décrivant les événements et leurs lieux.

Laissant de côté les graphes, supposons que l'office du tourisme de notre exemple actuel n'a pas encore décidé comment modéliser les données pertinentes sur les attractions, les événements, les services, etc. L'office envisage d'abord d'utiliser une structure tabulaire - en particulier des bases de données relationnelles - pour représenter les données requises, et bien qu'ils ne sachent pas précisément quelles données ils devront capturer, ils commencent à concevoir un schéma relationnel initial. Ils commencent par une table Event avec cinq colonnes :

Event (name, venue, type, start, end)

où name et start, forment ensemble la clé primaire de la table afin d'identifier de manière unique les événements récurrents. Mais au fur et à mesure qu'ils commencent à remplir les données, ils rencontrent divers problèmes : les événements peuvent avoir plusieurs noms (par exemple, dans différentes langues), les événements peuvent avoir plusieurs lieux, ils peuvent ne pas encore connaître les dates et heures de début et de fin des événements futurs, des événements peut avoir plusieurs types, et ainsi de suite. En abordant progressivement ces problèmes de modélisation à mesure que les données se diversifient, ils génèrent des identifiants internes pour les événements et adaptent leur schéma relationnel jusqu'à ce qu'ils aient :

EventName(id, name), EventStart(id, start), EventEnd(id, end) (1)

EventVenue(id, venue), EventType(id, type)

Avec le schéma ci-dessus, l'organisation peut désormais modéliser des événements avec 0 – n noms, lieux et types, et 0–1 dates de début et de fin (sans avoir besoin de valeurs nulles relationnelles / cellules vides dans les tableaux).

En fait, le schéma raffiné et flexible avec lequel le tableau se termine - illustré dans (1) - modélise un ensemble de relations binaires entre entités, qui peuvent en effet être considérées comme la modélisation d'un graphe. En adoptant à la place un modèle de données de graphe dès le départ, la carte pourrait renoncer à la nécessité d'un schéma initial, et pourrait définir n'importe quelle relation (binaire) entre n'importe quelle paire d'entités à tout moment.

1.1.1. Directed Edge labelled Graphs

Un directed edge-labelled graph est défini comme un ensemble de nœuds, comme : Santiago , Arica , EID16 , 2018-03-22 12:00, et un ensemble d'arêtes étiquetées dirigées entre ces nœuds, comme : Santa Lucía → city → Santiago. Dans le cas des graphes de connaissances, les nœuds sont utilisés pour représenter les entités et les arêtes sont utilisées pour représenter les relations (binaires) entre ces entités. La figure 1 donne un exemple de la façon dont l'office du tourisme pourrait modéliser certaines données d'événements pertinentes sous la forme d'un directed edge-labelled graph. Le graphe comprend des données sur les noms, les types, les dates-heures de début et de fin et les lieux des événements. L'ajout d'informations à un tel graphe implique généralement l'ajout de nouveaux nœuds et arêtes. Représenter des informations incomplètes nécessite simplement d'omettre une arête particulière; par exemple, le graphique ne définit pas encore une date-heure de début / fin pour le festival Food Truck.

La modélisation des données sous forme de graphique de cette manière offre plus de flexibilité pour l'intégration de nouvelles sources de données, par rapport au modèle relationnel standard, où un schéma doit être défini à l'avance et suivi à chaque étape. Bien que d'autres modèles de données structurées tels que les arbres (XML, JSON, etc.) offriraient une flexibilité similaire, les graphiques ne nécessitent pas d'organiser les données de manière hiérarchique. Ils permettent également de représenter et d'interroger les cycles (par exemple, notez le cycle dirigé dans les itinéraires entre Santiago, Arica et Viña del Mar).

Un modèle de données normalisé basé sur des directed edge-labelled graphs est le Resource Description Framework (RDF) (Cyganiak et al, 2014), qui a été

recommandé par le W3C. Le modèle RDF définit différents types de nœuds, y compris les identificateurs de ressources internationalisés (IRI) (Dürst and Suignard, 2005) qui permettent une identification globale des entités sur le Web; les littéraux, qui permettent de représenter des chaînes (avec ou sans balises de langue) et d'autres valeurs de type de données (entiers, dates, etc.); et les nœuds vides, qui sont des nœuds anonymes auxquels aucun identifiant n'est attribué (par exemple, plutôt que de créer des identifiants internes comme EID15, EID16, en RDF, nous avons la possibilité d'utiliser des nœuds vides).

1.1.2. Graphes Dataset

Bien que plusieurs directed edge-labelled graphs puissent être fusionnés en prenant leur union, il est souvent souhaitable de gérer plusieurs graphes plutôt qu'un graphe monolithique ; par exemple, il peut être avantageux de gérer plusieurs graphiques provenant de différentes sources, ce qui permet de mettre à jour ou d'affiner les données d'une source, de distinguer les sources non fiables des sources plus fiables, etc. Un graphe dataset se compose alors d'un ensemble de graphes nommés et d'un graphe par défaut. Chaque graphe nommé est une paire d'un ID de graphe et d'un graphe. Le graphe par défaut est un graphe sans ID et est référencé « par défaut » si aucun ID de graphe n'est spécifié. La figure 2 fournit un exemple où les événements et les itinéraires sont stockés dans deux graphes nommés, et le graphe par défaut gère les méta-données sur les graphes nommés. Nous soulignons que les noms de graphes peuvent également être utilisés comme nœuds dans un graphe. En outre, les nœuds et les arêtes peuvent être répétés sur les graphes, où le même nœud dans différents graphes se réfère généralement à la même entité, ce qui permet aux données sur cette entité d'être intégrées lors de la fusion des graphes.

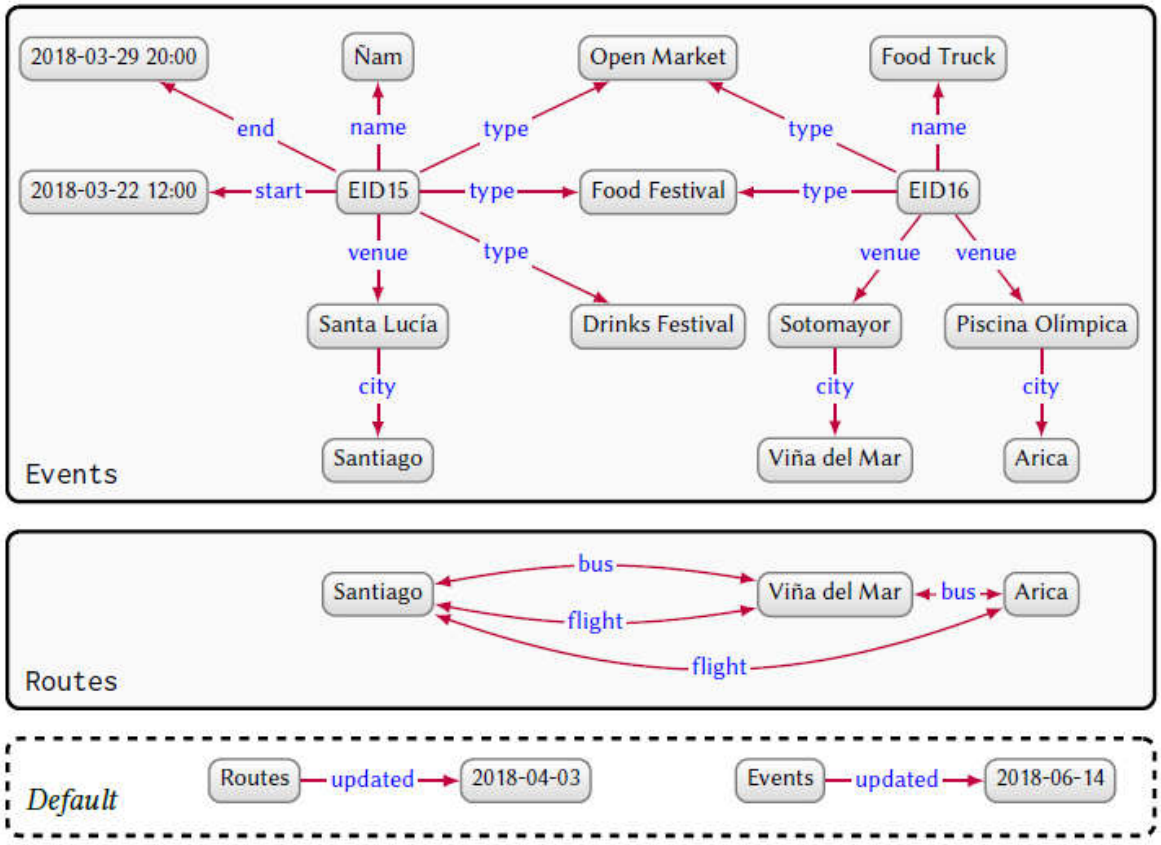


Figure 2:Ensemble de données graphes avec deux graphes nommés et un graphe par défaut décrivant les événements et les itinéraires

Un cas d'utilisation important pour les ensembles de données graphes est de gérer et d'interroger des données liées composées de documents interconnectés de graphiques RDF couvrant le Web. Lorsqu'il s'agit de données Web, le suivi de la source des données devient d'une importance capitale **(Piero et al, 2011)**.

1.1.3. Graphes de propriétés

Les graphes de propriétés ont été introduits pour offrir une flexibilité supplémentaire lors de la modélisation de relations plus complexes. Considérons d'intégrer des données entrantes qui fournissent des informations sur les compagnies qui proposent des tarifs sur quels vols, ce qui permet au conseil de mieux comprendre les itinéraires disponibles entre les villes (par exemple, sur les compagnies aériennes nationales).

Dans le cas de directed-edge labelled graphs, nous ne pouvons pas annoter directement une arête comme **Santiago → flight → Arica** avec la ou les sociétés

proposant cette route. Mais nous pourrions ajouter un nouveau nœud indiquant un vol, le connecter à la source, à la destination, aux entreprises et au mode, comme le montre la **Figure 3**. L'application de cette modélisation à toutes les données de la figure 1 impliquerait cependant une modification importante du graphe. Une autre option pourrait être de placer les vols de différentes entreprises dans différents graphiques nommés, mais si des graphiques nommés sont déjà utilisés pour suivre la source des graphiques (par exemple), cela pourrait devenir fastidieux.

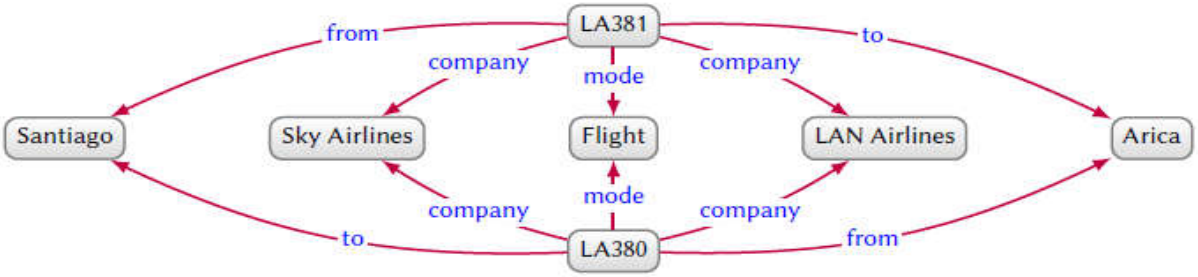


Figure 3: Directed edge-labelled graph avec des compagnies proposant des vols entre Santiago et Arica.

Le modèle de graphe de propriétés a donc été proposé pour offrir une flexibilité supplémentaire lors de la modélisation des données sous forme de graphe **(Angles et al, 2017)**. Un graphe de propriétés permet d'associer un ensemble de paires propriété-valeur et une étiquette aux nœuds et aux arêtes. La **Figure 4** en donne un exemple, montrant à nouveau les vols entre Santiago et Arica et les compagnies proposant ces itinéraires. Cette fois, nous utilisons des paires propriété-valeur sur les arêtes pour modéliser les entreprises, ainsi que la distance. Le type de relation est capturé par l'étiquette flight. Nous modélisons en outre que Santiago et Arica sont des villes utilisant une étiquette de nœud, ainsi que leur pays et si elles sont ou non la capitale en utilisant des paires propriété-valeur sur les nœuds.

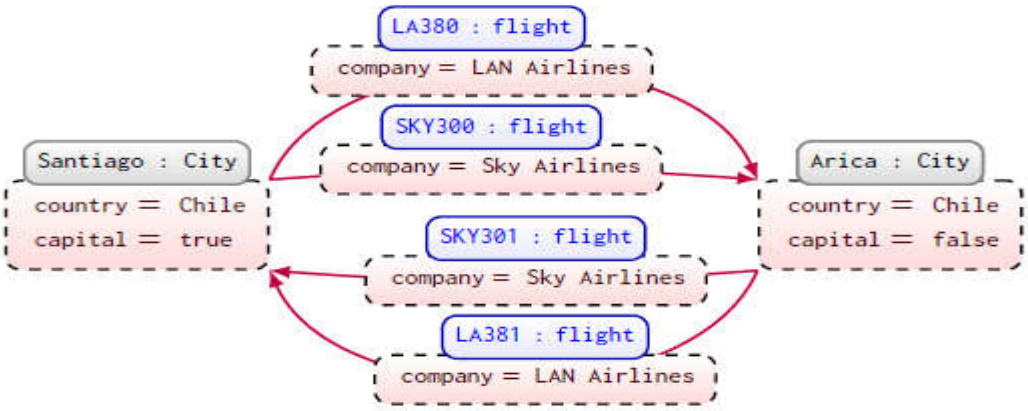
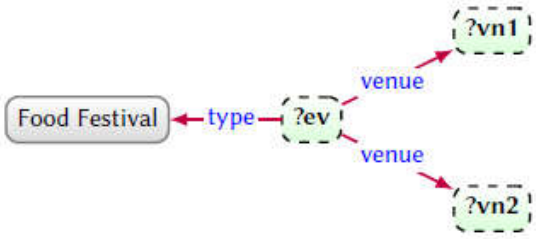


Figure 4: Graphe des propriétés avec les compagnies proposant des vols entre Santiago et Arica

Les graphes de propriétés sont les plus utilisés dans les bases de données de graphes populaires, telles que Neo4j (Angles et al, 2017). Lors du choix entre les modèles de graphes, il est important de noter que les graphes de propriétés peuvent être traduits vers / à partir de graphes à arêtes dirigées et / ou d'ensembles de données de graphes sans perte d'informations (Angles et al, 2019) (par exemple, la figure 4). En résumé, les Directed edge-labelled graphs offrent un modèle plus minimal, tandis que les graphes de propriétés en offrent un plus flexible. Souvent, le choix du modèle sera secondaire par rapport à d'autres facteurs pratiques, tels que les implémentations disponibles pour différents modèles, etc.



?ev	?vn1	?vn2
EID16	Piscina Olímpica	Sotomayor
EID16	Sotomayor	Piscina Olímpica
EID16	Piscina Olímpica	Piscina Olímpica
EID16	Sotomayor	Sotomayor
EID15	Santa Lucía	Santa Lucía

Figure 5: Modèle de graphe (à gauche) avec des mappages générés sur le graphe de la figure 1 (à droite)

1.2. Quering

Un certain nombre de langages pratiques ont été proposés pour interroger les graphes **(Angles et al, 2017)**, y compris le langage d'interrogation SPARQL pour les graphes RDF (Harris et al, 2013); et Cypher **(Francis et al, 2018)**, Gremlin **(Rodriguez, 2015)** et G-CORE **(Angles et al, 2018)** pour interroger les graphes

de propriétés. À la base de ces langages de requête se trouvent des primitives courantes, y compris des modèles de graphes (de base), des opérateurs relationnels, des expressions de chemin, et plus encore (Angles et al, 2017). Nous décrivons maintenant ces fonctionnalités de base pour interroger les graphiques à leur tour, en commençant par les modèles de graphes.

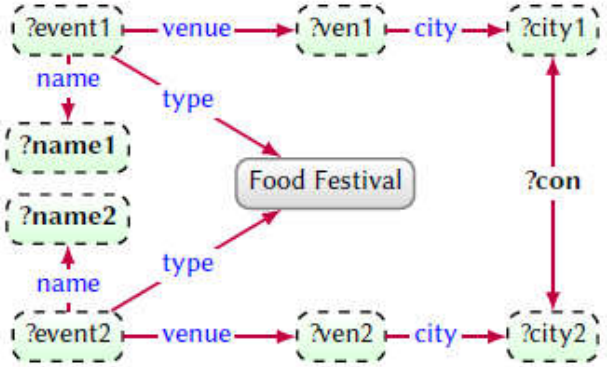
1.2.1. Modèles de graphes

Au cœur de chaque langage de requête structuré pour les graphes se trouvent des modèles de graphes (de base) **(Angles et al, 2017)**, qui suivent le même modèle que le graphe de données interrogé, autorisant en outre des variables comme termes. Les termes dans les modèles de graphes sont ainsi divisés en constantes, telles que Arica ou venue, et en variables, que nous préfixons avec des points d'interrogation, telles que ?event ou? rel. Un modèle de graphe est ensuite évalué par rapport au graphe de données en générant des mappages à partir des variables du modèle de graphe vers des constantes dans le graphe de données de sorte que l'image du modèle de graphe sous le mappage (en remplaçant les variables par les constantes assignées) soit contenue dans les données graphe.

Dans la **Figure 5**, nous fournissons un exemple de modèle de graphique recherchant les lieux des festivals gastronomiques, ainsi que les mappages possibles générés par le modèle de graphe par rapport au graphique de données de la figure 1. Dans certains des mappages présentés (les deux derniers répertoriés), plusieurs variables sont mappées sur le même terme, ce qui peut être souhaitable ou non selon l'application. Par conséquent, un certain nombre de sémantiques ont été proposées pour évaluer les modèles de graphes **(Angles et al, 2017)**, parmi lesquels les plus importantes sont: *la sémantique basée sur l'homomorphisme*, qui permet de mapper plusieurs variables sur le même terme de sorte que tous les mappages illustrés sur la **Figure 5** soient considérés comme des résultats; et *la sémantique isomorphe*, qui nécessite que les variables sur les nœuds et / ou les arêtes soient mappées à des termes uniques, excluant ainsi les trois derniers mappages de la **Figure 5** des résultats. Différents langages pratiques adoptent une sémantique différente pour évaluer des modèles de graphes où, par exemple, **SPARQL** adopte une sémantique basée sur l'homomorphisme, tandis que **Cypher** adopte une sémantique basée sur l'isomorphisme sur les arêtes.

1.2.2. Modèles de graphes complexes

Un modèle de graphe transforme un graphe d'entrée en un tableau de résultats (comme le montre la **Figure 5**). Nous pouvons alors envisager d'utiliser l'algèbre relationnelle pour combiner et / ou transformer ces tableaux, formant ainsi des requêtes plus complexes à partir d'un ou plusieurs modèles de graphes. Rappelons que l'algèbre relationnelle se compose d'opérateurs unaires qui acceptent une table d'entrée, et les opérateurs binaires qui acceptent deux tables d'entrée. Les opérateurs unaires incluent la projection (π) pour générer un sous-ensemble de colonnes, la sélection (σ) pour générer un sous-ensemble de lignes correspondant à une condition donnée et le changement de nom des colonnes (ρ).



?name1	?con	?name2
Food Truck	bus	Food Truck
Food Truck	bus	Food Truck
Food Truck	bus	Ñam
Food Truck	flight	Ñam
Food Truck	flight	Ñam
Ñam	bus	Food Truck
Ñam	flight	Food Truck
Ñam	flight	Food Truck

Figure 6: Requête conjonctive (à gauche) avec des mappages générés sur le graphe de la figure 1 (à droite)

Les opérateurs binaires incluent union (∪) pour fusionner les lignes de deux tables en une seule table, différence (-) pour supprimer les lignes de la première table présente dans la deuxième table et jointures (⋈) pour étendre les lignes d'une table avec des lignes de l'autre table qui satisfait une condition de jointure. Les conditions de sélection et de jointure incluent généralement les égalités (=), les inégalités (≤), la négation (¬), la disjonction (∨), etc. À partir de ces opérateurs, nous pouvons définir davantage d'autres opérateurs (syntaxiques), tels que l'intersection (∩) pour afficher les lignes dans les deux tables, l'anti-jointure (▷, aka n'existe pas) pour générer les lignes de la première table pour lesquelles il n'y a pas de jointure -lignes compatibles dans la deuxième table, jointure à gauche (alias facultatif) pour effectuer une jointure mais en conservant les lignes de la première table sans ligne compatible dans la deuxième table, etc.

Les modèles de graphes peuvent alors être exprimés dans un sous-ensemble d'algèbre relationnelle (à savoir π, σ, ρ, ⋈). En supposant, par exemple, une seule relation ternaire G (s, p, o) représentant un graphe - c'est-à-dire une table G avec trois colonnes s, p, o - la requête de la **figure 5** peut être exprimée en algèbre relationnelle comme:

$$\pi_{ev,vn1,vn2}(\sigma_{p=\texttt{type}\wedge o=\texttt{Food Festival}\wedge p_1=p_2=\texttt{venue}}(\rho_{s/ev}(G \bowtie \rho_{p/p_1,o/vn1}(G) \bowtie \rho_{p/p_2,o/vn2}(G))))$$

où ⋈ désigne une jointure naturelle, ce qui signifie que l'égalité est vérifiée sur les paires de colonnes de même nom dans les deux tables (ici, la jointure est donc effectuée sur la colonne sujet s). résultat de cette requête est une table avec une colonne pour chaque variable: ev, vn1, vn2. Cependant, toutes les requêtes utilisant π, σ, ρ et ⋈ sur G ne peuvent pas être exprimées sous forme de motifs de graphes; par exemple, nous ne pouvons pas choisir les variables à projeter dans un modèle de graphe, mais plutôt projeter toutes les variables non fixées à une constante.

1.2.3. Modèles de graphiques de navigation

Une caractéristique clé qui distingue les langages de requête graphique est la possibilité d'inclure des expressions de chemin dans les requêtes. Une expression de chemin r est une expression régulière qui permet de faire correspondre des chemins de longueur arbitraire entre deux nœuds, qui est exprimée comme une requête de *chemin régulier (x, r, y)*, où x et y peuvent être des variables ou des constantes (ou le même terme).

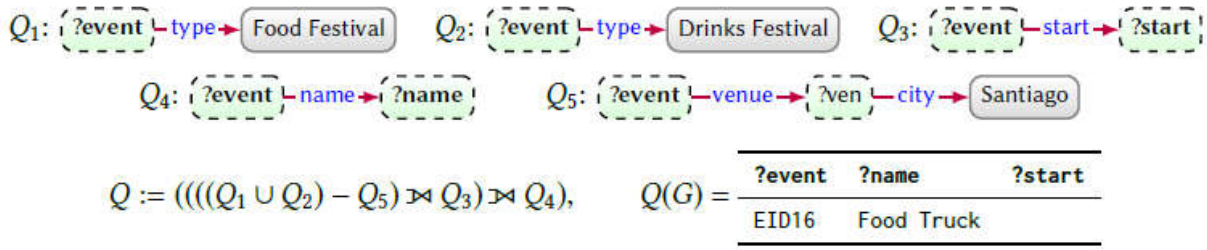


Figure 7: Modèle de graphique complexe (Q) avec mappages générés sur le graphe de la figure 1

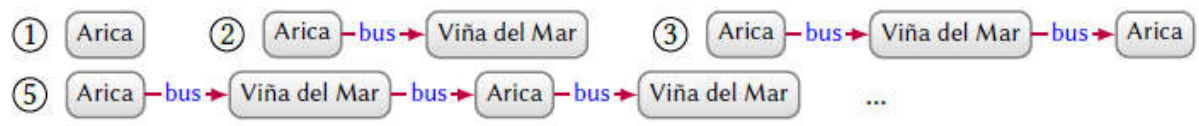
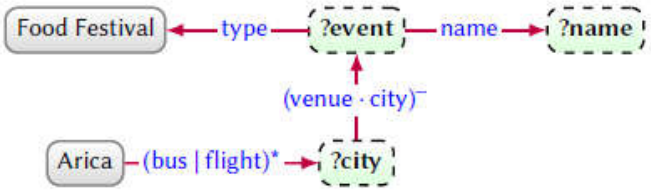


Figure 8: Quelques chemins possibles correspondant (Arica, bus*, ?City) sur le graphe de la figure 1 (Q(G))



?event	?name	?city
EID15	Ñam	Santiago
EID16	Food Truck	Arica
EID16	Food Truck	Viña del Mar

Figure 9: Modèle de graphique de navigation (à gauche) avec mappages générés sur le graphique de la figure 1 (à droite)

L'expression de chemin de base est où r est une constante (une étiquette d'arête).

2. SCHÉMA, IDENTITÉ, CONTEXTE

Dans cette section, nous décrivons diverses améliorations et extensions du graphe de données (relatives au schéma, à l'identité et au contexte) qui fournissent des structures supplémentaires pour accumuler des connaissances. Désormais, nous nous référons à un graphe de données comme une collection de données représentées sous forme de nœuds et d'arêtes en utilisant l'un des modèles discutés dans la section précédente. Nous désignons un graphe de connaissances comme un graphe de données potentiellement enrichi de représentations de schéma, d'identité, de contexte, d'ontologies et / ou de règles. Ces représentations supplémentaires peuvent être intégrées dans le graphe de données ou superposées au-dessus. Les représentations du schéma, de l'identité et du contexte sont discutées dans ce qui suit.

2.1. SCHÉMA

L'un des avantages de la modélisation des données sous forme de graphes - par rapport, par exemple, au modèle relationnel - est la possibilité de renoncer ou de reporter la définition d'un schéma. Cependant, lors de la modélisation de données sous forme de graphes, des schémas peuvent être utilisés pour prescrire une structure de haut niveau et / ou une sémantique que le graphique suit ou devrait suivre. Nous discutons de trois types de schémas de graphes : sémantiques, de validation et émergent.

2.1.1. Schéma sémantique.

Un schéma sémantique permet de définir la signification des termes de haut niveau (ou *vocabulaire* ou *terminologie*) utilisés dans le graphe , ce qui facilite le raisonnement sur les graphes utilisant ces termes. En regardant la figure 1, par exemple, nous pouvons remarquer certains regroupements naturels de nœuds basés sur les types d'entités auxquels ils se réfèrent. On peut donc décider de définir des *classes* pour désigner ces regroupements, tels que Event, City, etc. En fait, **la Figure 1** illustre déjà trois classes de bas niveau - Open Market, Food Market, Drinks Festival - regroupement des entités similaires avec une arête étiquetée **type**. On peut observer par la suite des relations naturelles entre certaines de ces classes que l'on aimerait capturer. Dans la figure 10, nous présentons une hiérarchie de classes pour les événements où les enfants sont définis comme des *sous-classes* de leurs parents de telle sorte que si nous trouvons une arête EID15 → **type** → Food Festival dans notre graphique, nous pouvons également en déduire que **EID15 → type → Festival** et **EID15 → type → Event**.

En plus des classes, nous pouvons également souhaiter définir la sémantique des étiquettes d'arête, aka propriétés. En revenant à la **Figure 1**, nous pouvons considérer que les propriétés city et venue sont des *sous-propriétés* d'une propriété plus général location, de sorte que, étant donné un bord Santa **Lucía-city→Santiago**, par exemple, nous pouvons également en déduire que Santa **Lucía- location→Santiago**. Nous pouvons également considérer, par exemple, que le bus et le vol sont tous deux des sous-propriétés d'une propriété plus générale connects to. À ce titre, les propriétés peuvent également former une hiérarchie. Nous pouvons définir en outre le domaine des propriétés, en indiquant la ou les classes d'entités pour les nœuds à partir desquels les arêtes avec cette propriété s'étendent.

Une norme importante pour définir un schéma sémantique pour les graphes (**RDF**) est la norme **RDF** Schema (**RDFS**) **(Brickley and Guha, 2014)**, qui permet de définir des sous-classes, des sous-propriétés, des domaines et des plages parmi les classes et propriétés utilisées dans un graphe RDF, où les définitions peuvent être sérialisées sous forme de graphique. Nous illustrons la sémantique de ces fonctionnalités dans le tableau 1 et fournissons un exemple concret des définitions de l**a Figure 11** pour un échantillon de termes utilisés dans l'exemple en cours d'exécution. Ces définitions peuvent ensuite être intégrées dans un

graphe de données. Plus généralement, la sémantique des termes utilisés dans un graphe peut être définie avec beaucoup plus de profondeur que ce que l'on voit ici, comme le supporte le standard OWL (Web Ontology Language) **(Hitzler et al, 2012)** pour les graphes RDF.

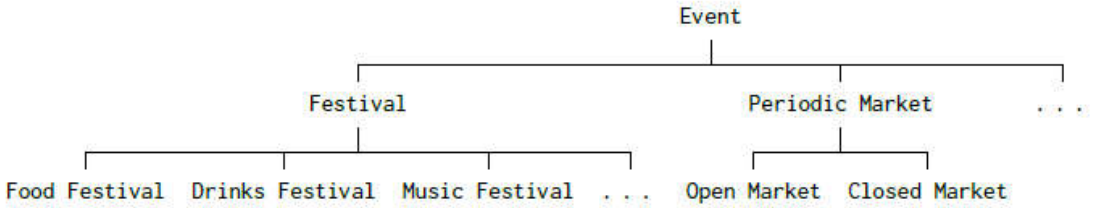


Figure 10: Exemple de hiérarchie de classes pour Event

Tableau 1.

Tableau 1:Définitions des caractéristiques de sous-classe, de sous-propriété, de domaine et de range dans les schémas sémantiques

Feature	Definition	Condition	Example
SUBCLASS	c — subc. of → d	x — type → c implies x — type → d	City — subc. of → Place
SUBPROPERTY	p — subp. of → q	x — p → y implies x — q → y	venue — subp. of → location
DOMAIN	p — domain → c	x — p → y implies x — type → c	venue — domain → Event
RANGE	p — range → c	x — p → y implies y — type → c	venue — range → Venue

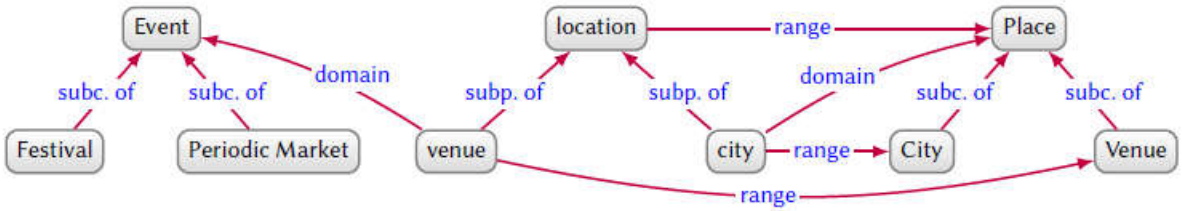


Figure 11: Exemple de graphe de schéma décrivant les sous-classes, sous-propriétés, domaines et ranges

2.1.2. Validation du schéma.

Lorsque les graphiques sont utilisés pour représenter des données diverses et incomplètes à grande échelle, **l'OWA** est le choix le plus approprié pour une sémantique par défaut. Mais dans certains scénarios, nous souhaitons peut-être garantir que notre graphe de données - ou des parties spécifiques de celui-ci - sont en un certain sens «complets».

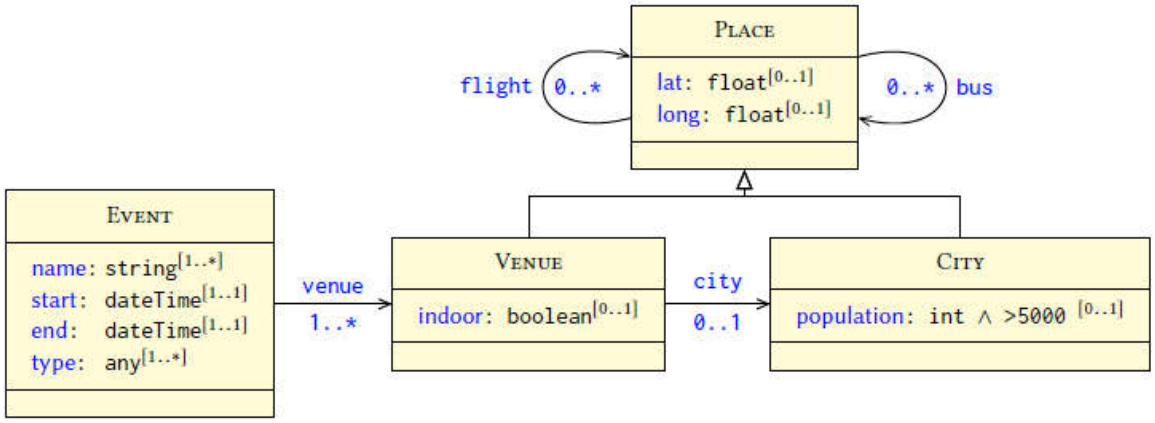


Figure 12: Exemple de graphique de formes représenté sous forme de diagramme de type UML

Revenant à la **Figure 1**, par exemple, nous souhaitons peut-être nous assurer que tous les événements ont au moins un nom, un lieu, une date de début et une date de fin, de sorte que les applications utilisant les données - par exemple, une application qui envoie des notifications d'événements aux utilisateurs - peuvent garantir qu'ils disposent des informations minimales requises. En outre, nous pouvons souhaiter nous assurer que la ville d'un événement est déclarée comme étant une ville (plutôt que de déduire qu'il s'agit d'une ville). Nous pouvons définir ces contraintes dans un schéma de validation et valider le graphe de données par rapport au schéma résultant, en listant les violations de contraintes (le cas échéant). Ainsi, alors que les schémas sémantiques permettent de déduire de nouvelles données graphes, la validation des schémas permet de valider les données graphes existantes.

Une manière standard de définir un schéma de validation pour les graphiques consiste à utiliser des formes **(Knublauch and Kontokostas, 2017)**. Une forme cible un ensemble de nœuds dans un graphe de données et spécifie des contraintes sur ces nœuds. La cible de la forme peut être définie de plusieurs manières, par exemple en ciblant toutes les instances d'une classe, le domaine ou la plage d'une propriété, le résultat d'une requête, les nœuds connectés à la cible d'une autre forme par une propriété donnée, etc. Des contraintes peuvent alors être définies sur les nœuds ciblés, par exemple pour restreindre le nombre ou les types de valeurs prises sur une propriété donnée. Un graphe de formes est formé d'un ensemble de formes interdépendantes.

Les graphes de formes peuvent être représentés sous forme de diagrammes de classes de type **UML**, où la **Figure 12** illustre un exemple de graphe de formes

basé sur la figure 1, définissant des contraintes sur quatre formes interdépendantes. forme - indiquée par une boîte comme Place, Event, etc., est associé à un ensemble de contraintes. Les nœuds se conforment à une forme si et seulement s'ils satisfont toutes les contraintes définies sur la forme. chaque boîte de forme est soumise à des contraintes sur le nombre (par exemple, [1 .. *] désigne un à plusieurs, [1..1] désigne précisément un, etc.) et les types (par exemple, string, dateTime, etc.) de nœuds auxquels les nœuds conformes peuvent se rapporter avec une propriété (par exemple, name, start, etc.). Une autre option consiste à placer des contraintes sur le nombre de nœuds conformes à une forme particulière que le nœud conforme peut associer à une propriété (générant ainsi des arêtes entre les formes).

2.1.3. Schéma émergent.

Les schémas sémantiques et de validation nécessitent qu'un expert du domaine spécifie explicitement les définitions et les contraintes. Cependant, un graphe de données présentera souvent des structures latentes qui peuvent être automatiquement extraites comme un schéma émergent **(Pham et al, 2015)**.

Un cadre souvent utilisé pour définir un schéma émergent est celui des graphes de quotient, qui partitionnent des groupes de nœuds dans le graphe de données selon une relation d'équivalence tout en préservant certaines propriétés structurelles du graphe. En prenant la **Figure 1**, nous pouvons distinguer intuitivement différents types de nœuds en fonction de leur contexte, tels que les nœuds d'événements, qui se lient aux nœuds de lieu, qui à leur tour se lient aux nœuds de la ville, et ainsi de suite. Afin de décrire la structure du graphe, on pourrait considérer six partitions de nœuds: événement, nom, lieu, classe, date-heure, ville. En pratique, ces partitions peuvent être calculées en fonction de la classe ou de la forme du nœud.

La fusion des nœuds de chaque partition en un nœud tout en préservant les arêtes conduit au graphe de quotient illustré à la **Figure 13** : les nœuds de ce graphe de quotient sont les partitions des nœuds du graphe de données et l'arête $X\text{-}y\rightarrow Z$ est dans le graphe quotient si et seulement s'il existe $x \in X$ et $z \in Z$ tels que $x\text{-}y\rightarrow z$ est dans le graphe de données.

Figure 13: Exemple de graphe de quotient simulant le graphe de données de la figure 1

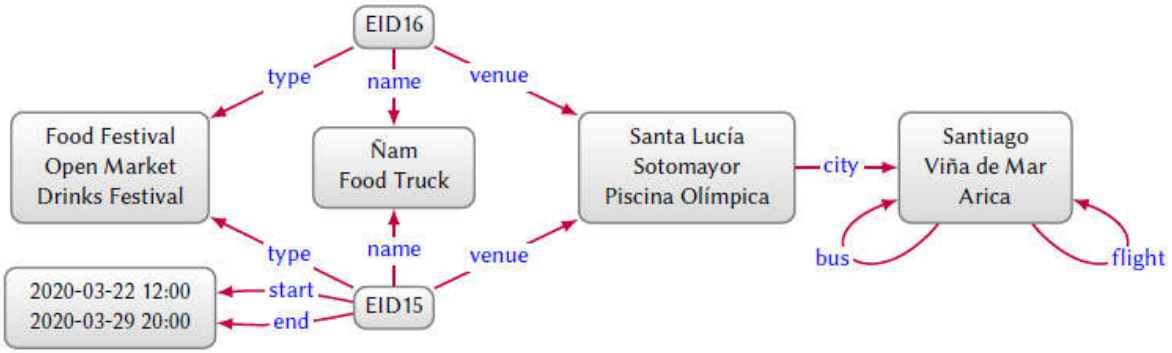


Figure 14: Exemple de graphe de quotient bisimilaire avec le graphe de données de la figure 1

Il existe de nombreuses façons de définir les graphes de quotient, en fonction non seulement de la façon dont les nœuds sont partitionnés, mais également de la manière dont les arêtes sont définies. Différents graphes de quotient peuvent fournir différentes garanties quant à la structure qu'ils préservent. Formellement, nous pouvons dire que chaque graphe quotient simule son graphe d'entrée (basé sur la relation de simulation d'appartenance d'ensemble entre les nœuds de données et les nœuds de quotient), ce qui signifie que pour tout x ∈ X avec x un nœud d'entrée et X un nœud de quotient, si x-y→z est une arête dans le graphe de données, alors il doit exister une arête X-y→Z dans le graphe quotient tel que z ∈ Z; par exemple, le graphique de quotient de la **Figure 13** simule le graphique de données de la figure 1. Cependant, ce graphique de quotient semble suggérer (par exemple) que l'EID16 aurait une date de début et de fin dans le graphique de données lorsque ce n'est pas le cas.

Il existe de nombreuses façons de définir les graphiques de quotient, selon la relation d'équivalence qui partitionne les nœuds. En outre, il existe de nombreuses façons de définir d'autres graphes similaires ou bisimilaires, en fonction de la relation de (bi) simulation qui préserve la structure du graphe de données **(Čebirić et al, 2019)**. Ces techniques visent à résumer le graphe de données dans une

topologie de niveau supérieur. Afin de réduire la surcharge mémoire du graphe de quotient, en pratique, les nœuds peuvent plutôt être étiquetés avec la cardinalité de la partition et / ou une étiquette de haut niveau (par exemple, event, city) pour la partition plutôt que de stocker les étiquettes de tous les nœuds de la partition.

Diverses autres formes de schémas émergents non basés sur un cadre de graphe de quotient ont également été proposées ; les exemples incluent des schémas émergents basés sur des tables relationnelles **(Pham et al, 2015)**, une analyse de concept formelle **(González and Hogan, 2018)**, etc. Des schémas émergents peuvent être utilisés pour fournir une vue d'ensemble compréhensible par l'homme du graphique de données, pour aider à la définition d'un schéma sémantique ou de validation, pour optimiser l'indexation et l'interrogation du graphique, pour guider l'intégration des graphiques de données, etc. .

2.2. Identité

Dans la **Figure 1**, nous utilisons des nœuds comme Santiago, mais à quel Santiago ce nœud se réfère-t-il ? Faisons-nous référence à Santiago du Chili, Santiago de Cuba, Santiago de Compostelle, ou peut-être parlons-nous du groupe de rock indépendant Santiago ? Sur la base de bords tels que Santa **Lucía-city→Santiago**, nous pouvons en déduire qu'il s'agit de l'une des trois villes mentionnées (pas le groupe de rock), et sur la base du fait que le graphique décrit les attractions touristiques au Chili, nous pouvons en déduire qu'il fait référence à Santiago du Chili. Sans plus de détails, cependant, les nœuds de désambiguïsation de cette forme peuvent s'appuyer sur des heuristiques sujettes à l'erreur dans des cas plus difficiles. Pour éviter une telle ambiguïté, nous pouvons d'abord utiliser des identifiants uniques au niveau mondial pour éviter les conflits de noms lorsque le graphe de connaissances est étendu avec des données externes, et deuxièmement, nous pouvons ajouter des liens d'identité externes pour lever l'ambiguïté d'un nœud par rapport à une source externe.

2.2.1. Identifiants globaux.

Supposons que nous souhaitions comparer le tourisme au Chili et à Cuba et que nous ayons acquis un graphique de connaissances approprié pour Cuba. Un des avantages de l'utilisation de graphiques pour modéliser des données est que nous pouvons fusionner deux graphiques en prenant leur union. Comme le montre la **Figure 15**, l'utilisation d'un nœud ambigu comme Santiago peut entraîner un conflit de noms : le nœud fait référence à deux villes différentes du monde réel

dans les deux graphiques, où le graphique fusionné indique que Santiago est une ville à la fois au Chili et à Cuba (plutôt que deux villes différentes). Un moyen pratique d'éviter de tels conflits de noms serait d'utiliser des espaces de noms comme chile : ou cuba : dans les graphiques correspondants, de sorte que des nœuds comme chile:Santiago et cuba:Santiago ne s'affronteront pas tant que des espaces de noms distincts sont utilisés.

Dans le contexte du Web sémantique, le modèle de données RDF va plus loin et recommande d'utiliser des identifiants Web globaux pour les nœuds et les étiquettes de périphérie. Cependant, plutôt que d'adopter les **URL (Uniform Resource Locators)** utilisés pour identifier l'emplacement des ressources d'information telles que les pages Web, RDF 1.1 propose d'utiliser les identificateurs de ressources internationalisés (IRI) pour identifier les ressources non informatives telles que les villes ou les événements. Ainsi, par exemple, dans la représentation RDF de **Wikidata (Vrandečić and Krötzsch, 2014)** tandis que l'URL **https://www.wikidata.org/wiki/Q2887** fait référence à une page Web qui peut être chargée dans un navigateur fournissant des méta-données lisibles par l'homme sur Santiago, l'IRI **http://www.wikidata.org/entity/Q2887** fait référence à la ville elle-même. Distinguer les identifiants des deux ressources (la page Web et la ville elle-même) évite les conflits de noms; par exemple, si nous utilisons l'URL pour identifier à la fois la page Web et la ville, nous pouvons nous retrouver avec un bord dans notre graphique, tel que (avec des étiquettes lisibles sous le bord):

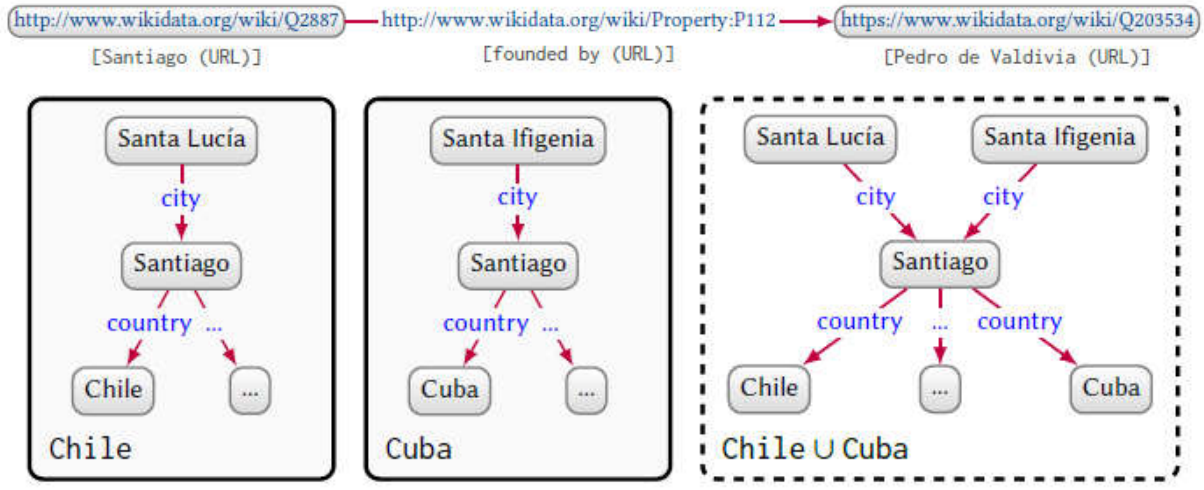
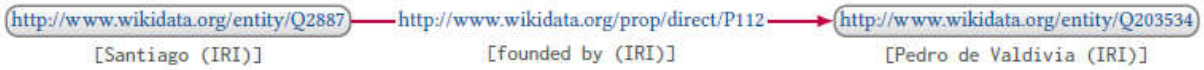


Figure 15:Résultat de la fusion de deux graphiques avec des identificateurs locaux ambigus

Un tel avantage laisse une certaine ambiguïté : Pedro de Valdivia était-il le fondateur de la page Web, ou de la ville ? L'utilisation d'IRI pour des entités distinctes des URL des pages Web qui les décrivent évite de tels cas ambigus, où Wikidata définit donc plutôt l'arête précédente comme suit:



en utilisant des IRI pour la ville, la personne et le fondateur de, distincts des pages Web qui les décrivent.

Si les IRI HTTP sont utilisés pour identifier les entités du graphe, lorsque l'IRI est recherchée (via HTTP), le serveur Web peut renvoyer (ou rediriger vers) une description de cette entité dans des formats tels que RDF. permet en outre aux graphes RDF de se lier à des entités liées décrites dans des graphes RDF externes sur le Web, donnant lieu à des données liées **(Berners-Lee, 2006)**, **(Heath and Bizer, 2011)**.

2.2.2. Liens d'identité externes.

Supposons que l'office du tourisme choisisse de définir le chile: namespace avec un IRI tel que **http://turismo.cl/entity/** sur un serveur Web qu'ils contrôlent, permettant des nœuds tels que chile: Santiago - un raccourci pour l'IRI http : //turismo.cl/entity/Santiago - à rechercher sur le Web. Si l'utilisation d'un tel schéma de dénomination permet d'éviter les conflits de dénomination, l'utilisation d'IRI n'aide pas nécessairement à ancrer l'identité d'une ressource. Par exemple, un graphe de connaissance géographique externe peut attribuer à la même ville l'IRI geo: SantiagoDeChile dans son propre espace de noms, où nous n'avons aucun moyen direct de savoir que les deux identifiants se réfèrent à la même ville. Si nous fusionnons les deux graphes de connaissances, nous nous retrouverons avec deux nœuds distincts pour la même ville.

Il existe un certain nombre de façons de fonder l'identité d'une entité. La première consiste à associer l'entité à des informations d'identification unique dans le graphique, telles que ses coordonnées géographiques, son code postal, l'année de sa création, etc. Une information supplémentaire supprime l'ambiguïté quant à la ville visée, offrant (par exemple) plus d'options pour faire correspondre la ville avec son analogue dans des sources externes. Une deuxième option consiste à utiliser des liens d'identité pour déclarer qu'une entité locale a la même identité qu'une autre entité coréférente trouvée dans une source externe; une instanciation

de ce concept peut être trouvée dans le standard OWL, qui définit la propriété owl: sameAs relative aux entités coreferent. En utilisant cette propriété, nous pourrions indiquer le bord chile: Santiago-owl: sameAs→geo: SantiagoDeChile dans notre graphe RDF, établissant ainsi un lien d'identité entre les nœuds correspondants dans les deux graphes. La sémantique de owl: sameAs définie par le standard OWL nous permet alors de combiner les données des deux nœuds.

2.2.3. Types de données.

Considérez les deux dates-heures à gauche de la **Figure 1** : comment attribuer des identifiants globaux à ces nœuds? Intuitivement, il n'aurait pas de sens, par exemple, d'attribuer des IRI à ces nœuds puisque leur forme syntaxique nous indique à quoi ils font référence : des dates et des heures spécifiques en mars 2020. Cette forme syntaxique est en outre reconnaissable par la machine, ce qui signifie qu'avec un logiciel approprié, nous pourrions ordonner ces valeurs par ordre croissant ou décroissant, extraire l'année, et ainsi de suite.

La plupart des modèles de données pratiques pour les graphiques permettent de définir des nœuds qui sont des valeurs de type de données. RDF utilise des types de données de schéma XML (XSD) **(Peterson et al, 2012)**, entre autres, où un nœud de type de données est donné sous forme de paire (l, d) où l est une chaîne lexicale, telle que "2020-03-29T20: 00: 00", et d est un IRI indiquant le type de données, tel que xsd: dateTime. Le nœud est alors noté "2020-03-29T20: 00: 00" ^^ xsd: dateTime. Les nœuds de type de données en RDF sont appelés littéraux et ne sont pas autorisés à avoir des arêtes sortantes. Les autres types de données couramment utilisés dans les données RDF incluent xsd: string, xsd: integer, xsd: decimal, xsd: boolean, etc. Dans le cas où le type de données est omis, la valeur est supposée être de type xsd: string. Les applications construites sur RDF peuvent alors reconnaître ces types de données, les analyser en objets de type de données et appliquer des vérifications d'égalité, une normalisation, un ordre, des transformations, un casting, selon leur définition standard. Dans le contexte des graphes de propriétés, Neo4j **(Miller, 2013)** définit également un ensemble de types de données internes sur les valeurs de propriété qui comprend des nombres, des chaînes, des booléens, des points spatiaux et des valeurs temporelles.

2.2.4. Lexicalisation.

Les identificateurs globaux pour les entités auront parfois une forme interprétable par l'homme, comme chile: Santiago, mais les chaînes d'identificateurs elles-mêmes n'ont aucune signification sémantique formelle. Dans d'autres cas, les

identificateurs globaux utilisés peuvent ne pas être interprétables par l'homme par conception. Dans **Wikidata**, par exemple, Santiago du Chili est identifié comme wd: Q2887, où un tel schéma a l'avantage de fournir une meilleure persistance et de ne pas être biaisé vers une langue humaine particulière. Par exemple, l'identifiant Wikidata pour Eswatini (wd: Q1050) n'a pas été affecté lorsque le pays a changé son nom de Swaziland, et ne nécessite pas de choisir entre les langues pour créer des IRI (plus lisibles) tels que wd: Eswatini (anglais), wd: eSwatini (Swazi), wd: Esuatini (espagnol), etc.

Comme les identificateurs peuvent être arbitraires, il est courant d'ajouter des arêtes qui fournissent une étiquette interprétable par l'homme pour les nœuds, comme wd: Q2887-rdfs: label→«Santiago», indiquant comment les gens peuvent se référer linguistiquement au nœud sujet. Les informations linguistiques de cette forme jouent un rôle important dans l'enracinement des connaissances, de sorte que les utilisateurs peuvent identifier plus clairement à quelle entité du monde réel un nœud particulier dans un graphe de connaissances fait réellement référence **(de Melo, 2015)**; il permet en outre de faire des références croisées d'étiquettes d'entité avec des corpus de texte pour trouver, par exemple, des documents qui parlent potentiellement d'une entité donnée **(Martínez-Rodríguez et al, 2020)**. Les libellés peuvent être complétés par des alias (par exemple, wd: Q2887-skos: altLabel→"Santiago de Chile") ou des commentaires (par exemple, wd: Q2887-rdfs: comment→"Santiago est la capitale du Chili") pour aider davantage à ancrer l'identité du nœud.

Les nœuds tels que «Santiago» désignent des chaînes littérales, plutôt qu'un identificateur. Selon le modèle de graphe spécifique, ces nœuds littéraux peuvent également être définis comme une paire (s, l), où s désigne la chaîne et l un code de langue; en RDF, par exemple, on peut indiquer chile: City-rdfs: label→"City" @en, chile: City-rdfs: label→"Ciudad" @es, etc., indiquant les libellés du nœud dans différentes langues. Dans d'autres modèles, la langue pertinente peut plutôt être spécifiée, par exemple via des métadonnées en périphérie. Les graphes de connaissances avec des étiquettes, alias, commentaires, etc. interprétables par l'homme (dans diverses langues) sont parfois appelés graphes de connaissances lexicalisés (multilingues) **(Bonatti et al, 2020)**.

2.2.5 Nœuds existentiels

Lors de la modélisation d'informations incomplètes, nous pouvons dans certains cas savoir qu'il doit exister un nœud particulier dans le graphe avec des relations

particulières avec d'autres nœuds, mais sans pouvoir identifier le nœud en question. Par exemple, nous pouvons avoir deux événements co-localisés au Chili: EID42 et Chili: EID43 dont le lieu n'a pas encore été annoncé. Une option consiste simplement à omettre les bords du lieu, auquel cas nous perdons l'information selon laquelle ces événements ont un lieu et que les deux événements ont le même lieu. Une autre option pourrait être de créer un nouvel IRI représentant le lieu, mais sémantiquement, cela devient impossible à distinguer du fait qu'il existe un lieu connu. Par conséquent, certains modèles de graphes permettent l'utilisation de nœuds existentiels, représentés ici par un cercle vide:

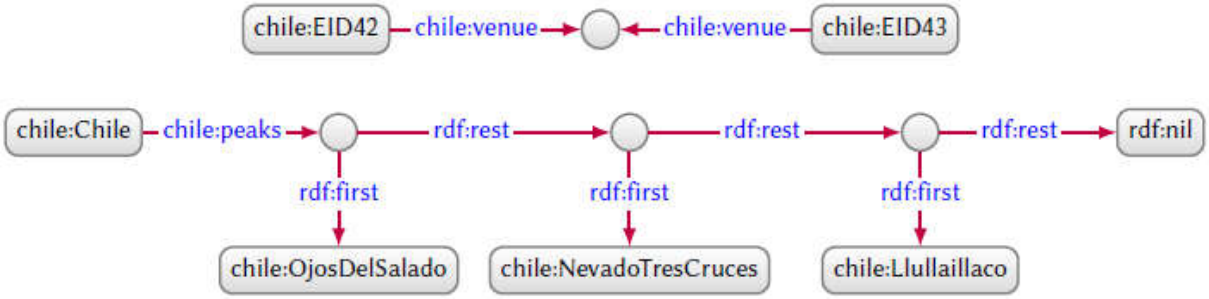


Figure 16: Liste RDF représentant les trois plus grands sommets du Chili, dans l'ordre

Ces arêtes indiquent qu'il existe un lieu commun pour le chili: EID42 et le chili: EID42 sans l'identifier. Les nœuds existentiels sont pris en charge dans RDF en tant que nœuds vides **(Cyganiak et al, 2014)**, qui sont également couramment utilisés pour prendre en charge la modélisation d'éléments complexes dans les graphes, comme les listes RDF **(Cyganiak et al, 2014)**, **(Hogan et al, 2014)**. La figure 16 illustre une liste RDF, qui utilise des nœuds vides dans une structure de liste chaînée pour coder l'ordre. Bien que les nœuds existentiels puissent être pratiques, leur présence peut compliquer les opérations sur les graphes, comme décider si deux graphiques de données ont la même structure de nœuds modulo existentiels **(Cyganiak et al, 2014)**. Par conséquent, des méthodes pour skolémiser les nœuds existentiels dans les graphes - en les remplaçant par des étiquettes canoniques - ont été proposées **(Hogan , 2017)**. D'autres auteurs appellent plutôt à minimiser l'utilisation de tels nœuds dans les données graphiques **(Cyganiak et al, 2014)**.

2.3. Contexte

De nombreux faits (sans doute tous) présentés dans le graphique de données de la **Figure 1** peuvent être considérés comme vrais par rapport à un certain contexte. En ce qui concerne le contexte temporel, Santiago n'existe en tant que ville que depuis 1541, les vols d'Arica à Santiago ont commencé en 1956, etc. En ce qui concerne le contexte géographique, le graphique décrit les événements au Chili. En ce qui concerne la provenance, les données relatives à l'EID15 ont été extraites - et sont donc dites vraies en ce qui concerne - la page Web de Ñam le 4 janvier 2020. D'autres formes de contexte peuvent également être utilisées. On peut en outre combiner des contextes, comme pour indiquer qu'Arica est une ville chilienne (géographique) depuis 1883 (temporelle) selon le traité d'Ancón (provenance).

Par contexte, nous nous référons ici à la portée de la vérité et parlons ainsi du contexte dans lequel certaines données sont considérées comme vraies **(Guha et al, 2004)**, **(McCarthy, 1993)**. Le graphique de la **Figure 1** laisse une grande partie de son contexte implicite. Cependant, rendre le contexte explicite peut permettre d'interpréter les données sous différents angles, par exemple pour comprendre ce qui était vrai en 2016, ce qui est vrai en excluant les pages Web trouvées plus tard comme contenant des données fausses, etc. Comme le montrent les exemples précédents, le contexte des données graphiques peuvent être considérés à différents niveaux: sur des nœuds individuels, des arêtes individuelles ou des ensembles d'arêtes (sous-graphes). Nous discutons maintenant de diverses représentations par lesquelles le contexte peut être rendu explicite à différents niveaux.

2.3.1. Représentation directe

La première façon de représenter le contexte est de le considérer comme des données non différentes des autres données. Par exemple, les dates de l'événement EID15 de la **Figure 1** peuvent être considérées comme représentant une forme de contexte temporel, indiquant la portée temporelle dans laquelle les arêtes telles que EID15-venue→Santa Lucía sont tenues vraies. Une autre option consiste à changer une relation représentée comme une arête, comme Santiago-flight→Arica, en un nœud, comme le montre la **Figure 3**, permettant d'attribuer un contexte supplémentaire à la relation. Alors que dans ces exemples, le contexte est représenté de manière ad hoc, un certain nombre de spécifications ont été proposées pour représenter le contexte sous forme de données d'une manière plus standard. Un exemple est *l'ontologie du temps* **(Cox et al, 2017)**, qui spécifie

comment les entités temporelles, les intervalles, les instants de temps, etc. - et les relations entre eux comme avant, chevauchements, etc. - peuvent être décrits dans les graphes RDF de manière interopérable. Un autre exemple est le modèle de données PROV **(Gil et al, 2013)**, qui spécifie comment la provenance peut être décrite dans les graphiques RDF, où les entités (par exemple, graphiques, nœuds, document physique) sont dérivées d'autres entités, sont générées et / ou utilisées par des activités (par exemple, extraction, paternité) et sont attribués à des agents (par exemple, des personnes, des logiciels, des organisations).

2.3.2. Réification

Souvent, nous souhaitons définir directement le contexte des arêtes elles-mêmes; par exemple, on peut souhaiter dire que Santiago-flight→Arica est valide depuis de 1956. Alors que nous pourrions utiliser le modèle de transformation de l'arête en nœud - comme illustré dans la **Figure 3** - pour représenter directement ce contexte, une autre option consiste à utiliser la réification, qui permet de faire des déclarations sur les déclarations de manière générique (ou dans le cas de un graphe, pour définir les arêtes autour des arêtes). Dans la **Figure 17**, nous présentons trois formes de réification qui peuvent être utilisées pour modéliser le contexte temporel sur l'arête mentionnée ci-dessus dans un graphe marqué par l'arête dirigée **(Hernández et al, 2017).**

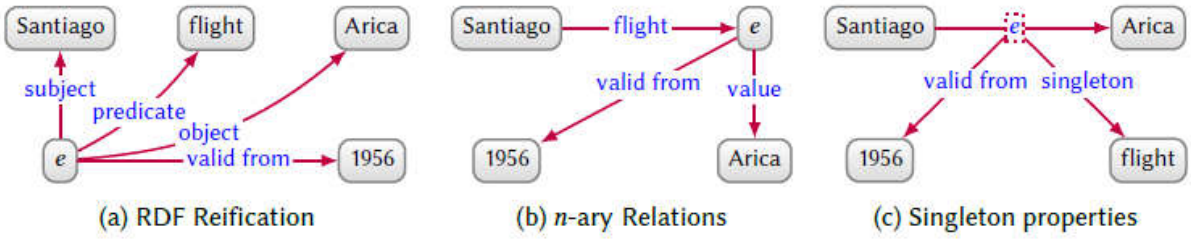


Figure 17: Trois représentations du contexte temporel sur une arête dans un graphe étiqueté à arête dirigée

Dans la **Figure 17,** nous présentons trois formes de réification qui peuvent être utilisées pour modéliser le contexte temporel sur l'arête susmentionnée dans un graphe orienté marqué par l'arête. Nous utilisons e pour désigner un identifiant arbitraire représentant l'arête elle-même vers laquelle les informations contextuelles peuvent être associé. Contrairement à une représentation directe, e représente une arête, pas un vol. La réification RDF **(Cyganiak et al, 2014)** (**Figure 17a)** définit un nouveau nœud e pour représenter le bord et le connecte au nœud source (via le sujet), au nœud cible (via l'objet) et à l'étiquette de bord

(via le prédicat) du bord. En revanche, les relations n-aires **(Cyganiak et al, 2014)** (**Figure 17b**) relient le nœud source de l'arête directement au nœud d'arête e avec l'étiquette de l'arête; le nœud cible de l'arête est alors connecté à e (via valeur). Enfin, les propriétés singleton (**Figure 17c)** utilisent plutôt e comme étiquette d'arête, la connectant à un nœud indiquant l'étiquette d'arête d'origine (via singleton). D'autres formes de réification ont été proposées dans la littérature, dont par exemple les NdFluents **(Giménez-García et al, 2017)**. En général, un bord réifié n'affirme pas le bord qu'il réifie; par exemple, nous pouvons réifier une arête pour déclarer qu'elle n'est plus valide. Nous nous référons aux travaux de Hernández et al **(Hernández et al, 2017)**. pour une comparaison plus approfondie des alternatives de réification et de leurs forces et faiblesses relatives.

2.3.3. Représentation de plus haute arité

Comme alternative à la réification, nous pouvons plutôt utiliser des représentations de plus haute arité pour modéliser le contexte. Reprenant le bord Santiago-flight→Arica, la **Figure 18** illustre trois représentations plus arites du contexte temporel. Tout d'abord, nous pouvons utiliser un graphe nommé (**Figure 18a**) pour contenir l'arête, puis définir le contexte temporel sur le nom du graphe. Deuxièmement, nous pouvons utiliser un graphe de propriétés (**Figure 18b**) où le contexte temporel est défini comme un attribut sur le bord. Troisièmement, nous pouvons utiliser RDF * (**Hartig, 2017**) (**Figure 18c**): une extension de RDF qui permet de définir des arêtes comme des nœuds. Parmi ces options, la plus flexible est la représentation graphique nommée, où nous pouvons attribuer un contexte à plusieurs arêtes à la fois en les plaçant dans un graphe nommé; par exemple, nous pouvons ajouter d'autres arêtes au graphe nommé de la figure 18a qui sont également valides à partir de 1956. L'option la moins flexible est RDF *, qui, en l'absence d'un identifiant d'arête, ne permet pas d'affecter différents groupes de valeurs contextuelles à une arête; par exemple, en considérant le bord Chili-président→M.Bachelet, si nous ajoutons quatre valeurs contextuelles à ce bord pour affirmer qu'il était valide de 2006 à 2010 et valide de 2014 à 2018, nous ne pouvons pas coupler les valeurs, mais plutôt devoir créer un nœud pour représenter différentes présidences (dans les autres modèles, nous aurions pu utiliser deux graphes nommés ou identifiants d'arête).

2.3.4. Annotations

Jusqu'à présent, nous avons discuté de la représentation du contexte dans un graphique, mais nous n'avons pas parlé de mécanismes automatisés pour raisonner sur le contexte; par exemple, s'il n'y a que des vols d'été saisonniers de Santiago à Arica, nous pouvons souhaiter trouver d'autres itinéraires de Santiago pour les événements d'hiver qui se déroulent à Arica. Alors que les dates pour les bus, les vols, etc. peuvent être représentées directement dans le graphique, ou en utilisant la réification, écrire une requête pour intersecter manuellement les contextes temporels correspondants sera fastidieux - voire pas du tout possible. Une autre alternative consiste à envisager des annotations qui fournissent des définitions mathématiques d'un domaine contextuel et des opérations clés possibles dans ce domaine qui peuvent ensuite être appliquées automatiquement.

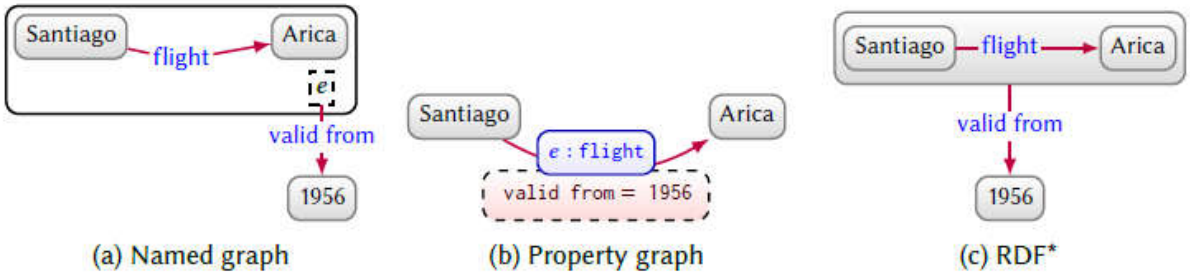


Figure 18: Trois représentations d'arité supérieure du contexte temporel sur une arête

Certaines annotations modélisent un domaine contextuel particulier; par exemple, Temporal RDF **(Gutiérrez et al, 2007)** permet d'annoter des arêtes avec des intervalles de temps, comme Chilie-président[2006, 2010]→M. Bachelet, tandis que Fuzzy RDF **(Straccia, 2009)** permet d'annoter des arêtes avec un degré de vérité tel que le Santiago- climat0,8→Semi- Aride, indiquant qu'il est plus ou moins vrai - avec un degré de 0,8 - que Santiago a un climat semi-aride.

2.3.5. Autres cadres contextuels

D'autres cadres ont été proposés pour la modélisation et le raisonnement sur le contexte dans les graphiques. Un exemple notable est celui des référentiels de connaissances contextuelles **(Homola and Serafini, 2012),** qui permettent d'assigner des (sous-) graphes individuels à leur propre contexte. Contrairement au cas des graphes nommés, le contexte est explicitement modélisé selon une ou plusieurs dimensions, où chaque (sous-) graphe doit prendre une valeur pour chaque dimension. Chaque dimension est en outre associée à un ordre partiel sur ses valeurs - par exemple, 2020-03-22 $\preceq$ 2020-03 $\preceq$ 2020 - permettant de sélectionner et de combiner des sous-graphiques valides dans des contextes à

différents niveaux de granularité. Schuetz et coll. **(Schuetz et al, 2020)** proposent de même une forme de traitement analytique en ligne contextuel (OLAP), basé sur un cube de données formé par des dimensions où les cellules individuelles contiennent des graphiques de connaissances. Des opérations telles que «slice-and-dice» (sélection des connaissances en fonction de dimensions données), ainsi que «roll-up» (agrégation des connaissances à un niveau supérieur) peuvent alors être prises en charge.

3. CRÉATION ET ENRICHISSEMENT

Dans cette section, nous discutons des principales techniques par lesquelles des graphiques de connaissances peuvent être créés et ensuite enrichis à partir de diverses sources de données héritées qui peuvent aller du texte brut aux formats structurés (et tout ce qui se trouve entre les deux). La méthodologie appropriée à suivre lors de la création d'un graphe de connaissances dépend des acteurs impliqués, du domaine, des applications envisagées, des sources de données disponibles, etc. De manière générale, cependant, la flexibilité des graphes de connaissances permet de démarrer avec un noyau initial qui peut être enrichi progressivement à partir d'autres sources selon les besoins (généralement en suivant une méthodologie Agile **(Hunt and Thomas, 2003)** ou «pay-as-you-go» **(Sequeda et al, 2019)**. Pour notre exemple courant, nous supposons que l'office du tourisme décide de construire un graphe de connaissances à partir de zéro, dans le but de décrire dans un premier temps les principales attractions touristiques - lieux, événements, etc. - au Chili afin d'aider les touristes en visite à identifier ceux qui les intéressent le plus. Le conseil décide de reporter l'ajout de données supplémentaires, comme les itinéraires de transport, les rapports de crime, etc., à une date ultérieure.

3.1. Collaboration humaine

Une approche pour créer et enrichir des graphiques de connaissances consiste à solliciter des contributions directes d'éditeurs humains. Ces éditeurs peuvent être trouvés en interne (par exemple, les employés de l'office du tourisme), en utilisant des plates-formes de crowdsourcing, par des mécanismes de rétroaction (par exemple, des touristes ajoutant des commentaires sur les attractions), via des plates-formes d'édition collaborative (par exemple, un wiki d'attractions ouvert à éditions publiques), etc. Bien que la participation humaine entraîne des coûts

élevés, certains graphiques de connaissances importants ont été principalement basés sur des contributions directes d'éditeurs humains. Toutefois, selon la manière dont les contributions sont sollicitées, l'approche présente un certain nombre d'inconvénients clés, dus principalement à l'erreur humaine, au désaccord, au biais, au vandalisme, etc. défis concernant les licences, l'outillage et la culture. Les humains sont parfois plutôt employés pour vérifier et organiser des ajouts à un graphe de connaissances extrait par d'autres moyens (par exemple, à travers des jeux vidéo avec un but), pour définir des mappages de haute qualité à partir d'autres sources, pour définir un schéma de haut niveau approprié **(Keet, 2018)**, etc.

3.2. Sources de texte

Les corpus de textes - tels que ceux provenant de journaux, de livres, d'articles scientifiques, de médias sociaux, de courriels, de crawls Web, etc. - sont une source abondante d'informations riches. Cependant, extraire ces informations avec une grande précision et un rappel dans le but de créer ou d'enrichir un graphe de connaissances est un défi non trivial. Pour résoudre ce problème, des techniques de traitement du langage naturel (NLP) et d'extraction d'informations (IE) **(Grishman, 2012**) peuvent être appliquées. Bien que les processus varient considérablement selon les cadres d'extraction de texte, dans la figure 19, nous illustrons quatre tâches principales pour l'extraction de texte sur un exemple de phrase. Nous discuterons de ces tâches à tour de rôle.

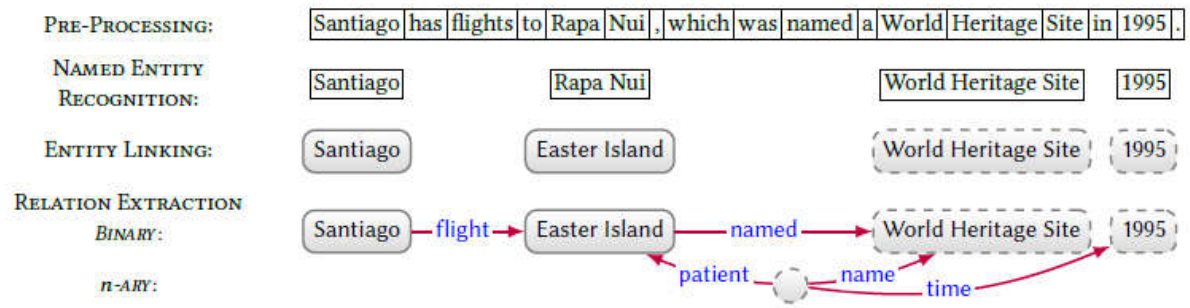


Figure 19: Exemple d'extraction de texte; les nouveaux nœuds du graphe de connaissances sont affichés en tirets

3.3. Sources de balisage

Le Web a été fondé sur des documents de balisage interconnectés dans lesquels des marqueurs (ou balises) sont utilisés pour séparer les éléments du document

(généralement à des fins de formatage). La plupart des documents sur le Web utilisent le langage HTML (HyperText Markup Language). La **Figure 20** présente un exemple de page Web HTML sur les sites du patrimoine mondial au Chili. Les autres formats de balisage incluent Wikitext utilisé par Wikipedia, TeX pour la composition, Markdown utilisé par les systèmes de gestion de contenu, etc. Une approche pour extraire des informations à partir de documents de balisage - afin de créer et / ou d'enrichir un graphe de connaissances - consiste à dépouiller les marqueurs (par exemple, balises HTML), ne laissant que du texte brut sur lequel les techniques de la section précédente peuvent être appliquées. Cependant, le balisage peut être utile à des fins d'extraction, où des variations des tâches susmentionnées pour l'extraction de texte ont été adaptées pour exploiter ce balisage **(Lockard et al, 2018).**

Figure 20: Exemple de document de balisage (HTML) avec code source (à gauche) et document formaté (à droite)

Nous pouvons diviser les techniques d'extraction pour les documents de balisage en trois catégories principales: les approches générales qui fonctionnent indépendamment du balisage utilisé dans un format particulier, souvent basées sur des wrappers qui mappent des éléments du document à la sortie; des approches ciblées qui ciblent des formes spécifiques de balisage dans un document, le plus souvent des tableaux Web (mais parfois aussi des listes, des liens, etc.); et des approches basées sur des formulaires qui extraient les données sous-jacentes à une page Web, selon la notion de Deep Web. Ces approches peuvent souvent bénéficier des régularités partagées par les pages Web d'un site Web donné, que ce soit en raison de conventions informelles sur la façon dont les informations sont publiées sur les pages Web, ou en raison de la réutilisation de modèles pour générer automatiquement du contenu sur les pages Web; par exemple,

intuitivement, alors que la page Web de la **Figure 29** concerne le Chili, nous trouverons probablement des pages pour d'autres pays suivant la même structure sur le même site Web.

3.4. Sources structurées

Une grande partie des données héritées disponibles au sein des organisations et sur le Web est représentée dans des formats structurés, principalement des tableaux - sous forme de bases de données relationnelles, de fichiers CSV, etc. - mais également des formats arborescents tels que JSON, XML, etc. documents de balisage, les sources structurées peuvent souvent être mappées à des graphes de connaissances où la structure est (précisément) transformée selon un mappage plutôt que (de manière imprécise) extraite. Le processus de mappage comprend deux étapes : 1) créer un mappage de la source vers un graphique, et 2) utiliser le mappage afin de matérialiser les données source sous forme de graphique ou de virtualiser la source (création d'une vue graphique sur les données héritées).

4. GRAPHIQUES DE CONNAISSANCES EN PRATIQUE

Dans cette section, nous discutons de certains des graphiques de connaissances les plus importants qui ont émergé ces dernières années. Nous commençons par discuter des graphiques de connaissances ouverts, qui ont été publiés sur le Web conformément aux directives et protocoles. Nous aborderons plus tard les graphiques de connaissances d'entreprise créés par des entreprises pour une large gamme d'applications.

4.1. Graphiques de connaissances ouverts

Par graphes de connaissances ouverts, nous faisons spécifiquement référence aux graphes de connaissances publiés dans le cadre de la philosophie Open Data, à savoir que « ouvert signifie que tout le monde peut librement accéder, utiliser, modifier et partager pour n'importe quel but (sous réserve, au plus, d'exigences qui préservent la provenance et l'ouverture) ». De nombreux graphiques de connaissances ouvertes ont été publiés sous la forme d'ensembles de données ouverts liés **(Heath and Bizer, 2011)**, qui sont des graphiques (RDF) publiés selon les principes des données liées suivant la philosophie des données ouvertes. La plupart des graphes de connaissances ouverts les plus importants - y compris DBpedia **(Lehmann et al, 2015)**, YAGO **(Suchanek et al, 2007)**, Freebase

(Bollacker et al, 2007) et Wikidata **(Vrandečić and Krötzsch, 2014)** - couvrent plusieurs domaines, représentant une grande diversité d'entités et de relations. La plupart des graphes de connaissances ouverts dont nous discutons dans cette section sont modélisés en RDF, publiés selon les principes des données liées, et offrent un accès à leurs données via des vidages (RDF), des recherches de nœuds (données liées), des modèles de graphes (SPARQL) et, dans certains cas, motifs de bord (fragments de motif triple).

4.2. Graphiques de connaissances d'entreprise

Diverses entreprises ont annoncé la création de «graphiques de connaissances d'entreprise» exclusifs avec une variété d'objectifs à l'esprit, notamment: l'amélioration des capacités de recherche **(Chang, 2018), (Hamad et al, 2018)**, **(Krishnan, 2018)**, la formulation de recommandations aux utilisateurs **(Chang, 2018)**, **(Hamad et al, 2018)**, mise en œuvre d'agents conversationnels / personnels **(Pittman et al, 2017),** amélioration de la publicité ciblée **(He et al, 2016),** renforcement de l'analyse commerciale **(He et al, 2016),** connexion des utilisateurs **(He et al, 2016)**, extension du support multilingue **(He et al, 2016)**, facilitation de la recherche et de la découverte, évaluation et atténuation des risques, le suivi des actualités et l'augmentation de l'automatisation des transports, parmi (beaucoup) d'autres. Bien que très diversifiés, ces graphiques de connaissances d'entreprise suivent certaines tendances de haut niveau, comme en témoigne la discussion de Noy et al. **(Noy et al, 2019):** (1) les données sont généralement intégrées dans le graphe de connaissances à partir de diverses sources externes et internes (impliquant souvent du texte); (2) le graphe de connaissances d'entreprise est souvent très volumineux, avec des millions, voire des milliards de nœuds et d'arêtes, posant des défis en termes d'évolutivité; (3) l'affinement du graphe de connaissances initial - ajout de nouveaux liens, consolidation des entités en double, etc. - est important pour améliorer la qualité; (4) les techniques pour maintenir le graphe de connaissances à jour avec le domaine sont souvent cruciales; (5) un mélange de représentations ontologiques et d'apprentissage automatique est souvent combiné ou utilisé dans différentes situations afin de tirer des conclusions du graphe de connaissances d'entreprise; (6) les ontologies utilisées ont tendance à être des taxonomies légères, souvent simples représentant une hiérarchie de classes ou de concepts.

5. CONCLUSION

Nous avons fourni une introduction complète aux graphiques de connaissances, qui ont reçu de plus en plus d'attention ces dernières années. Sous la définition d'un graphe de connaissances comme un graphe de données destiné à accumuler et à transmettre des connaissances du monde réel, dont les nœuds représentent des entités d'intérêt et dont les arêtes représentent les relations entre ces entités, nous avons discuté des modèles par lesquels les données peuvent être structurées sous forme de graphes ; représentations du schéma, de l'identité et du contexte; techniques pour tirer parti des connaissances déductives et inductives; méthodes pour la création, l'enrichissement, l'évaluation de la qualité et le raffinement des graphiques de connaissances; principes et normes de publication des graphiques de connaissances; et enfin, l'adoption de graphes de connaissances dans le monde réel

Chapitre II

La préservation de la vie privée

La préservation de la vie privée

Introduction

Avec les progrès des techniques du Web sémantique, actuellement une énorme quantité de données est disponible sur Internet. Ces données sont collectées et publiées par différentes sources (par exemple, les entreprises, les gouvernements) à de nombreuses fins, telles que des services, des statistiques, des tests, de la recherche, etc. Le Web sémantique permet l'intégration et la combinaison de ces données en fournissant des modèles standards tel Knowledge Graphs. Cependant, vu que davantage de données sont publiées et partagées, des informations sensibles telles que des maladies, des salaires ou des comptes bancaires sont également fournies et, par conséquent, compromettent la vie privée des entités (patients, utilisateurs, entreprises). Ainsi, pour protéger la vie privée des entités principales, il est n´nécessaire d'identifier, dans les données publiées, les informations permettant de découvrir directement ou indirectement la relation entre les entités principales et les informations sensibles. L'anonymisation est une réponse à ce besoin car elle permet la réutilisation des données tout en protégeant la vie privée. Notre contribution s'inscrit dans le cadre évalue les techniques de l'anonymisation sur les graphes de connaissances. Nous consacrons ce chapitre à la présentation de ce vaste domaine. Après un rappel de quelques concepts directement liés à ce mémoire, nous fournissons une vue d'ensemble de l'anonymisation, à savoir les différentes techniques applicables à l'anonymisation.

2. Différents niveaux de protection de la vie privée

On peut définir quatre propriétés principales pour la protection de la vie privée : L'anonymat, pseudonymat, Non-"chaînabilité" et Non-observabilité (**Yves Deswarte, 2004**).

- **Anonymat** : Requiert que d'autres utilisateurs ou sujets soient incapables de déterminer le véritable nom de l'utilisateur associé à un sujet, une opération ou un objet.
- **Pseudonymat** : C'est l'utilisation d'un pseudonyme au lieu du vrai nom.

- **Non-chaînabilité** : C'est l'impossibilité pour d'autres utilisateurs d'établir un lien entre les différentes opérations faites par un même utilisateur.
- **Non-observabilité** : Consiste à ce que des utilisateurs ou des sujets ne puissent pas déterminer si une opération est en cours d'exécution

3. Les principes fondamentaux de protection de la vie privée

Il existe quelques principes universels liés au respect de la vie privée comme la minimisation des données, souveraineté des données, consentement explicite et la transparence.

3.1. Minimisation des données

La première mesure de minimisation consiste que la seule information nécessaire pour compléter une application particulière devrait être collectée ou utilisée et pas plus **(Sébastien G,2012)**, par exemple le commerce électronique, impliquant un client, un marchand, un service de livraison, des banques. Le marchand n'a pas besoin en général de l'identité du client, mais doit être sûr de la validité du moyen de paiement. La société de livraison n'a pas besoin de connaître l'identité de l'acheteur, ni ce qui a été acheté (sauf les caractéristiques physiques), mais doit connaître l'identité et l'adresse du destinataire. La banque du client ne doit pas connaître le marchand ni ce qui est acheté, seulement la référence du compte à créditer, le montant, etc (**Yves Deswarte, 2004**).

3.2 Souveraineté des données

Lorsque des données personnelles se trouvent sur un site distant, c'est-à-dire une machine qui n'est pas sous le contrôle direct de la personne concernée (typiquement, un serveur d'une entreprise ou administration), soit pour un court moment (par exemple l'exécution d'une simple transaction), soit pour plus longtemps (par exemple des dossiers médicaux dans un hôpital), l'accès à ces données devrait être strictement limité à l'usage souhaité par leur propriétaire, c'est-à-dire la personne correspondant à ces données. Cela signifie que le propriétaire des données doit pouvoir imposer une politique de protection de la vie privée sur ses données et que le serveur qui conserve et traite ces données doit mettre en œuvre cette politique par des mécanismes de contrôle des accès à ces données. La politique en question peut définir des permissions et des interdictions précisant qui peut ou ne peut pas réaliser quelle opération sur ces données

personnelles, mais aussi des obligations précisant, par exemple, que les données expirent (et donc doivent être effacées) après un délai donné suivant la terminaison de la transaction, ou que la divulgation de ces données à un tiers doit être notifiée au propriétaire par courriel, etc. Bien sûr, la politique de vie privée imposée par le propriétaire des données doit être compatible avec la politique de sécurité qui protège les biens de l'entreprise et gouverne l'exécution de l'application, et donc les accès effectifs aux données. La compatibilité entre ces deux politiques doit être vérifiée avant la divulgation par l'utilisateur de ses données personnelles **(Yeves D, Sébastien G,2016)**.

3.3 Consentement explicite

Ça signifie qu'avant de collecter les données personnelles d'un individu, il faut lui demander son autorisation et lui expliquer comment elles seront utilsées **(Sébastien G,2012)**.

3.4 Transparence

Ça signifie que le système ne doit pas être considère comme une boite noire dans laquelle l'individu doit avoir une confiance aveugle **(Sébastien Gambs,2012)**.

4. Modèles de protection de la vie privée

Les efforts de recherche consacrés à la protection de la vie privée ont donné naissance à plusieurs modèles et variantes de modèles. À titre d'exemple, **(B. C. M. Fung et al. 2010)** recense non moins de quinze modèles. Dans cet article, nous allons décrire cinq types de modèles d'anonymisation, qui cherchent à cacher ou briser le lien existant entre une personne du monde réel, et ses données sensibles : la pseudonymisation, le *k*-anonymat, la *l*-diversité, la *t*-proximité et la *differential privacy*.

4.1 La pseudonymisation

La pseudonymisation consiste à supprimer les identifiants explicites et leur remplacement éventuel par des pseudonymes c.à.d. à rajouter à chaque enregistrement un nouveau champ, appelé *pseudonyme*. Pour créer ce pseudonyme, on utilise souvent une *fonction de hachage* que l'on va appliquer à l'un des champs identifiants (par exemple le numéro de sécurité sociale), qui est un type de fonction particulier qui rend impossible (ou tout du moins extrêmement

difficile) le fait de déduire la valeur initiale. On voit ainsi que deux entités possédant des informations sur une même personne, identifiée par son numéro de sécurité sociale, pourraient partager ces données de manière anonyme en *hachant* cet identifiant. Il est également possible d'utiliser tout simplement une fonction aléatoire pour générer un identifiant unique pour chaque personne, mais nous verrons plus bas que cela ne résout pas tous les problèmes.
Le gros avantage de la pseudonymisation est qu'il n'y a aucune limite sur le traitement subséquent des données. Tant que l'on traite des champs qui ne sont pas directement identifiants, on pourra exécuter exactement les mêmes calculs qu'avec une base de données non-anonyme. Ainsi, on montre dans la **Figure 21** un exemple de calcul de la moyenne d'âge pour une pathologie donnée. L'utilisation de données pseudonymisées ne nuit pas à ce calcul.

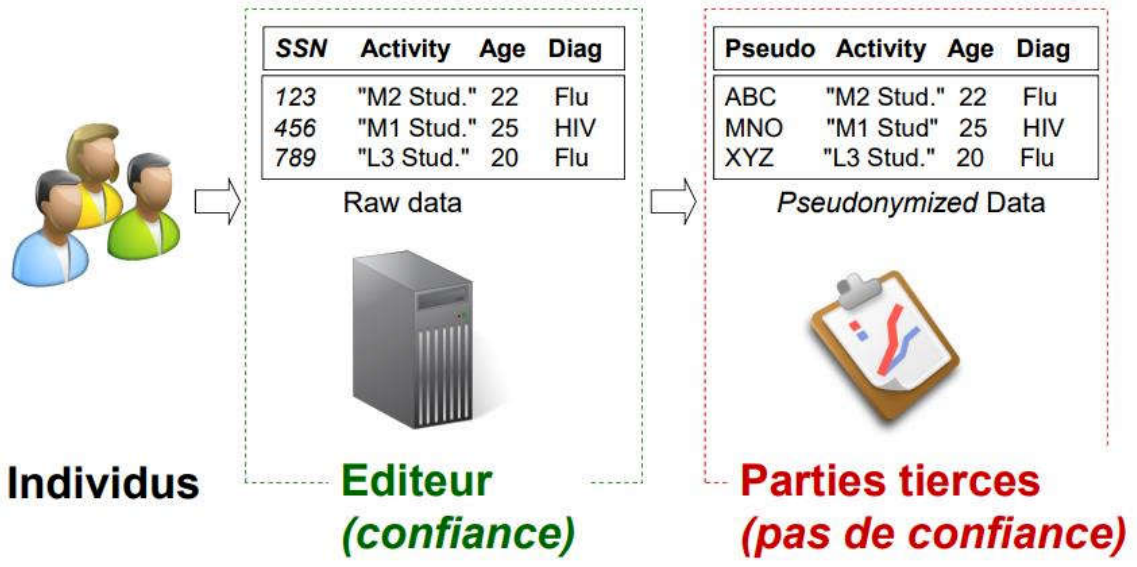


Figure 21: Pseudonymisation et exemple de calcul

Toutefois, la pseudonymisation n'est pas reconnue comme un moyen d'anonymisation, car la pseudonymisation rend vulnérables les enregistrements dans lesquels une partie de la donnée permet de ré-identifier l'individu concerné : la combinaison d'autres champs peut permettre de retrouver l'individu concerné. Sweeney[6] l'a mis en évidence aux Etats-Unis en 2001 en croisant deux bases de données, une base de données médicale pseudonymisée et une liste électorale avec des données nominatives. Le croisement a été effectué non pas sur des champs directement identifiants, mais sur un triplet de valeurs : code postal, date de

[6] https://doi.org/10.1142/S0218488502001648

naissance et sexe, qui est unique pour environ 80% de la population des Etats-Unis ! Elle a ainsi pu relier des données médicales à des individus (en l'occurrence le gouverneur de l'Etat). **(L. Sweeney)**

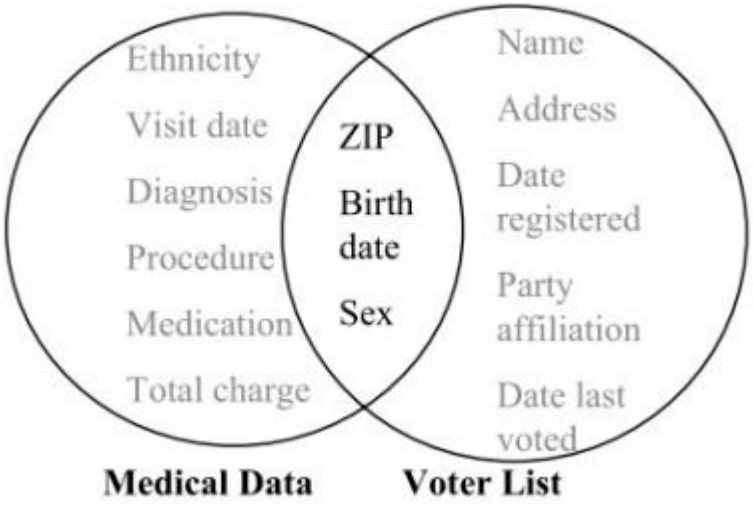


Figure 22: Un exemple de recoupement d'une base anonyme (source Sweeney 2002)

4.2. Le *k*-anonymat

Le modèle de k-anomymat est le premier modèle proposé dans la littérature **(Samarati et Sweeney 1998**). Lorsqu'il est mis en œuvre, ce modèle offre l'assurance que chaque n-uplet de valeurs de quasi-identifiants apparaît au moins k fois dans la table à publier. Ainsi, une table qui répond à ce type de modèle est dite «k-anonyme» , **Sweeney** a proposé la technique de *k*-anonymat. Celle-ci va flouter la possibilité de lier un *n*-uplet anonyme à un *n*-uplet non anonyme de la manière suivante :

1) déterminer les ensembles d'attributs (appelés *quasi*identifiants) qui peuvent être utilisés pour croiser les données anonymes avec des données identifiantes ;

2) réduire le niveau de détail des données de telle sorte qu'il y a au moins *k n*-uplets différents qui ont la même valeur de *quasi*-identifiant, une fois celui-ci généralisé (on dit alors que les individus font partie de la même classe *d'équivalence*). « Généraliser » signifie en fait « enlever

un degré de précision » à certains champs. Ainsi, il est impossible d'être sûr à plus d'une chance sur *k* qu'on a bien lié un individu donné avec son *n*-uplet anonyme. L'avantage du *k*-anonymat est que l'analyse des données continue de fournir des résultats exacts, à ceci près qu'on ne peut pas dissocier les individus d'un groupe.

Toutefois, Certains cas peuvent aussi apparaître : si tous les individus d'une classe d'équivalence possèdent les mêmes valeurs sur un champ intéressant l'attaquant, alors celui-ci sera capable d'identifier cette valeur.

4.3 La *l*-diversité

L'objectif de la l-diversité est de s'assurer que chaque groupe k-anonyme est également assez divers ou assez varié, c'est-à-dire avec suffisamment de valeurs sensibles distinctes à l'intérieur, ceci afin d'empêcher la déduction par homogénéité de la valeur sensible d'une personne. Chaque classe de k individus doit donc être associée à au moins l valeurs sensibles dites bien représentées. Cela veut dire qu'il faut avoir suffisamment de valeurs différentes, ou bien suffisamment de valeurs qui apparaissent souvent du point de vue statistique dans l'ensemble de la population.

4.4. La *t*-proximité

Bien qu'une table puisse être protégée par le principe de la l-diversité, il est possible pour un adversaire d'obtenir des informations au sujet d'un attribut sensible dès lors qu'il dispose d'informations sur la distribution globale de cet attribut. Pour contrer cela, la t-proximité (« t-closeness ») a été proposée par **(N. Li, Li, et Venkatasubramanian 2007)**. Sachant qu'il est impossible d'empêcher un adversaire d'avoir des informations sensibles globales sur une population, le principe de la t-proximité a pour objectif de limiter la capacité de cet adversaire à déduire des informations sensibles sur des individus ciblés. Pour ce faire, il fait en sorte que la distribution de l'attribut sensible au sein de n'importe quelle classe d'équivalence soit proche de la distribution globale de l'attribut. En d'autres termes, il introduit le concept de distance entre ces deux distributions et propose que cette distance ne dépasse pas le seuil t. Ainsi, plus t est petit, plus la possibilité d'inférence de l'adversaire est réduite.

Age	Sexe	Département	Pathologie	Nombre d'individus
<45	M	75	Grippe	400
<45	M	75	Rhume	800
>45	M	75	Grippe	500
>45	M	75	Rhume	1000
<35	F	75	Grippe	300
<35	F	75	Rhume	600
>35	F	75	Grippe	600
>35	F	75	Rhume	1200
...				

Figure 23:t-proximité

La t-proximité souffre de plusieurs problèmes, le plus important étant sans doute son utilité ! En effet, il parait évident d'exploiter des données *k*-anonymes ou même *l*-diverses pour découvrir des corrélations entre des données appartenant au quasi-identifiant et des données sensibles. Toutefois, le but même de la t-proximité est de réduire au maximum ces corrélations, puisque toutes les données sensibles de chaque classe d'équivalence vont se ressembler ! Ainsi, comme on le voit dans la **Figure 23**, la *t*-proximité permet surtout de répondre à la question suivante : *comment partitionner mes données de telle sorte que toutes les partitions se ressemblent en termes de distribution ?* Par exemple, si on imagine une base de données nationale sur des pathologies, comment regrouper les départements, classes d'âge et sexes, de telle sorte qu'on ait la même distribution des pathologies dans chaque sous-groupe. On peut s'interroger du jeu de données qui résulte de cette opération lorsqu'on souhaite précisément réaliser une analyse qui fait ressortir les facteurs qui différencient les individus

4.5 La confidentialité différentielle (Différential Privacy)

Nous concluons ce survol des techniques d'anonymisation par la confidentialité différentielle, une méthode très en vogue dans les milieux de la recherche en informatique depuis quelques années, car contrairement aux méthodes précédentes, elle est la seule à donner des garanties formelles, c'est-à-dire des preuves mathématiques, sur la possibilité de borner les informations qu'on peut apprendre sur les individus. Cette méthode introduit un échantillonnage des données vraies (avec une probabilité α), et une génération de données fictives avec une probabilité $\beta << \alpha$ (mais ces données doivent naturellement rester réalistes...). Les garanties formelles sont cruciales, et permettent de quantifier le risque de ré-identification des n-uplets, d'où l'engouement pour cette méthode. En effet, en observant le jeu de données anonymes, l'information qu'on peut obtenir sur le fait qu'un n-uplet soit vrai ou faux est doublement bornée : on n'est jamais sûr qu' un n-uplet soit vrai avec une probabilité supérieure à α, ni qu'il soit faux avec une probabilité inférieure à β.

4.6 Conclusion

Nous avons présenté dans ce chapitre un état de l'art sur la protection des données personnelles, En commençant par une définition des données à caractère personnelle, ainsi que les risques relatifs à la vie privée, les attaques sur vie privée et enfin nous avons terminé par quelque approche de protection. Dans le chapitre suivant nous présenterons la partie pratique de notre travail.

Chapitre III

IMPLEMENTATION ET EXPÉRIMENTATION

IMPLEMENTATION ET EXPÉRIMENTATION

Introduction

L'opposition entre une donnée qui permet d'identifier une personne et une donnée anonyme n'est pas une opposition absolue. C'est pourquoi il existe plusieurs méthodes d'anonymisation, plus ou moins efficaces. On utilise souvent aujourd'hui la « k-anonymisation », la « l-diversité », ou la « confidentialité différentielle ». Dans ce chapitre, nous présentons notre approche qui se base essentiellement sur l'approche « l-diversité » pour améliorer les résultats **(SHWAN Khaled,2019)** d'anonymisation des graphes de connaissances. Bien que, les différentes techniques sont à juger à la fois sur la sécurité qu'elles procurent, et sur ce qu'elles laissent subsister comme analyses possibles, notre but ici est de balayer les lacunes déjà discuter dans le chapitre précédent (voir section 4) de la technique « k-anonymisation » concernant cette problématique. Dans ce qui suit, nous proposons une évaluation pour valider cette contribution. L'expérimentation est réalisée en utilisant un ensemble de métriques d'évaluation. Ces dernières permettent de juger notre approche en se basant sur le matching entre le graphe résultant après l'application de la technique anonymisation avec celui de l'attaquant.

Le reste du ce chapitre est organisé comme suit : la section 2 illustre l'environnement de la programmation, suivie par la concrétisation avec une architecture simplifiée de notre proposition. La section 5 est une évaluation de notre approche. Tandis que la dernière section conclue le chapitre.

2. Environnement de programmation

Il existe plusieurs outils et langages pour implémenter notre application web pour interroger les graphes de connaissances. Parmi ces outils nous avons utilisé Jena. Et parmi les langages nous avons utilisés : **Pyhton L'environnement de**

développement (IDE) : IDLE Python **Application Programming Interface** : Tkinter **Package Python** : RDFLib/rdflib, NetworkX

4.1.1 JENA

Framwork jena Apache Jena (ou Jena en bref) est un environnement de travail open source en Java, pour la construction d'application web sémantique. JENA permet de manipuler des documents RDF, RDFS, OWL et SPARQL. Il fournit un moteur d'inférences permettant des raisonnements sur les ontologies. JENA est maintenant sous Apache Software Licence. **(BELA 15)**

2.1. Jeux de requêtes :SPARQL

Stocker une quantité immense de données de façon structurée n'aurait aucun intérêt s'il était irréalisable et impossible de pouvoir accéder à ces données. Alors, il est indispensable d'avoir un langage d'interrogation, à l'instar des autres langages tels que SQL. Pour le Web sémantique, c'est SPARQL qui représente le langage standard d'interrogation de graphes RDF.

SPARQL, est l'acronyme de ***S**imple **P**rotocol* And ***R**DF **Q**uery **L**anguage*, est une recommandation du W3C depuis 15 janvier 2008. Il joue le rôle d'un pont entre les technologies du Web sémantique (dont RDF), et les plateformes Web déjà existantes. Il est une API universelle d'accès aux données.

SPARQL est, d'une part, un protocole et un langage de requête permettant l'accès aux données RDF. Il est aussi un protocole d'accès comme un service Web

(SOAP : Simple Object Access Protocol), et également, un langage de présentation des résultats (XML).

Il ne prend pas en charge l'inférence en elle-même. Il ne fait rien de plus que de prendre les descriptions de ce que l'application veut, sous la forme d'une requête, et renvoie le résultat sous forme de graphe RDF. Par ailleurs, SPARQL peut être utiliser afin d'exprimer des requêtes sur différentes sources de données.

La requête SPARQL officielle adopte quatre formes diverses :

- Requête de la forme SELECT, renvoie la valeur de la variable, qui peut être attachée par un modèle de requête équivalent/correspondant ;
- Requête ayant la forme ASK, renvoie vrai si la requête correspond aux données et faux sinon ;
- Requête de la forme CONSTRUCT, retourne les réponses qui satisfont un ensemble de contraintes, sous forme de graphe RDF. La structure du graphe retourné est décrite par un patron (ou template) dans la requête. Elle est comparable à une vue matérialisée dans les SGBDR ;
- Requête de la forme DESCRIBE, renvoie un graphe RDF décrivant une ressource RDF particulière. Parmi les fonctionnalités du langage SPARQL, nous citons :
- FILTERS : contraint les résultats de la requête à uniquement ceux où l'expression du filtre est évaluée à TRUE ;
- OPTIONAL : puisque les données RDF sont des données semi-structurées, lorsqu'une requête est exécutée, elle n'échoue jamais même si les données n'existent pas. Ceci est réalisé grâce à la clause OPTIONAL ;
- LIMIT : met une limite au nombre de résultats de requête retournés ;
- ORDER BY : cette clause est utilisée pour classer (dans l'ordre croissant ou décroissant) les résultats de la requête ; - DISTINCT : est utilisée pour éliminer les doublons présents dans le résultat de la requête ;
- REGEX : cet opérateur appelle la fonction de correspondance pour faire correspondre le texte avec un patron d'expression régulière ;
- UNION : combine des patrons graphiques

2.2. Jeux de données

Dans ce mémoire nous avons choisi de présenter les résultats sur un grand jeu de données : KBpedia[7]. D'autres expérimentations ont été réalisées avec des jeux de données plus petits qui ont montré des résultats similaires. Toutefois nous n'avons

[7] ://fr.wikipedia.org/wiki/DBpedia. [Online ; accessed 15-June-2020].

pas pour ceux-là réalisé une ´évaluation complète comme cela a été fait pour le fragment KBpedia que nous considérons. KBpedia Le jeu de données utilisé pour nos expérimentations est un fragment de KBpedia. Il s'agit d'un jeu de données réel puisque le projet communautaire s'en occupant a pour but de délivrer des documents RDF représentant le contenu encyclopédique disponible sur Internet en annotant sémantiquement les pages

3. Implémentation

Formalisation du problème : Nous formalisons notre problème de la façon suivante :
Etape1 : Avant de procéder vers l'application de l'anonymisation, nous devons construire notre graphe initial RDFlib. Ce qui est réalisé à partir de la source des graphes de connaissance KBpedia.
Etape 2 : Dans cette étape nous contentons de présenter un exemple réduit pour bien illustrer l'amélioration apporter par notre choix de technique d'anonymisation vis-à-vis la technique déjà choisi par nos collègues dans les travaux ultérieurs.

A. Application du K-anonyma

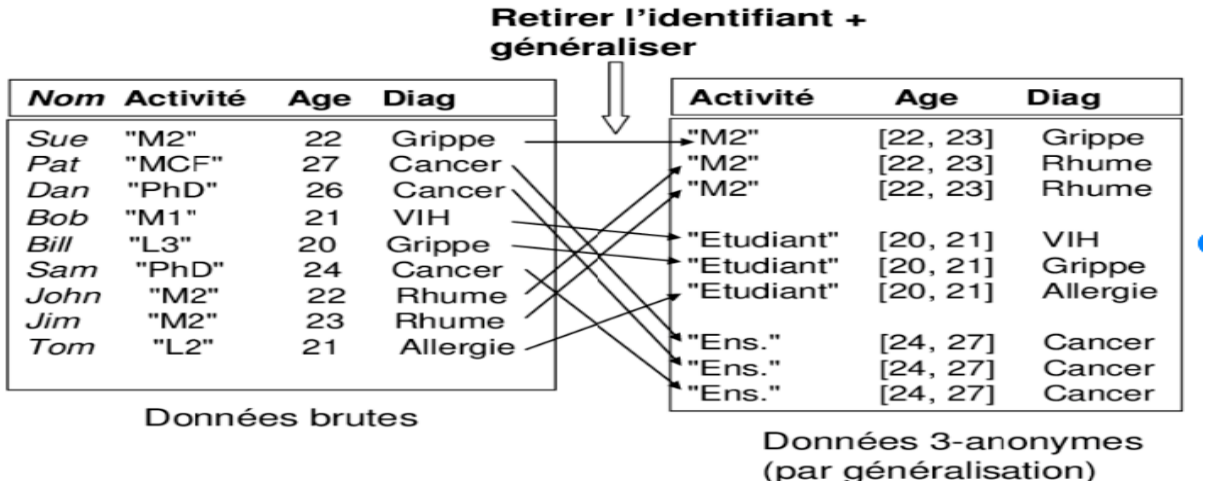


Figure 24: Anonymisation d'une table sur des données universitaires

Dans la **Figure24**, nous montrons un exemple de généralisation des champs activité et âge d'une base de données médicales sur des étudiants et enseignants d'une université. Les étudiants sont identifiés par leur niveau d'étude (L3, M1, etc.), qui se généralise en « étudiant », et les enseignants par leur position académique (doctorant, maître de conférences, etc.), qui se généralise en « enseignant ». Nous traçons dans cette Figure l'origine de chaque n-uplet flouté.
On peut justifier nos critiques par l'exemple suivantes, en considérant les données de la Figure 1, on peut déduire qu'un enseignant ayant un âge entre 24 et 27 ans a forcément le cancer. Si on sait que Sam est un doctorant de 24 ans, alors on peut en déduire qu'il a le cancer.

Enfin, un problème technique important subsiste pour réaliser le k-anonymat : être capable de déterminer les généralisations à effectuer pour produire les quasi-identifiants, ce qui peut être fait soit par un expert humain qui connaît le domaine, ou bien par un calcul informatique, souvent très coûteux pour une base de données réelle.

B. Application du l-diversité

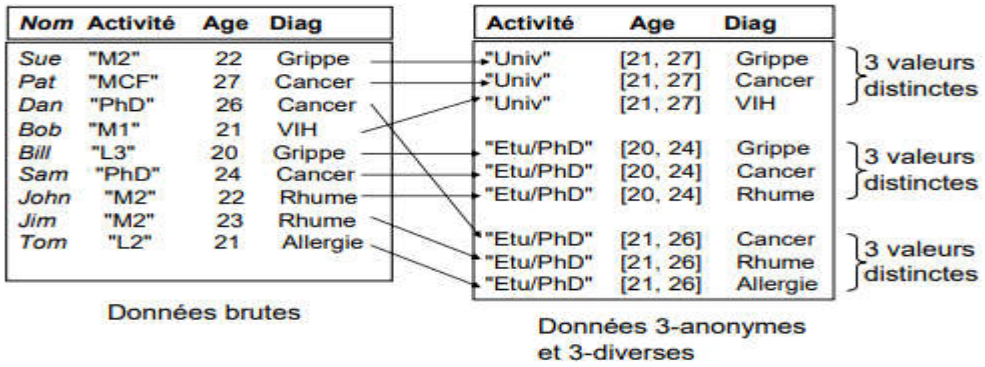


Figure 25: Données l-diverses

Comme on l'a vu à la **Figure 25**, il est possible de déduire des informations dans certains cas pathologiques, sans faire le moindre croisement, par exemple si tous les individus d'une classe possèdent la même valeur sensible. Le modèle de la l-diversité répond à ce problème, en rajoutant une contrainte supplémentaire sur les classes d'équivalence : non seulement au moins *k n*-uplets doivent apparaître dans une classe d'équivalence, mais en plus le champ sensible associé à la classe d'équivalence doit prendre au moins l valeurs distinctes. Dans l'exemple de la **Figure 25**, on voit que pour constituer de telles classes on doit parfois regrouper ensemble des étudiants et des enseignants. Leur activité est alors désignée de

façon encore plus générale (« université »). Notons qu'on peut également lister les valeurs possibles, par exemple avoir une modalité « Étudiant ou Doctorant » (Etu/PhD).

3.1 Présentation des interfaces graphiques

Figure 26 : Interface de l'application

L'interface principale permet de charger une ontologie pour faire une conversion aux formats RDF, créer des nouvelles triplets RDF à partir de source de donnée.

Dans cette interface Il y a deux méthodes d'anonymisation (k-anonymity-généralisation et l-diverses). Après avoir appuyé sur le bouton d'anonymisation il faut faire l'évaluation pour spécifier quels sont les attributs identifiant, quasi-identifiant et les attributs insensibles. Puis Confirmer pour voir les données anonymes.

4. Expérimentation

Avant d'entamer l'évaluation de notre approche, nous présentons la distance de Jaccard. Cette dernière, serve comme une métrique de similarité entre l'ensemble des nœuds du graphe cible et celui du graphe de l'attaquant.

4.1 Similarité de Jaccard

Jaquard L'indice de Jaccard est en effet bien adapté pour déterminer la similarité entre deux ensembles, l'ensemble des valeurs de l'instance x pour la propriété k et l'ensemble des valeurs de l'instance y pour la propriété k. Formellement, si x et y sont deux instances qui ont pour valeur respectivement Pk(x) = {x1, x2, ..., xn} et Pk(y) = {y1, y2, ..., ym}

pour la propriété k, la valeur de similarité est :

$$\begin{aligned} SIM_k(x,y) &= SIM_{jaccard}(P_k(x), P_k(y)) \\ &= \frac{|P_k(x) \cap P_k(y)|}{|P_k(x) \cup P_k(y)|} \\ &= \frac{|\{x_1, x_2, \ldots, x_n\} \cap \{y_1, y_2, \ldots, y_n\}|}{|\{x_1, x_2, \ldots, x_n\} \cup \{y_1, y_2, \ldots, y_n\}|} \end{aligned}$$

On constate aisément que la similarité entre deux instances est symétrique et toujours comprise entre 0 et 1, avec un maximum à 1 lorsque les ensembles comparés sont identiques.

5. Évaluation

Notre expérimentation s'étale sur la démarche suivante : un calcule de matching entre un graphe d'attaquant construit à partir de la source KBpedia et le graphe résultant après l'application de notre proposition pour l'anonymisation. Les résultats obtenus servent à comparer l'exactitude (rapport des correspondances correctes) de similarité entre les graphes.

Numéro	Nombre des nœuds a Matching	Taux de succès de Matching
1	19	0,536
2	97	0,623
3	128	0.602
4	256	0.741
5	512	0.661

Tableau 2:le résultat de l'évaluation

Les résultats montrent que la technique d'anonymisation **l-diverses** est plus efficace par rapport aux résultats obtenus par la méthode k-anonymat, par ailleurs cette performance est dû l'efficacité de la méthode **l-diverses** par rapport au K-anonymat.

5. Conclusion

Dans ce chapitre, nous avons présenté une approche d'anonymisation **l-diverses** appliquée sur un graphe de connaissance. L'objectif principale de cette implémentation est de balayer les lacunes découvertes dans les travaux ultérieurs qui ont étaient proposés dans la même optique. Les résultats ont été plus ou moins satisfaisants vue le taux d'anonymisation réalisé. Par conséquent, l'expérimentation a prouvé l'applicabilité de notre proposition pour remédier les défauts de l'anonymisation par la technique K-anonymat.

IMPLEMENTATION ET EXPÉRIMENTATION

Conclusion et perspectives

La problématique de la protection des données à caractère personnel fait l'objet de nombreux écrits. Chaque seconde, des données sont encodées, manipulées et conservées. Ces données, souvent à caractère personnel, se doivent d'être protégées. Régie par différents règlements, directives et lois, l'utilisation de ces données doit être protégée de différentes façons. Dans ce mémoire nous allons travailler plus particulièrement sur l'anonymisation des données pour la protection de celles-ci. Après un examen du cadre légal, nous allons définir la démarche à suivre pour anonymiser les données de façon fiable.

La précision de notre approche de protection dépend de la mesure de similarité. Étant donné que les ensembles de données graphe de connaissance sont hétérogènes et liés sont fournis en utilisant plusieurs vocabulaires, Comme perspectives nous envisageant d'appliquer les similarités sémantiques, afin de comparer des ressources similaires ayant des propriétés différentes.

BIBLIOGRAPHIE

(Angles et al, 2017): Renzo Angles, Marcelo Arenas, Pablo Barceló, Aidan Hogan, Juan L. Reutter, and Domagoj Vrgoc. 2017. Foundations of Modern Query Languages for Graph Databases. ACM Computing Surveys 50, 5 (2017), 68:1–68:40. https: //doi.org/10.1145/3104031.

(Angles et al, 2018): Renzo Angles, Marcelo Arenas, Pablo Barceló, Peter A. Boncz, George H. L. Fletcher, Claudio Gutierrez, Tobias Lindaaker, Marcus Paradies, Stefan Plantikow, Juan F. Sequeda, Oskar van Rest, and Hannes Voigt. 2018. G-CORE: A Core for Future Graph Query Languages, See [112], 1421–1432.

(Angles et al, 2019): Renzo Angles, Harsh Thakkar, and Dominik Tomaszuk. 2019. RDF and Property Graphs Interoperability: Status and Issues, See [237], 11. http://ceur-ws.org/Vol-2369/paper01.pdf

(BENA ,2017) Mr BENABDALLAH Ali. « Construction semi-automatique des ontologies à partir des documents textuels arabes ».Thèse de doctorat, université abou-bekr belkaid – Tlemcen, 2017.

(Berners-Lee, 2006): Tim Berners-Lee. 2006. Linked Data. W3C Design Issues. https://www.w3.org/DesignIssues/LinkedData.html.

(Brickley and Guha, 2014) : Dan Brickley and R. V. Guha. 2014. RDF Schema 1.1, W3C Recommendation 25 February 2014. W3C Recommendation. World Wide Web Consortium. https://www.w3.org/TR/2014/REC-rdf-schema-20140225/.

(Bollacker et al, 2007): Kurt Bollacker, Robert Cook, and Patrick Tufts. 2007. Freebase: A Shared Database of Structured General Human Knowledge. In Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada. AAAI Press, 1962–1963.

(Bonatti et al, 2020): Piero Andrea Bonatti, Stefan Decker, Axel Polleres, and Valentina Presutti. 2018. Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Seminar 18371). Dagstuhl Reports 8, 9 (2018), 29–111.

BIBLIOGRAPHIE

(Chang, 2018): Spencer Chang. 2018. Scaling Knowledge Access and Retrieval at Airbnb. AirBnB Medium Blog. https://medium.com/airbnb-engineering/scaling-knowledge-access-and-retrieval-at-airbnb-665b6ba21e95.

(Cox et al, 2017): Simon Cox, Chris Little, Jerry R. Hobbs, and Feng Pan. 2017. Time Ontology in OWL, W3C Recommendation 19 October 2017. W3C Recommendation / OGC 16-071r2. World Wide Web Consortium and Open Geospatial Consortium. https://www.w3.org/TR/2017/REC-owl-time-20171019/

(Čebirić et al, 2019): Šejla Čebirić, François Goasdoué, Haridimos Kondylakis, Dimitris Kotzinos, Ioana Manolescu, Georgia Troullinou, and Mussab Zneika. 2019. Summarizing semantic graphs: a survey. The Very Large Data Base Journal 28, 3 (2019), 295–327.

(Cyganiak et al, 2014): Richard Cyganiak, David Wood, and Markus Lanthaler. 2014. RDF 1.1 Concepts and Abstract Syntax, W3C Recommendation 25 February 2014. W3C Recommendation. World Wide Web Consortium. https://www.w3.org/TR/2014/RECrdf11- concepts-20140225/.

(De Melo, 2015) : Gérard de Melo. 2015. Lexvo.org: Informations relatives aux langues pour le cloud de données liées linguistiques. Semantic Web Journal 6, 4 (7 août 2015), p. 393–400.

(Dürst and Suignard, 2005): Martin Dürst and Michel Suignard. 2005. Internationalized Resource Identifiers (IRIs). RFC 3987. Internet Engineering Task Force. http://www.ietf.org/rfc/rfc3987.txt

(Francis et al, 2018) : Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. 2018. Cypher: An Evolving Query Language for Property Graphs, See [112], 1433–1445.

(González and Hogan, 2018): Larry González and Aidan Hogan. 2018. Modelling Dynamics in Semantic Web Knowledge Graphs with Formal Concept Analysis. In Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018, Pierre-Antoine Champin, Fabien L. Gandon, Mounia

(Giménez-García et al, 2017) : José M. Giménez-García, Antoine Zimmermann, and Pierre Maret. 2017. NdFluents: An Ontology for Annotated Statements with Inference Preservation, See [47], 638–654.

BIBLIOGRAPHIE

(Guha et al, 2004): Ramanathan V. Guha, Rob McCool, and Richard Fikes. 2004. Contexts for the Semantic Web. In The Semantic Web - ISWC 2004: Third International Semantic Web Conference, Hiroshima, Japan, November 7-11, 2004. Proceedings (Lecture Notes in Computer Science), Frank van Harmelen, Sheila McIlraith, and Dimitri Plexousakis (Eds.), Vol. 3298. Springer, 32–46.

(Grishman, 2012): Ralph Grishman. 2012. Information Extraction: Capabilities and Challenges. Technical Report. NYU Dept. CS. Notes prepared for the 2012 International Winter School in Language and Speech Technologies.

(Gil et al, 2013): Yolanda Gil, Simon Miles, Khalid Belhajjame, Daniel Garijo, Graham Klyne, Paolo Missier, Stian Soiland-Reyes, and Stephan Zednik. 2013. PROV Model Primer, W3C Working Group Note 30 April 2013. W3C Working Group Note. World Wide Web Consortium. https://www.w3.org/TR/2013/ NOTE-prov-primer-20130430/

(Gutiérrez et al, 2007) : Claudio Gutiérrez, Carlos A. Hurtado, and Alejandro A. Vaisman. 2007. Introducing Time into RDF. IEEE Transactions on Knowledge and Data Engineering 19, 2 (2007), 207–218.

(Hamad et al, 2018): Ferras Hamad, Isaac Liu, and Xian Xing Zhang. 2018. Food Discovery with Uber Eats: Building a Query Understanding Engine. Uber Engineering Blog. https://eng.uber.com/uber-eats-query-understanding/.

(Harris et al, 2013): Steve Harris, Andy Seaborne, and Eric Prud'hommeaux. 2013. SPARQL 1.1 Query Language, W3C Recommendation 21 March 2013. W3C Recommendation. World Wide Web Consortium. https://www.w3.org/TR/2013/REC-sparql11- query-20130321/

(Hartig, 2017): Olaf Hartig. 2017. Foundations of RDF* and SPARQL* – An Alternative Approach to Statement-Level Metadata in RDF. In Proceedings of the 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web, Montevideo, Uruguay, June 7-9, 2017 (CEUR Workshop Proceedings), Juan L. Reutter and Divesh Srivastava (Eds.), Vol. 1912. Sun SITE Central Europe (CEUR), 11. http://ceur-ws.org/Vol-1912/paper12.pdf

(He et al, 2016): Qi He, Bee-Chung Chen, and Deepak Agarwal. 2016. Building The LinkedIn Knowledge Graph. LinkedIn Blog. https://engineering.linkedin.com/blog/2016/10/building-the-linkedin-knowledge-graph.

BIBLIOGRAPHIE

(Heath and Bizer, 2011): Tom Heath and Christian Bizer. 2011. Linked Data: Evolving the Web into a Global Data Space (1st Edition). Synthesis Lectures on the Semantic Web: Theory and Technology, Vol. 1. Morgan & Claypool. 136 pages.

(Hernández et al, 2017): Daniel Hernández, Aidan Hogan, and Markus Krötzsch. 2015. Reifying RDF: What Works Well With Wikidata?. In Proceedings of the 11th International Workshop on Scalable Semantic Web Knowledge Base Systems co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, October 11, 2015 (CEUR Workshop Proceedings), Thorsten Liebig and Achille Fokoue (Eds.), Vol. 1457. Sun SITE Central Europe (CEUR), 32–47. http://ceur-ws.org/Vol-1457/SSWS2015_paper3.pdf

(Hogan et al, 2014) : Aidan Hogan, Marcelo Arenas, Alejandro Mallea et Axel Polleres. 2014. Tout ce que vous avez toujours voulu savoir sur les nœuds vides. Journal of Web Semantics 27-28 (2014), 42–69. https://doi.org/10.1016/j.websem.2014.06.004

(Hogan, 2017) : Aidan Hogan. 2017. Canonical Forms for Isomorphic and Equivalent RDF Graphs: Algorithms for Leaning and Labelling Blank Nodes. ACM Transactions on the Web 11, 4 (2017), 22:1–22:62. https://doi.org/10.1145/3068333

(Homola and Serafini, 2012): Martin Homola and Luciano Serafini. 2012. Contextualized Knowledge Repositories for the Semantic Web. Journal of Web Semantics 12 (2012), 64–87.

(Hunt and Thomas, 2003): Andy Hunt and Dave Thomas. 2003. The Trip-Packing Dilemma. IEEE Software 20, 3 (2003), 106–107.

(Heath and Bizer, 2011): Tom Heath and Christian Bizer. 2011. Linked Data: Evolving the Web into a Global Data Space (1st Edition). Synthesis Lectures on the Semantic Web: Theory and Technology, Vol. 1. Morgan & Claypool. 136 pages.

(Hitzler et al, 2012): Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph. 2012. OWL 2 Web Ontology Language Primer (Second Edition), W3C Recommendation 11 December 2012. W3C Recommendation. World Wide Web Consortium. https://www.w3.org/TR/2012/REC-owl2-primer-20121211/

BIBLIOGRAPHIE

(Knublauch and Kontokostas, 2017): Holger Knublauch and Dimitris Kontokostas. 2017. Shapes Constraint Language (SHACL), W3C Recommendation 20 July 2017. W3C Recommendation.World WideWeb Consortium. https://www.w3.org/TR/2017/REC-shacl-20170720/

(Keet, 2018) : C. Maria Keet. 2018. An Introduction to Ontology Engineering. College Publications.

(Krishnan, 2018) : Arun Krishnan. 2018. Making search easier: How Amazon's Product Graph is helping customers find products more easily. Amazon Blog. https://blog.aboutamazon.com/innovation/making-search-easier.

(Lehmann et al, 2015): Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web Journal 6, 2 (2015), 167–195.

(Lockard et al, 2018): Colin Lockard, Xin Luna Dong, Prashant Shiralkar, and Arash Einolghozati. 2018. CERES: Distantly Supervised Relation Extraction from the Semi-Structured Web. Proceedings of the VLDB Endowment 11, 10 (2018), 1084–1096.

(Li, Ninghui, Tiancheng Li, et Suresh Venkatasubramanian, 2007) : « t-closeness: Privacy beyond k-anonymity and l-diversity ». In Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on, 106–115. IEEE. http://ieeexplore.ieee.org/abstract/document/4221659/.

(L. Sweeney): " k-anonymity: a model for protecting privacy" *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems,* 10(5), 2002.

(Martínez-Rodríguez et al, 2020): Jose L. Martínez-Rodríguez, Aidan Hogan, and Ivan Lopez-Arevalo. 2020. Information Extraction meets the Semantic Web: A Survey. Semantic Web Journal 11, 2 (2020), 255–335.

(Maxime th et al,2020) : Maxime Thouvenot, Olivier Curé, Lynda Temal, Sarra Ben Abbès, Philippe Calvez, « Anonymisation de graphes de connaissances par anatomisation », EGC (Extraction et Gestion des Connaissances), Bruxelles, Belgique, 2020.

BIBLIOGRAPHIE

(Miller, 2013): Justin J. Miller. 2013. Graph Database Applications and Concepts with Neo4j. In Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA March 23rd-24th, 2013. AIS eLibrary, Article 24, 7 pages. https://aisel.aisnet.org/sais2013/24

(McCarthy, 1993): John McCarthy. 1993. Notes sur la formalisation du contexte. Dans les actes de la 13e Conférence internationale conjointe sur l'intelligence artificielle. Chambéry, France, 28 août - 3 septembre 1993, Ruzena Bajcsy (Ed.). Morgan Kaufmann, 555–562.

(Noy et al, 2019): Natasha F. Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. 2019. Industry-scale Knowledge Graphs: Lessons and Challenges. ACM Queue 17, 2 (2019), 20.

(Peterson et al, 2012): David Peterson, Shudi Gao, Ashok Malhotra, C. M. Sperberg-McQueen, Henry S. Thompson, and Paul V. Biron. 2012. W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes, W3C Recommendation 5 April 2012. W3C Recommendation. World Wide Web Consortium. https://www.w3.org/TR/2012/REC-xmlschema11-2-20120405/

(Pham et al, 2015): Minh-Duc Pham, Linnea Passing, Orri Erling, and Peter A. Boncz. 2015. Deriving an Emergent Relational Schema from RDF Data, See [168], 864–874.

(Pittman et al, 2017): R. J. Pittman, Amit Srivastava, Sanjika Hewavitharana, Ajinkya Kale, and Saab Mansour. 2017. Cracking the Code on Conversational Commerce. eBay Blog. https://www.ebayinc.com/stories/news/cracking-the-code-on-conversationalcommerce/.

(Piero et al, 2011): Piero A. Bonatti, Aidan Hogan, Axel Polleres, and Luigi Sauro. 2011. Robust and scalable Linked Data reasoning incorporating provenance and trust annotations. Journal of Web Semantics 9, 2 (2011), 165–201.

Lalmas, and Panagiotis G. Ipeirotis

(Eds.). ACM Press, 1175–1184.

(Rodriguez, 2015) : Marko A. Rodriguez. 2015. The Gremlin graph traversal machine and language. In Proceedings of the 15th Symposium on Database Programming Languages, Pittsburgh, PA, USA, October 25-30, 2015, James Cheney and Thomas Neumann (Eds.). ACM Press, 1–10.

BIBLIOGRAPHIE

(Samarati**, Pierangela, et Latanya Sweeney, 1998)** : « Protecting privacy when disclosing information: kanonymity and its enforcement through generalization and suppression ». Technical report,SRI International. http://epic.org /privacy/reidentification / Samarati_Sweeney_paper.pdf.

(Schuetz et al, 2020): Christoph Schuetz, Loris Bozzato, Bernd Neumayr, Michael Schrefl, and Luciano Serafini. 2020. Knowledge Graph OLAP: A Multidimensional Model and Query Operations for Contextualized Knowledge Graphs. Semantic Web Journal (2020). (Under open review).

(Sequeda et al, 2019): Juan F. Sequeda, Willard J. Briggs, Daniel P. Miranker, and Wayne P. Heideman. 2019. A Pay-as-you-go Methodology to Design and Build Enterprise Knowledge Graphs from Relational Databases, See [179], 526–545.

(Straccia, 2009) : Umberto Straccia. 2009. A Minimal Deductive System for General Fuzzy RDF. In Web Reasoning and Rule Systems, Third International Conference, RR 2009, Chantilly, VA, USA, October 25-26, 2009, Proceedings (Lecture Notes in Computer Science), Axel Polleres and Terrance Swift (Eds.), Vol. 5837. Springer, 166–181.

(SHWAN, 2019) : SHWAN Khaled, (2019). La Préservation de la Vie Privée dans le Web de Données. Mémoire de Master de l'université Dr. TAHAR MOULAY. SAIDA,

(Suchanek et al, 2007): Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: A core of semantic knowledge unifying WordNet and Wikipedia. In Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007, Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy (Eds.). ACM Press, 697–706.

(Vrandečić and Krötzsch, 2014): Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. Communications of the ACM 57, 10 (2014), 78–85.

(**Yves Deswarte, 2004**) : Yves Deswarte. Intelligence ambiante et protection de la vie privée,cours, 2004.

(Sébastien G,2012) : Sébastien Gambs. Introduction à la protection de la vie privée, cours, 2012.

BIBLIOGRAPHIE

(Yeves D, Sébastien G,2016) : Yves Deswarte , Sébastien Gambs. Protection de la vie privée : Principes et technologies, CNRS ; France Université de Toulouse,2016.