Automatic Generation of Complex Ontology Alignments

Elodie Thiéblin

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Elodie THIÉBLIN

Le 21 octobre 2019

Génération automatique d'alignements complexes d'ontologies

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Informatique et Télécommunications**

Unité de recherche :
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Ollivier HAEMMERLé et Cassia TROJAHN DOS SANTOS

Jury

Mme Nathalie Pernelle, Rapporteure
M. Jérôme Euzenat, Rapporteur
M. Pascal Hitzler, Examinateur
M. Ollivier Haemmerlé, Directeur de thèse
Mme Cassia Trojahn dos Santos, Co-directrice de thèse

Acknowledgements

For accepting to review my thesis and for their constructive remarks, I thank Jérôme Euzenat and Nathalie Pernelle. I also thank Pascal Hitzler who accepted to be on my jury.

For communicating their love of research, sharing their insights on my Ph.D. subject, I thank Cassia Trojahn and Ollivier Haemmerlé.

During three years as a Ph.D. student, I have had the chance to meet and be part of the Semantic Web community through conferences, workshops and countless e-mail exchanges. I thank every member of this community for sharing their knowledge, as well as their love of Japanese karaokes, fish dishes, jam sessions, otter observation, *etc.*

For the organisation of the OAEI complex track, and discussions on what exactly is a "Contribution", I would like to thank Michelle Cheatham, Ondrej Zamazal and Lu Zhou.

For her help with the Taxon dataset, I thank Catherine Roussey.

Merci aux contribuables de l'État Français et au service public de m'avoir permis de réaliser cette thèse.

Merci à l'UT2J d'être une fac vivante, et à son département de Maths-Info de m'avoir permis d'enseigner pendant 3 ans.

Un grand merci à Anne Hernandez d'avoir eu la patience de traquer et d'éliminer les fautes d'anglais dans mes articles.

À mes collègues et ami·e·s, Nicolas, Nathalie, Fabien, Caroline, sur qui j'ai pu compter en toute circonstance, pour une relecture de dernière minute, un repas au Onador ou des séminaires plus ou moins improvisés autour d'une bière.

À mes ami·e·s d'enfance, de l'INSA, de Lindex, du théâtre, de la musique, à Ping,

À ma famille, de me soutenir depuis toujours,

À Pierre, de croire en moi, toujours,

Merci.

Résumé en français

Introduction

Le web de données liées (Linked Open Data, LOD)[1] est composé de nombreux entrepôts de données. Ces dernières sont décrites par différents vocabulaires (ou ontologies) ayant chacune une terminologie et une modélisation propre, ce qui peut être source d'hétérogénéité.

Pour lier et rendre les données du web de données liées interopérables, les alignements d'ontologies établissent des correspondances entre les entités desdites ontologies. Par exemple $\langle$ o_1*:Personne* , o_2*:Person* , $\equiv$ $\rangle$ est une correspondance. Il existe de nombreux systèmes d'alignement qui génèrent des correspondances simples, *i.e.*, ils lient une entité à une autre entité. Toutefois, pour surmonter l'hétérogénéité des ontologies, des correspondances plus expressives sont parfois nécessaires. Par exemple, la correspondance $\langle$ o_1*:PapierAccepté* , $\exists o_2$*:aPourDecision.*o_2*:Acceptation* , $\equiv$ $\rangle$ lie le concept de papier accepté dans l'ontologie source o_1 au concept abstrait qui représente les papiers ayant une acceptation pour décision dans l'ontologie cible o_2. Trouver de telles correspondances est un travail fastidieux qu'il convient d'automatiser.

Dans le cadre de cette thèse, une approche d'alignement complexe (*i.e.*, approche qui génère automatiquement des alignements complexes d'ontologies) est proposée. Deux hypothèses permettent de simplifier le problème. La première vise à réduire l'espace de recherche des correspondances dans les ontologies en se concentrant sur les besoins en connaissance d'un utilisateur. La seconde est que les ontologies alignées comportent au moins une instance commune pour chaque besoin en connaissance de l'utilisateur. L'approche compare le besoin utilisateur exprimé dans l'ontologie source avec le voisinage des instances communes dans la base de connaissances cible et en extrait un sous-graphe qui deviendra le membre cible de l'ontologie. Le déroulement de cette approche est détaillé dans la sous-section *Une approche d'alignement complexe orientée besoin en connaissance* de ce résumé. Comparée aux approches de la littérature, celle proposée ici ne se limite pas à la détection de correspondances suivant un patron précis ni ne nécessite une quantité importante d'instances communes. D'autre part, les approches de la littérature tentent d'aligner la totalité des ontologies même quand l'utilisateur ne s'intéresse qu'à une partie de leur champ.

Le domaine des alignements complexes est relativement récent et peu de travaux abordent la problématique de leur évaluation. Pour pallier ce manque, nous proposons un système d'évaluation automatique fondé sur de la comparaison d'instances. Ce système est complété par un jeu de données artificiel sur le domaine des conférences pour former un banc d'évaluation. L'approche proposée a été automatiquement évaluée sur ce jeu de données. Pour se confronter à un cas

[1] `https://www.lod-cloud.net/`

tiré directement du web de données liées, l'approche a été manuellement évaluée sur des bases de connaissances publiques dont le champ commun est la taxonomie des plantes.

Dans ce résumé, nous présentons l'essentiel des travaux qui sont détaillés dans ce tapuscrit. Nous présentons d'abord les ontologies, leur hétérogénéité et leurs alignements, puis nous récapitulons l'état de l'art sur la génération d'alignements complexes. Nous détaillons l'approche d'alignement proposée puis nous présentons le banc d'évaluation. Enfin nous décrivons brièvement les résultats des expérimentations et finissons par discuter le travail effectué et les perspectives.

Ontologies hétérogènes et alignements

Les ontologies permettent de conceptualiser et représenter la connaissance d'un domaine. Elles sont constituées de classes, relations, valeurs et représentent des instances. Par exemple *Camille* est une instance de type *Personne*, et *Personne* est une classe d'une ontologie.

Le World Wide Web consortium (W3C) a défini des standards pour représenter la connaissance et les ontologies. RDF est standard pour décrire les instances. Chaque instance est décrite par un ensemble de triplets. Tous les triplets mis bout à bout donnent un graphe, qu'on peut appeler un graphe de connaissance. OWL et RDFS sont des standards qui permettent d'exprimer les ontologies décrivant les données en RDF. Les graphes RDF peuvent être interrogés à l'aide du langage de requête SPARQL.

Dans le web de données liées, les ontologies sont souvent construites indépendamment et bien qu'elles puissent modéliser des connaissances similaires, la manière dont elles le font peut varier.

En effet, la création d'ontologies est souvent guidée par des **questions de compétence** [Grüninger & Fox 1995, Suárez-Figueroa *et al.* 2012, Ren *et al.* 2014, Dennis *et al.* 2017]. Ces questions formulées en langage naturel vont donner forme à l'ontologie.

Considérons les questions de compétence *Quels sont les papiers acceptés ?* et *Quels papiers ont pour décision une acceptation ?*. Ces deux questions de compétence représentent la même connaissance : le fait qu'un papier soit accepté ou non ; mais la manière dont ces questions sont formulées diffère. Ainsi, la première donnerait lieu à une classe o_1*:PapierAccepté* tandis que la seconde suppose l'existence d'une relation o_2*:aPourDécision* liant un o_2*:Papier* à une o_2*:Décision*, cette décision pouvant être une o_2*:Acceptation*. Ces deux fragments d'ontologies avec des instances sont représentés dans la Figure 1.

Nous venons de voir que les ontologies peuvent être hétérogènes et ce dès leur conception. Pourtant, un des buts du web de données liées est justement de surmonter ces hétérogénéités et de rendre les instances et outils utilisant ces ontologies interopérables.

Les alignements d'ontologies ont été introduits pour surmonter les hétérogénéités

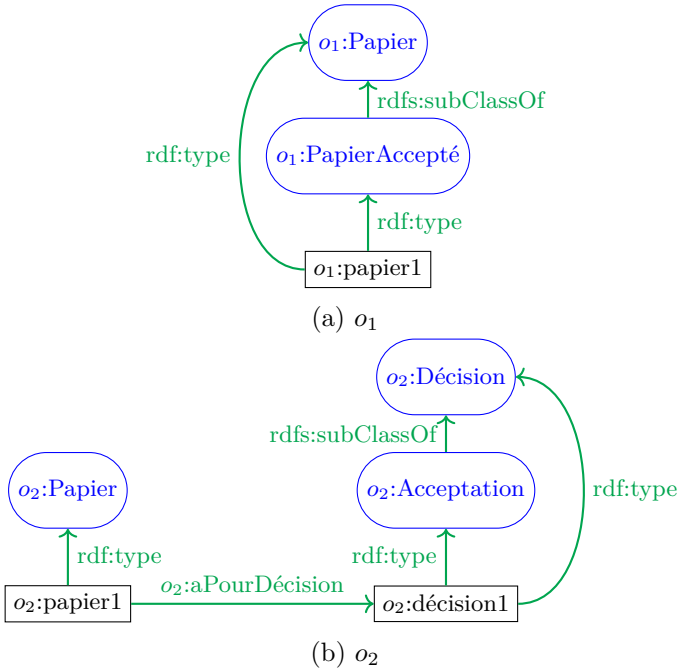


Figure 1: Fragments d'ontologies hétérogènes. La légende du schéma est présentée dans l'annexe A.3.

entre ontologies. Un alignement est un ensemble de correspondances qui lient des fragments d'ontologies entre eux. Chaque correspondance est un triplet qui lie une expression e_{o_1} d'une ontologie source o_1 à une expression e_{o_2} d'une ontologie cible o_2 par une relation (*e.g.*, $\equiv$) : $\langle\ e_{o_1}\ ,\ e_{o_2}\ ,\ \equiv\ \rangle$. On parle d'ontologie source et d'ontologie cible car un alignement peut être orienté.

On distingue deux sortes de correspondances : les correspondances simples et les correspondances complexes.

Les **correspondances simples** lient deux entités représentées par leurs identifiants. Par exemple $\langle\ o_1{:}Papier\ ,\ o_2{:}Papier\ ,\ \equiv\ \rangle$ est une correspondance simple.

Les **correspondances complexes** complètent les correspondances simples en proposant plus d'expressivité. Les membres des correspondances complexes peuvent en effet être des constructions logiques comme $\exists\ o_2{:}aPourDécision.o_2{:}Acceptation$ ou utiliser des fonctions de transformation de valeur comme $convertirEnEuros(o_2{:}prixEnDollars)$. Ainsi, $\langle\ o_1{:}Papier\ ,\ \exists\ o_2{:}aPourDécision.o_2{:}Acceptation\ ,\ \equiv\ \rangle$ et $\langle\ o_1{:}prixEnEuros\ ,\ convertirEnEuros(o_2{:}prixEnDollars)\ ,\ \equiv\ \rangle$ sont des correspondances complexes.

Aligner manuellement deux ontologies est une tâche fastidieuse. Pour cette raison, des approches automatique d'alignement ont été proposées dans la littérature.

Etat de l'art

Les approches d'alignement d'ontologies foisonnent, comme en témoigne l'OAEI (Ontology Alignment Evaluation Initiative)[2], initiative d'évaluation d'alignements entre ontologies. Chaque année, de nouveaux systèmes d'alignement y participent sur les jeux de données qu'elle fournit.

Les approches d'alignement complexes d'ontologies sont, elles, moins nombreuses. La première tâche d'alignement complexe de l'OAEI a été proposée en 2018 [Thiéblin *et al.* 2018a]. Toutefois, des approches d'alignements entre autres types de modèles (bases de données relationnelles, modèles conceptuels, *etc.*) ont été proposées dans la littérature. Chaque approche étudiée est détaillée dans le Chapitre 3. Nous présentons ici les différentes familles d'approches d'alignement complexe.

Les catégories d'approches d'alignements proposées par [Euzenat & Shvaiko 2013] sont applicables aux aligneurs (*i.e.*, approches d'alignement) simples et complexes. Parmi les approches de l'état de l'art, certaines utilisent des ressources formelles comme des ontologies ou des alignements externes, d'autres utilisent des ressources informelles comme des formulaires web. Des aligneurs traitent les labels de manière syntaxique (en chaîne de caractères) tandis que d'autres appliquent des techniques et propriétés linguistiques comme la lemmatisation, la synonymie, *etc.* Les instances décrites par les ontologies sont utilisées par certains aligneurs.

La spécificité des aligneurs complexes est la forme des correspondances qu'ils produisent. Ceci a un impact sur le cœur de leur fonctionnement. En effet, des aligneurs choisissent de fixer la forme des deux membres de leurs correspondances : ils utilisent un patron de correspondance. D'autres fixent un des membres, par exemple le membre source et cherchent un membre cible qui lui correspond. Enfin, certaines approches cherchent des correspondances sans fixer aucun membre *a priori*.

Nous proposons dans le Chapitre 3 une classification des approches d'alignement en fonction des types de structures qui guident la détection de leurs correspondances. Comme précédemment décrit, certains aligneurs utilisent des patrons de correspondance. D'autres utilisent des répétitions de patrons dans les membres de leurs correspondances. D'autres encore utilisent de la recherche de chemins ou d'arbres couvrants. Enfin, certaines approches n'utilisent pas de structure définie.

Bien que plusieurs approches d'alignement complexe soient présentées dans l'état de l'art, l'évaluation de ces aligneurs a été peu étudiée. Dans le Chapitre 3 sont listés les bancs de tests et les stratégies d'évaluation pour les aligneurs simples et complexes. Si la plupart des évaluations d'alignement simple sont automatisées, c'est loin d'être le cas pour l'alignement complexe.

[2] `http://oaei.ontologymatching.org/`

Une approche d'alignement complexe orientée besoin en connaissance

Le problème d'alignement complexe est par nature non-trivial. En effet, il est difficile de dénombrer l'ensemble de toutes les possibilités de correspondances complexes entre deux ontologies, tandis que l'ensemble des correspondances simples possibles et le produit scalaire de leurs entités respectives.

Pour simplifier le problème, nous nous basons sur deux hypothèses. La première consiste à réduire l'espace des correspondances possibles en se focalisant sur les besoins en connaissance d'un utilisateur. Cette hypothèse va à l'encontre des approches existantes qui tentent d'aligner la totalité des ontologies même si l'utilisateur n'est intéressé que par une sous-partie de leur champ. La seconde est que les ontologies à aligner décrivent quelques instances communes. Cette hypothèse est moins restrictive que pour certaines approches qui nécessitent un grand nombre d'instances communes.

Pour représenter le besoin en connaissance, nous introduisons les **questions de compétence pour alignement**. Cette notion est directement inspirée des questions de compétence utilisées pour la conception d'ontologies. Leur différence est que les questions de compétence pour alignement (CQA) sont vouées à être couvertes par deux ontologies ou plus, et qu'elles ne définissent ni leur champ, ni leur structure. D'autre part, le champ d'une CQA est limité par l'intersection des champs des ontologies qu'elle couvre. Une CQA peut être traduite en requêtes SPARQL, et à la différence des questions de compétence, par une requête SPARQL pour chaque ontologie couverte. Ainsi, la question de compétence *Quels sont les papiers acceptés ?* devient en SPARQL:

o_1 `SELECT ?x WHERE { ?x a o1:PapierAccepté. }`

o_2 `SELECT ?x WHERE { ?x o2:aPourDécision ?y. ?y a o2:Acceptation. }`

Nous distinguons trois types de CQA en fonction de l'arité de leurs réponses attendues. Les CQA **unaires** attendent un ensemble d'instances, *e.g.*, *Quels sont les papiers acceptés ?* est une question unaire. Les CQA **binaires** attendent un ensemble de paires d'instances ou de valeurs, *e.g.*, *Quelle est la décision associée à chacun des papiers ?* est une CQA binaire. Les CQA **n-aires** attendent un ensemble de tuples d'instances ou de valeurs, *e.g.*, *Quelle est la décision associée à chaque papier et à quelle date fut-elle donnée ?* est une CQA n-aire. Comme les ontologies sont majoritairement composées de classes (prédicats unaires), interprétées comme des ensembles d'instances, et de relations (prédicats binaires), interprétées comme des ensembles de paires d'instances, nous nous concentrons sur les CQA unaires et binaires.

Dans le cadre de l'approche, nous complétons notre première hypothèse en considérant que l'utilisateur est capable d'exprimer son besoin sous forme de requêtes SPARQL dans l'ontologie source. Nous présentons le déroulé de l'approche sur un exemple dans la Figure 2. Dans la suite, les étapes de l'approche sont référencées

par leur numéro (*e.g.*, ①). Les étapes de l'approche sont détaillées dans le Chapitre 4.

À partir de la CQA sous forme de requête SPARQL, l'approche extrait une formule logique ①. Prenons l'exemple de la requête `SELECT ?x WHERE { ?x a o1:PapierAccepté. }`. La formule logique correspondante est o_1*:PapierAccepté*. La couche lexicale associée aux éléments de cette formule logique est extraite ②. Par exemple, o_1*:PapierAccepté* a pour label *"papier accepté"*. Les instances décrites par cette requête sont également récupérées ; nous les appelons les réponses à la CQA ③. Par exemple, o_1*:papier1* est une réponse à notre CQA.

L'approche tente ensuite de trouver des instances communes aux réponses et aux instances de la base de connaissances cible ④. Par exemple, l'instance de la base cible o_2*:papier1* est équivalente à o_1*:papier1* qui elle-même fait partie des réponses sources. Les instances cibles ainsi trouvées sont décrites, c'est-à-dire que pour chacune de ces instances, le sous-graphe RDF la décrivant est récupéré ⑤. Les labels des entités du sous-graphe ⑥ sont ensuite comparés à la couche lexicale de la CQA ⑦, puis le sous-graphe est élagué pour garder seulement les parties similaires à la CQA ⑧. Par exemple, le sous-graphe composé des trois triplets : $\langle$ o_2*:paper1 , *o_2*:aPourDécision , *o_2*:décision1* $\rangle$, $\langle$ o_2*:décision1 , rdf:type ,* o_2*:Décision* $\rangle$ et $\langle$ o_2*:décision1 , rdf:type ,* o_2*:Acceptation* $\rangle$ est élagué car o_2*:Acceptation* est plus proche syntaxiquement de *"papier accepté"* que o_2*:décision1* ou o_2*:Décision*. Suite à cela, nous transformons le sous-graphe élagué en une formule logique. Dans l'exemple traité ici, la formule logique obtenue est $\exists o_2$*:aPourDécision.*o_2*:Acceptation*. L'approche recherche des contre-exemples, c'est-à-dire des instances communes aux deux ontologies qui sont décrites par la formule logique obtenue dans la base de connaissances cible mais qui ne sont pas décrites par la CQA dans la base de connaissances source ⑨. Cette formule est ensuite filtrée par un seuil sur une mesure de similarité ⑩ et mise en correspondance avec celle obtenue de la CQA ⑪ : $\langle$ o_1*:PapierAccepté ,* $\exists o_2$*:aPourDécision.*o_2*:Acceptation ,* $\equiv \rangle$

La manière dont le sous-graphe décrivant les réponses est obtenu varie lorsque l'approche traite les CQA binaires. En effet, les réponses sont des paires d'instances. Par exemple, *Qui a écrit un papier donné ?* a pour réponse *(*o_1*:papier1,* o_1*:personne1)*. Une instance équivalente de la base cible est cherchée pour chaque instance de la réponse source. Si de telles instances sont trouvées, l'approche va chercher un chemin entre elles dans le graphe RDF cible.

Une des limitations de l'approche proposée est qu'elle nécessite au moins une instance (une réponse) commune entre les deux ontologies pour chaque CQA. En revanche, en comparaison avec d'autres approches comme [Walshe *et al.* 2016, Parundekar *et al.* 2010, Parundekar *et al.* 2012, Hu *et al.* 2011] qui nécessitent un nombre important d'instances communes, notre approche est moins contraignante.

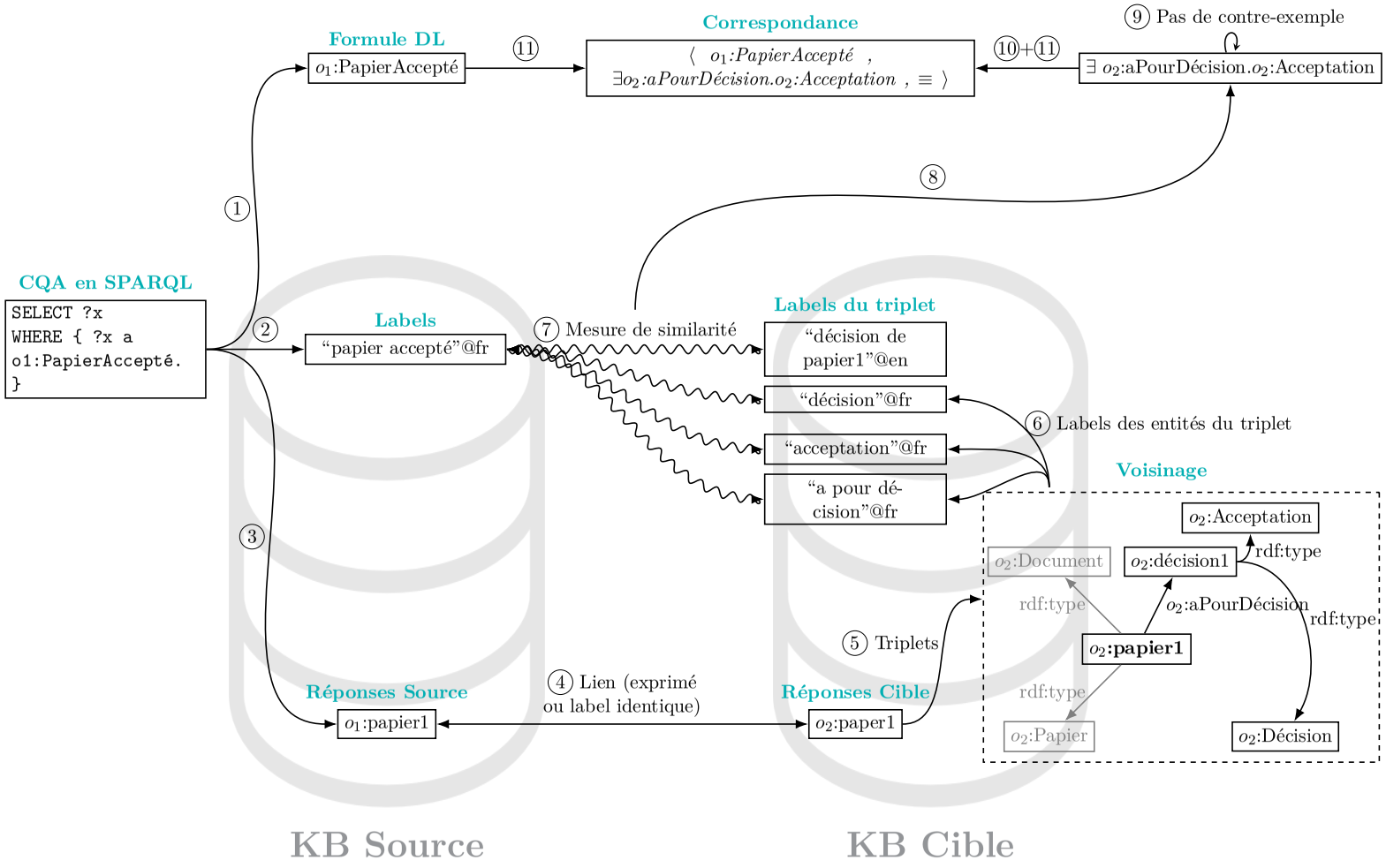


Figure 2: Schéma de l'approche pour l'exemple

Banc d'évaluation d'alignements complexes automatique

Le champ de recherche de l'évaluation des alignements complexes est assez récent. Bien que des jeux d'évaluation aient été proposés récemment [Zhou *et al.* 2018, Thiéblin *et al.* 2018b], aucun système ne permet à ce jour d'évaluer automatiquement des alignements sur ceux-ci.

Dans le Chapitre 5, nous analysons ce qui rend l'évaluation des alignements complexes difficile. Cette analyse est faite en disséquant le déroulement générique d'une évaluation d'alignement. Nous concluons l'analyse en ayant identifié le fait que la comparaison des correspondances complexes est difficilement automatisable. Ce problème a lieu pour trouver quelles correspondances comparer entre elles et ensuite pour trouver à quel point deux correspondances se ressemblent.

Nous proposons un système d'évaluation fondé sur des questions de compétence pour alignement (CQA). Les CQA servent de référence pour le système d'alignement : une CQA source a une CQA cible équivalente. L'utilisation des CQA permet de mettre une limite à l'alignement de référence. En effet, il est très difficile de prouver qu'un alignement complexe de référence couvre toutes les correspondances complexes possibles. Dans l'évaluation, la CQA source est automatiquement traduite grâce à l'alignement évalué. Le système de réécriture de requêtes présenté dans [Thiéblin *et al.* 2016] a été étendu pour ce faire. Les résultats de la requête ainsi réécrite sur la base cible sont comparés à ceux de la CQA cible de référence. En agrégeant les résultats, nous obtenons un score de **couverture de CQA**. Celui-ci est un indicateur de la capacité de l'alignement à couvrir les besoins d'un utilisateur (exprimé en CQA). Ce score est assimilable à la métrique "rappel" utilisée dans la recherche d'information. En effet, il permet d'estimer le fait que l'alignement évalué ait un champ suffisamment large pour couvrir une référence.

Pour équilibrer ce score, une **précision intrinsèque** est calculée. Les instances décrites par les membres source et cible de chaque correspondance sont comparées. Ce score de précision nécessite que les ontologies soient peuplées similairement. Contrairement à la métrique classique de "précision" de la recherche d'information, la précision intrinsèque n'utilise pas de référence.

La moyenne harmonique de la précision intrinsèque et de la couverture de CQA est calculée afin d'agréger les deux scores.

Un système d'évaluation nécessite un jeu de données sur lequel s'exécuter. Pour compléter cette proposition, nous avons choisi de réutiliser le jeu d'évaluation Conférence [Šváb Zamazal *et al.* 2005, Šváb Zamazal & Svátek 2017] largement utilisé dans l'OAEI et décliné en plusieurs variantes [Cheatham & Hitzler 2014, Thiéblin *et al.* 2018b]. Ce jeu de données ne contient pas d'instance, c'est pourquoi nous l'avons peuplé artificiellement. L'autre raison de proposer un peuplement artificiel de ces ontologies est qu'il est ainsi possible de le contrôler.

Nous utilisons des CQA pour peupler le jeu de données Conférence. De cette manière, nous nous assurons de l'interprétation des ontologies et de la cohérence de leurs instances entre elles. En tout, 152 CQA ont été créées par un expert en considérant le scénario de l'organisation d'une conférence. Les instances du

peuplement et les CQA sont inspirées du site Web de la conférence ESWC 2018[3]. Toutes les CQA ne sont pas couvertes par toutes les ontologies. Un sous-ensemble de 100 CQA a été séléctionné comme référence pour le système d'évaluation. Les statistiques de ce jeu de données ont été pseudo-aléatoirement reproduites dans les jeux de données artificiels. Six jeux de données émanent de ce peuplement ayant entre 0% et 100% d'instances communes.

Expérimentations

Nous utilisons le banc de test précédemment présenté pour évaluer notre approche d'alignement. Différentes variantes de l'approche sont évaluées et comparées. Dans chaque variante, un paramètre de l'approche change : le seuil de la fonction de similarité lexicale, le nombre d'instances communes prises en compte dans les réponses, le type de lien entre instances communes (liens explicitement déclarés ou absence de lien explicites), *etc.* Une variante ne prend pas de CQA en entrée mais génère des requêtes en suivant des patrons. Les résultats sont présentés en détail dans le Chapitre 6. La variante qui génère des requêtes au lieu d'utiliser des CQA fournies par l'utilisateur obtient un moins bon score de couverture de CQA. Bien que l'utilisation des mêmes CQA dans la génération de l'alignement que pour son évaluation soit discutable, les besoins en connaissance d'un utilisateur sont mieux couverts lorsque l'approche les prend en compte. La précision intrinsèque de l'approche en précision-orientée et en rappel-orientée (deux métriques définies dans [Ehrig & Euzenat 2005]) est entre 60% et 80%. La couverture de CQA en précision-orientée et en rappel-orientée est d'environ 80%. Les moyennes harmoniques dans ces deux métriques sont entre 70% et 80%.

Les résultats de l'approche sont également comparés à des alignements de référence simples [Šváb Zamazal *et al.* 2005] et complexes [Thiéblin *et al.* 2018b] sur ce jeu de données et au résultat d'approches d'alignement [Ritze *et al.* 2010, Faria *et al.* 2018]. Notre approche obtient en moyenne une précision intrinsèque inférieure à celle des alignements de référence mais une meilleure couverture de CQA. Ceci vient en partie du fait que ces alignements de référence ont une expressivité limitée. L'approche de [Ritze *et al.* 2010] obtient un bon score de précison (75%) mais permet de couvrir peu de CQA (entre 40% et 44%). Celle de [Faria *et al.* 2018] obtient une précision entre 40% et 62% pour une couverture de CQA d'environ 45%.

L'approche a ensuite été évaluée sur un ensemble de bases de connaissances issues du web de données liées dont le champ commun est la taxonomie des plantes. Bien qu'ayant un champ commun, ces ontologies n'ont pas exactement les mêmes instances. De plus, certaines de ces bases de connaissances ne sont pas peuplées de manière cohérente, c'est-à-dire que la même connaissance peut être représentée différemment sur deux instances différentes au sein de la même base de connaissances. Par exemple, dans DBpedia, le fait qu'un taxon soit de rang taxonomique *genre* est représenté par une propriété *dbp:genus* pour certaines instances,

[3] `https://2018.eswc-conferences.org/`

par un lien vers l'instance *dbr:Genus* dans d'autres, par l'existence de la propriété *dbp:genusAuthority*, *etc.* Il était donc difficile de créer des requêtes exhaustives afin d'automatiser leur comparaison.

Le détail des résultats est donné dans le Chapitre 6.

Conclusion et perspectives

Dans ce tapuscrit, nous présentons une classification et un état de l'art détaillé sur la génération d'alignements complexes et leur évaluation. Nous proposons une approche d'alignement complexe qui se base sur les besoins en connaissance d'un utilisateur. Nous proposons un système automatique d'évaluation d'alignements complexes et le complétons par un jeu de données. L'approche a été évaluée sur différents jeux de données et les résultats y sont consignés et analysés.

L'approche que nous proposons se base sur des hypothèses assez fortes, notamment le fait qu'un utilisateur sache traduire des CQA en requêtes SPARQL. Dans le futur, nous envisageons d'explorer la génération de requête SPARQL à partir de questions en langage naturel. La nécessité d'au moins une instance commune est aussi une limitation. Nous voudrions explorer la possibilité d'une approche ne nécessitant pas d'instances communes. Une des critiques que l'on peut faire à l'approche est de générer les alignements par morceaux (par CQA) au lieu de faire un alignement total une fois. Nous envisageons par la suite de nous intéresser à la façon dont notre approche pourrait s'intégrer à un dépôt d'alignement pour créer les alignements de manière incrémentale [Zhdanova & Shvaiko 2006]. En effet, les correspondances générées pour le besoin utilisateur pourraient être validées puis ajoutées à un tel dépôt. Si les correspondances couvrent le besoin d'un nouvel utilisateur, elles seraient donc disponibles, sinon l'utilisateur pourrait exprimer son besoin et en générer de nouvelles.

Le système d'évaluation proposé est limité par la qualité des systèmes de réécriture de requêtes à partir d'alignements complexes. Les correspondances *(c:c)* sont particulièrement difficiles à traiter, ce qu'il serait intéressant d'approfondir. Nous souhaitons analyser notre système d'évaluation d'alignements complexes plus en détail, notamment en le comparant à d'autres systèmes, comme par exemple un système fondé sur une comparaison sémantique (un tel système n'existe pas encore à notre connaissance).

Les alignements complexes peuvent être utilisés pour croiser des données en faisant de la réécriture ou de la fédération de requêtes. Ils peuvent également être intégrés dans des clés de liage pour aligner des instances. Les clés de liage peuvent nécessiter des alignements complexes. Nous pourrions envisager une approche qui se base sur des instances communes pour générer des alignements complexes et sur ces alignements complexes pour trouver plus d'instances communes.

Grâce à l'approche proposée, nous pouvons générer des alignements complexes qui rendent interopérables les bases de connaissances et les outils utilisant les ontologies. Toutefois, il est nécessaire de garder à l'esprit que rendre tout système in-

teropérable n'est pas sans danger. Bien que l'interopérabilité soit souhaitée dans de nombreux domaines pour la portabilité des données, faciliter les démarches administratives, *etc.*, elle peut poser des problèmes éthiques. Il est notamment préférable de garder les données à caractère personnel cloisonnées.

Table of contents

Chapter 1

Introduction

The World Wide Web (WWW), also known as the Web, makes resources available on the Internet. Everyone with an Internet connection can access these resources and the information they contain. At the beginning of the Web, the resources were mostly limited to HyperText Markup Language (HTML) static pages, whose content is intended for human interpretation.

In order to make the resources and their information interpretable by machines, Tim Berners-Lee, the Web inventor, conceived the Semantic Web. He wrote in "Weaving the Web" [Berners-Lee & Fischetti 2000]:

> *I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A "Semantic Web", which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines.*

The Semantic Web emerged with the underlying concepts of knowledge representation and interoperability [Berners-Lee *et al.* 2001]. The first key step was to represent a resource and its associated knowledge so that machines can interpret it and reason over it: ontologies were introduced for this purpose. The second key step was to make the resources and their knowledge interoperable. Even if two agents use different ontologies, they should be able to communicate with each other.

Following the vision of Tim Berners-Lee, the World Wide Web Consortium (W3C)[1] has worked on a set of standards to make the Semantic Web possible. In the past years, many knowledge repositories have been created following the W3C standards and published on the Web. They are often linked to each other, and considered as a whole, they make up the Linked Open Data (LOD) cloud. Each repository (or knowledge base) includes an ontology which formalises the knowledge it contains. Even if the ontologies are created to make knowledge interpretable to machines, they are most of the time created by human beings. We (human beings) use language to formulate our thought and knowledge [Boroditsky 2011]. Language is full of ambiguity, polysemy and synonymy. Therefore, the ontologies we create are heterogeneous: the same piece of knowledge can be represented in many different ways. Making the knowledge interoperable as envisioned by Tim Berners-Lee can be achieved by ontology alignments [Euzenat & Shvaiko 2013]. Alignments draw a list of similar, equivalent or related concepts between knowledge bases. Finding

[1] `https://www.w3.org`

alignments is called ontology matching and it can be a fastidious task, especially for large ontologies. Many automatised ontology matching approaches proposed in the literature deal with simple alignment generation. They link identifiers of entities to each other, *e.g.*, they find correspondences like $\langle$ o_1*:Paper* , o_2*:Paper* , $\equiv$ $\rangle$ which declares equivalent the *Paper* concepts of the ontologies o_1 and o_2. However, more expressive correspondences may be needed to overcome the heterogeneity between ontologies. They are called complex correspondences, *e.g.*, $\langle$ o_1*:AcceptedPaper* , o_2*:Paper* $\sqcap$ $\exists o_2$*:hasDecision.*o_2*:Acceptance* , $\equiv$ $\rangle$ declares equivalent the concept of *accepted paper* from o_1 to the defined concept of the paper having an acceptance decision captured by a logic formula in o_2.

The need for complex correspondences has been identified in various domains and applications, such as cultural heritage, where some data integration or data translation applications are based on complex correspondences [Szekely *et al.* 2013, de Boer *et al.* 2012, Kakali *et al.* 2007, Nurmikko-Fuller *et al.* 2015]. To tackle the issue, complex alignment generation systems have been developed and used [Szekely *et al.* 2013], or complex correspondences have been manually created [Kakali *et al.* 2007, Nurmikko-Fuller *et al.* 2015]. In the agronomic domain, complex alignments have been used to cross-query linked open data repositories [Thiéblin *et al.* 2017]. In the biomedical domain, complex alignments have been used to build a consensual model from heterogeneous terminologies [Jouhet *et al.* 2017]. Moreover, complex alignments between medical ontologies have been published [Fung & Xu 2012, Giannangelo & Millar 2012]. In most of these works, complex alignments are manually created, a process which requires both expertise and time.

Despite the variety of automatic matching approaches, most of them aim at fully aligning two ontologies, *i.e.*, the output alignment aims at fully covering the common scope of the two ontologies. However, a user may not need as much coverage as he or she may be interested by only a part of the ontology scope. The proposed matcher relies on this assumption, more precisely, it requires the user to express its knowledge needs as Competency Questions for Alignment.

The work presented in this thesis discusses the automatisation of their discovery in complex alignment generation systems. It proposes a classification of complex alignment generation approachs, followed by a complex alignment generation approach. The matchers, just as the other systems, must be characterised and compared on the quality of their output. For this reason, a complex alignment evaluator is also proposed in this thesis.

The research question this thesis aims at studying is the following:

Do Competency Questions for Alignment help fostering complex alignments?

We decompose it into more precise questions which we will try to answer in the experimentation chapter.

Is one common instance per Competency Question for Alignment enough evidence to generate complex correspondences?

What is the impact of the number of support answers on the alignment quality?

What is the impact of the quality of the instance links on the generated alignments quality?

Can Competency Questions for Alignment improve the precision of generated alignments?

Does similarity reassessment based on counter-examples improve the quality of the generated alignments?

Can Competency Questions for Alignment improve the run-time performance of complex ontology matching?

What is the impact of the Competency Questions for Alignment on the type of output correspondence?

The main contributions of this thesis are i) a survey of complex alignment generation approaches ii) a complex alignment generation approach based on Competency Questions for Alignment iii) an evaluation system based on instance comparison. These contributions are presented in the following chapters:

Chapter 2 provides background information about knowledge, ontologies in the Semantic Web and their heterogeneities. It also defines ontology alignments and presents their applications, representation formats and evaluation strategies.

Chapter 3 surveys ontology and schema complex alignment generation approaches because ontology matching is often associated with schema matching [Shvaiko & Euzenat 2005]. It also surveys existing ontology matching evaluation processes for simple and complex alignment generation approaches.

Chapter 4 presents our proposition of a complex alignment generation approach. It relies on Competency Questions for Alignment to take into account the knowledge needs of the user.

Chapter 5 analyses the process of matching evaluation, the challenges for complex alignment generation evaluation. It then presents our proposition of an automatic complex alignment benchmark composed of an evaluation system and its associated dataset.

Chapter 6 details the experimentations and evaluation results of the proposed matcher.

Chapter 7 discusses the limitations and perspectives of the contributions of this thesis: a complex alignment generation approach classification, a complex alignment generation approach and a complex alignment benchmark.

Chapter 2

From heterogeneous ontologies to complex alignments

Content

The Web makes interlinked human-readable resources accessible. Making these resources available and readable by both machines and humans is a challenge that the Semantic Web endeavours to take up [Berners-Lee *et al.* 2001]. To achieve such a goal, the knowledge contained in the Web resources and their links must be formalised. For this reason, the W3C proposes formats and technologies to implement the Semantic Web.

Nowadays, the Semantic Web formalisms of the W3C are used in the the LOD, in which knowledge repositories are published and interlinked.

At the core of the Semantic Web, ontologies describe resources in a formal way. Ontologies are mostly created by human beings. Even if the ontology designers follow guidelines such as the NeOn Methodology [Suárez-Figueroa *et al.* 2012], the resulting ontologies can be heterogeneous. One of the main aspects of the Semantic Web (and of the LOD cloud) is the interoperability of the knowledge repositories. To

ensure that, ontology alignments have been introduced [Euzenat & Shvaiko 2013]. Their aim is to record the similarities between ontologies. The correspondences which compose ontology alignments can be simple or complex. Complex correspondences allow for more expressiveness than simple correspondences. Nevertheless, finding alignments is fastidious and automatising this task is the focus of the *ontology matching* domain. Automatic ontology matching approaches have been proposed in the literature. They however mostly focus on finding simple correspondences. Hence, complex ontology matching is still a challenge. Another stake is the evaluation of matching approaches. Indeed, a user may need a matcher which suits its use-case.

In this Chapter, we define ontologies with regard to knowledge representation, we present the main formalisms of the W3C and how they apply in the LOD cloud in Section 2.1. The ontologies which describe the resources can be heterogeneous. Even though the same piece of knowledge appears in two ontologies, its labels or its modelling can be different. We present the types of heterogeneity and introduce ontology alignments as a solution for interoperability in Section 2.2. Ontology alignments come with challenges such as their generation (ontology matching), their representation, or their evaluation which are also introduced in Section 2.2.

2.1 Knowledge, ontologies and the Semantic Web

One of the challenges of the Semantic Web is to represent the knowledge and make it accessible. Before going any further, we define the notion of **knowledge**.

2.1.1 Data, information and knowledge

Ontologies represent general knowledge or knowledge associated with a given field. As discussed in [Rowley 2007] and [Zins 2007], the notions of data, information and knowledge are not consensual. We propose a definition of knowledge with respect to data and information inspired from [Amarger 2015].

Data is a raw data, a value with no signification. For example, *42* is raw data of type integer.

Information is a data associated with its context. For example, the data *42* associated with its context becomes a piece of information: *Camille has 42 hens.*

Knowledge is a formalisation of the information in a knowledge representation language. Associated with other pieces of knowledge, new knowledge can be inferred. The piece of knowledge *Camille has 42 hens* formalised into a knowledge representation language can be associated with the piece of knowledge *People having more than 10 hens are gallinophile.* The following piece of knowledge can be inferred from the two previous ones: *Camille is gallinophile.*

We distinguish two kinds of knowledge: assertional knowledge and ontological knowledge. An assertional piece of knowledge is the formalisation of a fact or information. For example, *Camille has 42 hens* is part of the assertional knowledge. As a complement, ontological knowledge represents the general rules of the domain, *i.e.*, it is a **conceptualisation** of the domain. In our previous example, *People having more than 10 hens are gallinophile* is part of the ontological knowledge.

2.1.2 Information and knowledge representation models

We introduced that knowledge is the formalisation of information in its context; and that there are two kinds of knowledge: assertional and ontological.

The ontological part of a knowledge base constitutes what we call here an **ontology**. [Studer *et al.* 1998] proposes the following definition by merging those of [Gruber 1993] and [Borst 1997]: "An ontology is a formal, explicit specification of a shared conceptualization." This definition completes ours by stating that the conceptualisation should express a shared view between several parties, a consensus rather than an individual view. Also, such a conceptualisation should be expressed in a formal machine readable format. Later, [Guarino *et al.* 2009] revisited the notions of conceptualisation, formal and explicit specification, and the importance of "shared" in the definition.

There are more or less expressive ways to represent the ontologies, that we call **models**. The expressiveness of the models depends on the variety of constraints and axioms they can represent. In [Uschold & Gruninger 2004, Euzenat & Shvaiko 2013], the models have been ranked by expressiveness.

We thereafter give a list of models and describe their expressiveness.

Table schemata A table schema is a flat schema instantiated as tabular data. The table schema refers to the name of the table columns (also called attributes).

Relational database schemata (RDB) Relational database schemata require the data to be organised as relations implemented by tables. The name of each relation is given, as well as the names and types of the relations' attributes. This model includes the notions of primary key and foreign key providing the links between the relations.

Document-oriented schemata (DOS) Document type Definition (DTD), Extensible Markup Language (XML) schemata and JavaScript Object Notation (JSON) schemata define the structure of documents (XML or JSON documents). These document-oriented schemata include elements, attributes and types. Elements can be either complex when specifying nested sub-elements, or simple when specifying built-in data types, such as string, for an element or attribute.

Conceptual models (CM) Conceptual models include entity-relationship models, used to abstract a relational database schema, and UML class diagrams, used

to abstract object-oriented programs and databases. The entities of these models describe the internal structure of domain objects. The entities can be organised as a hierarchy. Moreover, these models can also express relations (associations) with a multiplicity of constraints between the entities.

(Formal) ontologies Formal ontologies are axiomatised theories. Their entities are most often classes, object properties, data properties, instances and values. The expressiveness of the ontology's axioms is limited to the fragment of logic they implement (*e.g.*, $\mathcal{SHIN}, \mathcal{SROIQ}$, two-variable First-Order Logic (FOL)). Even though various ontology languages have been proposed in the past, the Web Ontology Language (OWL), a W3C standard [McGuinness & Van Harmelen 2004], is now widely used. The variants of OWL implement different logic fragments such as $\mathcal{SHIF}$, $\mathcal{SHOIN}$ or $\mathcal{SROIQ}$. Other ontology languages such as DAML+OIL [Horrocks 2002] or Conceptual Modelling Language (CML) [Schreiber *et al.* 1994] implement a fragment of Description Logic (DL).

In Chapter 3, we refer to all the above-mentioned models as **knowledge representation models** regardless of their expressiveness.

2.1.3 Ontology entities and semantics

[Guarino 1998] classifies ontologies according to their "level of generality", in particular:

Top-level ontologies describe very general concepts (*e.g.*, space, time, object), which are independent of a particular problem or domain. These ontologies, also named upper or foundational ontologies [Semy *et al.* 2004], are equipped with a rich axiomatic layer;

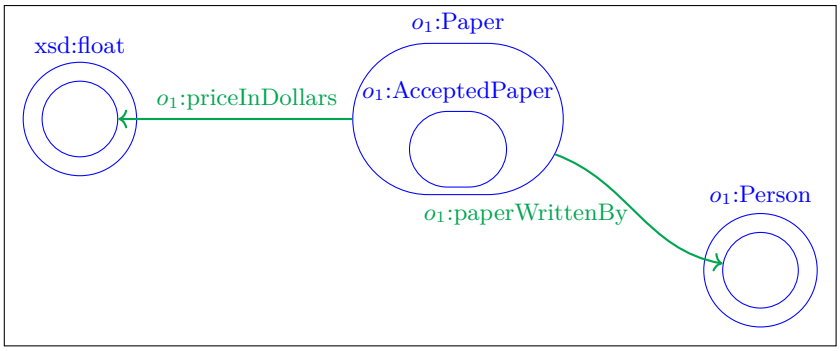
Domain ontologies and **task ontologies** that describe the entities and other information related to a generic domain (*e.g.*, biology or aeronautic), or a generic task or activity (*e.g.*, diagnosis) by specialising the concepts represented in top-level ontologies; and finally

Application ontologies which describe the roles played by domain entities when performing an activity (which are, respectively, described by domain and activity ontologies).

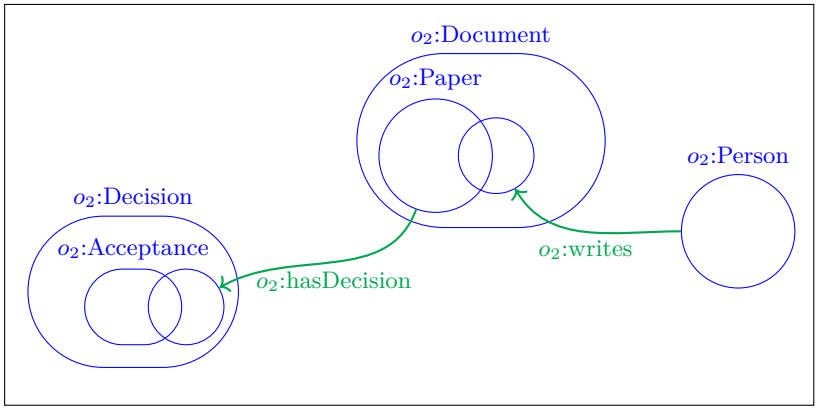
The interpretation of the ontologies' entities is restrained by the axioms of the ontology. Figure 2.1 shows three different ontologies about conference organisation. The legend used for the schemata of this manuscript is described in Appendix A.3.

The ontologies are represented in an ontology language. Most of these languages share the same types of entities and implement a fragment of two-variable FOL which can be represented with DL [Baader *et al.* 2009]. The most common entities shared by the ontology languages are:

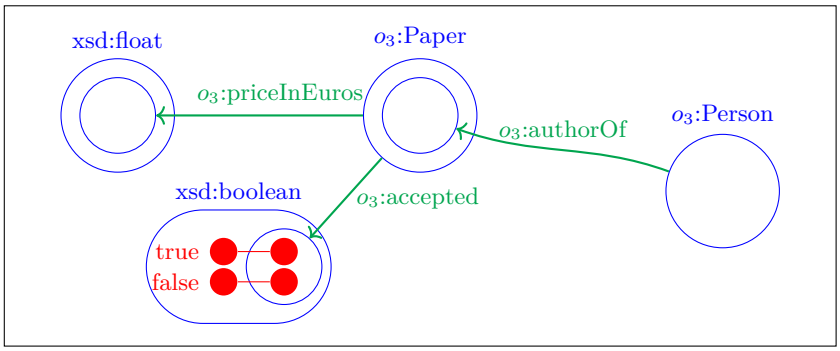
Classes (or concepts) which are interpreted as a set of individuals in the domain. o_1*:AcceptedPaper* is a class.



(a) o_1



(b) o_2



(c) o_3

Figure 2.1: Example ontologies

Table 2.1: Syntax and semantics of DL expressions [Baader *et al.* 2003]

Description	Syntax	Semantics
Everything	$\top$	$\mathcal{D}$
Empty	$\bot$	$\emptyset$
Concept	C	$C^{\mathcal{I}} \subseteq \mathcal{D}$
Role	R	$R^{\mathcal{I}} \subseteq \mathcal{D} \times \mathcal{D}$
Individual	a	$a^{\mathcal{I}} \in \mathcal{D}$
Intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Complement	$\neg C$	$\mathcal{D} \setminus C^{\mathcal{I}}$
Universal restriction	$\forall R.C$	$\{x \in \mathcal{D} \vert \forall y((x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}})\}$
Existential restriction	$\exists R.C$	$\{x \in \mathcal{D} \vert \exists y(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
Cardinality restriction	$= \ n \ R.C$	$\{x \in \mathcal{D} \vert \#\{y \in C^{\mathcal{I}} \wedge (x,y) \in R^{\mathcal{I}}\} = n\}$
Min Cardinality restriction	$\leq \ n \ R.C$	$\{x \in \mathcal{D} \vert \#\{y \in C^{\mathcal{I}} \wedge (x,y) \in R^{\mathcal{I}}\} \leq n\}$
Max Cardinality restriction	$\geq \ n \ R.C$	$\{x \in \mathcal{D} \vert \#\{y \in C^{\mathcal{I}} \wedge (x,y) \in R^{\mathcal{I}}\} \geq n\}$
Role composition	$R \circ S$	$\{(x,y) \vert \exists z((x,z) \in R^{\mathcal{I}} \wedge (z,y) \in S^{\mathcal{I}}\}$
Inverse	R^{-}	$\{(x,y) \vert (y,x) \in R^{\mathcal{I}}\}$
Domain restriction	*dom(C)*	$C^{\mathcal{I}} \times \mathcal{D}$
Range restriction	*range(C)*	$\mathcal{D} \times C^{\mathcal{I}}$
Concept instance	$a : C$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
Role instance	$(a,b) : R$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
Entity equivalence	$E \equiv E'$	$E^{\mathcal{I}} = E'^{\mathcal{I}}$
Entity subsumption	$E \sqsupseteq E'$	$E^{\mathcal{I}} \supseteq E'^{\mathcal{I}}$

Relations which are interpreted as sets of pairs of individuals in the domain. o_1*:paperWrittenBy* is a relation.

Individuals or instances are interpreted as a particular individual of a domain.

Data types are subset of the domain which specify values. *xsd:boolean* is a data type.

Data values (or literal values) are simple values. *"true"* is a data value of type boolean.

A **knowledge base** is a tuple $KB = \langle C, I, R, T, V, A \rangle$, where C is a set of classes, I a set of individuals, R a set of relations, T a set of datatypes, V a set of values and A a set of axioms (formulae) between the entities from the before-listed sets. A knowledge base is composed of a terminologic part ($\mathcal{T}box$), called **ontology** and an assertional part ($\mathcal{A}box$) called **population**.

The $\mathcal{T}box$ is $\langle C, R, T, V, \mathcal{T}(A) \rangle$ and the $\mathcal{A}box$ is $\langle I, V, A \setminus \mathcal{T}(A) \rangle$, where $\mathcal{T}(A)$ is the set of axioms which define and describe the entities from C, R, T. V contains the label values of the other entities; for this reason it is in both the $\mathcal{T}box$ and $\mathcal{A}box$.

Table 2.1 shows syntax and semantics of DL expressions which will be used in the remainder of this thesis. In the table, $\langle \mathcal{I}, \mathcal{D} \rangle$ is an interpretation in which $\mathcal{I}$ is the interpretation function and $\mathcal{D}$ is the domain of interpretation.

2.1.4 Knowledge in the Semantic Web

Ontologies conceptualise a domain and can be formalised as logic theories. They are at the core of the Semantic Web to describe the resources available on the Web. To make ontologies readable by machines and compliant with the Web standards, the W3C has provided different standards, formats and recommendations. In particular, the main principles of the Semantic Web, also known as Linked Data (LD)[1] principles, state that:

- Entities (or resources) are identified by dereferencable Uniform Resource Identifiers, which means that they can be accessed with HyperText Transfer Protocol (HTTP).
- Entities are represented with W3C standards.
- Entities are connected together by crawlable links.

In this section, we present the W3C standards used for representing knowledge.

The base of the Semantic Web are URIs. A Uniform Resource Identifier (URI) is a string which uniquely identifies a resource (or entity). A URI can be represented as a prefixed name. For example the URI *http://dbpedia.org/ontology/Species* can be represented as *dbo:Species* with *dbo = http://dbpedia.org/ontology/* the prefixed

[1] `https://www.w3.org/DesignIssues/LinkedData.html`

namespace. All the namespaces in this thesis are recorded in Appendix A.2. Internationalised Resource Identifiers are an extension of URIs as they can contain non-ASCII characters.

Resources identified by URIs may be described using the Ressource Description Framework (RDF), the W3C standard used to represent knowledge[2]. RDF is based on triples $\langle$ *subject , predicate , object* $\rangle$ in which the *subject* is a URI or a blank node, the *predicate* is a URI and the *object* can be a literal value (*e.g.*, string, integer), a blank node or a URI. The *object* is a value associated with the *subject* by the *predicate*. For example, in the triple $\langle$ o_1*:paper1 ,* o_1*:paperWrittenBy ,* o_1*:person1* $\rangle$, o_1*:person1* is a value of o_1*:paperWrittenBy* for o_1*:paper1*.

Put all together, the RDF triples become a labelled graph, called **RDF graph**. The *subject* and *object* of a triple are the labels of the vertices and its *predicate* the label of the arc between them.

RDF on its own is not expressive enough to describe logical axioms in ontologies. Therefore, ontology languages have been developed to enrich RDF expressions with additional meaning. First, RDF Schema (RDFS)[3] the ontology language associated with RDF, was proposed. Then, OWL[4] completed RDFS by providing more expressiveness.

A knowledge repository makes its content available to human and machines through its querying. Thus, RDF graphs can be queried with SPARQL Protocol And RDF Query Language (SPARQL)[5]. A SPARQL query contains a set of triple patterns called a **basic graph pattern**. The basic graph pattern can contain logical operators such as UNION, MINUS, *etc.* Triple patterns are like RDF triples except that each of the subject, predicate and object may be a variable. For example, $\langle$ *?x ,* o_1*:paperWrittenBy ,* o_1*:person1* $\rangle$ is a triple pattern, *?x* is a variable.

A basic graph pattern matches a subgraph of the RDF data when RDF terms from that subgraph may be substituted for the variables. SPARQL queries can retrieve information (*e.g.*, SELECT, ASK or DESCRIBE queries) or they can add or remove triples from the RDF graph (*e.g.*, CONSTRUCT, INSERT or DELETE queries).

The result of a SPARQL SELECT query is the set of bindings of the variables which appeared in its SELECT clause and which were matched in the basic graph patterns. For example the query `SELECT ?x WHERE { ?x o1:paperWrittenBy o1:person1.}` run over the RDF graph composed of the triple $\langle$ o_1*:paper1 ,* o_1*:paperWrittenBy ,* o_1*:person1* $\rangle$ outputs the binding *?x* = o_1*:paper1*.

2.1.5 The Linked Open Data cloud

One of the most famous implementation of the W3C standards is the LOD cloud, also called the "Web of data"[6]. The LOD is an initiative to reduce the isolation of

[2] `https://www.w3.org/RDF/`
[3] `https://www.w3.org/TR/rdf-schema/`
[4] `https://www.w3.org/TR/owl2-overview/`
[5] `https://www.w3.org/TR/rdf-sparql-query/`
[6] `https://www.w3.org/standards/semanticWeb/`

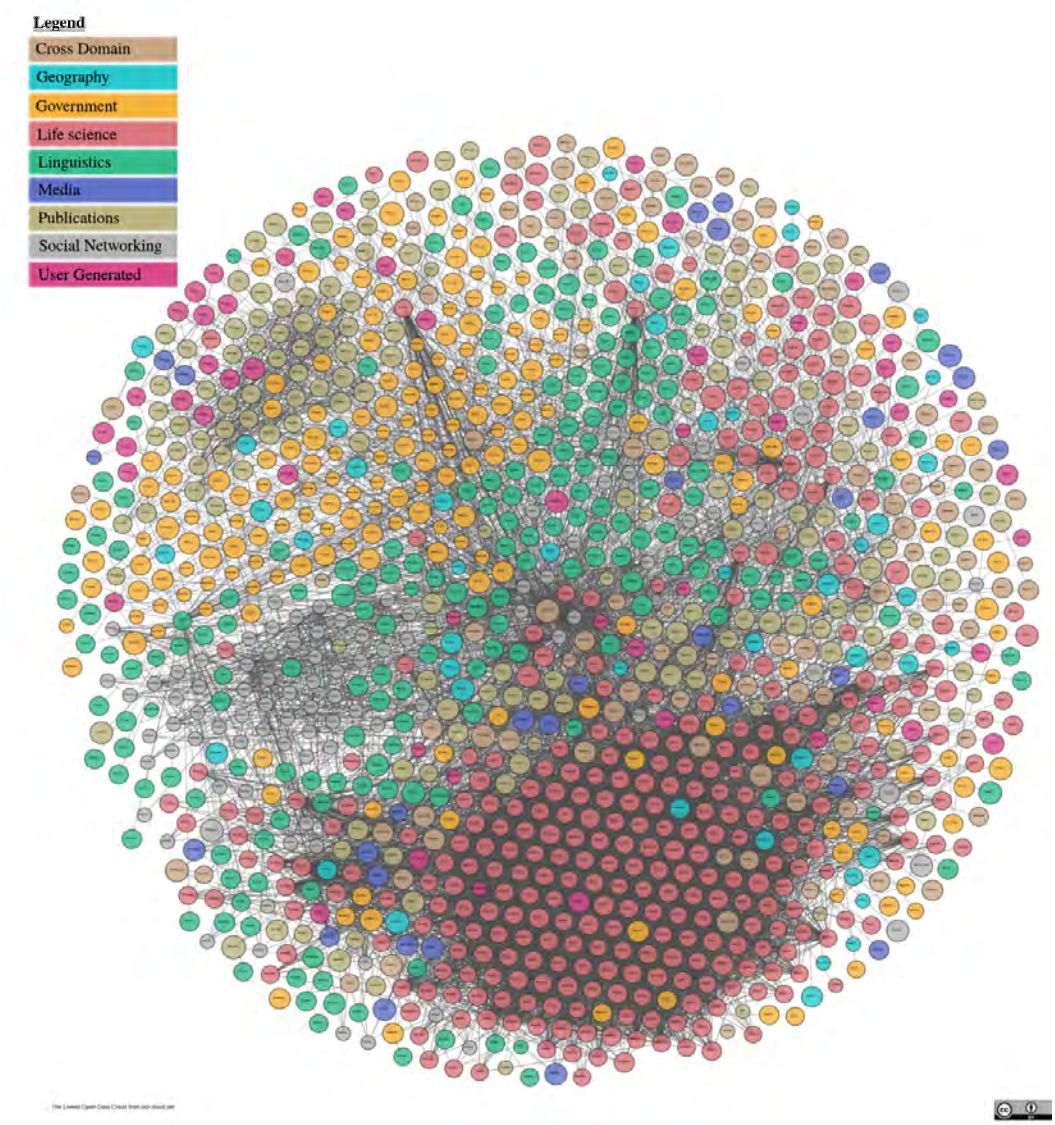


Figure 2.2: LOD cloud in January 2019 (`https://www.lod-cloud.net/`)

knowledge repositories. The LOD cloud is a set of knowledge repositories following the LOD guidelines: they are on the Web, they reuse the Semantic Web formats (RDF, URIs, *etc.*) and they are linked to one another.

The number of LOD cloud repositories increases each year. Figure 2.2 shows the LOD cloud in January 2019. Each repository of the LOD cloud is a knowledge base and its content can be shared without duplication.

The links between the repositories connect similar or equivalent resources. Therefore, the knowledge about a given resource can be extended by each knowledge base which describes its equivalent. For example, the resource *Gallus gallus* (*i.e.*, species of chicken) can be described in two knowledge bases providing complementary information, *e.g.*, one about its natural environment and another about its use in culinary recipes.

However, the links between the LOD repositories are not as prevalent as they could be. [Ermilov *et al.* 2016] states that less than 10% of the LOD entities are actually linked. The *owl:sameAs* links represent that two URIs refer to the same object. These links are commonly used but are sometimes erroneous [Halpin *et al.* 2015]. Moreover, the ontologies of the LOD cloud were often created independently. This makes the task of linking them harder.

The creation process of the ontologies can result in different ways of modelling the domain of interest. For that reason, we analyse how they are created, in particular how competency questions in natural language can shape them.

2.1.6 Competency Questions: specifying knowledge needs

To create an ontology, the experts must define its scope, *i.e.*, the knowledge requirements it will cover. The concept of Competency Question (CQ) has been introduced in ontology authoring to formalise the knowledge requirements of an ontology. CQs are *ontology's requirements in the form of questions the ontology must be able to answer* [Grüninger & Fox 1995]. The competency questions have been integrated in ontology engineering methodologies such as NeOn [Suárez-Figueroa *et al.* 2012].

In [Ren *et al.* 2014], a CQ in natural language can be expressed and translated into a SPARQL query. The authors define a set of characteristics to analyse competency questions based on both the natural language question and its associated SPARQL query. The work of [Ren *et al.* 2014] was corroborated by a recent study [Dennis *et al.* 2017] on how users' interpretation of the CQs match the CQs' author's intentions. The following characteristics are those defined by [Ren *et al.* 2014].

Question type A CQ can be a *selection* question (what, when, which, *etc.*), a *binary* question (yes or no), or a *counting* question (how many). The binary and counting question have a corresponding selection question: "Is this paper accepted ?" and "How many accepted papers are there ?" have for selection question "What are the accepted papers?".

Element visibility The elements of a CQ can be *implicit* or *explicit* in the natural

language query. For example, in the CQ "What is the decision of this paper ?" *Decision* and *Paper* are explicit, the relation *hasDecision* is implicit.

Question polarity A CQ can be asked in a *positive* or *negative* manner, *e.g.*, "Which papers are not rejected?" is negative.

Predicate arity The main predicate of a CQ query may take one or more arguments. A *unary* predicate takes one argument, a *binary* predicate takes two arguments, an *n-ary* predicate takes 3 or more arguments. For example, the main predicate of the CQ "What are the accepted papers?", is a unary predicate: "accepted papers". The main predicate of the CQ "What are the papers with an acceptance decision?" is "has decision" which is a binary predicate.

Relation type The predicate of a SPARQL query is either a *data property* or an *object property*.

Modifier A CQ can contain restrictions on its values or entities (at least, the most, more than, *etc.*).

Domain independent element A CQ can contain *temporal* or *spatial* elements.

As we can see in the predicate arity example, the same piece of knowledge can be expressed in different manners. Both questions "What are the accepted papers?" and "What are the papers with an acceptance decision?" expect a list of accepted papers. However, their expression in natural language can lead to a different representation in the ontologies. The first CQ has an *Accepted Paper* underlying concept as in o_1 in Figure 2.1(a) while the second implies the existence of a relation between a *Paper* and a *Decision* that can be an *Acceptance* as in o_2 in Figure 2.1(b). Following the idea that language shapes the thought [Boroditsky 2011], language shapes the ontologies, and this results in heterogeneity.

2.2 Ontology alignments

In order to overcome ontology heterogeneity, ontology alignments were introduced.

2.2.1 Ontology heterogeneities

Ontologies can be heterogeneous in various ways. We present different levels of heterogeneity inspired from [Visser *et al.* 1997, Klein 2001, Euzenat & Shvaiko 2013].

Syntactic heterogeneity (same name in [Euzenat & Shvaiko 2013], language level mismatch in [Klein 2001]) occurs when two ontologies are not expressed in the same ontology language. For example, an OWL ontology and a CML ontology are syntactically different.

Terminological heterogeneity (same name in [Euzenat & Shvaiko 2013], terminological mismatch in [Klein 2001], part of explication mismatch in

[Visser *et al.* 1997]) occurs when the terms used to describe a concept vary from ontology to ontology. The use of different natural languages or the use of synonyms cause terminological heterogeneity. For example, *Paper* and *Papier* or *Paper* and *Article*.

Modelling heterogeneity (style of modelling mismatch in [Klein 2001], part of explication mismatch in [Visser *et al.* 1997]) occurs when the modelling choices of two ontologies differ. For example, the distinction between two classes can be modelled with a qualifying attribute or by introducing a separate class (*e.g.*, $\exists o_1$*:acceptedPaper.{true}* and o_2*:AcceptedPaper*); an event can be modelled as an interval of time or a point in time.

Conceptual heterogeneity (conceptualisation mismatch in [Visser *et al.* 1997] and [Klein 2001]) occurs when there is a difference between the scopes of ontologies modelling the same domain of interest. First, there can be a **scope coverage heterogeneity**. For example, with two ontologies about conference organisation, one can focus on the social events and the other on the paper acceptance status. Then, the **granularity heterogeneity** occurs when one ontology models a domain with more details than another. Finally, the **perspective heterogeneity** occurs when two ontologies differ by the perspective under which the objects they describe should be used. For example, when two ontologies model authors for their conference registration payment or their scientific contribution. In comparison with [Euzenat & Shvaiko 2013], we make a distinction between the way the concepts are expressed (modelling heterogeneity) and the scope of the ontologies (conceptual heterogeneity).

Semiotic heterogeneity (same name in [Euzenat & Shvaiko 2013] or pragmatic heterogeneity [Bouquet *et al.* 2004]) concerns the different human interpretations of two ontologies. Different individuals or communities may interpret the same ontology in different ways in different contexts. For example, in a context of formalisation, an initial user can express an ontology as a class hierarchy with some first order formulae. If the whole ontology is then expressed only as first order formulae, the initial user may not understand the equivalence between the two expressions of the same ontology. This is not a matter of syntactic heterogeneity because first order formulae are allowed in the initial formalism.

2.2.2 Expressions

We define expressions, which is a preliminary for the definition of alignment and correspondence.

A **simple expression** is composed of a single entity represented by its unique identifier (*e.g.*, a URI for ontologies in the Semantic Web). For example, the URI o_1*:Paper* is a simple expression of o_1.

A **complex expression** is composed of at least one entity on which a constructor or a transformation function is applied. For example, $\exists o_3$*:accepted.*$\{true\}$ is a

complex expression which represents all the papers having the value *true* for the o_3*:accepted* property. The constructor used here is a value restriction constructor.

A **constructor** is a logic constructor (*e.g.*, union, intersection, inverse) or a restriction constructor (*e.g.*, cardinality restriction, type restriction, value restriction).

A **transformation function** is a function that modifies the values of a literal field. It can be an aggregation function (*e.g.*, string concatenation, sum of integers), a conversion function (*e.g.*, metric conversion), *etc.*

2.2.3 Simple and complex alignments

Ontology alignments can reduce the terminological, modelling and conceptual heterogeneity of ontologies. In this section, we give the definition of simple and complex correspondences and alignments.

A **correspondence** c_i is a tuple $\langle\ e_{o_1}\ ,\ e_{o_2}\ ,\ r\ ,\ n\ \rangle$. e_{o_1} and e_{o_2} are the **members** of the correspondence. They can be simple or complex expressions with entities from respectively o_1 and o_2:

- if the correspondence is **simple**, both e_{o_1} and e_{o_2} are simple expressions;
- if the correspondence is **complex**, at least one of e_{o_1} or e_{o_2} is a complex expression;
- r is a relation, *e.g.*, equivalence ($\equiv$), more general ($\sqsupseteq$), more specific ($\sqsubseteq$), holding between e_{o_1} and e_{o_2}.
- additionally, a value n (typically in [0,1]) can be associated with c_i indicating the degree of confidence that the relation r holds between e_{o_1} and e_{o_2}.

One can indicate if each member of the correspondence is a simple expression, noted s, or a complex expression, noted c.

A simple correspondence is always *(s:s)* whereas a complex correspondence can be *(s:c)*, *(c:s)* or *(c:c)*. The *(1:1), (1:n), (m:1), (m:n)* notations have been used for the same purpose in the literature [Rahm & Bernstein 2001, Zhou *et al.* 2018] (*1* for s and m or n for c). However they can be mistaken for the alignment arity or multiplicity [Euzenat 2003].

We provide some examples of complex correspondences based on the definitions above and the motivating example ontologies (Figure 2.1).

c_1 = $\langle$ o_1*:Person* , o_3*:Person* , $\equiv$ $\rangle$ is an *(s:s)* simple correspondence.

c_2 = $\langle$ o_1*:priceInDollars* , *changeRate*(o_3*:priceInEuros*) , $\equiv$ $\rangle$ is an *(s:c)* complex correspondence with a transformation function: *changeRate*.

c_3 = $\langle$ $\exists o_2$*:hasDecision*.o_2*:Acceptance* , o_1*:AcceptedPaper* , $\equiv$ $\rangle$ is a *(c:s)* complex correspondence with constructors.

$c_4 = \langle$ *o1:paperWrittenBy* , *o3:authorOf*$^-$, $\equiv \rangle$ is an *(s:c)* complex correspondence with the *inversion* constructor.

$c_5 = \langle \exists$*o3:accepted.*$\{true\}$, $\exists$*o2:hasDecision.o2:Acceptance* , $\equiv \rangle$ is a *(c:c)* complex correspondence with constructors.

An **ontology alignment** is directional between a source ontology o_1 and a target ontology o_2, denoted $A_{o_1 \to o_2}$. $A_{o_1 \to o_2}$ is a set of correspondences $A_{o_1 \to o_2} = \{c_1, c_2, ..., c_n\}$. In opposition to a **simple alignment**, a **complex alignment** contains at least one complex correspondence. Complex alignments can reduce modelling heterogeneity while simple alignments cannot. The links in the LOD cloud are mostly simple correspondences with an equivalence link. Most of them occur at *Abox* level. For this reason, the terminological and modelling heterogeneities between the LOD cloud are far from being covered.

2.2.4 Ontology matching

As defined in [Euzenat & Shvaiko 2013], **ontology matching** is the process of generating an ontology alignment A between two ontologies: a source ontology o_1 and a target ontology o_2. The matching process can take other parameters such as an existing alignment (anchor alignment), matching parameters (weights, thresholds, *etc.*) and external resources.

The **matching process** can be seen as a function f which returns an alignment $A'_{o_1 \to o_2}$ from a pair of ontologies o_1 and o_2, an anchor alignment $A_{o_1 \to o_2}$, a set of matching parameters p and external resources r.

$$A'_{o_1 \to o_2} = f(o_1, o_2, A_{o_1 \to o_2}, p, r)$$

This definition of *pairwise* matching process can be however extended to cover multiple ontologies. A **holistic** matching process considers more than two ontologies together without a source or target distinction [Grütze *et al.* 2012, Megdiche *et al.* 2016].

$$A' = f(o_1, o_2, \ldots, A, p, r)$$

On the other hand, **compound** matching is the process of matching one or more source ontologies to one or more target ontologies. This process is *pairwise* between the union of the source ontologies and the union of the target ontologies [Oliveira & Pesquita 2018].

$$A'_{\{o_1, o_2, \ldots\} \to \{o_3, o_4, \ldots\}} = f(\{o_1, o_2, \ldots\}, \{o_3, o_4, \ldots\}, A_{\{o_1, o_2, \ldots\} \to \{o_3, o_4, \ldots\}}, p, r)$$

Complex ontology matching is the process of generating a complex alignment between ontologies.

complex alignment generation is more difficult than simple alignment generation. Indeed, the **matching space** between two ontologies is the set of all the

possible correspondences between them. In simple alignment generation, the matching space is limited to the product of the source entities and the target entities. If o_1 contains m entities and o_2 contains n entities, the matching space is mn. In complex alignment generation, the matching space is larger as the complex expressions can include any number of entities, constructors and transformation functions.

2.2.5 Alignment representation formats

Alignments and correspondences can be represented as logic formulae or rules between two (or more) ontologies. Formats dedicated to describe axioms, rules or queries such as OWL, Semantic Web Rule Language (SWRL) or SPARQL can be used to represent alignments. Alignments expressed in these formats often have a designated application, *e.g.*, an alignment expressed in OWL is often used for the task of ontology merging.

In parallel, dedicated alignment representation formats have been proposed. The Alignment format which is proposed in the Alignment API [David *et al.* 2011] has become a reference in the ontology matching community. Expressive and Declarative Ontology Alignment Language (EDOAL) [Euzenat *et al.* 2007] is an extension of the Alignment format to represent the complex correspondences between OWL ontologies.

In this section, we present the languages originally designed to describe axioms or rules outside the specific scope of alignment representation (Section 2.2.5.1). Then, we introduce the dedicated languages (Section 2.2.5.2). We present the expression of correspondence c_3 from the examples in Section 2.2.3 in some of the introduced languages.

Table 2.2 gives a summary of the complex alignment formats presented in this section with their context of application. For instance, alignments represented in OWL are usually used for the task of ontology merging.

The alignment generation approaches presented in Chapter 3 use formats described in this section.

2.2.5.1 Generic representations

Rules and axioms

OWL The Web Ontology Language (OWL) [McGuinness & Van Harmelen 2004] can represent complex alignments as axioms involving logic constructors and entities from the source and target ontologies. These axioms form a merging ontology. The expressiveness of the correspondences in OWL (taking into account the expressiveness of the aligned ontologies) is restricted to the $\mathcal{SROIQ}$ logic for decidability reasons. The correspondence c_3 represented in the XML concrete Syntax of OWL is given in Figure 2.3.

Table 2.2: Complex alignment formats, "Logic" shows whether the format can represent logic constructors, "Transformations" shows whether the format can represent value transformation functions

Format	Type of knowledge representation models	Logic	Transformation	Alignment context application
Web-PDDL	FOL onto to FOL onto	√		Data integration
OWL	OWL onto to OWL onto	√		Ontology merging
SWRL	OWL onto to OWL onto	√	√	Data integration
SQL	RDB to RDB	√	√	Querying, Data translation
XQuery	XML to XML	√	√	Querying, Data translation
SPARQL	OWL onto to OWL onto	√	√	Querying, Data translation
EDOAL	OWL onto to OWL onto	√	√	Generic
XeOML	OWL onto to OWL onto	√	√	Generic
SBO	DAML+OIL onto to DAML+OIL onto	√	√	Data translation
SPIMBench	RDF to RDF	√	√	Data translation
R2RML	RDB to RDF	√	√	Data translation
RML	CSV, XML, JSON to RDF	√	√	Data translation
xR2RML	mixed formats(CSV, RDB, XML, JSON) to RDF	√	√	Data translation
D2RML	RDB, CSV, XML, JSON to RDF	√	√	Data translation
XSLT	XML to XML	√	√	Data translation

Web-PDDL The Web-PDDL [Dou 2008] is a strongly typed FOL (first-order logic) language. It allows the use of variables, constants, conditions, logical constructors and quantifiers. The predicates and constants take the form of URIs. An example of Web-PDDL for representing the correspondence c_3 is given in Figure 2.4.

SWRL The Semantic Web Rule Language (SWRL) [Horrocks *et al.* 2004] helps to define rules, in the form of FOL Horn-rules, between OWL ontologies. These rules provide flexibility thanks to the use of variables in the definition of the rules. This language comes with an XML Concrete Syntax to express the rules as XML documents. SWRL can be extended by *built-ins* based on the XQuery and XPath built-ins. These built-ins express transformation functions. An example of SWRL representing the correspondence c_3 is given in Figure 2.5.

Other logic syntaxes such as DataLog, RIF, *etc.* using URIs as predicates can be used to express logic formulae. Even if they were originally meant to express these formulae inside one ontology, they can be used to express correspondences when involving IRIs from more than one ontology.

Query languages

Alignments can be directly represented through semantically equivalent queries (or views) of their data. These query languages can use filters (or equivalent) to express transformation functions inside a query.

```
<owl:Class rdf:about="&o1;AcceptedPaper">
  <owl:equivalentClass>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:about="&o2;hasDecision"/>
      </owl:onProperty>
      <owl:someValuesFrom>
        <owl:Class rdf:about="&o2;Acceptance"/>
      </owl:someValuesFrom>
   </owl:Restriction>
  </owl:equivalentClass>
</owl:Class>
```

Figure 2.3: c_3 in OWL XML concrete syntax

```
(forall (x) (iff (is @o1:AcceptedPaper x)
   (exists( y - @o2:Acceptance) (@o2:hasDecision x y))))
```

Figure 2.4: c_3 in Web-PDDL

SQL Structured Query Language (SQL) is the language for querying relational databases.

XSLT, XPath, XQuery XQuery is the query language for XML documents. XSLT (eXtensible Stylesheet Language Tansformations) [Kay 2017] is a language with an XML concrete syntax. This language describes rules to transform a source tree (XML document) into a target tree (XML document). This language is based on transformation patterns and reuses XPath expressions. XPath (XML Path Language) defines expressions with logical operators and transformation functions over XML nodes. The XPath functions can be reused in other alignment languages.

SPARQL SPARQL is the RDF query language, a W3C standard. The following queries represent c_3 as equivalent SPARQL SELECT queries (as in Figure 2.7) and a SPARQL CONSTRUCT query (as in Figure 2.8).

2.2.5.2 Dedicated alignment representations

Various formats have been proposed to represent alignments between two different knowledge representation models. A survey on ontology alignment formats is presented in [Scharffe 2009].

EDOAL EDOAL [Euzenat *et al.* 2007] is an extension of the Alignment format to represent the complex correspondences between OWL ontologies. This language

```
<ruleml:imp>
  <ruleml:_rlab ruleml:href="#c3"/>
    <ruleml:_body>
      <swrlx:individualPropertyAtom  swrlx:property="&o2;
         hasDecision">
        <ruleml:var>x</ruleml:var>
        <ruleml:var>y</ruleml:var>
      </swrlx:individualPropertyAtom>
      <swrlx:classAtom>
        <owlx:Class owlx:name="&o2;Acceptance"/>
        <ruleml:var>y</ruleml:var>
      </swrlx:classAtom>
    </ruleml:_body>
    <ruleml:_head>
      <swrlx:classAtom>
        <owlx:Class owlx:name="&o1;AcceptedPaper"/>
        <ruleml:var>x</ruleml:var>
      </swrlx:classAtom>
    </ruleml:_head>
</ruleml:imp>
```

Figure 2.5: c_3 in SWRL

is based on correspondence patterns [Scharffe 2009] and can be processed by the Alignment API [David *et al.* 2011]. The Alignment format can be extended by other languages to express complex correspondences. c_3 is represented in EDOAL in Figure 2.6.

XeOML XeOML [Pazienza *et al.* 2004] is a language which represents alignments for ontologies and can be extended to other kinds of knowledge representation models. It is based on an XML schema (*Abstract Mapping* schema) to describe the structure of an alignment and is completed by two other schemata (*Ontology Element Definition* and *Mapping Definition*).

SBO MAFRA [Maedche *et al.* 2002, Silva & Rocha 2003] is a framework for constructing and editing DAML+OIL ontology alignments. The alignment representation part of the framework is based on the Semantic Bridge Ontology (SBO). This (not maintained) ontology provides a vocabulary to express complex correspondences with logical constructors and some transformation functions such as string concatenation.

SPIMBench The SPIMBench vocabulary was defined in an instance matching benchmark [Saveta *et al.* 2015]. It allows for the description of data transforma-

```
<map>
  <Cell>
    <entity1>
      <edoal:Class rdf:about="&o1;AcceptedPaper" />
    </entity1>
    <entity2>
      <edoal:AttributeDomainRestriction>
        <edoal:onAttribute>
          <edoal:Relation rdf:about="&o2;hasDecision" />
        </edoal:onAttribute>
        <edoal:exists>
          <edoal:Class rdf:about="&o2;Acceptance" />
        </edoal:exists>
      </edoal:AttributeDomainRestriction>
    </entity2>
    <measure rdf:datatype="&xsd;float">1.0</measure>
    <relation>Equivalence</relation>
  </Cell>
</map>
```

Figure 2.6: c_3 in EDOAL

tion between ontologies. These transformations include logic rules (based on OWL axioms) and value transformation functions.

In the area of OBDA (Ontology-Based Data Access) [Xiao *et al.* 2018], different formats to express correspondences between relational databases and RDF datasets have been proposed in the literature. A comprehensive review of different formats can be found in [Hert *et al.* 2011]. Here, the W3C RDB to RDF Mapping Language (R2RML) format and some of its extensions are briefly introduced.

R2RML R2RML is a W3C format [Das *et al.* 2012] used to represent correspondences between relational databases and RDF datasets. R2RML correspondences are expressed as RDF datasets. A few string operations can be expressed in the correspondences. The R2RML correspondences show how the data from the source schema should be transformed into the target ontology.

RML The RML language [Dimou *et al.* 2014] extends the R2RML format by allowing other kinds of data sources such as XML schema, JSON, or tabular data (CSV). The FnO ontology [De Meester *et al.* 2016] can be used in RML to describe transformation functions in the correspondences.

xR2RML The xR2RML language [Michel *et al.* 2015] extends the R2RML format by allowing the description of correspondences of mixed formats in the source schema. For example, if a JSON object is the value of a cell in a relational database.

D2RML The D2RML language [Chortaras & Stamou 2018] is based on R2RML and RML, allowing conditional case statements and programming inside the correspondences.

2.2.6 Ontology mediation applications

Ontology alignments are used in ontology mediation applications such as query rewriting, data translation, ontology merging and data integration [Euzenat & Shvaiko 2013]. The examples of the following scenarii are based on c_3 from Section 2.2.3.

Query rewriting Query rewriting consists of translating a query written in terms of the source ontology o_1 into a query written in terms of the target ontology o_2. For example, in Figure 2.7 a query over o_2 can be rewritten over o_1 using the correspondence c_3.

```
SELECT DISTINCT ?x WHERE {
  ?x a o1:AcceptedPaper.
}

SELECT DISTINCT ?x WHERE {
  ?x o2:hasDecision ?y.
  ?y a o2:Acceptance.
}
```

Figure 2.7: A source query and its rewritten equivalent using c_3.

Instance translation In an instance translation scenario, an alignment is used to translate the instances from the source ontology into the target ontology. The SPARQL CONSTRUCT query in Figure 2.8 can translate the instances described by o_2 into instances described by o_1 with correspondence c_3.

```
CONSTRUCT{ ?x a o1:AcceptedPaper. }
WHERE{ ?x o2:hasDecision ?y.
       ?y a o2:Acceptance.}
```

Figure 2.8: Construct query for instance translation with c_3.

Ontology merging Ontology merging is the process of creating a new ontology from two existing ontologies. The new ontology includes the two ontologies of

the alignment as well as axioms translating the correspondences contained in the alignment itself. For example, if o_2 and o_1 were merged with an alignment containing only c_3, the resulting ontology would contain all the axioms from o_1, all the axioms from o_2 and the axiom $o_1{:}AcceptedPaper \equiv \exists o_2{:}hasDecision.o_2{:}Acceptance$.

Data integration In a data integration scenario, various dataset or knowledge bases are queried at once and a broker returns the federated results. The distinction between data integration and instance (or data) translation is that, in a data integration process, there is no transformation of the data. A data integration application can be data querying without loading the data in a central repository [Euzenat & Shvaiko 2013].

2.2.7 Alignment evaluation

Evaluating ontology matching approaches is crucial to characterise and compare them. Such an evaluation can be performed on various dimensions. We call these the evaluation strategies and list them thereafter.

Tool-oriented This refers to the evaluation of the system performance in terms of run-time and memory usage. It is often performed over ontologies of different sizes and levels of expressiveness. Most Ontology Alignment Evaluation Initiative (OAEI) tracks include this kind of evaluation.

Controlled input Evaluation of the generated alignment given different (controlled) inputs. Such an evaluation was proposed for the GeoLink and Hydrography datasets of the OAEI Complex track [Thiéblin *et al.* 2018a]. Given a list of entities, the system should be able to find the correct (complex) construction involving these entities.

Output-oriented Evaluation of the output alignment. It can be intrinsic, extrinsic or task-oriented.

- **Intrinsic** The quality of an alignment can be measured based on its intrinsic characteristics. [Meilicke & Stuckenschmidt 2008] evaluates the logical coherence as marker of quality of an alignment. [Solimando *et al.* 2014a] considers that a good alignment should not violate the conservativity principle.
- **Extrinsic** An alignment can be evaluated with regard to a manually created reference alignment. Most OAEI tracks adopt this alignment evaluation strategy.
- **Task-oriented** The quality of an alignment can also be assessed regarding its suitability for a specific task or application. In the Ontology Alignment for Query Answering (OA4QA) task [Solimando *et al.* 2014b], alignments have been evaluated over a query rewriting scenario.

Following the definition of "benchmark" as a standard by which something can be measured or judged[7], an alignment **benchmark** is composed of a dataset and an evaluation system.

An **evaluation dataset** is a set of ontologies (with or without instances) with a common scope. Other resources can be added to the dataset such as input queries, an input alignment, *etc.* The evaluation dataset can also contain a reference alignment (for extrinsic evaluation), reference queries (for query rewriting task-oriented evaluation), *etc.*

An **evaluation system** implements an evaluation strategy and outputs a score for an evaluated alignment over an evaluation dataset.

For the moment, there is no automatic benchmark for output-oriented evaluation of complex alignments.

2.3 Conclusion

In this chapter, we have presented ontologies which represent knowledge and their implementation in the Semantic Web. The ontologies are heterogeneous and ontology alignments can make them interoperable. We defined the different types of alignments (simple, complex), of matching processes (pairwise, holistic, compound). The main alignment expression formats have been listed; many dedicated ontology alignment formats have been proposed so far but none of them has been officially adopted or endorsed by the W3C.

Complex ontology alignments extend the expressiveness of simple alignments and therefore help overcome the modelling heterogeneity between ontologies. Manually finding each complex correspondence is a fastidious task. That is why, in this thesis, we propose an automatic complex alignment generation approach.

With the development of a complex alignment generation approach comes its evaluation, as discussed in this chapter, there are many evaluation strategies. In this thesis, we also propose an automatic benchmark for complex alignment evaluation.

In the following chapter, we present existing works on automatic and semi-automatic approach for complex alignment generation, and on complex alignment evaluation.

[7] American Heritage® Dictionary of the English Language, Fifth Edition. S.v. "benchmark." Retrieved January 7 2019 from `https://www.thefreedictionary.com/benchmark`

Chapter 3

State of the art on complex alignment

Content

So far, the complex alignment generation approaches have been examined under generic perspectives which apply to simple and complex alignment generation. In this Chapter, we study their specificities. The work presented in this section was published in [Thiéblin *et al.* 2019].

First, we propose a classification of the complex alignment generation approaches which extends existing classifications in Section 3.1. This classification focuses on the specificities of complex alignment generation approaches. Following this classification, we review existing complex alignment generation techniques in ontology and schema matching in Section 3.2.

Evaluating matchers is an important task because it allows for their characterisation and their comparison to one another. With this in mind, we analyse the alignment evaluation initiatives in terms of dataset and evaluation strategies in Section 3.3.

3.1 Structure-based classification of complex alignment generation approaches

Ontology matching approaches have been classified in various surveys [Kalfoglou & Schorlemmer 2003, Noy 2004, De Bruijn *et al.* 2006, Otero-Cerdeira *et al.* 2015, Rahm & Bernstein 2001, Doan & Halevy 2005, Shvaiko & Euzenat 2005, Euzenat & Shvaiko 2013]. These classifications however do not address the specificities of the complex approaches. After presenting the main existing classifications of ontology matching approaches in Section 3.1.1, we introduce axes for the classification of complex alignment generation approaches in Section 3.1.2.

3.1.1 Classifications of ontology matching approaches

Euzenat and Shvaiko [Shvaiko & Euzenat 2005, Euzenat & Shvaiko 2013] define three matching dimensions: input, process and output which will be the guiding thread to present the classifications below. Most classifications so far focused on input and process dimensions [Rahm & Bernstein 2001, Shvaiko & Euzenat 2005, Euzenat & Shvaiko 2013, Noy 2004, Doan & Halevy 2005].

Regarding the input dimension, the *instance vs ontology* classification (called *instance vs schema* in [Rahm & Bernstein 2001]) divides the matchers into those which deal with information from the $\mathcal{TB}ox$ and those which deal with the $\mathcal{AB}ox$. [Rahm & Bernstein 2001] also consider as input the type of auxiliary information used by the approaches (thesaurii, *etc.*).

For the process dimension, [Rahm & Bernstein 2001] propose classification axes such as *element vs structure*, *linguistic vs constraint-based*. All of these classification axes are put together into a taxonomy.

This classification -[Rahm & Bernstein 2001]) has been developed and extended by [Shvaiko & Euzenat 2005, Euzenat & Shvaiko 2013]. For instance, they distinguish whether an input is considered syntactically or semantically by the approach. The two-way taxonomy ends in basic approach strategies (*e.g.*, string-based, model-based, formal resource-based).

The classification of schema matching techniques of [Doan & Halevy 2005] separates *rule-based* techniques from *learning-based* techniques. Considering both input and process dimensions, *rule-based* techniques only exploit schema-level information in specific rules while *learning-based techniques* may exploit data instance information with machine-learning or statistical analysis.

[Noy 2004] proposes two main categories of ontology matching approaches: in the first, the matching process is guided by a top-level ontology from which the source and target ontologies derive; in the second, the matching process uses heuristics or machine-learning techniques.

Regarding the output dimension of the matching approaches, [Rahm & Bernstein 2001] consider the output alignment arity as a characteristic of the approaches which could be integrated into its taxonomy.

In sum, among the ontology matching classifications so far, that of [Euzenat & Shvaiko 2013] is the most extensive (all the others can be represented in this classification). However, even if considered, the output dimension of the matching approaches is rarely a basis for classification, whereas it becomes of interest when considering complex correspondences.

More generally, the classifications of ontology matching cited above do not address the specificities of the complex alignment generation problem. The characteristics of the processes leading to the generation of complex correspondences need to be studied, in particular the kind of structure guiding the discovery of correspondences. The next section presents classification axes for complex ontology matching approaches.

3.1.2 Classification of complex alignment generation approaches

The specificities of the complex alignment generation approaches rely on their output and their process. These are the two axes of the proposed classification. In this section, the different types of output (types of correspondences) and the structures used in the process to guide the correspondence detection are presented (guiding structures).

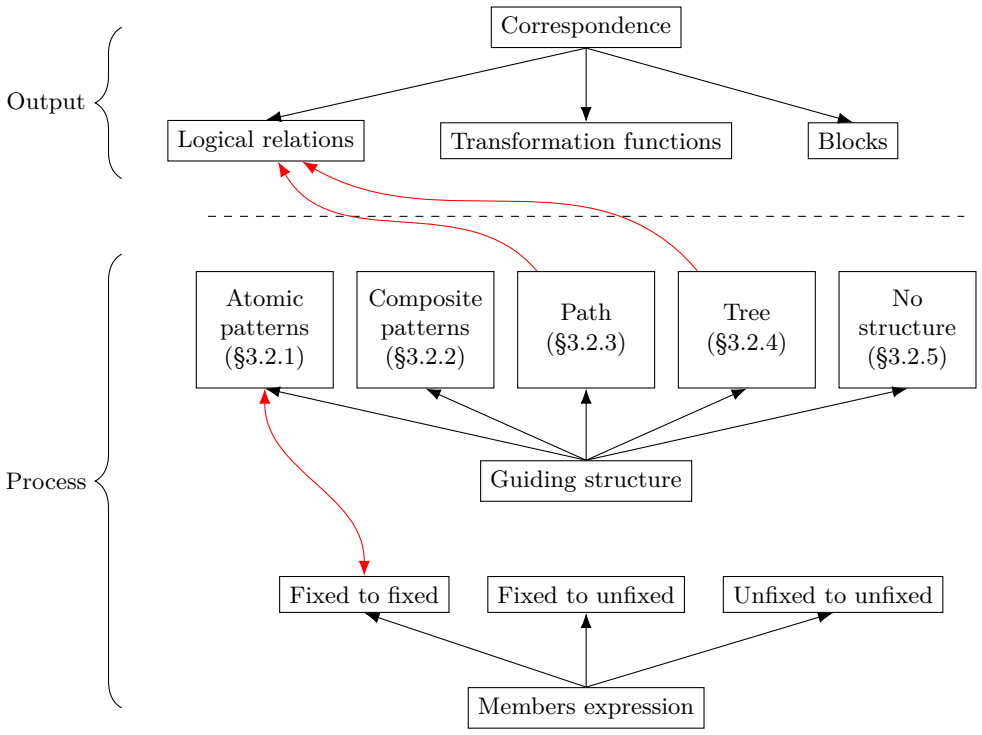


Figure 3.1: Two axes to characterise the complex alignment generation approaches: output and process. The subsumptions between the categories are represented with red arrows: *e.g.,* path and tree-based categories output logical relation correspondences.

Type of correspondence. The correspondences (output of the matching approaches) are divided into three main categories according to their type: *logical relations*, *transformation functions* and *blocks*. The **logical relations** category stands for correspondences in which complex members are expressed with logical constructors only. In contrast, the **transformation functions** category includes the approaches that generate correspondences with *transformation functions* in its members. The **blocks** correspondences gather entities using a grouping constructor in their members (clusters of entities), not specifying a semantic relation between them. For example, consider the following correspondences:

1. $\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}accepted.\{true\}\ ,\ \equiv\ \rangle$
2. $\langle\ o_1{:}priceInDollars\ ,\ changeRate(o_2{:}priceInEuros)\ ,\ \equiv\ \rangle$
3. $\langle\ \{o_1{:}Paper, o_1{:}Person\}\ ,\ \{o_2{:}Paper, o_2{:}Person\}\ ,\ \equiv\ \rangle$

Correspondence 1 is a logical relation correspondence, correspondence 2 is a transformation function correspondence and correspondence 3 is a block correspondence. No precise relation is specified between the entities involved in the third correspondence. It cannot therefore be classified as logical relation or transformation function correspondence. Note that in theory, a correspondence could have members expressed with transformation functions combined with logical constructors but no approach able to generate such a kind of correspondence was found. However, some approaches are able to generate both types independent of each other. An example of this correspondence expressed would be: $\langle\ o_1{:}Paper \sqcap o_1{:}priceInDollars\ ,\ o_2{:}Paper \sqcap changeRate(o_2{:}priceInEuros)\ ,\ \equiv\ \rangle$.

Guiding structures. These categories aim at classifying the (complex) matching approaches based on their process dimension. It focuses on the structure on which the process generating the correspondences relies:

- **Atomic patterns** The approaches in this category consider the correspondence as an instantiation of an atomic pattern, such as those defined by Scharffe [Scharffe 2009]. An atomic pattern is a template of a correspondence. A template can represent logical relation or transformation function correspondences. For example, an approach looking for correspondences following this exact pattern: $\langle\ o_1{:}A\ ,\ \exists o_2{:}b.o_2{:}C\ ,\ \equiv\ \rangle$ falls into this category and in the *logical relation* type of correspondence. An approach searching for $\langle\ o_1{:}a\ +\ o_1{:}b\ ,\ o_2{:}c\ ,\ \equiv\ \rangle$ falls into this category and in the *transformation function* type of correspondence.
- **Composite patterns** The approaches in this category aim at finding repetitive compositions of an atomic pattern. As for the atomic patterns, the composite patterns can represent both logical relation and transformation function correspondence patterns. For example, an approach looking for correspondences of the form $\langle\ o_1{:}A\ ,\ o_2{:}B \sqcup o_2{:}C \sqcup o_2{:}D \sqcup \ldots\ ,\ \equiv\ \rangle$, where $o_1{:}A$,

o_2*:B, o_2:C, o_2:D, etc.* are classes and the number of unions in the target member of correspondences is not *a-priori* defined by the approach, falls into this category. Correspondences representing string concatenation of an unlimited number of properties also fall into this category and in the *transformation function* type of correspondence.

- **Path** The approaches in this category detect the correspondences using path-finding algorithms. The resulting correspondence is a property path in o_1 put in relation with a path in o_2. For example, an approach looking for a path between two pairs of aligned instances described by o_1 resp. o_2 falls into this category.

- **Tree** The approaches in this category rely on tree structures inside the ontologies for correspondence detection. The ontologies are either considered as a tree or a tree-like structure is sought in an ontology graph. For example, when an XML schema is considered as a tree and the approach consists of finding the smallest equivalent tree in an ontology.

- **No structure** Contrary to the other approaches, the approaches of this category do not rely on a structure to guide the correspondence generation. Instead, they discover correspondences more freely.

The structures are used to guide the matching process, and therefore impact the structure of the output correspondences. However, a given correspondence, for example $\langle$ o_1*:AcceptedPaper* , $\exists o_2$*:acceptedBy.*$\top$, $\equiv$ $\rangle$, could be obtained by an approach based on atomic patterns with the pattern $\langle$ A , $\exists b.\top$, $\equiv$ $\rangle$, by an approach based on composite patterns such as $\langle$ A , $\exists b.\top \sqcup \exists c.\top \sqcup \ldots$, $\equiv$ $\rangle$ or by an approach with no guiding structure.

We chose to divide the approaches relying on patterns into two categories: atomic patterns and on composite patterns. The approaches in the latter category look for a repetition of a smaller pattern.

The *member expression pre-definition* specifies whether one of the members of the correspondence is assigned a fixed structure or not before the process. Three types of pre-definition are possible: fixed to fixed, fixed to unfixed and unfixed to unfixed.

- The **fixed to fixed** category includes the matching approaches that always produce correspondences with fixed member expressions. Atomic pattern-based approaches generate fixed to fixed correspondences as both members' expressions are defined by the pattern. As shown in Figure 3.1, this category is strongly correlated to the Atomic-pattern guiding structure category.

- The **fixed to unfixed** member expression category covers the matching approaches for which one of the members of the correspondence will always follow the same expression template, while the expression of the other member may vary. For example, an approach aiming at finding for each property

of an ontology a corresponding property path in the other ontology falls into this category: one of the members will always be one property while the other will be a path of a-priori undefined length.

- The **unfixed to unfixed** member expression category includes the approaches that output correspondences whose members have an undefined expression beforehand. For example, an approach aiming at finding similar paths in two ontologies falls into this category: both members have a-priori undefined length.

A matching approach can exploit many different matching strategies to find complex correspondences. In the following, the matching strategies are classified on their guiding structure. Therefore, the same approach can appear in multiple sections.

Some correlations can be noted as depicted in Figure 3.1: a path or tree-based approach will only output logical correspondences. There is also an equivalence between the *fixed to fixed* category and the *atomic pattern* category.

The choice of this guiding structure-based classification was made because guiding structures specific to complex alignment generation. Not only do they guide the matching process, but the correspondence structure derives directly from them. Other classifications were considered before this choice:

- A classification per type of knowledge representation model but it would not show the similarities between the matching systems even though they do not deal with the same type of knowledge representation model;
- A classification per type of correspondence output but this was not structuring enough;
- The classification from [Euzenat & Shvaiko 2013] but most complex alignment generation approaches combine many of those basic matching techniques;
- A classification per type of entity (concepts, properties, *etc.*) dealt with by the matchers but this was not specific to complex alignment.

In some way, the structure-based classification can be considered as a specialisation of the graph-based techniques category in the classification of [Euzenat & Shvaiko 2013].

3.2 Complex alignment generation approaches

The following sections present the approaches according to our classification. Although these sections are organised according to the guiding structure (Figure 3.1), a reference to the kind of output and member expression pre-definition is made in the text. The approaches are detailed in paragraphs with titles following a template: *Name [ref] Type of knowledge representation models, [(s:c), (c:s), (c:c)].*

3.2.1 Atomic patterns

Table 3.1: Atomic patterns used in the presented approaches. A, C are classes, a, b, c are properties, V is a value (instance or literal)

Name	Form	Example
Class by attribute type (CAT)	$\langle\ o_1{:}A\ ,\ \exists o_2{:}b.o_2{:}C\ ,\ \equiv\ \rangle$	$\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}hasDecision.o_2{:}Acceptance\ ,\ \equiv\ \rangle$
Class by attribute inverse type (CIAT)	$\langle\ o_1{:}A\ ,\ o_2{:}C \sqcap \exists o_2{:}b.\top\ ,\ \equiv\ \rangle$	$\langle\ o_1{:}AcceptedPaper\ ,\ o_2{:}Paper \sqcap \exists o_2{:}hasAcceptance.\top\ ,\ \equiv\ \rangle$
Class by attribute value (CAV)	$\langle\ o_1{:}A\ ,\ \exists o_2{:}b.\{V\}\ ,\ \equiv\ \rangle$	$\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}accepted.\{true\}\ ,\ \equiv\ \rangle$
Class by attribute existence (CAE)	$\langle\ o_1{:}A\ ,\ \exists o_2{:}b.\top\ ,\ \equiv\ \rangle$	$\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}acceptedBy.\top\ ,\ \equiv\ \rangle$
Property chain (PC)	$\langle\ o_1{:}a\ ,\ o_2{:}b \circ o_2{:}c\ ,\ \equiv\ \rangle$	$\langle\ o_1{:}reviewedBy\ ,\ o_2{:}hasReview \circ o_2{:}reviewWrittenBy\ ,\ \equiv\ \rangle$
Inverse Property (IP)	$\langle\ o_1{:}a^-\ ,\ o_2{:}b\ ,\ \sqsupseteq\ \rangle$	$\langle\ o_1{:}write^-\ ,\ o_2{:}writtenBy\ ,\ \sqsupseteq\ \rangle$
Class Intersection	$\langle\ o_1{:}A\ ,\ o_2{:}B \sqcap o_2{:}C\ ,\ \equiv\ \rangle$	$\langle\ o_1{:}AuthorAndReviewer\ ,\ o_2{:}Author \sqcap o_2{:}Reviewer\ ,\ \equiv\ \rangle$

Atomic patterns are used in approaches to detect logical relations as well as transformation functions. Table 3.1 presents several atomic correspondence patterns. Table 3.2 shows the atomic patterns of the correspondences which guide

Table 3.2: Atomic patterns per approach

Work	*Patterns*
[Ritze *et al.* 2009]	Class by Attribute Type, Class by Inverse Attribute Type, Class by Attribute Value, Property Chain
[Ritze *et al.* 2010]	Inverse Property, Class by Attribute Type, Class by Inverse Attribute Type, Class by Attribute Value
AMLC [Faria *et al.* 2018]	Class by Attribute Type, Class by Attribute Existence
[Oliveira & Pesquita 2018]	Class Intersection
[Rouces *et al.* 2016]	Linguistic patterns of FrameBase
Bayes-ReCCE [Walshe *et al.* 2016]	Class by Attribute Value
OAT [Chondrogiannis *et al.* 2014]	Combinations of predefined expressions
KAOM [Jiang *et al.* 2016]	Linear Regression
iMAP [Dhamankar *et al.* 2004]	Conversion functions predefined, basic arithmetic properties
BootOX [Jiménez-Ruiz *et al.* 2015]	RDB schema properties to OWL axioms

the state-of-the-art approaches of this category.

The atomic pattern-based approaches have different strategies for the definition of their patterns. For instance, some rely on the patterns defined by one of the ontologies to align [Rouces *et al.* 2016], other approaches have their own pattern library [Walshe *et al.* 2016, Dhamankar *et al.* 2004, Jiang *et al.* 2016, Ritze *et al.* 2009, Jiménez-Ruiz *et al.* 2015, Chondrogiannis *et al.* 2014, Faria *et al.* 2018]. Two main detection techniques appear: structuro-

linguistic conditions (called *matching patterns* defined in [Šváb Zamazal 2010]) [Ritze *et al.* 2009, Ritze *et al.* 2010, Rouces *et al.* 2016, Jiménez-Ruiz *et al.* 2015, Chondrogiannis *et al.* 2014, Faria *et al.* 2018], and statistical measures [Walshe *et al.* 2016, Jiang *et al.* 2016, Dhamankar *et al.* 2004]. These approaches are detailed below.

Ritze *et al.* [Ritze *et al.* 2009, Ritze *et al.* 2010] *OWL ontology to OWL ontology, (s:c)* In [Ritze *et al.* 2009, Ritze *et al.* 2010], Ritze *et al.* propose a set of matching conditions to detect correspondence patterns: *Class by Attribute Value*, *Class by Attribute Type*, *Class by Inverse Attribute Type*, *Inverse Properties* and *Property Chain* defined by Scharffe [Scharffe 2009] (*cf.* Table 3.1). The conditions are based on the labels of the ontology entities, the structures of these ontologies and the compatibility of the data-types of data-properties. The matching conditions to detect these patterns are an input to the matching algorithm. The user can add new matching conditions to detect other patterns.

The first approach[1] [Ritze *et al.* 2009] detects the modifier and head-noun of a label. In the matching conditions, string similarity (Levenshtein distance) is used to detect a potential relation between two entities (*e.g.*, *Acceptance* is similar to *Accepted*). The second version[2] of the matching conditions [Ritze *et al.* 2010] refines the syntactic part of the previous work by introducing linguistic analysis such as detection of antonymy, active form. Various linguistic analysis features are studied and incorporated in the matching conditions. In Example 1, the simplified matching conditions to detect correspondences fitting the inverse property pattern (see Table 3.1) states that if the verb phrase of the label of a source property o_1*:a* is the active voice of the verb phrase of a label of a target property o_1*:b, then* $\langle\ o_1$*:a*$^-$ *,* o_2*:b* $, \sqsubseteq\ \rangle$ *is a probable correspondence.*

Example 1 *Conditions:* $\langle\ o_1$*:a*$^-$ *,* o_2*:b* $, \sqsubseteq\ \rangle$ *iff verb(*o_1*:a) = active-voice (verb(*o_2*:b))*
Correspondence: $\langle\ o_1$*:writePaper*$^-$ *,* o_2*:writtenBy* $, \sqsubseteq\ \rangle$
because "write" is the active-form of "written"

The structural matching conditions are the same for both approaches. Example 1 is extended with structural constraints on the range and domain of o_1*:a (e.g.,* o_1*:write)* and o_2*:b (e.g.,* o_2*:writtenBy)* : the domain of o_1*:a (e.g.,* o_1*:Person)* should be subsumed by the range of o_2*:b (e.g.,* o_2*:Person)* and the range of o_1*:a* (*e.g.*, o_1*:Paper*) should be subsumed by the domain of o_2*:b* (*e.g.*, o_2*:Document*). The subsumption between ranges and domains of the two properties can be detected by inference on the ontologies' structure linked by the simple reference alignment or by a hypernymy relation between the labels. In the example, the necessary subsumptions are: $\langle\ o_1$*:Person* $, o_2$*:Person* $, \sqsubseteq\ \rangle$ and $\langle\ o_1$*:Paper* $, o_2$*:Document* $, \sqsubseteq\ \rangle$.

[1] `http://dominique-ritze.de/complex-mappings/`
[2] `https://code.google.com/archive/p/generatingcomplexalignments/downloads/`

AMLC [Faria *et al.* 2018] *OWL ontology to OWL ontology, (s:c)* AMLC (Complex AgreementMakerLight) is the complex version of the AML (AgreementMakerLight) system. It relies on lexical similarity and structural conditions to detect correspondence patterns. This approach is similar to that in [Ritze *et al.* 2009]. Two types of patterns are sought: Class by Attribute Existence and Class by Attribute Type (*cf.* Table 3.1).

Oliveira and Pesquita [Oliveira & Pesquita 2018] *OWL ontology to OWL ontology, (s:c)* The approach proposed in [Oliveira & Pesquita 2018] looks for *compound* correspondences which in their target member involve entities from more than one ontology. The sought correspondences follow the pattern $\langle\ o_1{:}A\ ,\ o_2{:}B \sqcap o_3{:}C\ ,\ \equiv\ \rangle$ in which $o_1{:}A$, $o_2{:}B$ and $o_3{:}C$ are classes from a source ontology o_1, and two target ontologies o_2 and o_3. The approach is based on a similarity measure between the labels of the source and target classes. In a first step, the source classes are aligned to the classes of a first target ontology (*e.g.*, o_2). Each of these correspondences is given a similarity score based on how the labels of the target classes overlap with the label of the source class. The correspondences are filtered over this similarity. The labels of the source class are reduced to the difference between the source and target classes' labels from the previously obtained correspondence. Finally, the source-reduced labels are matched with those of the second target ontology (*e.g.*, o_3) based on how this new label allows for the covering of the total source label.

Example 2 *A source class $o_1{:}AuthorAndReviewer$ with the label "author and reviewer" is first aligned to $o_2{:}Author$ which has the "author" label. The label of the source class is then reduced to "and reviewer" because of the correspondence in the previous step. In the last step, $o_3{:}Reviewer$ with the label "reviewer" is added to the correspondence because its label provides a good coverage of the reduced label "and reviewer". The output correspondence is: $\langle\ o_1{:}AuthorAndReviewer\ ,\ o_2{:}Author \sqcap o_3{:}Reviewer\ ,\ \equiv\ \rangle$*

Rouces *et al.* [Rouces *et al.* 2016] *OWL ontology to the FrameBase ontology (OWL), (s:c) (c:s)* Rouces *et al.* use FrameBase as a mediator ontology for complex alignment discovery. FrameBase is an ontology based on linguistic frames, seen as linguistic patterns in this approach. The approach identifies complex patterns in FrameBase from the linguistic patterns it describes. For each complex pattern identified, a corresponding *candidate property* is created (see Example 3). The names of the properties of the source ontology (the one to be aligned to FrameBase) are preprocessed, for example $o_1{:}birthDate$ becomes $o_1{:}hasBirthDate$. The properties of the source ontology are then aligned with simple alignments to the candidate properties created in FrameBase. The similarity of two properties is calculated based on a bag of words cosine from the tokenised property names. Once a source ontology property has been aligned to a created property of FrameBase, it is aligned to its corresponding pattern. The originality of this approach is that the correspondence

patterns on which it relies are encoded in one of the aligned ontologies (FrameBase). This approach is used in the Klint tool [Rouces *et al.* 2018] which provides a graphical interface for correspondence edition.

Example 3 *Created property: frame:hasBirthDate(s,o)*
Pattern:frame:BirthEvent(e) ∧ frame:hasSubject(e,s) ∧ frame:hasDate(e,o)
Source property preprocessing:
o_1*:birthDate* $\rightarrow$ o_1*:hasBirthDate*
Simple correspondence:
$\langle$ o_1*:hasBirthDate , frame:hasBirthDate ,* $\equiv \rangle$
Correspondence: $\langle$ o_1*:birthDate , frame:hasSubject*$^{-}$ $\circ$ *(dom(frame:BirthEvent)* $\sqcap$ *frame:hasDate) ,* $\equiv \rangle$

Bayes-ReCCE [Walshe *et al.* 2016] *OWL ontology to OWL ontology, (s:c)* This approach detects *Class Attribute Value Restrictions* correspondences. Bayes-ReCCE uses the properties of matched instances of two classes o_1*:AcceptedPaper* and o_3*:Paper*, with $\langle$ o_1*:AcceptedPaper ,* o_3*:Paper ,* $\sqsubseteq \rangle$ in a reference alignment. The matching problem is transformed into the feature-selection problem. The common instances are represented as binary vectors, each feature of the vector represents the presence of an *attribute-value* pair for a given instance. Feature-selection is the process of reducing the search space of features (here *attribute-value* pairs) to keep only relevant features for a model (here a classification). A score is given to each feature. Two metrics are used in the scoring process: information gain (with a closed-world assumption) and beta-binomial class prediction metric based on Bayesian probabilities (compliant with the open-world assumption). For each class, the top-k best features are returned to the user to choose from.

Example 4 *A reference alignment between* o_1 *and* o_3 *contains the correspondence* $\langle$ o_1*:AcceptedPaper ,* o_3*:Paper ,* $\sqsubseteq \rangle$*. The common instances of* o_1 *and* o_3 *described by* o_1*:AcceptedPaper in* o_1 *and* o_3*:Paper in* o_3 *are retrieved. The set of attribute-value pairs of each common instance is retrieved and becomes a feature in the feature-selection algorithm. If the attribute-value pair: (*o_3*:accepted,true) is selected by the algorithm, the correspondence* $\langle$ *(*o_1*:AcceptedPaper ,* $\exists o_3$*:accepted.*$\{true\}$ *,* $\equiv \rangle$ *is output.*

OAT [Chondrogiannis *et al.* 2014] *OWL ontology to OWL ontology, (s:c), (c:s), (c:c)* The Ontologies Alignment Tool (OAT) presented in [Chondrogiannis *et al.* 2014] is a semi-automatic complex alignment generation approach. The user can input correspondences through a graphical interface by instantiating correspondence patterns. For each of the two ontologies, the automatic matcher creates a set of expressions following a list of patterns (object property range restriction, inverse property, *etc.*). These expressions from the two ontologies are then compared by their entities' labels. If the similarity between two expressions is above a threshold, a correspondence putting these two expressions

together is suggested to the user who can validate or invalidate it. The confidence of the correspondence is then set to respectively 1 or 0 and propagated to the other correspondences. For example, the system finds that the domain restriction $dom(o_1{:}Paper) \sqcap o_1{:}hasAuthor$ is similar to the single property $o_2{:}writtenBy$, the following correspondence is output: $\langle\ dom(o_1{:}Paper) \sqcap o_1{:}hasAuthor\ ,\ o_2{:}writtenBy\ ,\ \equiv\ \rangle$. The system confidence associated with this correspondence is a weighted average of the similarity (or system confidence value) of the properties ($o_1{:}hasAuthor$, $o_2{:}writtenBy$), of their respective domains (or domain restrictions) ($o_1{:}Paper$, $o_2{:}Paper$) and ranges ($o_1{:}Author$, $o_2{:}Paper\_Author$). For example, the initial system confidence of $\langle\ o_1{:}Author\ ,\ o_2{:}Paper\_Author\ ,\ \equiv\ \rangle$ is 0.6. If the user validates this correspondence, it becomes 1. Then, the system confidence of $\langle\ dom(o_1{:}Paper) \sqcap o_1{:}hasAuthor\ ,\ o_2{:}writtenBy\ ,\ \equiv\ \rangle$ is updated to take into account this new range confidence value. The user can also manually add new correspondences.

iMAP [Dhamankar *et al.* 2004] *Relational database schema to relational database schema, (c:s)* The iMAP system [Dhamankar *et al.* 2004] uses a set of *searchers* to discover simple and complex correspondences between relational database schemata. The validity of each correspondence is then checked by a similarity estimator based on the columns' name similarity and a Naive-Bayes classifier trained on the target data. The correspondences are finally presented to a user who validates or invalidates them. Each searcher implements a specific strategy. Some of the searchers use atomic patterns for correspondence detection. For instance, the numeric, category and schema mismatch searchers look for correspondences fitting given atomic patterns. The patterns of the numeric searcher are equation templates given by the user or from previous matches. The category correspondence looks for equivalent attribute-value pairs for attributes having a small set of possible values. The schema mismatch searcher looks for correspondences in which an attribute of the source schema has a *true* value if it appears in a list of attributes in the target schema. Examples of category and schema mismatch correspondences are presented in Example 5. These searchers base their confidence in a correspondence on the data value distribution using the Kullback-Leibler divergence measure. The unit conversion searcher is based on string recognition rules in the attributes' names and data (such as "$", "hour", "kg", *etc.*). The searcher finds the best match function from a predefined set of conversion functions.

Example 5 *Category searcher correspondence between schemata describing papers and their acceptance status:*
$\langle\ \exists\ s_1{:}accepted.\{true\}\ ,\ \exists\ s_2{:}accepted.\{1\}\ ,\ \equiv\ \rangle$
Schema mismatch correspondence between schemata describing a conference participant status:
$\langle\ \exists s_1{:}actions.\{early\text{-}registration\}\ ,\ \exists s_2{:}early\text{-}registration.\{true\}\ ,\ \equiv\ \rangle$
This correspondence means that the target attribute s_2:early-registration is assigned a "true" value if "early-registration" appears in the list of the participant's actions

from the source schema.

KAOM [Jiang *et al.* 2016] *OWL ontology to OWL ontology, (s:c) (c:s) (c:c)* KAOM generates transformation function correspondences and logical relation correspondences. As the iMap's system [Dhamankar *et al.* 2004], KAOM implements different matching strategies: one for detecting transformation function correspondences, the other for logical relation correspondences. Here we present its transformation function correspondence detection approach, as it uses an atomic pattern. The logical relation correspondence approach is presented in Section 3.2.5. The atomic pattern used is a positive linear transformation function between numerical data properties $o_1{:}a$ and $o_2{:}b$ of respectively o_1 and o_2. A Kullback-Leibler divergence measure on the data values is used to define the coefficient $coeff$ of the linear transformation.

BootOX [Jiménez-Ruiz *et al.* 2015] *Relational database schema to OWL ontology, (c:s)* The BootOX approach [Jiménez-Ruiz *et al.* 2015] produces correspondences between a relational database schema and a target ontology via the creation of a "bootstrapped" ontology. The approach proceeds in two phases. In the first phase, an ontology is bootstrapped (created/extracted) from a relational database schema based on a set of patterns. For example, a non-binary relation table in the source schema produces a class in the bootstrapped ontology. The patterns used in this approach lead to the creation of axioms involving class restrictions in the bootstrapped ontology. R2RML correspondences between the relational database and its bootstrapped ontology are the result of this phase. This bootstrapped ontology is then aligned with the LogMap [Jiménez-Ruiz & Grau 2011] matcher to the target ontology. LogMap relies on linguistic and structural information to perform the matching. Put together, the transformation rules from RDB to ontology and the Logmap ontology alignment form a complex alignment between the RDB and the target ontology.

Other systems can bootstrap ontologies from relational database schemata [de Medeiros *et al.* 2015, Calvanese *et al.* 2017] but their aim is not to align the schema to an existing ontology. They are therefore out of the scope of this study. In this survey, BootOX is considered with its LogMap extension.

3.2.2 Composite patterns

Composite pattern-based approaches often focus on one or two patterns. Table 3.3 presents the different composite patterns detected by the approaches.

Some approaches iteratively construct the member(s) of the correspondence [Parundekar *et al.* 2012, Doan *et al.* 2003, Kaabi & Gargouri 2012, Warren & Tompa 2006, Dhamankar *et al.* 2004] (text searcher of iMap). Others first discover atomic pattern correspondences and merge them in a final (non-iterative) step [Parundekar *et al.* 2010, Arnold 2013].

Table 3.3: Composite patterns per approach. *A, B, C* are classes, *a, b, c, d* are properties, v_1, v_2, v_3, v_4 are values (instances or literals)

Work	*Composite pattern*	*Pattern form*
[Parundekar *et al.* 2012]	Disjunction of attribute-value pairs	$\langle\ \exists o_1{:}a.\{v_1\}\ ,\ \exists o_2{:}b.\{v_2,v_3,\dots\}\ ,\ \equiv\ \rangle$
[Parundekar *et al.* 2010]	Conjunction of attribute-value pairs	$\langle\ \exists o_1{:}a.\{v_1\} \sqcap \exists o_1{:}b.\{v_2\} \sqcap \dots\ ,\ \exists o_2{:}c.\{v_3\} \sqcap \exists o_2{:}d.\{v_4\} \sqcap \dots\ ,\ \equiv\ \rangle$
CGLUE [Doan *et al.* 2003]	Class unions	$\langle\ o_1{:}A\ ,\ o_2{:}B \sqcup o_2{:}C \sqcup \dots\ ,\ \equiv\ \rangle$
ARCMA [Kaabi & Gargouri 2012]	Class intersection	$\langle\ o_1{:}A\ ,\ o_2{:}B \sqcap o_2{:}C \sqcap \dots\ ,\ \sqsubseteq\ \rangle$
[Boukottaya & Vanoirbeek 2005]	String concatenation Subset merging	$\langle\ o_1{:}a\ ,\ \mathit{concatenation}(o_2{:}b,\ o_2{:}c,\dots)\ ,\ \equiv\ \rangle$ $\langle\ o_1{:}a\ ,\ o_2{:}b \sqcup o_2{:}c \sqcup \dots\ ,\ \equiv\ \rangle$
iMAP [Dhamankar *et al.* 2004]	String concatenation	$\langle\ o_1{:}a\ ,\ \mathit{concatenation}(o_2{:}b,\ o_2{:}c,\dots)\ ,\ \equiv\ \rangle$
Xu2003 [Xu & Embley 2003]	String concatenation Subset merging	$\langle\ o_1{:}a\ ,\ \mathit{concatenation}(o_2{:}b,\ o_2{:}c,\ \dots)\ ,\ \equiv\ \rangle$ $\langle\ o_1{:}a\ ,\ o_2{:}b \sqcup o_2{:}c \sqcup \dots\ ,\ \equiv\ \rangle$
Xu2006 [Xu & Embley 2006]	String concatenation Subset merging	$\langle\ o_1{:}a\ ,\ \mathit{concatenation}(o_2{:}b,\ o_2{:}c,\ \dots)\ ,\ \equiv\ \rangle$ $\langle\ o_1{:}a\ ,\ o_2{:}b \sqcup o_2{:}c \sqcup \dots\ ,\ \equiv\ \rangle$
[Warren & Tompa 2006]	String concatenation of attribute substrings	$\langle\ o_1{:}a\ ,\ \mathit{concatenation}(\mathit{substr}(o_2{:}b),\ \mathit{substr}(o_2{:}c),\dots)\ ,\ \equiv\ \rangle$
COMA++ [Arnold 2013]	String concatenation	$\langle\ o_1{:}a\ ,\ \mathit{concatenation}(o_2{:}b,\ o_2{:}c,\ \dots)\ ,\ \equiv\ \rangle$
[Wu *et al.* 2004]	Annotated sets of properties (is-a or aggregate)	$\langle\ \{o_1{:}a\}\ ,\ \mathit{is\text{-}a}\{o_2{:}b,\ o_2{:}c,\ \dots\}\ ,\ \equiv\ \rangle$; $\langle\ \{o_1{:}a\}\ ,\ \mathit{aggregate}\{o_2{:}b,\ o_2{:}c,\ \dots\}\ ,\ \equiv\ \rangle$
[Šváb Zamazal & Svátek 2009]	N-ary to N-ary N-ary to object property	Structure matching, see Fig. 3.2
PORSCHE [Saleem *et al.* 2008]	Bag of properties/blocks	$\langle\ \{o_1{:}a\}\ ,\ \{o_2{:}b,\ o_2{:}c,\dots\}\ ,\ \equiv\ \rangle$
DCM [He *et al.* 2004]	Bag of properties/blocks	$\langle\ \{o_1{:}a\}\ ,\ \{o_2{:}b,\ o_2{:}c,\dots\}\ ,\ \equiv\ \rangle$
HSM [Su *et al.* 2006]	Bag of properties/blocks	$\langle\ \{o_1{:}a,o_1{:}b,\dots\}\ ,\ \{o_2{:}c,\ o_2{:}d,\dots\}\ ,\ \equiv\ \rangle$

Approaches use graph-pattern matching either as detection conditions [Saleem *et al.* 2008, Boukottaya & Vanoirbeek 2005, Wu *et al.* 2004, Šváb Zamazal & Svátek 2009] or over the properties of a mediating ontology [Xu & Embley 2003, Xu & Embley 2006, Dhamankar *et al.* 2004] (iMap's date searcher). Finally, [He *et al.* 2004, Su *et al.* 2006] start by grouping schema attributes before matching the groups. Even though the holistic approaches [He *et al.* 2004, Su *et al.* 2006] produce block correspondences (of properties only), it has been decided that these two approaches are composite pattern driven as the *grouping* phase follows a repetitive pattern. Some approaches search for composite patterns inside a tree structure [Saleem *et al.* 2008, Xu & Embley 2003, Xu & Embley 2006, Boukottaya & Vanoirbeek 2005]. These approaches could also be classified into the tree-based category. However, as their matching process relies on the identification of a composite pattern in those trees, they were classified in this category. In [Šváb Zamazal & Svátek 2009], the approach detects and matches N-ary relation reifications between ontologies. The N-ary relation contains a repetitive pattern, therefore [Šváb Zamazal & Svátek 2009] was classified in this category.

Parundekar *et al.* [**Parundekar *et al.* 2012**] *OWL ontology to OWL ontology, (s:c) (c:s)* In this approach proposed by Parundekar *et al.*

[Parundekar *et al.* 2012], the type of correspondences sought is an *attribute-value* pair matched with an attribute and a union of its acceptable values. In the first step, the approach finds correspondences between *attribute-value* pairs from the linked instances of the two ontologies (instances linked with *owl:sameAs* predicate). The number of instances sharing both attribute-value pairs defines whether the correspondence has a subsumption or equivalence relation. The second step is, for each subsumption correspondence of the previous step, to merge in a union all the attribute-pairs with a common attribute. The relation of the new correspondence is then re-evaluated according to the number of instances for each member. The following example shows the two-step approach.

Example 6 *First step output:*
$\langle\ \exists o_1$*:accepted.*$\{true\}\ ,\ \exists o_2$*:hasStatus.*$\{accepted\}\ ,\ \sqsupseteq\ \rangle$
$\langle\ \exists o_1$*:accepted.*$\{true\}\ ,\ \exists o_2$*:hasStatus.*$\{camera\text{-}ready\}\ ,\ \sqsupseteq\ \rangle$
Second step output:
$\langle\ \exists o_1$*:accepted.*$\{true\}$, $\exists o_2$*:hasStatus.*$\{accepted\ ,\ camera\text{-}ready\}\ ,\ \equiv\ \rangle$

Parundekar *et al.* [Parundekar *et al.* 2010] *OWL ontology to OWL ontology, (s:c) (c:s) (c:c)* Parundekar *et al.* [Parundekar *et al.* 2010] look for conjunctions of *attribute-value* pairs, for instance correspondences of the form $\langle\ \exists o_1$*:a.*$\{v_1\}\ \sqcap\ \exists o_1$*:b.*$\{v_2\}\ \sqcap\ \ldots\ ,\ \exists o_2$*:c.*$\{v_3\}\ \sqcap\ \exists o_2$*:d.*$\{v_4\}\ \sqcap\ \ldots\ ,\ \equiv\ \rangle$ with o_1*:a, *o_1*:b, *o_2*:c, *o_2*:d* properties and the $v_{i\in n}$ constant values: instances or literals. The approach starts with pre-processing the two knowledge-bases described by o_1 and o_2. Only the common instances are kept. Properties that cannot contribute to the alignment are manually removed (*i.e.*, properties from a different domain than the common scope of the ontologies and inverse functional properties). A set of first correspondences (the seed hypotheses) are created between *attribute-value* pairs. An example of a seed hypothesis is $\langle\ \exists o_1$*:hasDecision.*$\{accept\}\ ,\ \exists o_2$*:accepted.*$\{true\}\ ,\ \equiv\ \rangle$. Starting from these seed hypotheses, the approach implements a heuristic in-depth-first exploration of the search space (all the possible conjunctions of attribute-value pairs). The search space is considered as a tree, the root being a seed hypothesis. Each node is an extended version of its parent: an attribute-value pair is added to one member of the parent (*e.g.*, $\exists o_1$*:submitted.*$\{true\}$ has been added to the source member of the seed hypothesis: $\langle\ \exists o_1$*:hasDecision.*$\{accept\}\ \sqcap\ \exists o_1$*:submitted.*$\{true\}\ ,$ $\exists o_2$*:accepted.*$\{true\}\ ,\ \equiv\ \rangle$). The search-tree is pruned following rules based on the variation of instances described by each member. For example if the attribute-value added in a node is too restrictive or if the support of the ancestor node is the same as the current node, the children of the current node are not explored. The final set of correspondences is filtered to avoid redundancy. The number of instances of each member will determine the correspondence's relation.

CGLUE [Doan *et al.* 2003] *DL ontology to DL ontology, (s:c)* The GLUE system [Doan *et al.* 2003] is specialised in detecting (s:s) correspondences between ontologies' classes using machine learning techniques such as joint probability distri-

bution. CGLUE, also presented in [Doan *et al.* 2003], is an extension of the GLUE system. It can detect (s:c) class unions in class hierarchies such as $\langle$ o_1*:Document* , o_2*:Paper* $\sqcup$ o_2*:Poster* $\sqcup \ldots$, $\equiv$ $\rangle$. To detect these unions, the authors make the assumption that the subclasses of a class represent a partition of this class. To find a correspondence to a source class o_1*:Document*, each class-union of o_2 is considered a potential candidate. The first candidates are the set of single classes of o_2. An adapted beam search finds the k best candidates according to a similarity score given by the GLUE system. The k best candidates are then expanded as unions with the classes of o_2 until no improvement is obtained on the similarity score.

ARCMA [Kaabi & Gargouri 2012] *OWL ontology to OWL ontology, (s:c)* Kaabi *et* Gargouri [Kaabi & Gargouri 2012] propose ARCMA (Association Rules Complex Matching Approach) to find correspondences of the form $\langle$ o_1*:A* , o_2*:B* $\sqcap$ o_2*:C* $\sqcap \ldots$, $\sqsubseteq$ $\rangle$. A set of terms is associated with each class: the terms are extracted from the annotations, labels, instance values, instance labels of this class and its subclasses. The detection of the correspondences rely on existing simple correspondences: each class of the right member (*o_2:B, o_2:C, ...)* must be equivalent to a parent of o_1*:A*. The correspondences are then filtered based on a value measuring how the sets of terms of each member overlap. The following example presents how a correspondence is detected by this approach.

Example 7 *Let o_1:AuthorAndReviewer be a subclass of o_1:Author and o_1:Reviewer. Simple correspondences, between o_1 and o_2 are given:*

- $\langle$ o_1*:Author* , o_2*:Author* , $\equiv$ $\rangle$
- $\langle$ o_1*:Reviewer* , o_2*:Reviewer* , $\equiv$ $\rangle$

With the overlap of terms associated with o_1:AuthorAndReviewer and the terms of respectively o_2:Author and o_2:Reviewer, the following correspondence can be output: $\langle$ o_1*:AuthorAndReviewer* , o_2*:Author* $\sqcap$ o_2*:Reviewer* , $\sqsubseteq$ $\rangle$

Boukottaya and Vanoirbeek [Boukottaya & Vanoirbeek 2005] *XML schema to XML schema, (s:c) (c:s) (c:c)* Boukottaya *et* Vanoirbeek [Boukottaya & Vanoirbeek 2005] propose an XML schema matching approach based on the schema tree and linguistic layer of the schema. This approach finds simple and complex correspondences. The complex correspondences follow a few patterns such as merge/split, union/selection and join. The first step calculates a similarity between nodes of the source and target schemata. A linguistic similarity is calculated. A datatype similarity is then computed for the linguistically similar nodes. The union/selection and merge/split correspondences are detected based on graph-mapping. Union/selection correspondences are detected when nodes have a common abstract type (based on their WordNet similarity) which matches a node from the other schema. Merge/split are computed when a leaf node matches a non-leaf node. The correspondences are filtered based on their *structural context*:

ancestors and children nodes. The access path of each node is written in the final correspondences.

Example 8 *If a node o_1:address of the source schema with children leaf nodes (o_1:street, o_1:city) matches a leaf node o_2:address of the target schema, then a concatenation of the children nodes can be matched to the target node:*
$\langle$ *concatenation(o_1:street, o_1:city)* , o_2:address , $\equiv \rangle$

If two nodes o_1:Journal-Article and o_1:Conference-Article from the source ontology have a common abstract super node (computed from WordNet): Article and that o_2:Article matches this super node, a union pattern is detected:
$\langle$ o_1:Journal-Article $\sqcup$ o_1:Conference-Article , o_2:Article , $\equiv \rangle$

COMA++ [Arnold 2013] *Document-oriented schema to document-oriented schema, (s:c)* As an improvement to the COMA system [Maßmann *et al.* 2011], Arnold [Arnold 2013] discusses a solution based on a lexical strategy on the schemata attribute names: *several (s:s)* attribute correspondences with the same attribute as target (or source), could be merged into a complex one. The initial approach generates simple correspondences with expressive relations such as meronymy *part-of* or holonymy *has-a* besides usual relations ($\sqsupseteq$, $\sqsubseteq$, $\equiv$). The extension for transforming the simple correspondences into a complex one can take into account the type of attribute (*e.g.*, concatenation for a string attribute or sum for a numeric attribute). The following example shows a complex correspondence inferred from simple correspondences.

Example 9 *Part-of correspondences with same target member:*

- $\langle$ s_1:firstName , s_2:fullName , part-of $\rangle$
- $\langle$ s_1:lastName , s_2:fullName , part-of $\rangle$

Aggregation in a new correspondence:
$\langle$ *concatenation(s_1:firstName, s_1:lastName)* , s_2:fullName , $\equiv \rangle$

iMAP [Dhamankar *et al.* 2004] *Relational database schema to relational database schema, (c:s)* As seen in the previous section, the iMAP system [Dhamankar *et al.* 2004] uses a set of *searchers* to discover simple and complex correspondences between relational database schemata. Some of the searchers use composite patterns for correspondence detection. For instance, the text searcher looks for correspondences between an attribute from the target schema and concatenation of string attributes from the source schema. This searcher starts from ranking all possible simple correspondences between attributes. For this, a Naive-Bayes classifier is trained on the target data values to classify whether a given value can be from the target attribute. The average score given by this classifier to a correspondence is used for the ranking. Once the k best simple correspondences are selected, the process is reiterated but with combinations of concatenations of

the selected source attribute and other source attributes as base correspondences. These new correspondences are scored, selected, and so on.

Another searcher implements a composite pattern search: the date searcher. It uses a date ontology as mediating schema containing date concepts (*e.g.*, date, month, year) and the relations between them (*e.g.*, concatenation, subset). The attributes of each schema are matched to the date ontology's entities and the relations between them are reported as transformation functions in the resulting correspondence. The date ontology contains the composite patterns which are discovered by simple graph matching.

Xu and Embley [Xu & Embley 2003, Xu & Embley 2006] *Conceptual model to conceptual model, (s:c) (c:s)* Xu and Embley [Xu & Embley 2003] propose a similar approach to iMap's date matcher. It uses a user-specified domain ontology as mediator between the two conceptual models to be aligned. This ontology contains relations between concepts such as composition, subsumption, *etc.* It is populated thanks to regular expressions applied on source and target data. Simple correspondences (equivalence or subsumption) are first detected using *recognition of expected value* techniques between the source conceptual model (resp. target) entities and the ontology's concepts. These simple correspondences are kept for the next phase if the number of common values between the conceptual model entity and the ontology concept are above a threshold.

The relation between the ontology concepts in simple correspondences will become the transformation functions between the attributes they are linked to. For example, *s:street s:city* are two entities from the source conceptual model and *t:address* is an entity from the target conceptual model. In the first matching phase, simple correspondences are drawn with concepts from the mediating ontology *o*:

- $\langle$ *o:Address* , *t:address* , $\equiv \rangle$
- $\langle$ *o:Street* , *t:street* , $\equiv \rangle$
- $\langle$ *o:City* , *t:city* , $\equiv \rangle$

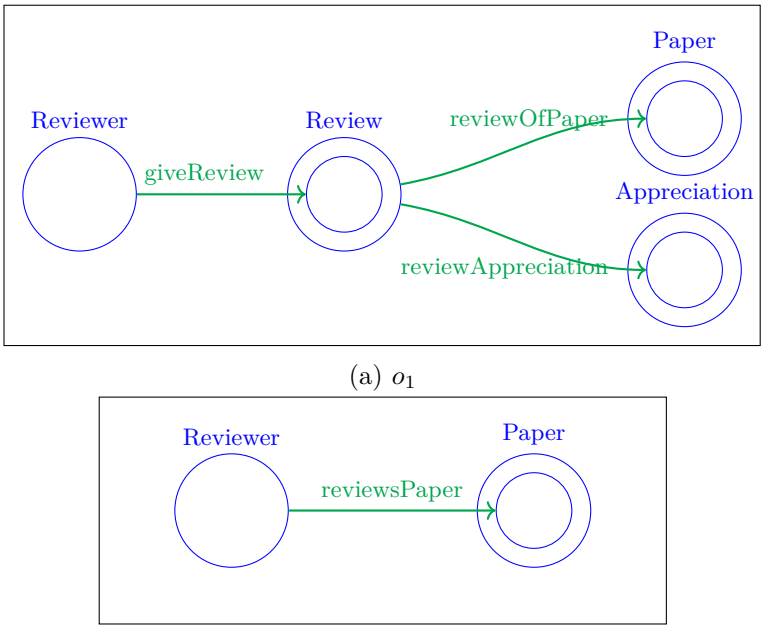
In *o*, the concept *o:Address* has a composition relation with the concepts *o:Street*, *o:City*. Therefore, the output complex correspondence will state that *t:address* is a string concatenation of *s:street* and *s:city*.

The later version of Xu and Embley's approach [Xu & Embley 2006] completes this work with two new confidence calculations for simple attribute matching. The two new calculations do not consider a mediating ontology.

Warren and Tompa [Warren & Tompa 2006] *Table schema to table schema, (c:s)* Warren and Tompa [Warren & Tompa 2006] focus on finding correspondences between string columns of tabular data. They deal with correspondences that translate a concatenation of column sub-strings. The approach starts by ranking the source columns according to the *q-grams* (sequence of *q* characters) of its values

found in the target column. Then it looks for matched instances (rows) according to a *tf-idf* formula on co-occurring q-grams. The source column that has the smallest editing distance from the target column is put in an initial translation rule. This translation rule is then iteratively refined with addition of sub-strings from other source columns.

Example 10 *A correspondence output by this approach could be:* $\langle$ *concatenation(substr(o_1:firstName,1),substr(o_1:lastName,6))* , *o_2:username* , $\equiv \rangle$, *with substr(x,n) a function giving the first n characters of the string x.*



(a) o_1

(b) o_2

Figure 3.2: N-ary relation pattern

Šváb-Zamazal and Svátek [Šváb Zamazal & Svátek 2009] *OWL ontology to OWL ontology, (s:c),(c:s),(c:c)* This approach is based on structural and naming conditions to detect N-ary relation patterns as defined by the Semantic Web Best Practice (SWBP)[3] in the aligned ontologies. First, reified N-ary relations are sought in the ontologies with the help of a lexico-structural pattern. The fragment of ontology represented in Figure 3.2(a) shows an N-ary relation between a reviewer, a paper and its review appreciation. This pattern consists in an intermediate concept (here o_1*:Review*) representing the relation between a domain o_1*:Reviewer* and N ranges o_1*:Appreciation*, o_1*:Paper*. Once the N-ary relations are detected in the source and target ontologies, a similarity measure is computed between the source and target patterns. This similarity is an aggregation of the label similarities of the concepts in the N-ary relations. If the similarity is above a threshold, a structure to

[3]https://www.w3.org/TR/swbp-n-aryRelations/

structure correspondence is created. The N-ary relations are also matched to object properties by comparing their labels and domain/range compatibility. Figure 3.2 shows an example of an N-ary relation (3.2(a)) and corresponding object properties (3.2(b)). The structure to structure correspondences cannot be interpreted.

PORSCHE [Saleem *et al.* 2008] *XML schema to XML schema, (s:c) (c:s)* PORSCHE (Performance ORiented SCHEma Matching) [Saleem *et al.* 2008] matches a set of XML *schema trees* (schemata with a single root) simultaneously. It is a holistic approach. This approach outputs a mediating schema (all the schemata merged) as well as correspondences from each source schema to the mediating schema. An initial mediating schema is chosen among the source schema trees. It is then extended by the approach. For each node of each schema, the approach tries to find a corresponding node in the mediating schema. The tokenised labels of the nodes are compared with the help of an abbreviation table. The context of a node is also taken into account for the merging, where the ancestors of the nodes must match. The pattern used for the detection of the complex correspondences is: if a non-leaf node (*e.g.*, s_1*:address*) is similar to a leaf node (*e.g.*, s_2*:address*), a (c:s) correspondence is created between the leaf node s_2*:address* and the leaf nodes descending from s_1*:address* (*e.g.*, s_1*:street*, s_1*:city*). The correspondences produced are coherent (leaves with leaves) but approximate. Indeed, the context of a node is not checked in the case of a (s:c) leaf-non-leaf correspondence. No transformation function is specified in the correspondence. They come as un-annotated sets of properties. For example, $\langle$ {s_1*:street,*s_1*:city*} , {s_2*:address*} , $\equiv \rangle$ could be an output correspondence.

The following two approaches are also holistic: they match many schemata simultaneously. They rely on Web query interfaces.

DCM [He *et al.* 2004] *Table schema to Table schema, (s:c) (c:s) (c:c)* DCM (Dual Correlation Mining) [He *et al.* 2004] is a holistic schema matching system. It aligns attribute names of Web Forms. It uses data-mining techniques (positive and negative correlation mining) on a corpus of Web query interfaces to discover complex correspondences. The approach uses attribute co-occurrence frequency as a feature for the correlation algorithm. The first step of the algorithm is to mine frequently co-occurring attributes from the Web query interfaces. These attributes are put together as *groups* (*e.g.*, {s_1*:firstName, *s_1*:lastName*}). In the second step, each set of co-occurring attributes (*e.g.*, {s_1*:firstName, *s_1*:lastName*}) is put in correspondence with sets of attributes which do not often co-occur with them (*e.g.*, {s_2*:author*}). The correspondences are then filtered according to their confidence (negative co-occurrence) value, or aggregated if they have a common attribute: if $\langle$ {s_1*:firstName, *s_1*:lastName*} , {s_2*:author*} , $\equiv \rangle$ and $\langle$ {s_2*:author*} , {s_3*:writer*} , $\equiv \rangle$, then $\langle$ {s_1 :*firstName, *s_1*:lastName*} , {s_2 :*author*} , {s_3 :*writer*} , $\equiv \rangle$. As this approach is holistic, the correspondences are not limited to two members.

A holistic approach reduces the bias of one-to-one schema matching as errors

can be overcome by the number of correct correlations mined. However, only the attributes present on the Web query interfaces can be involved in the correspondences.

HSM [Su *et al.* 2006] *Table schema to Table schema, (s:c) (c:s) (c:c)* HSM (Holistic Schema Matching) [Su *et al.* 2006] is similar to DCM [He *et al.* 2004] as it considers schema matching as a whole. It finds synonyms and grouping attributes based on their co-occurrence frequency and proximity in the Web query interfaces. Two scores are computed between attributes : synonym scores (the confidence that two fields may refer to the same concept or *thing*) and grouping scores (confidence that two concepts are complementary to one-another). The algorithm then goes through the synonym scores in decreasing order and adds new correspondences to the alignment. If an attribute is a synonym of an attribute already involved in a correspondence, it may be added to an existing group of attributes according to its grouping score with them.

Example 11 *The approach explores the synonyms: s_1:firstName is found to be a synonym of s_2:author. The following groups are formed:*
$\langle$ {s_1:*firstName*} , {s_2:*author*} , $\equiv$ $\rangle$
Then, s_1:lastName is found to be a synonym of s_2:author. Because s_1:lastName and s_1:firstName have a good grouping score, s_1:lastName is added to the correspondence as follows:
$\langle$ {s_1:*firstName, s_1:lastName*} , {s_2:*author*} , $\equiv$ $\rangle$

Wu *et al.* [Wu *et al.* 2004] *Table schema to table schema, (s:c) (c:s)* Wu *et al.* [Wu *et al.* 2004] propose a clustering approach to find attribute correspondences based on Web query interfaces. It considers the hierarchical structure of an HTML form. It also considers the values taken in the table rows as the *domain* of an attribute.

The first step consists in finding complex correspondences of the form *(s:c)* or *(c:s)* in which the attribute in the simple member is called the *singleton attribute* and the attributes in the complex member, the *grouped attributes*. Two types of correspondences are sought: *aggregate* and *is-a*. An *aggregate* correspondence shows a value concatenation: $\langle$ {*date*} , *aggregate*{*day,month,year*} , $\equiv$ $\rangle$. A *is-a* correspondence shows a union, sum, *etc.* of these values: $\langle$ {*passengers*} , *is-a*{*adults,children,seniors*} , $\equiv$ $\rangle$. The detection conditions of these correspondences are based on the hierarchy of the Web form attributes: the label of the parent node of the grouped attributes must be similar to that of the singleton attribute. For *is-a*, the grouped attributes' domains must be similar to the singleton's, whereas for *aggregate*, the domain of each grouped attribute must be similar to a subset of the singleton attribute's domain.

A clustering technique then computes simple correspondences in a holistic manner between the interfaces. Simple correspondences and preliminary complex cor-

respondences are merged. Other complex correspondences may be inferred from this merging phase. Even if the simple alignment generation process is holistic, the detection of the complex correspondences is made interface to interface. Thus, the output correspondences are schema to schema.

The final step of the approach is user refinement. The system asks the user questions to refine the alignment and tune the parameters of the clustering algorithm and similarity calculation.

3.2.3 Path

A specificity of the path-based approaches is that they all rely on simple correspondences (at instance or ontology level). Some of them discover these simple correspondences themselves as a preliminary step [Qin *et al.* 2007, Dou *et al.* 2010], others take them as input [An *et al.* 2012, Yan *et al.* 2001, Miller *et al.* 2000, An & Song 2008]. Most approaches perform the path search on the graph-like or tree-like structure of the schemata/ontologies directly whereas [An & Song 2008] creates a *mapping graph* on which the search will be performed.

An *et al.* [An *et al.* 2012] *Document-oriented schema to CML ontology, (s:c)* An *et al.* [An *et al.* 2012] map a Web query form to a CML ontology. The attribute and fieldset names of the form are transformed into a *form tree* (derived from HTML), similar to an XML schema tree. The algorithm takes the form tree, the ontology and simple correspondences between the form tree and the ontology as input. The first step of the algorithm is to find for each edge of the form tree between two nodes, all subgraphs G_i (as minimum spanning Steiner trees) in the ontology (*e.g.*, in a book related Web form an edge can link the node *s:book* to its sub-node *s:author*). The subgraphs are property chains in the target ontology between two nodes (classes) *o:Book* and *o:Author* such that $\langle$ *s:book , o:Book ,* $\equiv \rangle$ and $\langle$ *s:author , o:Author ,* $\equiv \rangle$ are two simple correspondences given in the input. The goal of the algorithm is to output the most (or k-most) probable subgraphs for the given form tree. To compute the probability of a subgraph given a form tree, a model is trained with machine learning techniques. The training corpus is composed of Web query interfaces annotated with the target ontology. The model is based on a Naive Bayesian approach and *m-estimate probabilities* to approximate the sub-graph probability given a form tree.

Clio [Miller *et al.* 2000, Yan *et al.* 2001] *Relational database schema to relational database schema, (s:c) (c:s) (c:c)* Based on structural information of relational database schemata, the Clio system[4] [Miller *et al.* 2000, Yan *et al.* 2001] is one of the first systems to consider the creation of complex correspondences between schemata. The user must input *value correspondences*: functions linking one or many attributes (*e.g.*, $\langle$ s_1*:Parent1.Salary +* s_1*:Parent2.Salary ,*

[4] `http://www.almaden.ibm.com/cs/projects/criollo/`

s_2*:Student.FamilyIncome , ≡* ⟩). Used for populating target schemata with source data, it provides the user with a framework for alignment creation. Clio discovers formal queries from these *value correspondences*. The formal queries are defined step-by-step with the user by presenting him or her with potential *query graphs* between attributes: trees from the data source schema structure. Clio helps the user find simple, path relation and value transformation correspondences with data visualisation, data walk and data chase. The alignments are automatically transformed into SQL queries. The SQL queries transform the source data into target schema. The user can refine and extend the alignments (queries) with filters and joins. The Clio system is user-oriented: the user intervenes at every step of the matching process. What Clio does automatically is find the path between the attributes and tables to complete the input *value correspondences*. It also automatically transforms the correspondences into SQL queries.

Ontograte [Qin *et al.* 2007, Dou *et al.* 2010] *OWL ontology to OWL ontology, (c:c)* OntoGrate [Qin *et al.* 2007] is a framework that mines *frequent queries* and outputs them as conjunctive first-order logic formulae. The system can deal with ontology matching [Qin *et al.* 2007] and was adapted to relational database schema matching in [Dou *et al.* 2010] by transforming the relational database schema into a *database ontology*. In OntoGrate, the first step of the matching algorithm is to *generate simple correspondences* at ontology level. An *object reconciliation* phase then aligns instances from source and target knowledge bases. The instance correspondences from the object reconciliation fuel the simple correspondence generation. The algorithm iterates on both steps (simple correspondence generation and object reconciliation) until no new instance correspondence or simple correspondence is discovered. Once the simple correspondences are found, a *group generator* process generates groups of entities closely related to a source property. The group generation is done by exploring the ontology graph and finding a path between entities (*e.g.*, classes) linked by a simple property/property correspondence (the property/property correspondence can be data-property/data-property or object-property/object-property). The path-finding algorithm is an exploration algorithm of the two ontology graphs where classes are the nodes and properties (object properties, data properties, subclass relations and super-class relations) are the edges. The ontology graphs are explored until two nodes, one in the source path and one in the target path, are found which were matched in the first steps of the matching process. The final step of the matching process is Multi-Relational Data Mining (MRDM) to retrieve *frequent queries* among the matched instances for the given *entity groups*. If the support of a query is above a threshold, the query is considered *frequent* and kept. The frequent queries are then refined and formalised into first-order logic formulae.

Example 12 *The simple alignment generation phase computed:*

- ⟨ o_1*:Person , *o_2*:Person , ≡* ⟩

- $\langle$ o_1:*email* , o_2:*contactEmail* , $\equiv$ $\rangle$

However, the last correspondence is wrong and it is considered incomplete because

- o_1:*Person is the domain of* o_1:*email*
- o_2:*Paper is the domain of* o_2:*contactEmail*

The group entity algorithm starts with the following entity groups:

- *source: {*o_1:*Person,* o_1:*email}*
- *target: {*o_2:*Paper,* o_2:*contactEmail}*

The process searches both ontologies so that two equivalent classes can be found in the groups: the o_2:*writes property and its domain* o_2:*Paper are added to the target group:*

- *source: {*o_1:*Person,* o_1:*email}*
- *target: {*o_2:*Person,* o_2:*writes,* o_2:*Paper,* o_2:*contactEmail}*

If the matched instances give the entity groups enough support, the following correspondence is output:
$\langle$ *dom(*o_1:*Person)* $\sqcap$ o_1:*email , (dom(*o_2:*Person)* $\sqcap$ o_2:*writes)* $\circ$ *(dom(*o_2:*Paper)* $\sqcap$ o_2:*contactEmail)* , $\equiv$ $\rangle$

An and Song [An & Song 2008] *CML ontology to CML ontology, (c:c)* An and Song [An & Song 2008] introduce the concept of mapping graph between two ontologies in CML language. This process relies on a simple alignment between the concepts of the ontologies. The first step of the approach is to generate the mapping graph between the ontologies. The nodes of a mapping graph represent pairs of concepts from the two ontologies. For example *(*o_1:*Reviewer,* o_2:*Reviewer) and (*o_1:*Paper,* o_2:*Paper) are two nodes of the mapping graph.* The weighted edges of the mapping graph are defined according to the presence and nature of the relations between the concerned concepts in the conceptual models. Once the mapping graph is generated, a Dijkstra algorithm is used to find the smallest path (with maximum weights) between nodes that appear in an input simple alignment. If the simple alignment states that $\langle$ o_1:*Reviewer* , o_2:*Reviewer* , $\equiv$ $\rangle$ and $\langle$ o_1:*Paper* , o_2:*Paper* , $\equiv$ $\rangle$, then the approach will look for a path between *(*o_1:*Reviewer,* o_2:*Reviewer) and (*o_1:*Paper,*o_2:*Paper).*

Example 13 *If* $\langle$ o_1:*Reviewer* , o_2:*Reviewer* , $\equiv$ $\rangle$ *and* $\langle$ o_1:*Paper* , o_2:*Paper* , $\equiv$ $\rangle$ *are two correspondences in an input alignment, a path between the nodes (*o_1:*Reviewer,*o_2:*Reviewer) and (*o_1:*Paper,*o_2:*Paper) of the mapping graph will be sought. The mapping graph edges are products of the source and target relations, as*

well as identity, subclass-of, part-of properties. A path in the mapping graph could be as follows, where the nodes are marked between parenthesis `( )` *and the edges between brackets* `- -[]- ->`*.*

```
(o1:Reviewer,o2:Reviewer)
        --[o1:reviewerOf,o2:writesReview]-->
        (o1:Paper,o2:Review)
              --[Identity,o2:reviewOf]-->
                    (o1:Paper,o2:Paper)
```

The correspondence translating this path is $\langle$ *dom(o_1:Reviewer)* $\sqcap$ *o_1:reviewerOf* $\sqcap$ *range(o_1:Paper) , (dom(o_2:Reviewer)* $\sqcap$ *o_2:writesReview* $\sqcap$ *range(o_2:Review))* $\circ$ *(o_2:reviewOf* $\sqcap$ *range(o_2:Paper)) ,* $\equiv$ $\rangle$*.*

3.2.4 Tree

While some approaches [An *et al.* 2005b, An *et al.* 2005a, Knoblock *et al.* 2012] rely on a *semantic tree* derived from the schema, the approaches focusing on structural transformations between two trees (addition of a node, deletion of an attribute, *etc.*) such as [Fletcher & Wyss 2009, Hartung *et al.* 2013] often rely on tree-structure. However, they are out of the scope of this study as they are part of the ontology evolution field. Other approaches such as [Nunes *et al.* 2011, de Carvalho *et al.* 2013] use tree-based algorithms such as genetic programming. However they do not consider the schemata or ontologies as trees and therefore are not classified in this category.

MapOnto [An *et al.* 2005b, An *et al.* 2005a] *Relational database schema to CML ontology [An et al. 2005b], XML schema to OWL ontology [An et al. 2005a], (c:c)* MapOnto[5] [An *et al.* 2005b, An *et al.* 2005a], a work of An *et al.* is inspired from Clio in terms of path finding and tree construction. The approaches focus on aligning a source schema to a target ontology. Two approaches were proposed: a relational database schema to ontology [An *et al.* 2005b] and an XML schema to ontology [An *et al.* 2005a]. Both approaches take simple correspondences between the schema attributes and the ontology data-properties as input. These matching techniques construct a conjunctive first-order formula composed of target ontology entities to match a table (relational database) or *element trees* (XML) from the source schema. The production of the logical formula (presented as a *semantic tree* in [An *et al.* 2005b]) differs between the two approaches because of the different nature of the schemata. However, both approaches look for the smallest tree spanning all the attributes of the schema. A set of the most "reasonable" alignments are output for the user to choose among. These techniques output (c:c) correspondences because a whole table (or *element tree*) is transformed in each correspondence.

[5] `http://www.cs.torOnto.edu/semanticWeb/mapOnto/`

Example 14 *Let PAPERS(id,title,accepted) be a table from a relational database schema. The following translation rule to map the PAPERS table to an ontology o can be output by this approach:*
PAPERS(id,title,author) ⇒ o:Paper(x) ∧ o:paperId(x,id) ∧ o:title(x,title) ∧ o:Author(y) ∧ o:authorOf(x,y) ∧ o:name(y,author)

KARMA [Knoblock *et al.* 2012, Wu & Knoblock 2015] *Table schema, Relational database schema, XML schema, JSON schema to OWL ontology, (s:c),(c:s),(c:c)* KARMA[6] [Knoblock *et al.* 2012, Wu & Knoblock 2015] is a semi-automatic relational database schema to ontology matching system. Other types of structured data such as JSON or XML files can be processed by KARMA: they are transformed into a relational data model in a first step following a few rules. KARMA has two parts: a structured data to ontology matching part presented in [Knoblock *et al.* 2012] and a programming-by-example algorithm [Wu & Knoblock 2015] to create data transformation functions which falls in the *No structure* category. The structured data to ontology approach is similar to those of An *et al.* [An *et al.* 2005b, An *et al.* 2005a] as it is based on a Steiner-tree algorithm and outputs FOL-like formula as alignments (as in example 14). It can be categorised as *Tree* based and will output *(c:c)* correspondences. The matching process is articulated in 4 steps during which the user can intervene to correct or refine the correspondences. The first step consists in finding correspondences between the columns of one of the source database tables and the target ontology. The ontology member of the correspondence can be a class or a pair of property-domain or subclass of domain. These correspondences are found using a conditional random field trained with labelled data (column names, values and associated ontology entity). The training labelled data can be obtained from previous user assignments or generated using feature vectors based on the names and values of the columns. The second step consists in constructing a graph linking the ontology entities from the previous step together by using object properties and hierarchical relations of the ontology. The reachable classes from the ontology are added as nodes of the graph. The user can edit the graph by changing the correspondences with the ontology, edges of the graphs. The user can also generate multiple instances of a class. In the third step, a Steiner-tree algorithm looks for the minimum-weight tree in the graph that spans all nodes. Finally, the computed Steiner-tree is transformed into a FOL-like formula as target member of the correspondence (as a translation rule). The translation rule from Example 14 could be output by KARMA.

3.2.5 No structure

The approaches described in this section do not follow any of the above structures. While [Hu *et al.* 2011] is based on Inductive Logic Programming and builds its correspondences in an *ad hoc* manner, [Jiang *et al.* 2016] uses Markov Logic Networks for combinatorial exploration, [Hu & Qu 2006] uses classifying techniques to

[6] `https://github.com/usc-isi-i2/Web-Karma`

generate block correspondences, [Dhamankar *et al.* 2004] uses a numeric searcher using context-free grammar for equation discovery, [Wu & Knoblock 2015] applies a user-driven programming-by-example strategy and finally [Nunes *et al.* 2011, de Carvalho *et al.* 2013] use genetic programming to combine data value transformation functions.

Hu *et al.* [Hu *et al.* 2011] *OWL ontology to OWL ontology, (s:c)* The approach proposed by Hu *et al.* [Hu *et al.* 2011] uses Inductive Logic Programming (ILP) techniques to discover complex alignments. This technique is inspired by Stuckenschmidt *et al.* [Stuckenschmidt *et al.* 2008]. The approach is based on the common instances of a source and a target ontology. It outputs Horn-rules of the form $A \wedge B \wedge C \wedge ... \rightarrow D$ with $A, B, C...$ source entities represented as first-order predicates and D a target entity as a first-order predicate. The Horn-rule contains two parts: the body on the left side of the implication and the head on the right side. Three phases compose the approach. In the first, the instances of the two ontologies are matched. In the second called *data-tailoring*, instances and attributes from their context (relations, data-properties, other linked instances, *etc.*) are chosen for each target entity. The purpose of this phase is to eliminate irrelevant data. The last phase is the *mapping learning* phase. For each target entity, a new Horn-rule is created with this target entity as head predicate. Then iteratively, the predicate with the highest information gain score is added to the body of the Horn-rule. During this process, the variables of the Horn-rule are bound according to the instances and their context. The information gain metric involved in the process is based on the number of facts (instances or instance pairs) which support the correspondence or not.

Example 15 *At the first iteration of the process, the head of the Horn-rule is set to a predicate (unary or binary) and the body of the Horn-rule is empty. Let us consider the case when the Horn-rule head is a binary predicate:*

$\forall x,y, \rightarrow o_2$*:reviewerOf(x,y)*

All possible pairs of common instances are classified as positive binding or negative binding with regards to whether they instantiate o_2*:reviewerOf or not. The predicate with the biggest information gain (calculated from the positive and negative bindings) over the instance pairs is added to the body of the Horn-rule:*

$\forall x,y, \exists z,$ o_1*:writesReview(x,z)* $\rightarrow$ o_2*:reviewerOf(x,y)*

and in the next iteration:

$\forall x,y, \exists z,$ o_1*:writesReview(x,z)* $\wedge$ o_1*:Paper(y)* $\rightarrow$ o_2*:reviewerOf(x,y)*

and so on until no more positive binding is left to find or the number of predicates in the Horn-rule body has reached a threshold:

$\forall x,y, \exists z,$ o_1*:writesReview(x,z)* $\wedge$ o_1*:reviewOfPaper(z,y)* $\wedge$ o_1*:Paper(y)* $\rightarrow$ o_2*:reviewerOf(x,y)*

which translates as the correspondence:

$\langle$ o_1*:writesReview* $\circ$ (o_1*:reviewOfPaper* $\sqcap$ *range(*o_1*:Paper))* , o_2*:reviewerOf* , $\sqsubseteq$ $\rangle$

KAOM [Jiang *et al.* 2016] *OWL ontology to OWL ontology, (s:c) (c:s) (c:c)* KAOM (Knowledge Aware Ontology Matching) is a system proposed by Jiang *et al.* [Jiang *et al.* 2016]. It uses Markov Logic Network as a probabilistic framework for ontology matching. The Markov Logic formulae presented in this approach use the entities of the two ontologies (source and target) as *constants*, the relations between entities and the input *knowledge rules* as *evidence*. The *knowledge rules* can be axioms of an ontology or they can be specified by the user. They do not have to be semantically exact. To handle numerical data-properties, KAOM proposes two methods to find positive linear transformations between rules. These methods are based on the values that the data-properties take in a given knowledge base (the distribution of the values or a way to discretise them). The correspondence patterns and conditions presented by Ritze *et al.* [Ritze *et al.* 2009, Ritze *et al.* 2010] can be translated into knowledge rules and therefore used into Markov Logic formulae. The *knowledge rules* can be obtained in various ways as was shown in the experiments where decision trees, association rules obtained from an *a priori* algorithm or manually written rules were translated as *knowledge rules* for three different test cases.

Example 16 *A knowledge rule could be "Many reviewers are also authors of paper", which would be in* o_1 *(*$\rightsquigarrow$ *seen as a "is often true" relation):* o_1*:Reviewer* $\rightsquigarrow$ $\exists o_1$*:authorOf.*o_1*:Paper. The same knowledge rule expressed in* o_2 *would be:* $\exists o_2$*:reviewerOf.*$\top$ $\rightsquigarrow$ o_2*:Author. Based on these knowledge rules, two candidate correspondences can be:* $\langle$ o_1*:Reviewer* , $\exists o_2$*:reviewerOf.*$\top$, $\equiv$ $\rangle$ *and* $\langle$ $\exists o_1$*:authorOf.*o_1*:Paper* , o_2*:Author* , $\equiv$ $\rangle$.

iMAP [Dhamankar *et al.* 2004] *Relational database schema to relational database schema, (c:s)* As seen previously, the iMAP system [Dhamankar *et al.* 2004] uses a set of *searchers* to discover simple and complex correspondences between database schemata. The overlap numeric searcher uses the LAGRAMGE algorithm for equation discovery based on overlapping data. This algorithm uses a context-free grammar to define the search space of the arithmetic equations and executes a beam-search to find a suitable correspondence. The output of this search space is then stored as a pattern for the numeric searcher.

Nunes *et al.* [Nunes *et al.* 2011] *OWL ontology to OWL ontology, (c:s)* Genetic programming can be used for finding complex correspondences between data-properties. It can combine and transform the data-properties of an ontology to match a property of another ontology. Nunes *et al.* [Nunes *et al.* 2011] propose a genetic programming approach for numerical and literal data property matching. The correspondences generated are (c:s) as n data-properties from the source ontology are combined to match a target data-property. The source data-properties are chosen from a calculated estimated mutual information (EMI) matrix. Each individual of the genetic algorithm is a tree representing the combination operations over data properties. The elementary operations used for combination are

concatenation or split for literal data-properties and basic arithmetic operations for numerical data-properties (sum, multiplication, *etc.*). The fitness of a solution is evaluated on the values given by this solution and the values expected (based on matched instances) using a Levenshtein distance.

de Carvalho *et al.* [de Carvalho *et al.* 2013] *Table schema to table schema, (s:c) (c:s) (c:c)* De Carvalho *et al.* [de Carvalho *et al.* 2013] apply a genetic algorithm to alignments as its "individuals". Each "individual" is a set of correspondences. Each correspondence is a pair of *tree functions* made of elementary operations (as for Nunes *et al.* [Nunes *et al.* 2011]) and having source (resp. target) attributes as leaves. Constraints over the correspondences have been defined: a schema attribute cannot appear more than once in a correspondence, crossover and mutation can only be applied to attributes of the same data type, the number of correspondences in an alignment is fixed *a priori*. Mutation and cross-over operations occur at the correspondence's tree-level when parts of two *tree functions* are swapped, or changed. The fitness evaluation function of the schema alignments (individuals) is the sum of the fitness score of its correspondences. The fitness score of a correspondence can be calculated in two ways: *entity-oriented* with the similarity of matched instances (the data must be overlapping) or *value-oriented* with the similarity of all transformed source instances and target instances. The similarity metric for each correspondence is chosen by an expert. Compared to the approach of Nunes et al. [Nunes *et al.* 2011] it can detect *(c:c)* correspondences thanks to its internal modelling. However the process may require more iterations than [Nunes *et al.* 2011].

KARMA [Knoblock *et al.* 2012, Wu & Knoblock 2015] *Table schema, Relational database schema, XML schema, JSON schema to OWL ontology, (s:c),(c:s),(c:c)* The programming-by-example algorithm of KARMA (approach presented in the *Tree category) creates data transformation functions. It* considers the transformation functions as programs divided into subprograms to be applied to the data to transform it. At the beginning of the process, an example of source data (a table cell or row value) is given to the user and he or she gives what he or she expects as a result. This first pair of values constitutes an example and a program (transformation function) is then synthesised and applied to the other instances of the data. The user iteratively corrects the wrongly translated data, giving new examples from which the process refines its program by detecting and changing incorrect subprograms. The basic operations (or segments) of a program or subprogram are string operations (substring, concatenation, recognizing a number, *etc.*). As the input and the output of the process can cover one or many columns of the source and target tables, this part of KARMA can output *(s:c), (c:s)* or *(c:c)* correspondences.

Example 17 *A first example "PaperABC written by AuthorTT strong accept 2016" from the source database is given to the user. The user gives the expected value "Pa-*

perABC (2016)". This first pair of values constitutes an example and a program (transformation function) is synthesised. For example, out of all the possible programs (called hypotheses) one could be:

```
transform(val):
        pos1=val.indexOf(START,WORD,1)
        pos2=val.indexOf(WORD,BNK,1)
        pos3=val.indexOf(BNK,NUM,1)
        pos4=val.indexOf(NUM,END,1)
        output= val.substr(pos1,pos2)+‘ (’+val.subtr(pos3
           ,pos4)+‘)’
        return output
```

where indexOf(LEFT, RIGHT, N) takes the left and right context of the occurrence and N denotes the n-th occurrence. START is the beginning of the value, END its end. WORD represents a ([A-Za-z]+) string, NUM a number, BNK a whitespace. This program is then applied to the other instances of the data. The user iteratively corrects the wrongly translated data, giving new examples from which the process refines its program (the hypothesis space will be reduced).

BMO [Hu & Qu 2006] *OWL ontology to OWL ontology, (s:c) (c:s) (c:c)* BMO (Block Matching for Ontologies) focuses on matching sets of entities (classes, relations or instances) called blocks. This approach is articulated into four steps. The first step is the construction of virtual documents for each entity of both ontologies: the annotations and all triples in which an entity occurs are gathered into a *document*. The second one computes a *relatedness matrix* by calculating the similarity between each vectorised virtual document. In the third step, the *relatedness matrix* is used to apply a partitioning algorithm: this algorithm is recursively applied to the set of ontology entities. At the end of this algorithm, the similar entities are together in the same block while dissimilar entities are in distinct blocks. The final step consists in finding the optimal alignment given a number of blocks. Ontology entities which are in the same block can be separated into o_1 and o_2 to obtain a correspondence. As the blocks can contain any type of entity, it is not considered as a composite pattern.

3.2.6 Summary of the survey on complex alignment generation approaches

The proposed classification is based on two main axes, the output (type of correspondence) and process (guiding structure) dimensions of the approaches. The following tables present the approaches in the order in which they first appear in the survey.

Table 3.4 summarises the type of knowledge representation models aligned by the approaches and the additional input. Most approaches require external output such as matched instances or a simple alignment. This table shows the va-

Table 3.4: Input of the approaches: type of aligned knowledge representation model and type of additional input information

Approach	Type of Knowledge Representation Model	Additional Input
[Ritze *et al.* 2009]	OWL ontology to OWL ontology	simple alignment
[Ritze *et al.* 2010]	OWL ontology to OWL ontology	simple alignment (opt.)
AMLC [Faria *et al.* 2018]	OWL ontology to OWL ontology	simple alignment
[Oliveira & Pesquita 2018]	OWL ontology to OWL ontology	
[Rouces *et al.* 2016]	OWL ontology to OWL ontology	
Bayes-ReCCE [Walshe *et al.* 2016]	OWL ontology to OWL ontology	matched instances
OAT [Chondrogiannis *et al.* 2014]	OWL ontology to OWL ontology	
iMAP [Dhamankar *et al.* 2004]	RDB schema to RDB schema	domain constraints and value distribution
KAOM [Jiang *et al.* 2016]	OWL ontology to OWL ontology	knowledge rules
BootOX [Jiménez-Ruiz *et al.* 2015]	RDB schema to OWL ontology	
[Parundekar *et al.* 2012]	OWL ontology to OWL ontology	matched instances
[Parundekar *et al.* 2010]	OWL ontology to OWL ontology	matched instances
CGLUE [Doan *et al.* 2003]	DL ontology to DL ontology	
ARCMA [Kaabi & Gargouri 2012]	OWL ontology to OWL ontology	simple alignment
[Boukottaya & Vanoirbeek 2005]	XML schema to XML schema	
(COMA++) [Arnold 2013]	Taxonomy to Taxonomy	
[Xu & Embley 2003]	Conceptual Model to Conceptual Model	domain ontology
[Xu & Embley 2006]	Conceptual Model to Conceptual Model	domain ontology
[Warren & Tompa 2006]	Table schema to Table schema	
[Šváb Zamazal & Svátek 2009]	OWL ontology to OWL ontology	
PORSCHE [Saleem *et al.* 2008]	XML schema to XML schema	abbreviation Table schema
DCM [He *et al.* 2004]	Table schema to Table schema	web query interfaces
HSM [Su *et al.* 2006]	Table schema to Table schema	web query interfaces
[Wu *et al.* 2004]	Table schema to Table schema	web query interfaces
[An *et al.* 2012]	Document-oriented schema to CML ontology	web query interfaces, simple correspondences web form-onto
Clio [Miller *et al.* 2000, Yan *et al.* 2001]	RDB schema to RDB schema	value correspondences
OntoGrate [Qin *et al.* 2007, Dou *et al.* 2010]	OWL ontology to OWL ontology	
[An & Song 2008]	CML ontology to CML ontology	simple alignment
MapOnto [An *et al.* 2005b]	RDB schema to CML ontology	attribute-data properties correspondences
MapOnto [An *et al.* 2005a]	XML schema to OWL ontology	attribute-data properties correspondences
KARMA [Knoblock *et al.* 2012, Wu & Knoblock 2015]	Table, RDB, XML, JSON schema to OWL ontology	examples for data transformation functions
[Hu *et al.* 2011]	OWL ontology to OWL ontology	
[Nunes *et al.* 2011]	OWL ontology to OWL ontology	
[de Carvalho *et al.* 2013]	Table schema to Table schema	
BMO [Hu & Qu 2006]	OWL ontology to OWL ontology	

Table 3.5: Output of the approaches: correspondence members form, types of correspondences and correspondence format

Approach	(s:c)	(c:s)	(c:c)	Logic	Transfo	Block	Correspondence format
[Ritze *et al.* 2009]	•			•			pseudo-DL
[Ritze *et al.* 2010]	•			•			EDOAL
AMLC [Faria *et al.* 2018]	•			•			EDOAL
[Oliveira & Pesquita 2018]	•			•			Not specified
[Rouces *et al.* 2016]	•			•			SPARQL construct
Bayes-ReCCE [Walshe *et al.* 2016]	•	•		•			EDOAL
Chondrogiannis2014 OAT [Chondrogiannis *et al.* 2014]	•	•	•	•			OWL, EDOAL
iMAP [Dhamankar *et al.* 2004]		•			•		equations
KAOM [Jiang *et al.* 2016]	•	•	•	•	•		pseudo-DL
BootOX [Jiménez-Ruiz *et al.* 2015]		•		•			R2RML
[Parundekar *et al.* 2012]	•	•		•			pseudo-DL
[Parundekar *et al.* 2010]	•	•	•	•			pseudo-DL
CGLUE [Doan *et al.* 2003]	•			•			Not specified
ARCMA [Kaabi & Gargouri 2012]	•			•			DL
[Boukottaya & Vanoirbeek 2005]	•	•	•	•	•		XSLT
COMA++ [Arnold 2013]	•	•			•		Not specified
[Xu & Embley 2003]	•	•		•	•		Not specified
[Xu & Embley 2006]	•	•	•	•	•		Not specified
[Warren & Tompa 2006]		•			•		SQL queries
[Šváb Zamazal & Svátek 2009]	•	•	•			•	Not specified
PORSCHE [Saleem *et al.* 2008]	•	•				•	Not specified
DCM [He *et al.* 2004]	•	•	•			•	sets
HSM [Su *et al.* 2006]	•	•	•			•	sets
[Wu *et al.* 2004]	•	•	•			•	sets
[An *et al.* 2012]	•			•			Not specified
Clio [Miller *et al.* 2000, Yan *et al.* 2001]	•	•	•	•			SQL views
OntoGrate [Qin *et al.* 2007, Dou *et al.* 2010]			•	•			DataLog, SWRL, Web-PDDL
[An & Song 2008]			•	•			FOL or SPARQL
MapOnto [An *et al.* 2005b]			•	•			FOL
MapOnto [An *et al.* 2005a]			•	•			FOL
KARMA [Knoblock *et al.* 2012, Wu & Knoblock 2015]	•	•	•	•	•		FOL
[Hu *et al.* 2011]		•		•			FOL
[Nunes *et al.* 2011]		•			•		equations
[de Carvalho *et al.* 2013]	•	•	•		•		equations
BMO [Hu & Qu 2006]	•	•	•			•	sets

Table 3.6: Process characteristics of the approaches based on the proposed classification

Approach	Guiding structure	fixed to fixed	fixed to unfixed	unfixed to unfixed	Ontology-level evidence	Instance-level evidence	Other
[Ritze *et al.* 2009]	Atomic patterns	•			•		
[Ritze *et al.* 2010]	Atomic patterns	•			•		
AMLC [Faria *et al.* 2018]	Atomic patterns	•			•		
[Oliveira & Pesquita 2018]	Atomic patterns	•			•		Compound
[Rouces *et al.* 2016]	Atomic patterns	•			•		
Bayes-ReCCE [Walshe *et al.* 2016]	Atomic patterns	•				•	
OAT [Chondrogiannis *et al.* 2014]	Atomic patterns	•			•		
iMAP [Dhamankar *et al.* 2004]	Atomic patterns, Composite patterns, No structure	•	•			•	
KAOM [Jiang *et al.* 2016]	Atomic patterns, No structure	•		•	•	•	
BootOX [Jiménez-Ruiz *et al.* 2015]	Atomic patterns	•			•		
[Parundekar *et al.* 2012]	Composite patterns		•			•	
[Parundekar *et al.* 2010]	Composite patterns			•		•	
CGLUE [Doan *et al.* 2003]	Composite patterns		•			•	
ARCMA [Kaabi & Gargouri 2012]	Composite patterns		•		•	•	
[Boukottaya & Vanoirbeek 2005]	Composite patterns		•		•		
COMA++ [Arnold 2013]	Composite patterns		•		•		
[Xu & Embley 2003]	Composite patterns	•	•		•	•	
[Xu & Embley 2006]	Composite patterns, Path to path	•	•	•	•	•	
[Warren & Tompa 2006]	Composite patterns		•			•	
[Šváb Zamazal & Svátek 2009]	Composite patterns		•	•	•		
PORSCHE [Saleem *et al.* 2008]	Composite patterns		•		•		Holistic
DCM [He *et al.* 2004]	Composite patterns			•	•		Holistic
HSM [Su *et al.* 2006]	Composite patterns			•	•		Holistic
[Wu *et al.* 2004]	Composite patterns		•		•		
[An *et al.* 2012]	Path to Path		•		•		
Clio [Miller *et al.* 2000, Yan *et al.* 2001]	Path to Path			•	•	•	
OntoGrate [Qin *et al.* 2007, Dou *et al.* 2010]	Path to Path			•	•	•	
[An & Song 2008]	Path to Path			•	•		
MapOnto [An *et al.* 2005b]	Tree to tree			•	•		
MapOnto [An *et al.* 2005a]	Tree to tree			•	•		
KARMA [Knoblock *et al.* 2012, Wu & Knoblock 2015]	Tree to tree, No structure		•	•	•	•	
[Hu *et al.* 2011]	No structure		•		•	•	
[Nunes *et al.* 2011]	No structure		•			•	
[de Carvalho *et al.* 2013]	No structure			•		•	
BMO [Hu & Qu 2006]	No structure			•	•		

Table 3.7: Classification of the complex matchers on [Euzenat & Shvaiko 2013]'s basic techniques

Approach	Formal resource-based	Informal resource-based	String-based	Language-based	Constraint-based	Taxonomy-based	Graph-based	Instance-based	Model-based
[Ritze *et al.* 2009]			•	•	•	•	•		
[Ritze *et al.* 2010]			•	•	•	•	•		
AMLC [Faria *et al.* 2018]			•		•	•	•		
[Oliveira & Pesquita 2018]			•						
[Rouces *et al.* 2016]	•		•	•			•		
Bayes-ReCCE [Walshe *et al.* 2016]	•						•	•	
OAT [Chondrogiannis *et al.* 2014]			•		•	•	•		
iMap [Dhamankar *et al.* 2004]	•		•		•		•	•	
KAOM [Jiang *et al.* 2016]	•		•		•			•	•
BootOX [Jiménez-Ruiz *et al.* 2015]				•	•		•		
[Parundekar *et al.* 2012]	•						•	•	
[Parundekar *et al.* 2010]	•						•	•	
CGLUE [Doan *et al.* 2003]			•			•		•	
ARCMA [Kaabi & Gargouri 2012]			•			•		•	
[Boukottaya & Vanoirbeek 2005]				•	•	•	•		
COMA++ [Arnold 2013]			•			•			
[Xu & Embley 2003]	•		•				•	•	
[Xu & Embley 2006]	•		•	•			•	•	
[Warren & Tompa 2006]								•	
[Šváb Zamazal & Svátek 2009]			•				•		
PORSCHE [Saleem *et al.* 2008]				•		•			
DCM [He *et al.* 2004]		•							
HSM [Su *et al.* 2006]		•							
[Wu *et al.* 2004]		•	•	•	•	•		•	
[An *et al.* 2012]		•					•		
Clio [Yan *et al.* 2001, Miller *et al.* 2000]					•		•	•	
OntoGrate [Qin *et al.* 2007, Dou *et al.* 2010]							•	•	
[An & Song 2008]	•					•	•		
MapOnto [An *et al.* 2005b]	•						•		
MapOnto [An *et al.* 2005a]	•						•		
KARMA [Knoblock *et al.* 2012, Wu & Knoblock 2015]		•	•				•	•	
[Hu *et al.* 2011]								•	
[Nunes *et al.* 2011]								•	
[de Carvalho *et al.* 2013]								•	
BMO [Hu & Qu 2006]			•				•		

riety of knowledge representation models for which complex alignment generation approaches have been proposed. Table 3.5 presents the output by the approaches: the correspondence members form, the type of correspondence and the output format. Few approaches can generate both logic construction and transformation function correspondences. Most approaches output correspondences as FOL rules, without following a particular format. The latest approaches ([Faria *et al.* 2018, Thiéblin *et al.* 2018c] published in 2018) output correspondences in EDOAL, which coincides with their participation in the OAEI complex track (*cf.* Section 3.3).

Table 3.6 presents the process of the approaches according to our classification. Most approaches are pattern-based (atomic or composite). Only a few approaches have no guiding structure. There is no direct correlation between the member expression (fixed to fixed, unfixed to unfixed, *etc.*) and the (s:c), (c:s) kind of correspondence.

In the Ontology Matching book [Euzenat & Shvaiko 2013], the basic matching techniques are classified as follows:

- *Formal resource-based*: rely on formal evidence: upper-level ontology, domain-specific ontology, linked data, linguistic frames, alignment
- *Informal resource-based*: rely on informal evidence: directory, annotated resources, Web forms
- *String-based*: use string similarity: name similarity, description similarity, global namespace
- *Language-based*: use linguistic techniques: tokenisation, lemmatisation, thesauri, lexicon, morphology
- *Constraint-based*: use internal ontology constraints: types, key properties
- *Taxonomy-based*: consider the specialisation relation of the ontologies: taxonomy, structure
- *Graph-based*: consider the ontologies as graphs: graph homomorphism, path, children, leaves, correspondence patterns
- *Instance-based*: compare sets of individuals: data analysis, statistics
- *Model-based*: use the semantic interpretation: SAT solvers, DL reasoners

The complex alignment generation approaches are described according to this classification in Table 3.7. The majority combine different matching techniques.

Few approaches are model-based (no semantic interpretation of the alignment). However, it is important to note that identifying the strategies based on Euzenat and Shvaiko's classification was not always straightforward.

Another way of classifying the approaches is with respect to the kind of evidence they exploit (ontology-level or instance-level), as done in different surveys

in the field. This classification was applied in two columns of Table 3.6. Most approaches use the ontology-level information as evidence. The approaches which output transformation functions mostly rely on instance-level information.

Overall, the approaches look for a way to reduce the matching space (*i.e.*, the space of possible correspondences between two ontologies). To do so, they rely on structures, common instances, external resources such as web query interfaces or a combinations of those. However, all of them intend to cover the full scope of the source and target ontologies.

3.3 Matching evaluation

The matching approaches presented in Section 3.2 deal with different knowledge and information representation models. Evaluation methods and datasets have been proposed to evaluate them. For example, the Illinois semantic integration archive[7] [Doan 2005] is a dataset of complex correspondences on value transformations (*e.g.*, string concatenation) in the inventory and real estate domain. This dataset only contains correspondences between schemata with transformation functions. The UIUC Web integration Repository [Chang *et al.* 2003] is a repository of schemata and query forms. XBenchMatch[8] [Duchateau *et al.* 2007] is a benchmark for XML schema matching. The RODI benchmark[9] [Pinkel *et al.* 2017] proposes an evaluation over given scenarii, R2RML correspondences between a database schema and an ontology. The benchmark relies on ontologies from the OAEI Conference dataset, Geodata ontology, Oil and gas ontology. The schemata are either derived from the ontologies themselves or curated on the Web. The RODI benchmark deals with R2RML alignment and uses reference SPARQL and SQL queries to assess the quality of the alignment.

In this section, we focus on the evaluation of ontology matching approaches. First we will present the datasets for ontology alignment evaluation (Section 3.3.1) then the evaluation strategies for simple alignment generation approaches (Section 3.3.2) and for complex alignment generation approaches (Section 3.3.3). The evaluation strategies are defined in Section 2.2.7.

3.3.1 Ontology alignment evaluation datasets

In the OAEI campaigns, the tracks evaluate matchers on various aspects. Most of these tracks focus on simple alignment generation approaches. The datasets of the tracks have specificities such as the size of the ontologies, their expressiveness, or their language (English, Spanish, *etc.*). In 2018, complex alignment evaluation datasets have been proposed [Thiéblin *et al.* 2018b, Zhou *et al.* 2018], leading to the first complex alignment generation track in the OAEI [Thiéblin *et al.* 2018a].

[7] http://pages.cs.wisc.edu/~anhai/wisc-si-archive/domains/
[8] https://perso.liris.cnrs.fr/fabien.duchateau/research/tools/xbenchmatch/
[9] https://github.com/chrpin/rodi

We give a list of the tracks of the OAEI 2018 as well as other complex alignment datasets.

Anatomy [Bodenreider *et al.* 2005] Running from the first OAEI campaign in 2005, the Anatomy dataset consists of a reference alignment between the Adult Mouse Anatomy and a part of the NCI Thesaurus describing the human anatomy.

LargeBio First proposed in the OAEI 2012 campaign, the LargeBio dataset contains reference alignments between the Foundational Model of Anatomy (FMA), SNOMED CT, and the National Cancer Institute Thesaurus (NCI). These ontologies are semantically rich and contain tens of thousands of classes. The reference alignments are derived from the Unified Medical Language System (UMLS) Metathesaurus.

Biodiv In the OAEI 2018 campaign, this dataset was proposed in a new track. It contains alignments between the Environment Ontology (ENVO) and the Semantic Web for Earth and Environment Technology Ontology (SWEET), and between the Flora Phenotype Ontology (FLOPO) and the Plant Trait Ontology (PTO). These ontologies are semantically rich and contain tens of thousands of classes.

Phenotype [Harrow *et al.* 2017] Running from the OAEI 2016 campaign, the phenotype dataset proposes alignments between ontologies about phenotype and diseases. The number of ontologies in this track is not the same for every campaign. A manual reference alignment between the ontologies of the dataset is proposed.

Conference [Šváb Zamazal *et al.* 2005, Šváb Zamazal & Svátek 2017] Running from 2009 in the OAEI, Conference track contains 16 ontologies about conference organisation. These ontologies were developed individually and are therefore heterogenous. Seven ontologies are involved in the available reference alignment (*ra1*): cmt, conference (sofsem), confOf (ConfTool), edas, ekaw, iasted and sigkdd. Even though this dataset has been largely used, it has only been partially populated in the OA4QA track (detailed below).

Multifarm [Meilicke *et al.* 2012] Proposed in 2012 and running in OAEI since then, this dataset evaluates cross-lingual version of the Conference dataset. The 7 ontologies from the *ra1* (the available reference alignment of conference) were translated into 9 other languages (total of 10 languages if we include English).

OA4QA [Solimando *et al.* 2014b] In the 2014 and 2015 OAEI campaigns, the OA4QA track included a dataset which extended Conference: some of the ontologies were synthetically populated and SPARQL queries over the ontologies

were provided for the evaluation. Only the classes covered by the 18 reference queries were populated and the creation of the synthetic $\mathcal{A}box$ has not been documented.

The simple alignment datasets cover a wide diversity of domain, ontology size, and particularities such as multilinguism or available queries. The first complex alignment track of the OAEI was proposed in 2018 [Thiéblin *et al.* 2018a]. Some datasets with complex alignments had been proposed earlier for custom evaluation of complex alignment generation approaches.

Conference Complex Task-Oriented [Thiéblin *et al.* 2018b] A first version of complex alignments between the conference ontologies from the OAEI long-running track has been proposed[10]. It contains two complex datasets over 5 ontologies of *ra1*. One, more expressive, is intended for query rewriting and the other for ontology merging. This dataset is limited to *(c:s)* and *(s:c)* correspondences.

Conference Complex Consensus This dataset was used in the OAEI 2018 complex track. It relies on a consensual complex alignment between 3 ontologies of the conference dataset. The methodology followed to create it is the one from [Thiéblin *et al.* 2018b] it is therefore limited to *(c:s)* and *(s:c)* correspondences. 3 conference experts created the correspondences individually. Then, 4 experts discussed them to reach a consensus.

Geolink [Zhou *et al.* 2018] This dataset is composed of two ontologies: the GeoLink Base Ontology (GBO) and the GeoLink Modular Ontology (GMO). The reference alignment[11] contains some *(c:c)* correspondences. This dataset was included in the OAEI 2018 complex track.

Hydrography This dataset contains alignment between 4 source ontologies (Hydro3, HydrOntology_native, HydrOntology_translated, and Cree) and a target ontology, the Surface Water Ontology (SWO). HydrOntology_translated is in Spanish while the other ontologies are in English. This dataset contains some *(c:c)* correspondences and was included in the OAEI 2018 complex track.

Taxon This dataset is composed of four ontologies which describe the classification of species: AgronomicTaxon, Agrovoc, DBpedia and TaxRef-LD. The common scope of these ontologies is plant taxonomy. This dataset extends the one proposed in [Thiéblin *et al.* 2017] by adding the TaxRef-LD ontology. Three of the four ontologies come from large-scale LOD repositories. They are therefore populated with many instances. In the final evaluation, only a subset of the instances were selected to avoid out of memory errors. This dataset also contains equivalent SPARQL queries derived from AgronomicTaxon's competency questions.

[10] `http://doi.org/10.6084/m9.figshare.4986368`
[11] `http://doi.org/10.6084/m9.figshare.5907172`

Compound [Pesquita *et al.* 2014, Oliveira & Pesquita 2018] This dataset contains compound alignments between ontologies from the Open Biological and Biomedical Ontology (OBO) Foundry. The correspondences follow the pattern $\langle\ o_1{:}A\ ,\ o_2{:}B \sqcap o_3{:}C\ ,\ \equiv\ \rangle$, with $o_1{:}A$, $o_2{:}B$ and $o_3{:}C$ classes from three different ontologies. Three ontologies were used as source and six ontologies were combined as target, resulting in six reference alignments. The reference alignments were automatically computed based on cross-products (sets of meaningful classes obtained by extending the classes from an ontology with the classes of another) [Pesquita *et al.* 2014].

[Parundekar *et al.* 2012] The approach of [Parundekar *et al.* 2012] estimated a reference alignment based on a recurring pattern (*Class by Attribute-Value*) between DBpedia and Geonames: $\langle\ \exists dbpedia{:}country.\{\texttt{theCountryInstance}\}\ ,\ \exists geonames{:}countryCode.\{\texttt{theCountryCode}\}\ ,\ \equiv\ \rangle$ where `theCountryInstance` is a country instance of DBpedia such as *dbpedia:Spain* and `theCountryCode` is a country code such as *"ES"*. The reference alignment was not explicitly created nor manually verified.

[Walshe *et al.* 2016] In [Walshe *et al.* 2016] the authors proposed an algorithm to create an evaluation dataset that is composed of a synthetic ontology containing 50 classes with known *Class by Attribute-Value* correspondences with DBpedia and 50 classes with no known correspondences with DBpedia. The synthetic ontology was then populated with DBpedia instances.

Table 3.8 summarises the datasets presented in this survey. Complex alignment datasets have emerged recently opening up new evaluation strategies.

3.3.2 Simple alignment generation evaluation

When writing about ontology matching evaluation, the Ontology Alignment Evaluation Initiative (OAEI)[12] cannot be missed. This yearly campaign evaluates and compares ontology matching approaches on different tracks (benchmarks). As seen in Section 3.3.1, each track dataset has a specificities: large ontologies, multi-lingual ontologies, *etc.* In this section, we focus on evaluation strategies implemented in these tracks and in other initiatives. We organise this section with the strategies defined in Section 2.2.7.

Tool-oriented In the 2018 OAEI campaign [Algergawy *et al.* 2018], the runtime was measured on all ontology matching tracks.

Controlled input In the interactive track of the OAEI 2018, the matching systems are given more or less correct input from a simulated user (called Oracle).

Output-oriented Assessing the alignment quality is at the center of ontology matching evaluation.

[12] `http://oaei.ontologymatching.org/`

Table 3.8: Ontology alignment evaluation datasets, their type of reference, the number of ontologies, if the ontologies have instances and the type of correspondences in the reference. If a distinction is made between source and target ontologies, it is written *source – target*: Hydrography has 4 source ontologies aligned to 1 target ontology.

Dataset	Reference	Nb Onto.	Instances	(s:s)	(s:c)-(c:s)	(c:c)
Anatomy	Alignment	2		√		
LargeBio	Alignment	3		√		
Phenotype	Alignment	2 + 2		√		
Biodiv	Alignment	2 + 2		√		
Conference	Alignment	16		√		
Multifarm	Alignment	7 × 10		√		
OA4QA	Alignment	7	√	√		
Conference complex T-O	Alignment	5		√	√	
Conference complex consensus	Alignment	3		√	√	
GeoLink	Alignment	2		√	√	√
Hydrography	Alignment	4 – 1		√	√	√
Taxon	Queries	4	√	√	√	√
Compound	Alignment	3 – 6			√	
[Parundekar *et al.* 2012]	Pattern	2	√		√	
[Walshe *et al.* 2016]	Alignment	2	√		√	

Intrinsic The coherence [Meilicke & Stuckenschmidt 2008] is measured in the Conference, LargeBio, Anatomy, Biodiv and Phenotype tracks. The conservativity [Solimando *et al.* 2014a, Solimando *et al.* 2017] is measured in the Conference track. These evaluations are automatic. In the LargeBio track, some correspondences are manually classified as true or false positive by an expert. This results in an intrinsic precision score.

Extrinsic All the simple ontology matching tracks of the OAEI 2018 use a reference alignment. Each correspondence is automatically classified as true positive or false positive with regard to this alignment. Precision, Recall and F-measure scores derive from this classification.

In the Phenotype track, the alignments are not only compared to a manually made reference alignment. They are also compared to a *voted* reference alignment which comes from the results of the other matching systems in the track.

Task-oriented The *OA4QA* track[13] [Solimando *et al.* 2014b] was proposed in the OAEI 2015 campaign. This track evaluates an alignment in a query answering scenario. First, the source and target ontologies are automatically merged using the output alignment. Then a query over

[13] `http://www.cs.ox.ac.uk/isg/projects/Optique/oaei/oa4qa/index.html`

the source ontology is run on this merged ontology. The answers to the query are compared to a set of reference answers. Precision, recall and F-measure scores are given to the alignment for each query. This track used a synthetically populated version of the *Conference* dataset and a set of manually constructed queries over these $\mathcal{A}boxes$.

[Hollink *et al.* 2008] proposes an "end-to-end" evaluation in which a set of queries are automatically rewritten using an evaluated alignment. The results of the queries are manually classified by relevance for a user on a 6-point scale. This evaluation was performed with two rewriting systems. If a source member e does not appear in any correspondence of the alignment, the *upwards* rewriting system will use super-classes of e which appear as source member in the alignment's correspondences and the *downwards* system will use subclasses of e. Three alignments were evaluated. For each alignment, 20 concepts were randomly selected to be queried and evaluated.

An alternative approach for evaluating query answering without using instances was proposed by [David *et al.* 2018], where queries are compared without instance data, by grounding the evaluation on query containment.

In the Semantic precision and recall metrics, the alignments are evaluated based on an ontology merging scenario [Euzenat 2007].

Simple ontology matching evaluation is widely automated. Comparing two simple correspondences consists in comparing their source member URI, their source target URI, their relation and their confidence value. The URI comparison can be performed by a string comparison.

In the alignment API, a query rewriting system which changes one source URI by its target equivalent has been implemented. [Hollink *et al.* 2008] proposes two query rewriting systems which rely on subsumption. The query rewriting systems also limit the adaptation of the task-oriented evaluation of complex alignments.

3.3.3 Complex alignment generation evaluation

Complex alignment generation evaluation has rarely been automated. The few automated cases are for specific types of correspondences. Most of the OAEI complex track evaluation is still manually performed because comparing two complex correspondences has not been automated yet. For example, $\langle\ o_1{:}Author\ ,\ \exists o_2{:}authorOf.\top\ ,\ \equiv\ \rangle$ is semantically equivalent to the correspondence $\langle\ o_1{:}Author\ ,\ \exists o_2{:}writtenBy^{-}.\top\ ,\ \equiv\ \rangle$. However, these two correspondences are syntactically different. Given this example, semantic precision and recall could integrate the fact that the two example expressions mean the same thing given that $o_2{:}authorOf$ is the inverse property of $o_2{:}writtenBy$. However, pattern-based alignment formats such as EDOAL (or OWL) may lead to other problems. For example, the correspondences $\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}acceptedBy.\top\ ,\ \equiv\ \rangle$ and $\langle\ o_1{:}AcceptedPaper\ ,$

≥ 1 o_2*:acceptedBy.*$\top$, $\equiv \rangle$ are equivalent but expressed using different constructors (respectively an existential restriction or a cardinality restriction over the o_2*:acceptedBy* property). This also complexifies the comparison of the two correspondences. The first Complex matching track of the OAEI was proposed in 2018 [Thiéblin *et al.* 2018a] but most of the evaluation was manual.

As in the previous section, we analyse the works with the evaluation strategies of Section 2.2.7.

Tool-oriented The runtime of the matchers was measured in the Taxon subtrack.

Controlled input In the GeoLink and Hydrography subtracks, the matchers were evaluated given different input. Given a list of entities, the system should be able to find the correct (complex) construction involving these entities. the matchers were evaluated with or without the entity list.

Output-oriented The evaluation of the complex correspondences is often manual.

Intrinsic In the Taxon subtrack, a precision score was given without a reference alignment. Each correspondence was manually classified as true positive or false positive by an expert. Most complex alignment generation approaches assessed the precision of their matcher this way [Ritze *et al.* 2009, Ritze *et al.* 2010, Parundekar *et al.* 2012, Walshe *et al.* 2016].

Extrinsic In the (complex consensus) Conference subtrack, only equivalent complex correspondences were evaluated because the matcher could take the simple reference alignment as input.

In the Hydrography and GeoLink subtracks, two types of correspondence evaluation were envisaged:

- Assess if the URIs used to express a correspondence are found. For example, for the reference correspondence $\langle$ o_1*:AcceptedPaper* , $\exists o_2$*:hasDecision.*o_2*:Acceptance* , $\equiv \rangle$, two pairs of URIs are expected: $\langle$ o_1*:AcceptedPaper* , o_2*:hasDecision* , $\equiv \rangle$ and $\langle$ o_1*:AcceptedPaper* , o_2*:Acceptance* , $\equiv \rangle$. If a matcher outputs $\langle$ o_1*:AcceptedPaper* , o_2*:Acceptance* , $\equiv \rangle$, it would have found half of the URI pairs. Precision and recall scores can be calculated. In this case, the precision is 1 (all pairs of URIs output are correct) and the recall is 0.5 (1 out of the 2 reference URI pairs are found). This evaluation was automated.
- Evaluate the complex correspondences as they are output. A semantic precision and recall was envisaged but no matcher was able to output complex correspondences in this track, so this strategy was not applied. Even if this strategy seems to be the most obvious, it did not seem to be implemented and it may be limited by the expressiveness of the alignment: the task of reasoning over ontologies more expressive than $\mathcal{SROIQ}$ may not be decidable [Horrocks *et al.* 2006].

Automatic evaluation of correspondences has already been performed on matchers such as [Parundekar *et al.* 2012, Walshe *et al.* 2016] which only output correspondences following one pattern. In [Walshe *et al.* 2016], the correspondences were then considered syntactically as strings to be compared to reference strings. [Parundekar *et al.* 2012] estimated the number of occurrences of the country instance-country code pattern between the ontologies and calculated a recall score based on this estimation.

Task-oriented In the Taxon subtrack, the alignments were manually analysed to see if they could cover the knowledge represented by a set of queries. This evaluation was manually performed because existing query rewriting systems [Makris *et al.* 2012, Correndo & Shadbolt 2011, Thiéblin *et al.* 2016] only deal with *(s:c)* correspondences.

In 2018, the first complex track of the Ontology Alignment Evaluation Initiative[14] was conducted. It included four datasets [Thiéblin *et al.* 2018a]: the Hydrography dataset, the GeoLink dataset, a Conference Complex Consensus, and the Taxon dataset. However, most of the evaluation on complex alignment was manual. In general, automatic evaluation has only been performed on alignments whose correspondences follow a single pattern.

For the complex ontology matching field to grow, an automatic evaluation of the complex alignments should be available.

3.3.4 Summary of the survey on alignment evaluation

Comparing complex correspondences is a difficult task. For this reason, the evaluation in the OAEI complex track is still manual. An alternative solution would be to base the comparison of the correspondences on instances but this would require regularly populated ontologies which might not be the case on the LOD (*e.g.*, DBpedia contains irregularities) nor on the usual OAEI datasets (*e.g.*, conference, anatomy are not populated).

The relation of the correspondences ($\equiv, \sqsupseteq, \sqsubseteq$) is not taken into account in the evaluation process as most matchers only consider equivalence. The confidence given to a correspondence is taken into account when dealing with weighted precision and recall.

Finally, measuring the suitability of the output alignment for a given application, as done for the OA4QA track of the OAEI [Solimando *et al.* 2014b] or the Taxon track of the complex OAEI track [Thiéblin *et al.* 2018a] could be further considered.

3.4 Conclusion

Interest in complex alignment has recently increased in the ontology matching community. This comes from the fact that applications needing interoperability find

[14] `http://oaei.ontologymatching.org/2018/complex/`

simple alignments not sufficient.

The study of the approaches in this survey shows that, contrary to what intuition may suggest, matching more expressive knowledge representation models does not imply applying more sophisticated techniques. Most approaches consider knowledge representation models as graphs, trees or pools of annotated data regardless of their expressiveness. These common representations lead to similar techniques over diverse knowledge representation models.

The proposed classification tried to capture some of the aspects described above, by focusing on the specificities of complex correspondences on two main axes. The first axis characterises the different types of output (type of correspondence) and the second the structures used in the process to guide the correspondence detection. With respect to this classification, some approaches adopt a *mono-strategy* (atomic patterns, for instance), while others can fall in diverse categories. Classifying some of the approaches into a specific category was not a simple task.

Some approaches rely on existing simple correspondences (at ontology or instance level) while others are able to discover complex correspondences without this kind of input. Other resources are used as evidence such as Web query interfaces, knowledge rules or linguistic resources such as WordNet. Another aspect of the approaches is related to the kind of correspondence relation they can output. As for simple alignment generation approaches, most works are still limited to generating equivalences. The semantics of the confidence of a correspondence are rarely considered.

While the use of instance data evidence is valuable for the matching process, statistical approaches are directly impacted by the quality of this data. They can be faced with the problem of sparseness or with a specific corpus distribution that leads to incorrect correspondences. For example, if o_1 is populated with most students aged 23, $\langle\ o_1{:}Student\ ,\ \exists o_2{:}age.\{23\}\ ,\ \equiv\ \rangle$ can be a valid correspondence for the instance-based matching algorithms. These approaches perform a generalisation: they extract general rules from instance data. To do so, they generally require large amounts of common instances. However, the approach of [Wu & Knoblock 2015] also performs a generalisation but it can rely on very few examples. The user iteratively gives and corrects instances and the approach extracts general rules from them. The use of extra information such as user input can make the generalisation more effective and avoid correspondences which show a specificity of the data such as $\langle\ o_1{:}Student\ ,\ \exists o_2{:}age.\{23\}\ ,\ \equiv\ \rangle$.

Most approaches are limited to pair-wise matching. Holistic and compound complex alignment generation approaches are scarce but may be needed in various domains, such as bio-medicine, where several ontologies describing different but related phenomena have to be linked together [Oliveira & Pesquita 2015]. As stated in [Pesquita *et al.* 2014], the increase in the matching space and the inherently higher difficulty to compute alignments pose interesting challenges to this task.

On a different matter, we observe that some correspondences are pragmatically coherent but not semantically equivalent. For example, $\langle\ o_1{:}Ticket\ ,\ o_2{:}Adult + o_2{:}Children + o_2{:}Senior\ ,\ \equiv\ \rangle$ is a practical correspondence for counting the number

of passengers. The semantic meaning of this correspondence is however questionable as a ticket and a passenger are not exactly the same thing. This raises questions about the notion alignment context, the domain under which an alignment holds. In [Bouquet *et al.* 2003] "context mappings" define to which extent an alignment is valid.

Moreover, user involvement is under-exploited in complex alignment generation. This aspect, related to the visualisation and edition of complex correspondences [Noy & Musen 2003, Dragisic *et al.* 2016], is an important issue to be addressed in the future.

Regarding the evaluation of complex alignments, automatic correspondence comparison remains an open issue. The perspective of a benchmark with a reference alignment, real-life ontologies populated with controlled instances and metrics based on these instances, would be a useful resource in the field. As the interpretation of an ontology can vary from user to user, having a consensual benchmark with correspondence confidences reflecting the agreement between annotators, as in [Cheatham & Hitzler 2014], could be also an interesting resource. Another direction would be to evaluate the complex alignments over a real-life application such as ontology merging, data translation or query rewriting. The suitability of the alignment for the given task could be automatically computed. The first OAEI complex track could –hopefully– stimulate new works on complex ontology matching, evaluation, visualisation, *etc.*

Chapter 4

Need-based complex alignment generation approach

Content

Complex alignment generation is a hard task. Indeed, the matching space (set all possible correspondences between two ontologies) is not $\mathcal{O}(mn)$ as for simple alignment generation (m and n being respectively the number of entities of the source and target ontologies), but higher than $\mathcal{O}(2^{mn})$. We decided to reduce it with two hypothesis.

The first hypothesis is that focusing on the user's need can reduce the search space. We make the assumption that the user does not need the alignment to cover the full scope of the ontologies and is able to express his or her knowledge needs as SPARQL queries. Representing the needs for knowledge is a problem addressed in the ontology authoring domain. The NeOn methodology [Suárez-Figueroa *et al.* 2012] recommends competency questions (CQs). In Section 4.1, we introduce the notion of Competency Question for Alignment (CQA) inspired from the CQs in ontology authoring to express the knowledge needs. This hypothesis goes in opposite to the existing complex alignment generation approaches which intend to cover the full common scope of the aligned ontologies. To simplify the

complex alignment generation problem, some approaches restrain the shape of their correspondences with patterns while others consider only the descriptions of common instances.

The second hypothesis is that for each knowledge need, the ontologies share at least one instance. In comparison with statistics-based approaches which often need large number of common instances, this approach can work with only one.

Finally, this approach focuses on finding correspondences with logical constructors and does not deal with transformation functions.

The proposed approach, called Complex Alignment Need-based Abox-based Relation Discovery (CANARD), is presented in this Chapter. The examples in this Chapter use the ontologies o_1 and o_2 from Figure 2.1 in Chapter 2. We start by introducing CQAs in Section 4.1, then we introduce the outline of the approach and two running examples in Section 4.2. We detail the main steps of the approach in Section 4.3 and finally, we position and discuss the approach in Section 4.4.

4.1 Competency Questions for Alignment (CQAs)

As introduced in Chapter 2, in order to formalise the knowledge needs of an ontology, CQ have been introduced in ontology authoring as *ontology's requirements in the form of questions the ontology must be able to answer* [Grüninger & Fox 1995] (Chapter 2).

A CQA is a competency question which should (in the best case) be covered by two or more ontologies, *i.e.*, it expresses the knowledge that an alignment should cover if both ontologies' scopes can answer the CQA. The first difference between a CQA and a CQ is that the scope of the CQA is limited by the intersection of its source and target ontologies' scopes. The second difference is that this maximal and ideal alignment's scope is not known *a priori*.

Like the CQs, a CQA can be expressed in natural language or as SPARQL SELECT queries. Most characteristics of CQs defined by [Ren *et al.* 2014] and presented in Section 2.1.6 also apply to CQAs. We adapt the notion of *predicate arity* into the **question arity**. The question arity of a CQA which represents the arity of the expected answers to a CQA was introduced in [Thiéblin *et al.* 2018c]:

- A *unary* question expects a set of instances or values, *e.g.*, *Which are the accepted paper? (paper1), (paper2).*
- A *binary* question expects a set of instances or value pairs, *e.g.*, *What is the decision of which paper? (paper1, accept), (paper2, reject).*
- An *n-ary* question expects a tuple of size 3 or more, *e.g.*, *What is the rate associated with which review of which paper? (paper1, review1, weak accept), (paper1, review2, reject).*

In opposition to CQ, the formulation of the CQA does not impact the shape of the ontologies or the SPARQL queries. CQAs apply to existing ontologies.

As introduced above, a CQA can be expressed as SPARQL queries (at least one by ontology which cover the CQA). For example, the CQA *What are the accepted papers?* becomes for o_2 and o_1:

o_2 `SELECT ?x WHERE {?x o2:hasDecision ?y.  ?y a o2:Acceptance.}`

o_1 `SELECT ?x WHERE {?x a o1:AcceptedPaper.}`

In the proposed approach, we restrain the question type to the *selection* type. Indeed, questions with a binary or counting type have a corresponding selection question. For example, the question *Is this paper accepted?* has a binary type: its answers can only be *True* or *False*. The question *How many papers are accepted?* is a counting question. These two questions have the same selection question: *What are the accepted papers?*.

We also restrain the question polarity to *positive* because a negative question implies that a positive information is being negated. For example, the question *Which people are not reviewers?* is a negation of the question *Who are the reviewers?*.

The CQAs we consider have no modifier. The question arity of the CQAs is limited to *unary* and *binary* because ontologies are mostly constructed using unary predicates (classes or class expressions) and binary predicates (object or data properties).

4.2 Overview of the approach

The proposed approach takes as input a set of CQAs in the form of SPARQL SELECT queries over the source ontology. It requires that the source and target ontologies have an *Abox* with at least one common instance for each CQA. The answer to each input query is a set of instances, which are matched with those of a knowledge base described by the target ontology. The matching is performed by finding the surroundings of the target instances which are lexically similar to the CQA. As stated before, CQAs for the approach are limited to *unary* (class expressions, set of single instances expected) and *binary* questions (relation expressions, pairs of instances), of *select* type, and no modifier. We make the assumption that the user knows the source ontology and is able to write each CQA into a SPARQL query on the source ontology. This assumption is however a limitation to the approach.

The approach is articulated in 11 steps, as depicted in Figure 4.1. The approach is based on **subgraphs** which are a **set of triples** for a unary CQA and a **property path** for a binary CQA. A lexical layer comparison is used to measure the similarity of the subgraphs with the CQA.

In the following two sections, we detail an example for a unary CQA in Section 4.2.1 and for a binary CQA in Section 4.2.2. These examples are based on the CQAs in Figure 4.2. The knowledge bases considered in the examples are depicted in Figure 4.3. They share common instances: o_1*:person1* and o_2*:person1*, o_1*:paper1* and o_2*:paper1*. Ontology o_1 represents the concept of *accepted paper* as a class while

o_2 models the same knowledge with a *has decision* property. The property *paper written by* is represented by a single property in o_1 while in o_2, the property *writes* links a *person* to a *document*. For more details about the semantics of o_1 and o_2, see Figure 2.1 from Chapter 2.

A criticism to this example is that two knowledge bases may not represent the same conference, therefore they may not share common paper instances. We argue that remark by considering that these two knowledge bases may have a different but overlapping scope. For example o_1 could focus on the event organisation part of a conference and o_2 on reviewer management.

To make our example easier to understand, there is only one instance in each knowledge base but there could be overlapping instances as long as there is one common instance. The idea behind the approach is to rely on a few examples (answers) to find a generic rule which describes more instances.

In the following sections, the CQAs are represented by the labels of their entities when needed. For example, the CQA *What are the accepted papers ?* represented by the SPARQL query in Figure 4.2(a) for o_1 can be represented by the label of o_1*:AcceptedPaper*, *"accepted paper"*.

The implementation set-up such as the threshold values, label comparison metrics are out of the scope of this Chapter and will be described in Chapter 6 (Section 6.1).

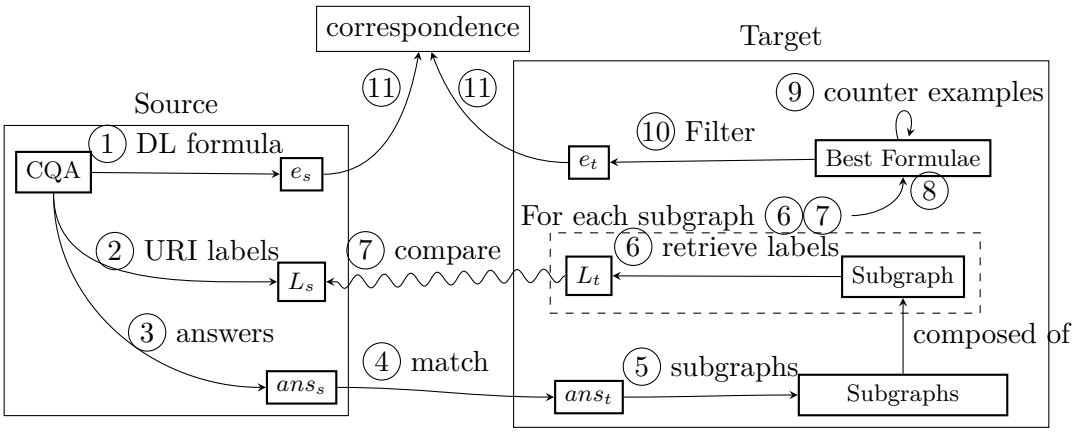


Figure 4.1: Schema of the general approach.

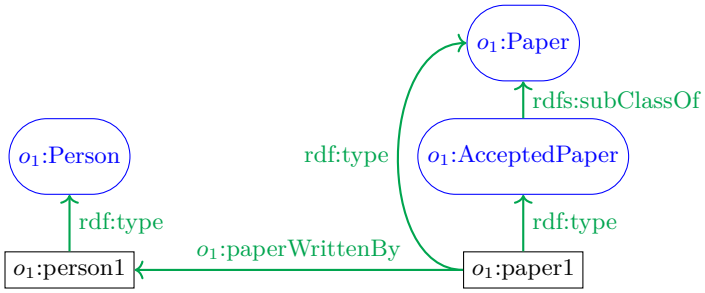
```
SELECT ?x WHERE {
  ?x a o1:AcceptedPaper.
}
```

(a) Source unary CQA

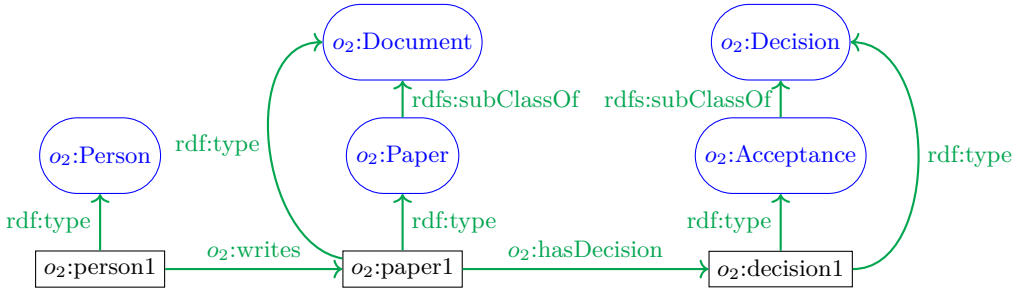
```
SELECT ?x ?y WHERE {
  ?x o1:paperWrittenBy ?y.
}
```

(b) Source binary CQA

Figure 4.2: Source CQAs as SPARQL queries



(a) Source knowledge base



(b) Target knowledge base

Figure 4.3: Source and target knowledge bases (*cf.* Appendix A.3)

4.2.1 Approach over a unary CQA

We instanciate Figure 4.1 for a unary CQA. The SPARQL CQA is that of Figure 4.2(a). The detail of the running example is presented in the following and depicted in Figure 4.4. We refer to the sections in which further details about the step are given.

(1) Represent the SPARQL CQA as a DL formula e_s (*e.g.*, o_1*:AcceptedPaper*) (Section 4.3.1).

(2) Extract lexical information from the CQA, L_s set labels of entities from the CQA (*e.g.*, "accepted paper").

(3) Retrieve source answers ans_s of the CQA (*e.g.*, o_1*:paper1*).

(4) Find equivalent or similar target answers ans_t to the source instances ans_s (*e.g.* o_1*:paper1* $\sim$ o_2*:paper1*) (Section 4.3.2).

(5) Extract the subgraphs of target answers (Section 4.3.3): for a unary query, this is the set of triples in which the target instances appear as well as the types (classes) of the subject or object of the triple (*e.g.* in DL, the description of o_2*:paper1* would contain $\langle$ o_2*:paper1 , *o_2*:hasDecision , *o_2*:decision1* $\rangle$, $\langle$ o_2*:decision1 , rdf:type ,* o_2*:Decision* $\rangle$ *and* $\langle$ o_2*:decision1 , rdf:type ,* o_2*:Acceptance* $\rangle$, see Figure 4.4.).

(6) For each subgraph, retrieve L_t the labels of its entities (*e.g.*, o_2*:hasDecision* $\rightarrow$ "decision", o_2*:decision1* $\rightarrow$ "decision for paper1", o_2*:Decision* $\rightarrow$ "decision").

(7) Compare L_s and L_t (Section 4.3.4).

(8) Select the subgraphs parts with the best similarity score, transform them into DL formulae (Section 4.3.3) and aggregate them (Section 4.3.5). In this example, the part of the subgraph which is the most similar to the CQA (in terms of label similarity) is o_2*:Acceptance*. The DL formula is therefore $\exists o_2$*:hasDecision.*o_2*:Acceptance*.

(9) Reassess the similarity of each DL formula based on their counter-examples (Section 4.3.6 and Section 4.3.7). The counter-examples are common instances of the two knowledge bases which are described by the target DL formula but not by the original CQA.

(10) Filter the DL formulae based on their similarity score (if their similarity score is higher than a threshold) (Section 4.3.8).

(11) Put the DL formulae e_s and e_t together to form a correspondence (*e.g.*, $\langle$ o_1*:AcceptedPaper* , $\exists$ o_2*:hasDecision.*o_2*:Acceptance* , $\equiv$ $\rangle$) and express this correspondence in a reusable format (*e.g.*, EDOAL). The confidence assigned to a correspondence is the similarity score of the DL formula computed.

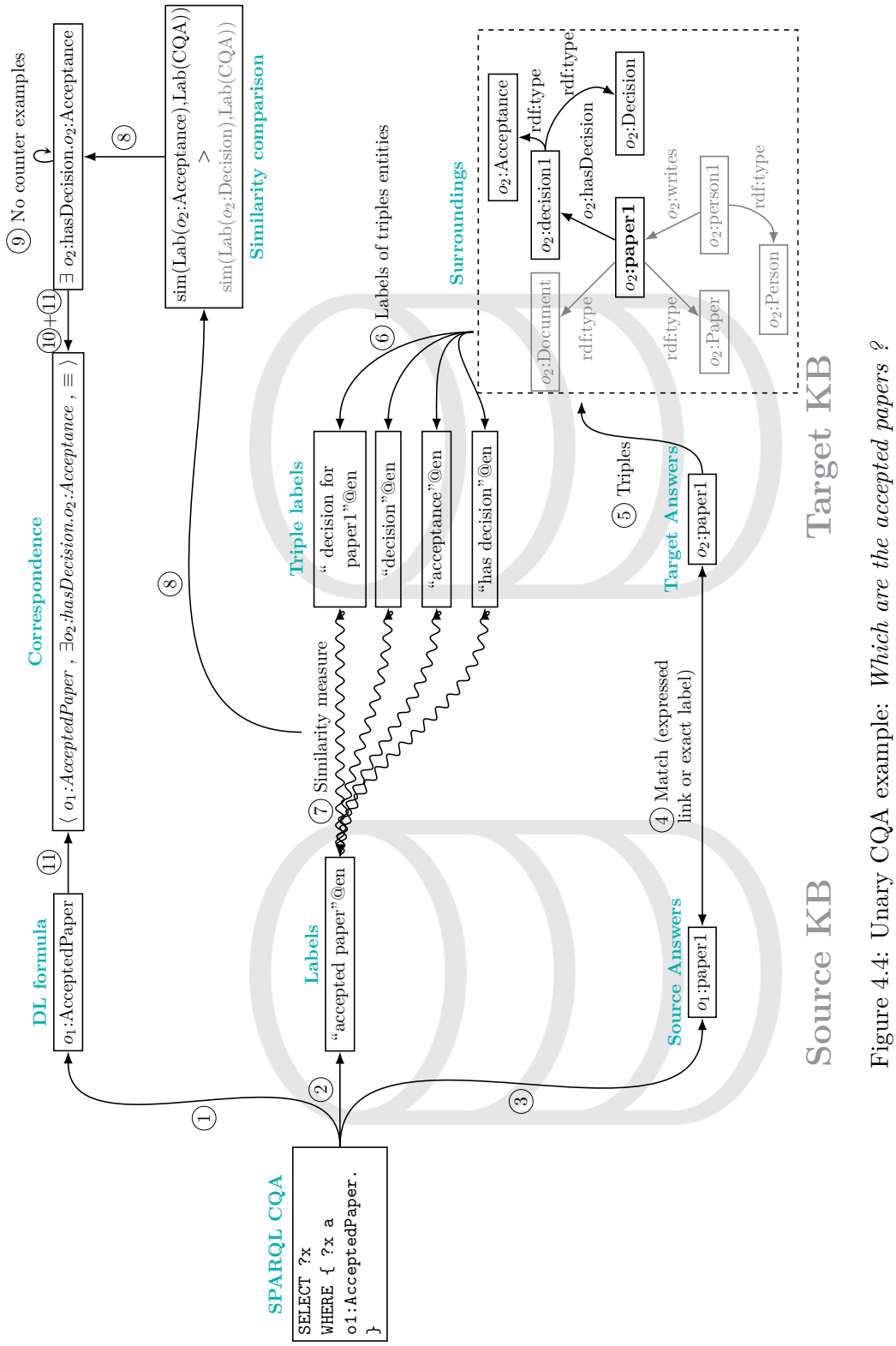


Figure 4.4: Unary CQA example: *Which are the accepted papers ?*

4.2.2 Approach over a binary CQA

Figure 4.5 instanciates the approach when dealing with the binary CQA from Figure 4.2(b) which retrieves the *"paper written by"*. The main difference with the case of unary CQAs is in Step (4) because the two instances of the pair answer are matched instead of one, Step (5) and Step (8) which deal with the subgraph extraction and pruning.

(1) Extract source DL formula e_s (*e.g.*, o_1*:paperWrittenBy*) from SPARQL CQA (Section 4.3.1). The CQA for this running example is shown in Figure 4.2(b).

(2) Extract lexical information from the CQA, L_s set labels of atoms from the DL formula (*e.g.*, "paper written by").

(3) Extract source answers ans_s of the CQA (*e.g.*, a pair of instances *(*o_1*:paper1,* o_1*:person1)*).

(4) Find equivalent or similar target answers ans_t to the source instances ans_s (*e.g.* o_1*:paper1* $\sim$ o_2*:paper1* and o_1*:person1* $\sim$ o_2*:person1*) (Section 4.3.2).

(5) Retrieve the subgraphs of target answers (Section 4.3.3): for a binary query, it is the set of paths between two answer instances as well as the types of the instances appearing in the path (*e.g.*, a path of length 1 is found between o_2*:paper1* and o_2*:person1*). The path is composed of only one property and there are no other instances than o_2*:paper1* and o_2*:person1* in this path. Their respective types are retrieved: (o_2*:Paper*,o_2*:Document*) for o_2*:paper1* and (o_2*:Person*) for o_2*:person1*.

(6) For each subgraph, retrieve L_t the labels of its entities (*e.g.*, o_2*:writes* $\rightarrow$ "writes", o_2*:Person* $\rightarrow$ "person", o_2*:Paper* $\rightarrow$ "paper", *etc.*).

(7) Compare L_s and L_t (Section 4.3.4).

(8) Select the subgraph parts with the best score, transform them into DL formulae (Section 4.3.3). Keep the best path variable types if their similarity is higher than a threshold. (*e.g.*, the best type for the instance o_2*:paper1* is o_2*:Paper* because its similarity with the CQA labels is higher than the similarity of o_2*:Document*).

(9) Reassess the similarity of each DL formula based on their counter-examples (Section 4.3.6 and Section 4.3.7).

(10) Filter the DL formulae based on their similarity score (if their similarity score is higher than a threshold) (Section 4.3.8) .

(11) Put the DL formulae e_s and e_t together to form a correspondence (*e.g.*, $\langle$ o_1*:paperWrittenBy* , *dom(*o_2*:Paper)* $\sqcap$ o_2*:writes*$^-$, $\equiv$ $\rangle$ and express this correspondence in a reusable format (*e.g.*, EDOAL). The confidence assigned to a correspondence is the similarity score of the DL formula computed.

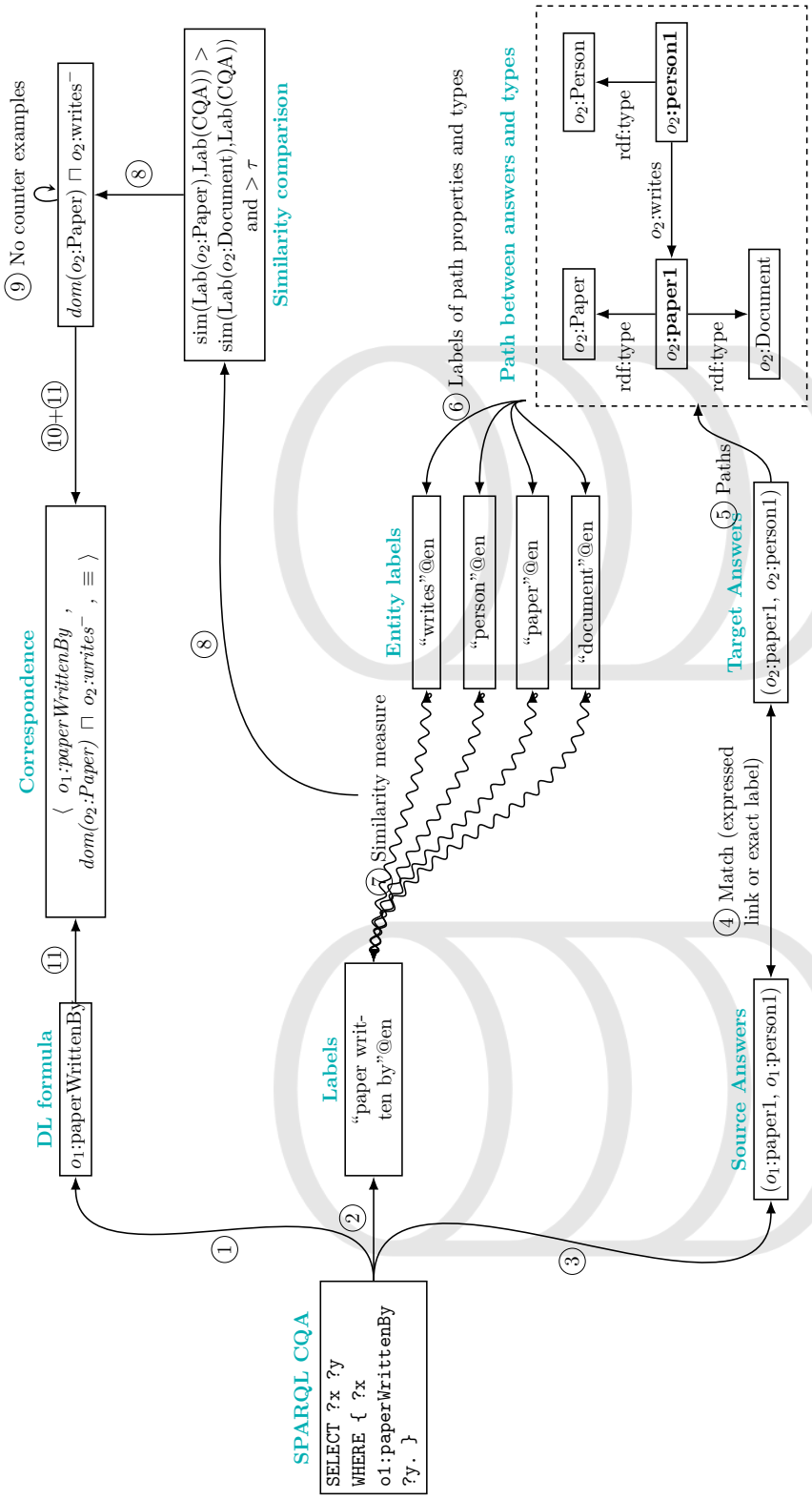


Figure 4.5: Binary CQA example: *Who is the author of which paper ?*

4.3 Main steps of the approach

In this section, we detail the steps (1), (4), (5), (7), (8), (9) and (10) of Figure 4.1 and illustrate them with examples.

4.3.1 Translating SPARQL CQAs into DL formulae

```
SELECT ?x WHERE {
  ?x o2:hasDecision ?y.
  ?y a o2:Acceptance.
}
```

Figure 4.6: SPARQL SELECT query with one variable in SELECT clause

In Step (1), in order to translate a SPARQL query (*e.g.*, in Figure 4.6) into a DL formula (*e.g.*, $\exists$ *o2:hasDecision.o2:Acceptance*), we start by translating it into a FOL formula and then transform it into a DL formula.

A SPARQL SELECT query (in the scope of our approach) is composed of a SELECT clause containing variable names and a basic graph pattern, *i.e.*, a set of triples with variables sometimes with constructors (such as UNION or MINUS).

First, the variables in the SELECT clause become the quantified variables of our formula. In unary CQAs, the SELECT clause contains one variable. In binary CQAs, the SELECT clause contains two variables. The SPARQL query of Figure 4.6, `?x` becomes the quantified variable of our formula: $\forall$ *x*.

Then, we parse the basic graph pattern to find what predicates apply to the quantified variables and add them to the formula. Each triple of the basic graph pattern is either a unary or a binary predicate. If new variables are added, we use an existential quantifier for them.
In the example, we find the triple $\langle$ `?x` , *o2*:hasDecision , `?y` $\rangle$. The FOL formula becomes $\forall$ *x,* $\exists$ *y, o2:hasDecision(x,y).*

We then recursively keep on exploring the basic graph pattern for each new variable introduced. After exploring the basic graph pattern for the variable `?y`, the FOL formula becomes $\forall$ *x,* $\exists$ *y, o2:hasDecision(x,y)* $\wedge$ *o2:Acceptance(y).*

At the end of the process, we transform the basic graph pattern into a DL formula, which can also be translated into an EDOAL formula as shown below.
$\forall$ *x,* $\exists$ *y, o2:hasDecision(x,y)* $\wedge$ *o2:Acceptance(y)* becomes in DL:
$\exists$ *o2:hasDecision.o2:Acceptance.* The FOL to DL equivalence is done as in [Borgida 1996].

4.3.2 Instance matching

In Step (4), the answers of the CQA over the source knowledge base which have been retrieved are matched with the instances of the target knowledge base. This instance matching phase relies on existing links (*owl:sameAs, skos:exactMatch,*

skos:closeMatch, etc.) if they exist. If no such link exists, an exact label match is performed.

When dealing with binary CQAs whose results are an instance-literal value pair, the instance is matched as before (existing links or exact labels), the literal value will be matched with an exactly identical value (the datatype is not considered) in the path finding step, detailed in Section 4.3.3.

4.3.3 Retrieving and pruning subgraphs

The whole approach relies on subgraphs, which are sets of triples from a knowledge base. We detail here how these subgraphs are found (Step (5)), pruned and transformed into DL formulae (Step (8)). The type of subgraphs for unary or binary CQAs is inspired from [Zheng *et al.* 2016], which proposes an approach to find equivalent subgraphs within the same knowledge base.

A unary CQA expects a set of single instances as answer. The subgraph of a single instance is composed of a triple in which the instance is either the subject or the object, and the types (classes) of the object or subject of this triple.

For example, *o2:paper1* is the subject of the triple *o2:paper1 o2:hasDecision o2:decision1* and *o2:decision1* has types (classes) *o2:Acceptance* and *o2:Decision*. A subgraph of *o2:paper1* is therefore composed of the three following triples:

1. ⟨ *o2:paper1 , o2:hasDecision , o2:decision1* ⟩
2. ⟨ *o2:decision1 , rdf:type , o2:Acceptance* ⟩
3. ⟨ *o2:decision1 , rdf:type , o2:Decision* ⟩

When comparing the subgraph with the CQA labels, if the most similar object (resp. subject) type is more similar than the object (resp. subject) itself, the type is kept. Let us consider the *accepted paper* CQA. The most similar type of the triple of the object is *o2:Acceptance*. Therefore, triple 3 is pruned.

The object of triple 1 is *o2:decision1* and the most similar object type to the CQA is *o2:Acceptance*. *o2:Acceptance* is more similar to the CQA than *o2:decision1*. Therefore, *o2:decision1* becomes a variable and triple 2 stays in the subgraph.

To translate a subgraph into a DL formula, we first translate this subgraph into a SPARQL query:

- The answer is transformed into a variable and put in the SELECT clause. In this example, *o2:paper1* becomes a variable `?x` in the SELECT clause: `SELECT ?x WHERE`.
- The instances of the subgraphs which are not kept are transformed into variables. In this example, *o2:decision1* becomes a variable `?y`.
- These transformations are applied to the selected triples of the subgraph which become the basic graph pattern of the SPARQL query. In this example, the SPARQL query is the one in Figure 4.6.

Finally, the SPARQL query is transformed into a DL formula by using the same process as that described in Section 4.3.1: $\exists o_2{:}hasDecision.o_2{:}Acceptance$.

A binary CQA expects a set of pairs of instances (or pairs of instance-literal value) as answer. Finding a subgraph for a pair of instances consists in finding a path between the two instances. The shortest paths are considered more accurate. Because finding the shortest path between two entities is a complex problem, paths of length below a threshold are sought. First, paths of length 1 are sought, then if no path of length 1 is found, paths of length 2 are sought, *etc.*

If more than one path of the same length are found, all of them go through the following process. When a path is found, the types of the instances forming the path are retrieved. If the similarity of the most similar type to the CQA is above a threshold, this type is kept in the final subgraph.

For example, for a "*paper written by*" CQA with the answer ($o_2{:}paper1$,$o_2{:}person1$) in the target knowledge, a subgraph containing the following triples is found:

1. $\langle\ o_2{:}person1\ ,\ o_2{:}writes\ ,\ o_2{:}paper1\ \rangle$
2. $\langle\ o_2{:}paper1\ ,\ rdf{:}type\ ,\ o_2{:}Paper\ \rangle$
3. $\langle\ o_2{:}paper1\ ,\ rdf{:}type\ ,\ o_2{:}Document\ \rangle$
4. $\langle\ o_2{:}person1\ ,\ rdf{:}type\ ,\ o_2{:}Person\ \rangle$

The most similar type of $o_2{:}person1$ is $o_2{:}Person$, which is below the similarity threshold. Triple 4 is then removed from the subgraph. The most similar type of $o_2{:}paper1$ is $o_2{:}Paper$. Triple 3 is therefore removed from the subgraph. $o_2{:}Paper$'s similarity is above the similarity threshold: triple 2 stays in the subgraph. The translation of a subgraph into a SPARQL query is the same for binary and unary CQAs. Therefore, the subgraph will be transformed into a SPARQL query and saved as the following DL formula: $dom(o_2{:}Paper) \sqcap o_2{:}writes^{-}$.

4.3.4 Label similarity

In Step (7), two sets of labels are compared and a similarity score is output from this comparison. Many lexical metrics have been proposed in the literature. [Cheatham & Hitzler 2013] survey their use in the ontology matching tasks.

Before their comparison, the labels can be preprocessed in a syntactic way (*e.g.*, tokenisation, normalisation, lemmatisation, stop word removal) or in a linguistic way (*e.g.*, synonymy, antonymy, translation in other languages, abbreviation expansion).

Then the comparison metrics (*e.g.*, TF-IDF, Levenshtein distance, Jaccard) can be classified in three axes: global versus local, set versus whole string, and perfect-sequence versus imperfect-sequence [Cheatham & Hitzler 2013]. A global

comparison considers extra label information such as labels from the whole ontologies whereas for local comparison, only the compared labels are needed as input. Set-based metrics rely on decomposed labels and compare the words to one another. The comparison of each word in the set-based metrics is performed using a whole string comparison metric. Perfect-sequence metrics require characters to occur in the same position in both strings while imperfect-sequence allow for a difference of character position.

The choice of the implementation of the approach are detailed in Chapter 6 in Section 6.1.

4.3.5 DL formula aggregation

In Step (8) of the approach, when dealing with unary CQAs, the DL formulae can be aggregated. It consists in transforming one or more formulae with a common predicate into a more generic formula. This aggregation only applies to formulae which contain an instance or a literal value and which were kept in the subgraph selection step. For example, this step would apply for a formula such as $\exists$ *o2:hasDecision.*{*o2:accept*}.

There are three steps to the aggregation. First, we create a first aggregated formula which we call the **extension** formula. It consists in merging the instances or literal values of the formulae with the same predicate into one set of values. Let us consider that through various answers to a CQA (*e.g.*, *o2:paper1*, *o2:paper2*, *etc.*), we encountered the following formulae:

- $\exists$ *o2:hasDecision.*{*o2:accept*}
- $\exists$ *o2:hasDecision.*{*o2:strongAccept*}
- $\exists$ *o2:hasDecision.*{*o2:weakAccept*}

The extension formula of these formulae is:

$\exists$ *o2:hasDecision.*{*o2:accept, o2:strongAccept, o2:weakAccept*}.

The extension formula of a formula which does not share its predicate with any other is the formula itself.

Then, an **intension** formula can be computed by replacing the set of values by the top class $\top$. The intension formula of the example formulae is:

$\exists$ *o2:hasDecision.*$\top$.

Finally, a choice is made between the extension or intension formulae based on the predicate similarity to the CQA. If the predicate is more similar than the values, the intension formula is kept. Otherwise, the extension formula is kept.

In our example, the extension formula $\exists$*o2:hasDecision.*{*o2:accept, o2:strongAccept, o2:weakAccept*} is kept.

We present two examples of initial formulae, with their respective intension and extension formulae in Table 4.1. These were obtained with the competency question *"accepted paper"*. In Table 4.1, the final formulae are in bold.

Applied to the examples of Table 4.1,

Table 4.1: Initial, extension, intension and final (in bold) formulae. The CQA considered is *"accepted papers"*.

Initial formulae	Extension	Intension
$\exists o_2$*:hasDecision.*$\{o_2$*:accept*$\}$ $\exists o_2$*:hasDecision.*$\{o_2$*:strongAccept*$\}$ $\exists o_2$*:hasDecision.*$\{o_2$*:weakAccept*$\}$	$\exists o_2$***:hasDecision.***$\{o_2$***:accept,*** o_2***:strongAccept,*** o_2***:weakAccept***$\}$	$\exists o_2$*:hasDecision.*$\top$
$\exists o_2$*:acceptedBy.*$\{o_2$*:person1*$\}$	$\exists o_2$*:acceptedBy.*$\{o_2$*:person1*$\}$	$\exists o_2$***:acceptedBy.***$\top$

- o_2*:accept*, o_2*:strongAccept* and o_2*:weakAccept* are more similar to the CQA than o_2*:hasDecision*. The extension form is chosen.

- o_2*:acceptedBy* is more similar (based on labels) to the CQA than o_2*:person1*. The intension form is chosen.

4.3.6 Calculating the percentage of counter-examples

In Step (9), the approach refines the DL formula similarity score by looking for counter-examples (details about the similarity score are given in Section 4.3.7). A **counter-example** is a common instance of the source and target ontologies which is described by the DL formula found by the approach in the target ontology but which is not described by the CQA in the source ontology.

For example, let us assume that the target formula e_t is o_2*:Paper* for the "accepted paper" CQA. From the target ontology, the answers o_2*:paper1,* o_2*:paper2,* o_2*:paper3* and o_2*:paper4* are retrieved from e_t and matched to the source instances respectively o_1*:paper1,* o_1*:paper2,* o_1*:paper3* and o_1*:paper4*. However, only o_1*:paper1* and o_1*:paper2* are accepted papers (and are described by the CQA) in the source ontology. Therefore o_1*:paper3* and o_1*:paper4* are counter-examples.

The percentage of counter-examples is computed as follows. The answers $ans_t^{e_t}$ described by the target subgraph (e_t) are retrieved from the target knowledge. These answers are matched to source instances: $ans_s^{e_t}$. The percentage of counter-examples is the proportion of common instances $ans_s^{e_t}$ which are not answers to the CQA ($\neg(ans_s^{cqa})$). The equation for the percentage of counter examples ($percCounterExamples$) is therefore:

$$percCounterExamples = \frac{|ans_s^{e_t} \sqcap \neg(ans_s^{cqa})|}{|ans_s^{e_t}|}$$

In the example, the percentage of counter-example is $\frac{2}{4} = 50\%$.

4.3.7 DL formula similarity

In Step (10), the formulae are filtered based on their similarity score with the CQA. The similarity score is a combination of:

Label similarity *labelSim* is the sum of the label similarity of each entity of the formula with the CQA.

Structural similarity *structSim*. This similarity was introduced to enhance some structural aspects in a formula. In the implementation of the approach, this value is set to 0.5 for a path between the two instances of the answer, and 0 for a unary CQA subgraph. Indeed, if the label similarity of the path is 0, the structural similarity hints that the fact that a path was found is a clue in favour of the resulting DL formula.

Percentage of counter examples *percCounterExamples* which is computed in Step (9) and detailed Section 4.3.6.

The similarity score is calculated with the following equation:

$$similarity = (labelSim + structuralSim) \times (1 - percCounterExamples) \quad (4.1)$$

The three examples below illustrate the computation of the similarity.

1. Similarity of $\exists o_2$*:hasDecision.*o_2*:Acceptance* with the unary CQA *"accepted paper"*.
   - $labelSim = 0.8 + 0.0$ because
     - *sim(labels(CQA), labels(*o_2*:hasDecision))* $= 0.0$
     - *sim(labels(CQA), labels(*o_2*:Acceptance))* $= 0.8$
   - $strucSim = 0.0$ because it is a unary CQA
   - $percCounterExamples = 0.0$

   The similarity of this DL formula is $similarity = (0.8 + 0.0) \times (1 - 0) = 0.8$
2. With the unary CQA *"accepted paper"* and the example from Section 4.3.6 (o_2*:Paper*), we have:
   - *labelSim= sim(labels(CQA), labels(*o_2*:Paper))* $= 0.6$
   - $structSim = 0$ because it is a unary CQA subgraph
   - $percCounterExamples = 0.5$ because there were 2 counter-examples out of four common instances between the target DL formula and the CQA

   The similarity of this DL formula is $similarity = (0.6 + 0.0) \times (1 - 0.5) = 0.3$
3. Let us consider an *"author of"* (o_1*:authorOf*) binary CQA. The path o_2*:writes* is found between two matched answer instances.
   - *labelSim= sim(labels(CQA), labels(*o_2*:writes))* $= 0.0$
   - $structSim = 0.5$ because it is a path between the instance answers
   - $percCounterExamples = 0.0$

   The similarity of this DL formula is $similarity = (0.0 + 0.5) \times (1 - 0.0) = 0.5$

4.3.8 DL formula filtering

In Step (10), the formulae are filtered. Only the DL formulae with a similarity higher than a threshold are put in a correspondence with the CQA DL formula. If for a given CQA, there is no DL formula with a similarity higher than the threshold, only the best DL formulae with a non-zero similarity are put in the correspondence. The best DL formulae are the formulae with the highest similarity score.

When putting the DL formula in a correspondence, if its similarity score is greater than 1, the correspondence confidence value is set to 1.

4.4 Positioning and conclusion

We propose a complex alignment generation approach based on CQAs. The CQAs define the knowledge needs of a user over two or more ontologies. The use of CQAs is both a strength of the approach as it allows for a generalisation over few instances and a limitation as it requires that the user is able to express her or his needs as SPARQL queries.

SPARQL CQA In our approach, CQAs are used as basic pieces of information which will be transformed as source members of correspondences. Their formulation in a SPARQL query over the source ontology is a limitation of the approach as a user would need to be familiar with SPARQL and the source ontology. However, in the scenario where someone wants to publish and link a knowledge base he or she created on the LOD cloud, this person is already familiar with the source ontology and can reuse the CQs of their own ontology. In other cases, one could rely on question answering systems which generate a SPARQL query from a question in natural language. This kind of system is evaluated in the Question Answering over Linked Data (QALD) open challenge [Unger *et al.* 2014].

Generalisation process Ontology matching approaches relying on the $\mathcal{A}box$of ontologies infer general statements from the instances, *i.e.*, they perform a generalisation[1]. This is the principle of *machine learning* in general and methods such as *Formal Concept Analysis* [Ganter *et al.* 2005] or *association rule mining* [Agrawal *et al.* 1993]. These generalisation processes however require a considerable amount of data (or instances). Approaches such as the ones from [Walshe *et al.* 2016, Parundekar *et al.* 2010, Parundekar *et al.* 2012, Hu *et al.* 2011] rely on large amounts of common ontology instances for finding complex correspondences. Few exceptions in ontology matching rely on few examples. For instance, the matcher of [Wu & Knoblock 2015] relies on example instances given by a user. With this information, the generalisation can be performed on few examples. The idea behind our approach is to rely on a few examples to find general

[1] 'They infer general statements or concepts from specific cases' (Oxford Online English Dictionary, "Generalisation" Retrieved June 3 2019 from `https://en.oxforddictionaries.com/definition/generalization`

Table 4.2: Approach positioning with regards to Tables 3.4, 3.5, 3.6 and 3.7

Approach	Type of Knowledge Representation Model	Additional Input
CANARD	OWL ontology to OWL ontology	CQAs

Approach	(s:c)	(c:s)	(c:c)	Logic	Transfo	Block	Correspondence format
CANARD	•	•	•	•			EDOAL, DL

Approach	Guiding structure	fixed to fixed	fixed to unfixed	unfixed to unfixed	Ontology-level evidence	Instance-level evidence	Other
CANARD	No structure, Path to path		•		•	•	

Approach	Formal resource-based	Informal resource-based	String-based	Language-based	Constraint-based	Taxonomy-based	Graph-based	Instance-based	Model-based
CANARD	•		•				•	•	

rules which would apply to more instances. In particular, the generalisation phase of our approach is guided by the CQA labels. Thanks to that, only one instance is sufficient for finding a correspondence. This would apply to knowledge bases which represent different contexts or points of view but whose ontologies are overlapping.

Classification of the approach We position our approach following the characteristics presented in Chapter 3 in Table 4.2. The name of the approach implementation is CANARD. CANARD can generate *(s:s), (s:c)* and *(c:c)* correspondences depending on the shape of the input CQA. It focuses on correspondences with logical constructors. The approach relies on a path to find the correspondences for binary CQAs. For the unary CQAs, we classify CANARD as *no structure* because it does not explicitly rely on atomic or composite patterns. The source member form is fixed before the matching process by the CQA but the target member form is unfixed, therefore we classify it as *fixed to unfixed*. CANARD relies on ontology and instance-level evidence. CANARD fits in the formal resource-based because it relies on CQAs and existing instance links, its implementation is string-based because of the label similarity metric chosen (see Section 6.1), it is also graph-based and instance-based.

To evaluate the proposed matcher, a complex alignment benchmark is needed, as introduced in the next chapter.

Chapter 5

Automatic evaluation of complex alignment

Content

In Chapter 3, we have seen that the automatic evaluation of complex alignments is still an open issue. In this chapter, we analyse where the difficulty of evaluation of complex alignments comes from. This analysis leads to the proposition of a complex alignment evaluation system and its associated dataset.

We started by analysing automatic output-oriented alignment evaluation in general. We focused on extrinsic evaluation systems (see Section 2.2.7), *i.e.*, evaluation systems which require a reference alignment or reference queries. We summarise our analysis into a generic workflow presented in Section 5.1. At each step of the workflow, we identify the difficulties which are imputable to complex alignments.

The analysis of the workflow and the various automatic evaluation possibilities discussed in Section 5.1 lead to the proposition of an instance-based evaluation system described in Section 5.2.2. This proposition consists in two measures. First, the CQA Coverage measure relies on pairs of equivalent SPARQL queries and measures

how well an alignment covers these queries. However, if an alignment covers every possible correspondence between two ontologies, its CQA Coverage score would be high even if a lot of its correspondences are incorrect. For this reason, we complete our evaluation proposal with an intrinsic *Precision* evaluation based on instances (Section 5.2.3). This instance-based Intrinsic Precision relies on ontologies populated with the same instances to compare the correspondences' member's interpretation. This metric balances the CQA Coverage like precision balances recall in information retrieval.

Since an evaluation system must be run on a dataset, we propose a new version of the Conference dataset with instances and CQAs in Section 5.3. We conclude by discussing limitations and positioning of our proposition in Section 5.4.

5.1 Evaluation workflow

As discussed in Section 3.3, ontology alignment evaluation is often performed by comparing a generated alignment to a reference one. Most of the OAEI tracks use this kind of evaluation. However, the reference of the evaluation can also take other forms such as merged ontologies with their transitive closure or equivalent queries (*i.e.*, a query over the source ontology and its equivalent for the target ontology). Even though these types of evaluation are developed and automated for simple alignments, complex alignments are still mostly manually evaluated [Thiéblin *et al.* 2018a]. The purpose of this section is to identify the difficulties inherent to complex alignment evaluation and discuss how they can be overcome. We start by dissecting the alignment evaluation process into a generic workflow in Section 5.1.1. We then present the specificities of simple (Section 5.1.2) and complex (Section 5.1.3) alignment evaluation.

5.1.1 Generic workflow

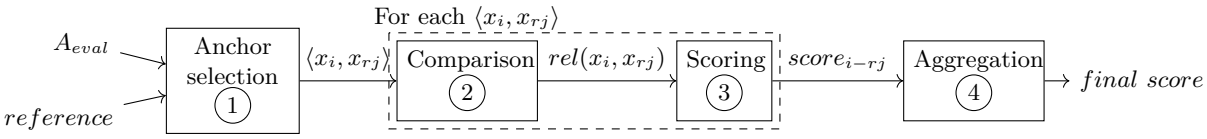


Figure 5.1: Evaluation process of the alignment A_{eval} with a generic $reference$

In this section, we analyse the alignment evaluation process with a reference, regardless of its type. Figure 5.1 presents the generic workflow resulting from this analysis. This workflow applies for simple and complex alignment evaluation.

Overall, the steps followed in the evaluation process are:

(1) **Anchor selection** The anchor selection step consists of outputting a pair of comparable objects $\langle x_i, x_{rj} \rangle$. x_i is an object related to the evaluated alignment

A_{eval} and x_{rj} is an object related to the reference $reference$. The objects depend on the type of reference. For example, if the reference is an alignment, x_i is a correspondence (c_i) from A_{eval}, x_{rj} is a correspondence (c_{rj}) from the reference alignment. If the reference is equivalent queries, x_i can be a query derived from A_{eval} and x_{rj} a reference query.

(2) **Comparison** The purpose of the comparison step is to output a relation $rel(x_i, x_{rj})$ for each pair previously obtained $\langle x_i, x_{rj}\rangle$. The relation can be an equivalence (*i.e.*, $x_i \equiv x_{rj}$), a subsumption, an overlap, a disjoint, *etc.* (this list can be extended according to the type of comparison performed). A similarity value can be associated with the relation. The comparison can be syntactic, semantic or instance-based as developed in Sections 5.1.2 and 5.1.3.

For correspondence comparison (if the reference is an alignment), $x_i = c_i = \langle e_i, e'_i, r_i, n_i\rangle$ and $x_{rj} = c_{rj} = \langle e_{rj}, e'_{rj}, r_{rj}, n_{rj}\rangle$. Each element of an evaluated correspondence should be compared to its counterpart in the reference correspondence. $rel(c_i, c_{rj})$ can be decomposed into the relations between the elements of c_i and c_{rj}: source members (e_i, e_{rj}), target members (e'_i, e'_{rj}), relations (r_i, r_{rj}) and confidence values (n_i, n_{rj}). A similarity score can be added to each relation between components.

$$rel(c_i, c_{rj}) = \begin{cases} rel(e_i, e_{rj}) \\ rel(e'_i, e'_{rj}) \\ rel(r_i, r_{rj}) \\ rel(n_i, n_{rj}) \end{cases} \tag{5.1}$$

(3) **Scoring** The scoring step associates a score with each relation found in the previous step. Thus, the scoring functions are directly impacted by the relation $rel(x_i, x_{rj})$ found between the objects. The scoring function gives a score between 0 (for incorrect) and 1 (for correct). Different scoring metrics such as relaxed, recall or precision-oriented metrics [Ehrig & Euzenat 2005], *etc.* have been proposed in the literature and implemented in the Alignment API [David *et al.* 2011].

The score can also be that which was associated with the relation found in the previous step. For example, if the comparison was syntactical and based on an edit distance, the edit distance value associated with $rel(x_i, x_{rj})$ can directly be used as $score_{i-rj}$. When using an instance-based comparison, a percentage of relevant instances can be associated with $rel(x_i, x_{rj})$ as in [Hollink *et al.* 2008]. We call the scores which use the similarity value obtained during the comparison phase, the **comparison value** scoring functions. When dealing with correspondences, their confidence value can also be incorporated into the score, as in the weighted precision and recall metrics.

The variety of scoring functions and all their possible combinations point out that there is not one consensual way to measure the compliance of an align-

ment with regard to a reference. There really is no "best scoring function" or "best metric". It depends on what the evaluation is supposed to measure. For example, if the evaluation measures how well an alignment allows for retrieving all results for a given query, regardless of the precision, a *recall-oriented* score can be applied [Ehrig & Euzenat 2005]. If the purpose of the evaluation is to measure the exactitude of an alignment, then a classical function (1 if correct, 0 if incorrect) can be applied.

④ **Aggregation** The scores are locally and globally aggregated to give the *final score*. The aggregations can be performed with different functions: best match, average, weighted average, *etc.*

The local aggregation aggregates all scores for a given object. There can be different local aggregations. For example, there can be an aggregation over the evaluated object and one over the reference object.

The global aggregation aggregates all the locally-aggregated scores. For example, if the local aggregation was performed over the reference object, all the reference objects were given a score. The reference object scores can be aggregated into a final score.

A final score locally aggregated over the evaluated objects is often referred to as the *precision* score. A final score locally aggregated over the reference objects is often referred to as the *recall* score.

The differences between simple and complex alignment evaluation lie in the Anchor selection ① and Comparison ② steps. We detail how they are performed for simple alignments in Section 5.1.2 and what are the challenges for their application to complex alignments in Section 5.1.3.

5.1.2 Workflow for simple alignment evaluation

Anchor selection The anchor selection step consists of outputting a pair of comparable objects. In a simple alignment, each correspondence consists of a pair of URIs linked by a relation and potentially a confidence.

The anchor selection can be performed by outputting all pairs of correspondences whose source member or target member are equivalent. As the source and target members of simple correspondences are URIs, an exact string match between the URIs is sufficient. Let us consider the evaluated correspondences c_1=⟨ *o*$_1$*:Paper* , *o*$_2$*:Paper* , ≡ ⟩ c_2=⟨ *o*$_1$*:Paper* , *o*$_2$*:Document* , ≡ ⟩ and the reference correspondence c_{r1}=⟨ *o*$_1$*:Paper* , *o*$_2$*:Paper* , ≡ ⟩. The pairs $\langle c_1, c_{r1}\rangle$ and $\langle c_2, c_{r1}\rangle$ are formed by comparing their source member to that of the reference correspondence.

In the case of reference queries, the anchoring phase consists of translating a source query based on the evaluated alignment. That means that the evaluated alignment is used for generating a query in terms of the target ontology translating a query in terms of the source ontology. The output pair consists of the generated query and the reference target one. For simple alignments, the query rewriting

system can consist in replacing each URI from the source query by an equivalent found in the evaluated alignment. For instance, the reference source query q_{rs} `SELECT ?x WHERE{?x a o1:Paper.}` gives :

- q_1 with c_1 `SELECT ?x WHERE{?x a o2:Paper.}`
- q_2 with c_2 `SELECT ?x WHERE{?x a o2:Document.}`

The reference query in this scenario is q_{rt}: `SELECT ?x WHERE{?x a o2:Paper.}`. The pairs $\langle q_1, q_{rt}\rangle$ and $\langle q_2, q_{rt}\rangle$ are formed.

Comparison The purpose of the comparison step is to output a relation $rel(x_i, x_{rj})$ for each pair previously obtained $\langle x_i, x_{rj}\rangle$. The comparison of the objects can be performed in a syntactic, semantic or instance-based manner.

A **syntactic** comparison compares the string representations of the objects. When dealing with simple alignments, the correspondences member URIs are compared. As the URIs are strings, a syntactical comparison is enough. This kind of comparison is the most common in the OAEI simple tracks. This kind of comparison is limited to stating whether objects (correspondences, queries, *etc.*) are equivalent, syntactically similar or different. In the example correspondence, the source, target members, and relations of c_1 and c_{r1} are syntactically equivalent. c_2 is syntactically different from c_{r1} because their target member differ. q_1 is syntactically equivalent to q_{rt}, q_2 is not.

A **semantic** comparison is based on reasoning rules. [Ehrig & Euzenat 2005] propose to compute whether a simple evaluated correspondence is more specific or more general than the reference one based on taxonomic inference. c_1 is semantically equivalent to c_{r1}. c_2 is more general than c_{r1} because *o_2:Paper* is a subclass of *o_2:Document*. In [David *et al.* 2018], a comparison between queries without instances can be performed based on inference rules. The semantic comparison does not depend on an ontology population. It can rely on existing reasoners and would work with every construction possible of the same axiom (inverse of inverse property, equivalent classes, *etc.*). q_1 is semantically equivalent to q_{rt}, q_2 is more general than q_{rt}.

An **instance-based** comparison is based on the interpretation of the objects in knowledge bases where the aligned ontologies have an associated $\mathcal{A}box$. The instance-based comparison only needs comparing sets of URIs. There is no expressiveness restriction for the evaluated alignment. The syntactic form of the correspondence does not matter. Therefore it can be used in the same manner for simple or complex correspondences. However, it fully relies on the ontologies' $\mathcal{A}box$. If the $\mathcal{A}box$ contains errors, or is irregular, the comparison results can be erroneous. If the target ontology o_2 is only populated with *o_2:Paper* instances (if there are no *o_2:Document* instances which are not *o_2:Paper*), then c_2 (resp. q_2) could be found equivalent to c_{r1} (resp. q_{rt}).

5.1.3 Workflow for complex alignment evaluation

The first issue when dealing with complex alignment evaluation is the creation of the reference. In [Thiéblin *et al.* 2018b], the proposed reference alignments were limited to *(s:s) (s:c)* and *(s:c)* correspondences and by the methodology. This way, a certain completude could be ensured: an evaluated correspondence which would fall into the reference alignment creation criteria could be classified as correct or incorrect. When dealing with no restriction on the shapes of the correspondences, it becomes hard to prove that a reference alignment covers every possible correct correspondence.

Anchor selection As introduced before, the anchor selection step consists of outputting a pair of comparable objects. In comparison with simple alignments, complex correspondence members are not limited to URIs. They therefore require more than a simple syntactic match.

When dealing with a reference alignment, the *(s:c)* or *(c:s)* correspondences can be anchored on their simple member. For example, the *(s:c)* evaluated correspondence $\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}acceptedBy.\top\ ,\ \equiv\ \rangle$ can be put in pair with the reference *(s:c)* correspondence $\langle\ o_1{:}AcceptedPaper\ ,\ \exists o_2{:}hasDecision.o_2{:}Acceptance\ ,\ \equiv\ \rangle$ because their source members are the same URI ($o_1{:}AcceptedPaper$). However, this kind of anchoring is not as easily applicable for *(c:c)* correspondences. For example, it is not clear if $\langle\ \exists o_3{:}accepted.\{true\}\ ,\ \exists o_2{:}hasDecision.\top\ ,\ \equiv\ \rangle$ and $\langle\ \exists o_3{:}accepted.\top\ ,\ \geq 1\ o_2{:}hasDecision.o_2{:}Acceptance\ ,\ \equiv\ \rangle$ should be put in a pair and compared.

In the case of reference queries, the anchoring phase consists of translating a source query based on the evaluated alignment. A query rewriting system dealing with complex correspondences is thus needed. In the literature, query rewriting systems only deal with *(s:c)* correspondences [Correndo & Shadbolt 2011, Makris *et al.* 2012, Thiéblin *et al.* 2016]: they translate a source URI into an equivalent construction based on the correspondence. Dealing with *(c:s)* and *(c:c)* correspondences for query rewriting remains a challenge.

Comparison The purpose of the comparison step is to output a relation for each pair of objects previously obtained. As for simple alignments, the comparison can be syntactic, semantic or instance-based.

A **syntactic** comparison for complex correspondences could measure how much effort should be done to transform an evaluated correspondence into the reference one. However correspondences which use different constructors, or different levels of factorisation can express the same meaning. A syntactic comparison also depends on the language in which the correspondences are expressed. Such a comparison strongly depends on the way the reference correspondences, queries, *etc.* are expressed.

For example, $\langle\ o_1{:}Author\ ,\ \exists o_2{:}authorOf.\top\ ,\ \equiv\ \rangle$ is semantically equivalent to the correspondence $\langle\ o_1{:}Author\ ,\ \exists o_2{:}writtenBy^{-}.\top\ ,\ \equiv\ \rangle$. However, these

two correspondences use different URIs in their constructors and thus are syntactically different. The correspondences $\langle$ *o1:AcceptedPaper* , $\exists$*o2:acceptedBy.*$\top$, $\equiv$ $\rangle$ and $\langle$ *o1:AcceptedPaper* , $\geq$ 1 *o2:acceptedBy.*$\top$, $\equiv$ $\rangle$ are equivalent but expressed using different constructors (respectively an existential restriction or a cardinality restriction over the *o2:acceptedBy* property). They are also syntactically different. A factorisation problem would consist in verifying that $\langle$ *o1:paperWrittenBy* , *dom(o2:Paper)* $\sqcap$ *o2:writes*$^-$, $\equiv$ $\rangle$ and $\langle$ *o1:paperWrittenBy* , *(o2:writes* $\sqcap$ *range(o2:Paper))*$^-$, $\equiv$ $\rangle$ are equivalent correspondences. The *inverse* constructor is factorised in the second correspondence. A syntactic comparison of queries is faced with the same problems: syntactically different SPARQL queries can share the same semantics.

A **semantic** comparison would then be an alternative solution. However, the expressiveness of the evaluated alignment with a semantic comparison is limited to $\mathcal{SROIQ}$ (the decidable fragment of OWL [Horrocks *et al.* 2006]). Correspondences with transformation functions could not be compared with such a comparison. The semantic query comparison proposed by [David *et al.* 2018] is based on query containment which can be based on inferences. However, it is also limited with regard to queries with transformation functions.

As discussed in Section 5.1.2, **instance-based** comparison is applicable for simple and complex alignments. However, it requires the knowledge bases to be regularly populated. Figure 5.2 shows examples of populated ontologies. The dataset D2 (in Figure 5.2(b)) is irregularly populated because the *o2:acceptedBy* property is only instantiated for *o2:paper1* whereas every accepted paper is supposed to be *o2:acceptedBy* someone[1].

Let us consider what an instance-based comparison would give on the dataset D1 and D2 of Figure 5.2. The anchoring step was performed on the source member of the correspondence, therefore, we compare the target members based on the returned instances. We consider the reference correspondence:

- $c_{r1} = \langle$ *o1:AcceptedPaper* , $\exists$*o2:hasDecision.o2:Acceptance* , $\equiv$ $\rangle$

The correspondence c_2 will be compared to c_{r1}:

- $c_2 = \langle$ *o1:AcceptedPaper* , $\exists$*o2:acceptedBy.*$\top$, $\equiv$ $\rangle$

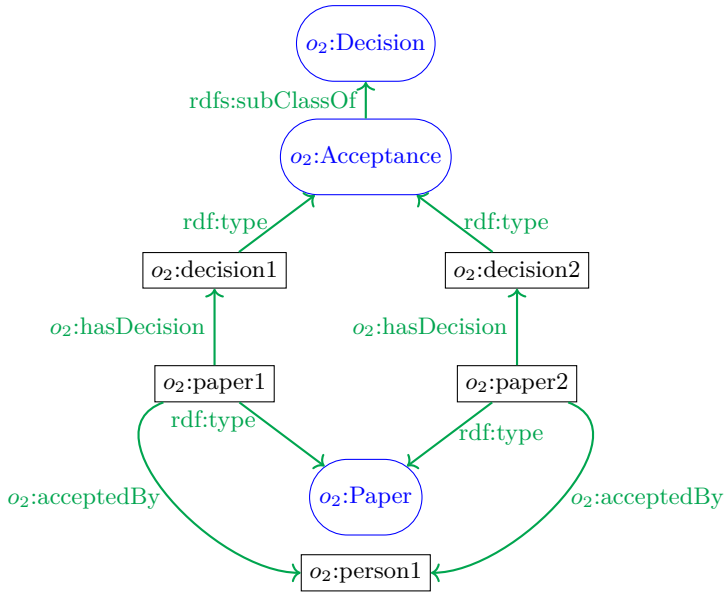
In D1, the instances described by the target member of the correspondences are:

- c_{r1}: (*o2:paper1*, *o2:paper2*)
- c_2: (*o2:paper1*, *o2:paper2*)

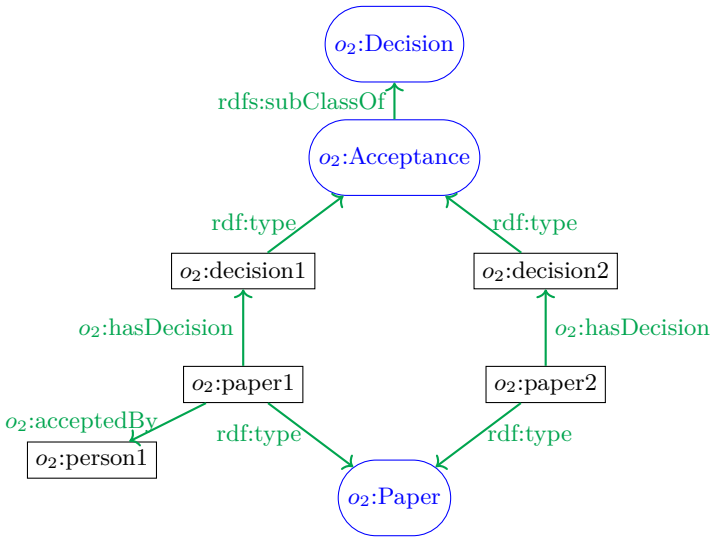
This leads to $c_2 \equiv c_{r1}$ which is correct (the only difference between the correspondences of the example is their target member, therefore, we write here the relation between the target member as the relation between the correspondences). D1 is suited for instance-based comparison.

In D2, the instances described by the target member of the correspondences are:

[1]The example ontology is inspired from the *cmt* ontology in the OAEI Conference dataset. See Section 3.3.1 for details about this dataset.



(a) D1 – Ontology with regular population for instance-based comparison



(b) D2 – Ontology with irregular population for instance-based comparison

Figure 5.2: The same ontology with different population (see Appendix A.3 for schema legend)

- c_{r1}: (o_2*:paper1*, o_2*:paper2*)
- c_2: (o_2*:paper1*)

This leads to $c_2 \sqsubseteq c_{r1}$ which is not correct. D2 implied errors in the comparison and was not suited for instance-based comparison.

We have identified that **reference creation**, **anchor selection** and **comparison** are the most difficult steps to automate for complex alignment. The instance-based comparison seems promising if run on a dedicated dataset. Using equivalent SPARQL CQAs (introduced in Section 4.1) as reference would ensure that the two compared objects are equivalent because they model the same piece of knowledge. The anchoring step could then be performed with query rewriting systems. However, query rewriting dealing with *(c:s)* and *(c:c)* remains a challenge. Finally, comparison can be performed based on instance sets relations.

5.2 Proposition of an instance-based evaluation system

5.2.1 Instance-based comparison and scoring

As stated before, instance comparison is more straightforward than syntactic or semantic comparison. In this section, we present the relations and scoring functions that will be used in the proposed metrics. Here, we reduce the comparison step to the comparison of two instance sets. This simplifies the process: only one relation will be computed instead of four as in Equation 5.1.

In this section, we present the relations and scoring functions that will be used in the CQA Coverage and Intrinsic Precision calculation.

Given a reference set of instances I_{ref} and an evaluated set of instances I_{ev}, two metrics are used to define their relation: QP and QR (respectively called query precision and query recall) in Equation 5.2. The possible relations between I_{ref} and I_{ev} are represented in Equation 5.3.

$$QP = \frac{|I_{ev} \cap I_{ref}|}{|I_{ev}|} \qquad QR = \frac{|I_{ev} \cap I_{ref}|}{|I_{ref}|} \tag{5.2}$$

$$rel(I_{ref}, I_{ev}) = \begin{cases} \equiv & \text{if } I_{ev} \equiv I_{ref} \textit{ i.e.}, QR = 1 \text{ and } QP = 1 \\ \sqsubseteq & \text{if } I_{ev} \subseteq I_{ref} \textit{ i.e.}, 0 < QR \leq 1 \text{ and } QP = 1 \\ \sqsupseteq & \text{if } I_{ev} \supset I_{ref} \textit{ i.e.}, QR = 1 \text{ and } 0 < QP \leq 1 \\ \between & \text{if } I_{ev} \cap I_{ref} \neq \emptyset \textit{ i.e.}, 0 < QR \leq 1 \text{ and } 0 < QP \leq 1 \\ \overset{\emptyset}{=} & \text{if } I_{ev} = I_{ref} = \emptyset \\ \perp & \text{if } I_{ev} \cap I_{ref} = \emptyset \text{ and } I_{ev} \cup I_{ref} \neq \emptyset \end{cases} \tag{5.3}$$

Based on the relation between the instance sets, the query precision (QP) and query recall (QR), we propose a set of scoring functions.

These different functions are complementary. The **classical** (Equation 5.4), **recall-oriented** (Equation 5.5) and **precision-oriented** (Equation 5.6) scoring functions are used in state-of-the-art works to emphasize whether the alignment favours precision or recall [Ehrig & Euzenat 2005]. We introduce the **overlap** metric to represent whether two queries have at least one common answer (Equation 5.7). The **not disjoint** metric gives a 1 score to all the overlapping queries and the queries where I_{ev} and $Iref$ are empty sets. The **query Fmeasure** scoring function (Equation 5.9) represents how close I_{ev} is to I_{ref}.

$$classical(I_{ref}, I_{ev}) = \begin{cases} 1 & \text{if } I_{ev} \equiv I_{ref} \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

$$recall\ oriented(I_{ref}, I_{ev}) = \begin{cases} 1 & \text{if } I_{ev} \sqsupseteq I_{ref} \\ 0.5 & \text{if } I_{ev} \sqsubseteq I_{ref} \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

$$precision\ oriented(I_{ref}, I_{ev}) = \begin{cases} 1 & \text{if } I_{ev} \sqsubseteq I_{ref} \\ 0.5 & \text{if } I_{ev} \sqsupseteq I_{ref} \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

$$overlap(I_{ref}, I_{ev}) = \begin{cases} 1 & \text{if } I_{ev} \between I_{ref} \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

$$not\ disjoint(I_{ref}, I_{ev}) = \begin{cases} 1 & \text{if } I_{ev} \between I_{ref} \text{ or } I_{ev} \overset{\emptyset}{=} I_{ref} \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

$$query\ Fmeasure(I_{ref}, I_{ev}) = 2 \times \frac{QR \times QP}{QR + QP} \tag{5.9}$$

Note that in the remainder of this manuscript, an instance set will be represented as such I_e^{KB}, e being the entity which represent the instances (formula or query) and KB the knowledge base in which this instance set was retrieved.

5.2.2 CQA Coverage

Computing a Recall for complex alignment is more complicated than computing a Precision. Indeed, in the latter case, the correspondences of an alignment can all be classified as "true positive" or "false positive" without a reference alignment. It is difficult to ensure that a reference complex alignment covers all possible correct correspondence, especially when including *(c:c)* correspondence. For that reason, we chose to evaluate how an alignment covers a set of knowledge needs represented by CQAs.

The reference for the CQA Coverage is a set of equivalent CQAs in the form of SPARQL queries. An evaluated alignment A will be used to rewrite each source

```
SELECT ?s WHERE {
  ?s a o1:AcceptedPaper.
}
```

(a) Query 1

```
SELECT ?s WHERE {
  ?s a o2:hasDecision ?o.
  ?o a o2:Acceptance.
}
```

(b) Query 2

Figure 5.3: Example SPARQL SELECT queries

CQA. The rewritten queries will then be compared to the reference target CQA. A best-match aggregation is locally performed to chose the rewritten query with the best *query Fmeasure* score (see Equation 5.9). Therefore, there is only one rewritten query left, $bestq_T$, for each pair of CQAs. A chosen scoring function is applied to each final query and these scores are averaged to give the CQA Coverage score.

In the following sections, we describe how each step of the workflow is implemented to compute the **CQA Coverage**.

The final equation of the CQA Coverage is shown in Equation 5.10. A is the evaluated alignment, cqa_{pairs} the set of pairs of CQAs as equivalent SPARQL queries, TKB is the target knowledge base, SKB the source knowledge base, f the chosen scoring function (chosen between Equations 5.4 – 5.9). *rewrite* is a query rewriting function described in 5.2.2.1.

$$\begin{aligned} coverage(A, cqa_{pairs}, SKB, TKB, f) = & \underset{\langle cqa_S, cqa_T\rangle \in cqa_{pairs}}{\text{average}} f(I^{TKB}_{cqa_T}, I^{TKB}_{bestq_T}) \\ bestq_T = & \underset{q_T \in rewrite(cqa_S, A, SKB)}{\text{argmax}} query\ Fmeasure(I^{TKB}_{cqa_T}, I^{TKB}_{q_T}) \end{aligned} \quad (5.10)$$

5.2.2.1 CQA query rewriting

As stated above, the reference in this kind of evaluation is a set of equivalent CQAs as SPARQL SELECT queries. Each source CQA cqa_S has an equivalent target CQA cqa_T.

In the anchoring step, each source cqa_S is rewritten using the generated alignment A. The rewriting phase outputs all the possible rewritten target queries from the rewriting systems as the set $Q_T = rewrite(cqa_S, A, SKB)$. For each rewritten query q_T in Q_T, a pair (q_T, cqa_T) is formed.

This strategy relies on query rewriting systems; we considered two of them, described below, in this work. None of these systems take account of the correspondence relation or confidence value.

System from [Thiéblin *et al.* 2016] The first system was proposed in [Thiéblin *et al.* 2016]. Each triple of cqa_S is rewritten using A. When the predicate or object of the triple appears as the source member of a correspondence in A, the target member of this correspondence is transformed into a SPARQL subgraph and put in the triple's place in the query. This system only deals with *(s:c)* correspondences. If a triple can be rewritten with different correspondences, all the possible combinations are added into Q_T. For example, consider the Query 1 from Figure 5.3(a) as CQA. It contains *o1:AcceptedPaper* which is the source member of the correspondences $c_1 = \langle$ o_1*:AcceptedPaper* , $\exists o_2$*:hasDecision.*o_2*:Acceptance* , $\equiv \rangle$. The rewritten query using c_1 is Query 2 from Figure 5.3(b).

This rewriting system cannot however work the other way around. For example, Query 2 from Figure 5.3(b) can not be rewritten with c_1.

Instance-based rewriting system The second system is based on instances. The instances $I^{SKB}_{cqa_S}$ of cqa_S are retrieved from the source knowledge base. For each correspondence c_i of A, the instances represented by its source member e_S are retrieved over the source knowledge base. If $I^{SKB}_{e_S} \equiv I^{SKB}_{cqa_S}$, then, the target member of c_i is transformed into a query and added to Q_T. For example Query 1 from Figure 5.3(a) retrieves a set of accepted paper instances in the o_1 ontology. This set of instances is then compared to the set of instances described by the source member of each correspondence. In this case, o_1*:AcceptedPaper* describes exactly the same set of instances as the source member of c_1. The target member of c_1 can therefore be transformed into Query 2 from Figure 5.3(b).

This rewriting system allows queries such as Query 2 from Figure 5.3(b) to be rewritten using the inverse of c_1 for example (the inverse of a correspondence is its equivalent except that the source member becomes the target member and vice-versa). It can deal with *(c:c)* correspondences but cannot combine correspondences in the rewriting process, *i.e.*, if more than one correspondence is needed to rewrite the query, the system can not deal with it. For example, the query: `SELECT ?x WHERE{?x o2:hasDecision ?y. ?y a o2: Acceptance. o2:person1 o2:writes ?x.}` cannot be rewritten with this query rewriting system as it would need to combine two correspondences: $\langle$ $\exists o_2$*:hasDecision.*o_2*:Acceptance* , o_1*:AcceptedPaper* , $\equiv \rangle$ and $\langle$ o_2*:writes* , o_1*:paperWrittenBy*$^-$, $\sqsupseteq \rangle$.

Of the existing rewriting systems dealing with complex correspondences, the one described in [Thiéblin *et al.* 2016] deals with most types of constructions. So far, no rewriting system dealing with *(c:c)* correspondences had been proposed in the literature. The instance-based rewriting system we propose can deal with them, but only if they are not combined together.

5.2.2.2 Instance-based anchoring, query comparison, scoring and aggregation

The rewriting phase outputs all the possible queries regardless of the correspondence relation. A lot of noise can therefore be introduced. Moreover, the same query can be output by both rewriting systems. Consequently, the anchoring step consists in rewriting the source query and selecting that with the best *query Fmeasure* score (Equation 5.9). This selected query is represented by $bestq_T$ in Equation 5.10 ($bestq_T = \underset{q_T \in rewrite(cqa_S, A, SKB)}{\text{argmax}} \; query\ Fmeasure(I^{TKB}_{cqa_T}, I^{TKB}_{q_T})$). This strategy protects the final score from the noise introduced by the query rewriting systems.

The instance-based comparison and scoring are performed as presented in Section 5.2.1 with $I^{TKB}_{bestq_T} = I_{ev}$ and $I^{TKB}_{cqa_T} = I_{ref}$. The scoring function $f \in \{$*classical, recall oriented, precision oriented, overlap, not disjoint, query Fmeasure*$\}$ is chosen and applied to the instance sets of $bestq_T$ and cqa_T: $f(I^{TKB}_{cqa_T}, I^{TKB}_{bestq_T})$.

If a source CQA could not be rewritten by the alignment, its scores are all 0.

An average function is then performed to aggregate the scores per CQA pair (*i.e.*, pair of SPARQL queries representing a CQA) into a final score.

Example Set of CQA pairs:

pair 1 Source CQA: `SELECT ?x WHERE { ?x a o1:AcceptedPaper. }`
Target CQA: `SELECT ?x WHERE { ?x o2:hasDecision ?y. ?y a o2:Acceptance.}`

pair 2 Source CQA: `SELECT ?x ?y WHERE { ?x o1:paperWrittenBy ?y.}`
Target CQA: `SELECT ?x ?y WHERE { ?y o2:writes ?x. ?x a o2:Paper.}`

We consider an alignment A composed of:

c_1 $\langle$ *o1:AcceptedPaper , o2:Paper ,* $\equiv \rangle$

c_2 $\langle$ *o1:AcceptedPaper ,* $\exists$ *o2:hasDecision.o2:Acceptance ,* $\equiv \rangle$

c_3 $\langle$ *o1:AcceptedPaper ,* $\exists$ *o2:hasDecision.*$\{$*o2:decision1*$\}$ *,* $\equiv \rangle$

c_4 $\langle$ *o1:AcceptedPaper ,* $\exists$ *o2:hasDecision.o2:Acceptance* $\sqcap \neg$ $\{$*o2:paper1*$\}$ *,* $\equiv \rangle$

Each correspondence c_i is used to rewrite Query 1, the rewritten query are marked q_i. Let us consider that Query 2 retrieves 100 answers from the target knowledge base, and the q_i retrieve respectively:

q_1 200 instances including the 100 from Query 2

q_2 100 instances, the same as the 100 from Query 2

q_3 1 instance, included in the 100 from Query 2

q_4 99 instances, included in the 100 from Query 2

The query with the best *query Fmeasure* score is q_2. Therefore it is kept as a result of a local aggregation. As $I_{q_2}^{TKB} \equiv I_{cqa_T}^{TKB}$, all the scoring functions output 1.0.

The source CQA from the second CQA pair could not be rewritten with A, therefore all scores are set to 0. The scores for each pair of CQAs are shown in Table 5.1.

Table 5.1: Scores of A for the CQA pairs

	classical	recall-oriented	precision-oriented	overlap	not disjoint	query Fmeasure
CQA pair 1	1	1	1	1	1	1
CQA pair 2	0	0	0	0	0	0
average	0.5	0.5	0.5	0.5	0.5	0.5

Aggregated, the CQA Coverage scores of A are:

- $coverage(A, cqa_{pairs}, SKB, TKB, classical) = 0.5$
- $coverage(A, cqa_{pairs}, SKB, TKB, recall\ oriented) = 0.5$
- $coverage(A, cqa_{pairs}, SKB, TKB, precision\ oriented) = 0.5$
- $coverage(A, cqa_{pairs}, SKB, TKB, overlap) = 0.5$
- $coverage(A, cqa_{pairs}, SKB, TKB, not\ disjoint) = 0.5$
- $coverage(A, cqa_{pairs}, SKB, TKB, query\ Fmeasure) = 0.5$

5.2.3 Instrinsic instance-based Precision

The CQA Coverage evaluation locally aggregates the results over the CQAs and not the rewritten queries because of the noise added by the rewriting systems. In return, an alignment with all the possible correspondences (correct and erroneous) between the source and target ontologies would obtain a good CQA Coverage score. To counterbalance the CQA Coverage score, we propose to measure the **intrinsic instance-based Precision** of an alignment.

In the OAEI 2018 Taxon evaluation [Thiéblin *et al.* 2018a, Algergawy *et al.* 2018], each correspondence of the alignment has been manually evaluated and classified as true positive or false positive. Here, this process is done automatically based on the comparison of instances. This however requires that the source and target ontologies are populated with the same instances. Even if all the classes of the ontologies were not populated, this metric can give pointers to the relevance of the correspondences.

For each correspondence $c_i = \langle\ e_S\ ,\ e_T\ ,\ \equiv\ \rangle$ in the evaluated alignment, the instances $I_{e_T}^{TKB}$ represented by the target member e_T are compared to the instance $I_{e_S}^{SKB}$ represented by the source member e_S.

The set of instances can then be compared and scored as shown in Section 5.2.1. We arbitrarily chose that the reference instance set (I_{ref}) is $I_{e_s}^{SKB}$ and the evaluated one (I_{ev}) is $I_{e_T}^{TKB}$. This decision affects the *recall-oriented* and *precision-oriented* scores which are directional.

The scores of the correspondences are then averaged to give the *Intrinsic Precision* score of the evaluated alignment A. Equation 5.11 shows the calculation of the Intrinsic Precision for an evaluated alignment A. TKB is the target knowledge base, SKB the source knowledge base, f the chosen scoring function (chosen between Equations 5.4 – 5.9).

$$precision(A, SKB, TKB, f) = \underset{\langle e_S, e_T \rangle \in A}{\text{average}} f(I_{e_S}^{SKB}, I_{e_T}^{TKB}) \tag{5.11}$$

The limitations of the intrinsic instance-based Precision are manifold. First, the relation of the correspondence is not taken into account in the comparison. Then, the population of the ontologies clearly impacts the score. For example, if an ontology class o_1*:Document* is only populated with *Paper* instances, and another o_2*:Document* is only populated with *Review* instances, the correspondence $\langle$ o_1*:Document* , o_2*:Document* , $\equiv$ $\rangle$ will have a 0 score for all the metrics we proposed.

On a dataset where two common classes are either populated with the same instances, not populated or share at least a subclass with the same instances, this metric may give a lower and upper bound for the precision of the alignment. The lower bound is given by the *classical* score in which only equivalent members are considered correct. The upper bound is given by the *not disjoint* score in which all correspondences with overlapping or empty members are considered correct.

Example We consider an alignment A composed of (as in the Example of Section 5.2.2):

c_1 $\langle$ o_1*:AcceptedPaper* , o_2*:Paper* , $\equiv$ $\rangle$

c_2 $\langle$ o_1*:AcceptedPaper* , $\exists$ o_2*:hasDecision.*o_2*:Acceptance* , $\equiv$ $\rangle$

c_3 $\langle$ o_1*:AcceptedPaper* , $\exists$ o_2*:hasDecision.*$\{o_2$*:decision1*$\}$, $\equiv$ $\rangle$

c_4 $\langle$ o_1*:AcceptedPaper* , $\exists$ o_2*:hasDecision.*o_2*:Acceptance* $\sqcap \neg \{o_2$*:paper1*$\}$, $\equiv$ $\rangle$

Let us consider that o_1*:AcceptedPaper* retrieves 100 instances in the source knowledge base, the target members of the correspondences above retrieve respectively:

c_1 200 instances including the 100 from o_1*:AcceptedPaper*

c_2 100 instances, the same as the 100 from o_1*:AcceptedPaper*

c_3 1 instance, included in the 100 from o_1*:AcceptedPaper*

c_4 99 instances, included in the 100 from o_1*:AcceptedPaper*

The scoring functions are applied to these correspondences and give the scores in Table 5.2.

Table 5.2: Scores of the correspondences of A

	classical	recall-oriented	precision-oriented	overlap	not disjoint	query Fmeasure
c_1	0	1	0.5	1	1	0.67
c_2	1	1	1	1	1	1
c_3	0	0.5	1	1	1	0.02
c_4	0	0.5	1	1	1	0.99
average	0.25	0.75	0.88	1	1	0.67

Aggregated, the Intrinsic Precision scores of A are:

- $precision(A, SKB, TKB, classical) = 0.25$
- $precision(A, SKB, TKB, recall\ oriented) = 0.75$
- $precision(A, SKB, TKB, precision\ oriented) = 0.88$
- $precision(A, SKB, TKB, overlap) = 1$
- $precision(A, SKB, TKB, not\ disjoint) = 1$
- $precision(A, SKB, TKB, query\ Fmeasure) = 0.67$

5.2.4 Harmonic Mean

To obtain a final score, the CQA Coverage and Intrinsic Precision are combined in an harmonic mean. The *HMean* score balances precision and coverage of an alignment as a F-measure balances precision and recall in information retrieval.

Equation 5.12 shows how the *HMean* scores are computed. A is the evaluated alignment, cqa_{pairs} a set of CQAs as pairs of equivalent SPARQL queries, SKB the source knowledge base, TKB the target knowledge base, and f the chosen scoring function, $f \in \{$*classical, recall-oriented, precision-oriented, overlap, not disjoint, query Fmeasure*$\}$.

$$HMean(A, cqa_{pairs}, SKB, TKB, f) = \\ 2 \times \frac{coverage(A, cqa_{pairs}, SKB, TKB, f) \times precision(A, SKB, TKB, f)}{coverage(A, cqa_{pairs}, SKB, TKB, f) + precision(A, SKB, TKB, f)} \quad (5.12)$$

Table 5.3: CQA coverage, Intrinsic Precision and HMean scores of A

	classical	recall-oriented	precision-oriented	overlap	not disjoint	query Fmeasure
CQA coverage	0.5	0.5	0.5	0.5	0.5	0.5
Intrinsic Precision	0.25	0.75	0.88	1	1	0.67
HMean	0.33	0.6	0.64	0.67	0.67	0.57

Example Based on the examples in Sections 5.2.2 and 5.2.3.

- $HMean(A, cqa_{pairs}, SKB, TKB, classical) = 0.33$
- $HMean(A, cqa_{pairs}, SKB, TKB, recall\ oriented) = 0.6$
- $HMean(A, cqa_{pairs}, SKB, TKB, precision\ oriented) = 0.64$
- $HMean(A, cqa_{pairs}, SKB, TKB, overlap) = 0.67$
- $HMean(A, cqa_{pairs}, SKB, TKB, not\ disjoint) = 0.67$
- $HMean(A, cqa_{pairs}, SKB, TKB, query\ Fmeasure) = 0.57$

5.3 Populated Conference dataset

In order to run the evaluation strategies presented in the previous sections (Sections 5.2.2 and 5.2.3), a dataset with instances and CQAs is required. For all the reasons listed in the previous sections, the instances of the ontologies must be controlled.

In Section 5.3.1, we present the methodology followed to create the evaluation dataset: the populated ontologies and associated CQAs. This population methodology is based on CQAs. They guide and help for an homogeneous interpretation of the ontologies. The methodology was applied to the OAEI Conference dataset described in Section 5.3.2. An artificial population of these ontologies was inspired by an actual conference dataset (Section 5.3.3). In Section 5.3.4, we extracted a set of evaluation CQAs from the CQAs used for the dataset population.

5.3.1 Dataset creation process

Here we propose a process to create a dataset on which the evaluation metrics can be applied. It relies on CQAs to ensure that the ontologies are regularly populated. The user defines the CQAs, creates a **pivot format** which covers them all and translates it into SPARQL queries for each ontology.

The proposed process has the following main steps:

1. Create a set of unary and binary CQAs based on an application scenario. For example, *Which are the accepted papers ?* (unary CQA), *Which paper was submitted to which conference ?* (binary CQA).

2. Create a pivot format which covers all the CQAs from step 1. The pivot format can be in JSON, for instance:

```
{"title": {{conftitle}},
 "papers": [{ "id": {{id}},
    "title": {{paptitle}},
    "authors": [{{auth1}}, ...],
    "type": {{paptype}},
    "decision":"accept" or "reject" },
    ... ],
  ...
}
```

3. For each ontology of the dataset, create SPARQL INSERT queries corresponding to the pivot format. Note that an ontology may not cover the totality of the pivot format, for instance:

```
INSERT DATA {
  {{pap}} a conference:Camera_ready_contribution.
  {{pap}} rdfs:label {{paptitle}}.
  ...  }
```

4. Instantiate the pivot format with an actual dataset or a synthetic dataset, for instance:

```
{"title": "ESWC",
 "papers": [{ "id": "10",
    "title": "User-Centric Ontology Population",
    "authors": ["K. Clarkson", ...],
    "type": "Research track",
    "decision":"accept" },
    ... ],
  ...
 }
```

5. Populate the ontologies with the instantiated pivot format using the SPARQL INSERT queries.

6. Run a reasoner to verify the consistency of the populated ontologies. If an exception occurs, try to change the interpretation of the ontology and iterate over steps 3 to 5.

7. Based on SPARQL INSERT queries, translate the CQAs covered by two or more ontologies as SPARQL SELECT queries.

In this process, the interpretation of the ontologies is the same for ontology population and the evaluation CQAs.

The creation of CQAs can be done by interviewing users and domain experts, as recommended in the NeOn methodology [Suárez-Figueroa *et al.* 2012] for competency question authoring. The CQAs can also derive from the competency questions which were used to design the ontologies of the dataset. In this implementation, however, only one expert created the CQAs.

We applied this process on the domain of conference organisation, to the Conference dataset.

5.3.2 Conference dataset

The dataset used here is the Conference dataset[2] proposed in [Šváb Zamazal *et al.* 2005]. It has been widely used [Šváb Zamazal & Svátek 2017], especially in the OAEI campaigns where it is a reference evaluation track. It is composed of 16 ontologies on the conference organisation domain and simple reference alignments between 7 of these ontologies. These ontologies were developed individually. The motivation for the extension of this dataset is that the ontologies were created from existing conference organisation tools or website, they are expressive and largely used for evaluation in the field. This dataset has been extended as in [Cheatham & Hitzler 2014]. The query-oriented evaluation benchmark OA4QA was also based on this dataset [Solimando *et al.* 2014b]. Furthermore, reference complex alignments for query rewriting and ontology merging tasks have been proposed over five ontologies of this dataset [Thiéblin *et al.* 2018b].

In the first OAEI complex track, an evaluation was proposed over a consensual complex alignment between three ontologies (*cmt*, *conference*, *ekaw*) [Thiéblin *et al.* 2018a]. Here, the five ontologies covered by [Thiéblin *et al.* 2018b] have been populated: *cmt*, *conference* (Sofsem), *confOf* (confTool), *edas* and *ekaw* (*cf.* Table 5.4).

Table 5.4: Number of entities by type of each ontology

	cmt	conference	confOf	edas	ekaw
Classes	30	60	39	104	74
Obj. prop.	49	46	13	30	33
Data prop.	10	18	23	20	0

[2] `http://oaei.ontologymatching.org/2018/conference/index.html`
`http://owl.vse.cz:8080/ontofarm/`

5.3.3 Populating the Conference ontologies

In order to create the CQAs and re-interpret the Conference ontologies, a conference organisation scenario has been considered. The list of CQA has first been established by examining a use case: the Extended Semantic Web Conference 2018 edition. The list of CQAs created from this use case has then been extended by exploring the conference ontologies scope.

The Extended Semantic Web Conference[3] (ESWC) is open review and its website provided a good base to analyse which information is needed for conference organisation. In order to create the artificial instances of the pivot format, the ESWC 2018 use case as well as data from Scholarly Data [Nuzzolese *et al.* 2016] were considered.

5.3.3.1 Re-interpreting the ontologies with a conference organisation scenario

As mentioned before, the first step of the population process was to create a list of CQAs and re-interpret the ontologies under the perspective of a conference organisation scenario. By analysing the ESWC 2018 website, a first list of CQAs was created. The methodology was followed based on this first list of CQAs. The pivot format was instantiated with the website data.

While running the Hermit [Shearer *et al.* 2008] reasoner in step 6 of the population process, several exceptions were encountered. For most of them, the problem was with the human interpretation of the ontology. For example, in the *cmt* ontology, *cmt:hasAuthor* is functional. Unlike primarily interpreted, this means that *cmt:hasAuthor* represents a "has first author" relationship between a *cmt:Paper* and a *cmt:Author*. Then, the SPARQL INSERT queries have been modified in order to fit the new interpretation of the ontology.

Two exceptions have been detected, which could not be resolved by a change of interpretation. In that case, the original ontologies have been slightly modified and the modifications acknowledged by the administrators of the original dataset:

- *cmt*: the relation *cmt:acceptPaper* between an *Administrator* and a *Paper* was defined as functional and inverse functional. This leads to an inconsistency when a conference administrator accepts more than one paper. *cmt:acceptPaper* has been changed to be only inverse functional. The Conference dataset administrator was already aware of this issue.

- *conference*: *conference:Contribution_1st_author* was disjoint with *conference:Contribution_co-author*, which lead to an inconsistency when a person was at the same time the first author of a paper and the co-author of another paper. The disjunction axiom from the ontology has then been removed. This issue was new to the Conference dataset administrator.

[3] `https://2018.eswc-conferences.org/`

If a CQA was not exactly covered by an ontology, the ontology would not be populated with its associated instances. This results in an uneven population of equivalent concepts in the ontologies. For example, considering the *ekaw* and *cmt* ontologies, which both contain a *Document* class. *"What are the documents?"* was not a CQA whereas *paper, review, web site* and *proceedings* were the focus of CQAs. While *ekaw:Document* class has for subclasses *ekaw:Paper*, *ekaw:Review*, *ekaw:Web_Site* and *ekaw:Conference_Proceedings*, *cmt:Document* has only two subclasses *cmt:Paper* and *cmt:Review*. *ekaw:Document* will, by consequence of its subclasses, be populated with paper, review, website and proceedings instances whereas *cmt:Document* will be populated with paper and review instances only.

We could have considered each class with exactly the same instances; for example, we could have populated *cmt:Document* with all the *Paper*, *Review*, *Web site* and *Conference proceedings* instances. Therefore, *cmt:Document* and *ekaw:Document* would share exactly the same instances. However, we chose to remain the closest to the original ontologies as possible. We opine that if a class was not explicitly present in an ontology, it was because their creators did not consider it in their requirements. This way, the instances also represent the conceptual mismatch between the ontologies.

5.3.3.2 Analysis of data on conference organisation

In order to create a population for the conference ontologies and make it close to actual scenarii, some figures from past conferences have been analysed. The information from the International Semantic Web Conference (ISWC) 2018 and the Extended Semantic Web Conference (ESWC) 2017 from Scholarly Data[4] complemented the ESWC 2018 website data for this analysis. Indeed, some information such as which program committee member reviewed which paper does not appear in Scholarly Data and the ESWC 2018 website did not show which person is affiliated to which organisation. Some points could be observed:

- percentage of accepted papers having at least a program committee member as author: 44% for ESWC 2017 and 59% for ISWC 2018
- distribution of the number of authors per submitted papers (ESWC 2018): 1 (6%), 2 (17%), 3 (29%), 4 (26%), 5 (9%), 6 (8%) ou 7-10 (2%)
- distribution of the number of collaborating institutions per accepted papers over scholarly data (global represents the statistics over all data from the scholarly data endpoint)

[4] `http://www.scholarlydata.org/`

nb inst.	global	ESWC 2017	ISWC 2018
1	56%	40%	40%
2	18%	16 %	30%
3	10 %	10 %	17%
4	6%	7 %	7%
5	5%	6%	5%
6+	between 0 and 2 %		

- distribution of the number of authors per accepted papers over scholarly data

nb auth.	global	ESWC 2017	ISWC 2018
1	12%	7%	13 %
2	21%	11%	14%
3	27%	28%	24%
4	19%	25%	23%
5	17%	17%	14%
6	5%	5%	6%
7+	between 0 and 4 %		

5.3.3.3 Artificially populated Conference ontologies

The first population of the ontologies with the ESWC 2018 data left some important knowledge un-represented. For example, the concepts of external reviewer, presenter of a paper, person affiliation which appeared important for a conference organisation were not available on the website. Always in the perspective of conference organisation, the conference ontologies were browsed to complete the list of CQAs with useful concepts. The pivot format and associated SPARQL INSERT queries were also extended to cover the new list of CQAs.

The next step was to artificially generate the pivot format instantiation. A size score between 1 and 10 is given to each conference. This score pseudo-randomly determines the number of submitted papers, program committee members, *etc.* as shown in Table 5.5.

The statistics from the ESWC 2018, ISWC 2018, ESWC 2017 datasets were globally reproduced: 50% of papers have at least one program committee member as author, the number of authors per paper is 1 (6%), 2 (17%), 3 (29%), 4 (26%), 5 (9%), 6 (8%) ou 7-10 (2%), the number of collaborating institutions is around 1 (40%), 2(30%), 3 (17%), 4 (7%), 5 (5%), 6(2%). These statistics are pointers, as the generation process is pseudo-random, these figures may vary in practice. Some proportions were arbitrarily chosen: 20% of the submitted papers are poster papers, and 20% are demo papers, the regular paper acceptance rate is in [0.1 – 0.7] and a poster/demo paper acceptance rate is in [0.4 – 1.0], 20% of the reviews are done by an external reviewer.

In order to evaluate statistics-based matchers on the benchmark, different sets of population were considered for the ontologies. The idea is to provide the same conference ontologies but with more or less common instances. To do so, 6 sets

Table 5.5: Number of submitted papers, pc members, *etc.* for a conference of size 1 and 10 (min – max values).

Number of	Size 1	Size 10
submitted papers	40 – 45	940 – 990
people	300 – 330	1830 – 2130
pc members	50 – 52	500 – 530
oc members	20 – 22	110 – 140
sc members	15 – 17	60 – 90
institutions	30 – 32	210 – 240
tutorials	1 – 2	10 – 11
workshops	1 – 2	19 – 20
tracks	1	6

of instance population with a more or less important common part were created. Each ontology is populated with different conferences[5] (with absolutely no common instance between the conferences (no common person, no common paper, *etc.*)). This ensures that there is a quantifiable common part and that the ontologies are regularly populated. 6 artificial datasets were created with 25 artificial conferences:

- 0 %: 5 different conferences per ontology
- 20 %: 1 common conference for all ontologies and 4 different conferences per ontology
- 40 %: 2 common and 3 different conferences
- 60 %: 3 common and 2 different conferences
- 80 %: 4 common and 1 different conference
- 100 %: 5 common conferences for all ontologies

Note that the percentage given in the name of the datasets is the percentage of common conference instances per ontology. As the size of each conference is different, the percentage of common instances (papers, authors, *etc.*) will not be same. In Table 5.6, the minimum and maximum percentage of the common paper instances is given for each dataset.

All the ontology concepts were not covered by the pivot CQAs. Table 5.7 shows the number of entities (by type of entity) of the ontologies which are populated in the datasets.

We verified that two equivalent classes in the populated ontologies would not obtain a disjoint ($\perp$) relation on this dataset. To do so, we used the reference alignment *ra1* from the original Conference dataset. We modified this alignment to take into account our interpretation of the ontologies. Two original correspondences were

[5] A *conference* here refers to the data related to a conference event.

Table 5.6: Percentage (min, max) of common submitted papers in the different datasets. The second line reads *"In the 20% dataset, the proportion of common paper instances is between 7 and 11 %"*. Which means that for one of the ontologies, the common part of paper instances represents 7% of all its paper instances. For another ontology, the common part of paper instances represents 11% of all its paper instances.

Dataset	Min	Max
0%	0%	0%
20 %	7%	11 %
40 %	29%	51%
60 %	40 %	57%
80 %	57%	84 %
100 %	100 %	100 %

Table 5.7: Number of populated entities by ontology. Number of populated entities / number of entities in the original ontology.

	cmt	conference	confOf	edas	ekaw
Classes	26 / 30	51 / 60	29 / 39	42 / 104	57 / 74
Obj. prop.	43 / 49	37 / 46	10 / 13	17 / 30	26 / 33
Data prop.	7 / 10	13 / 18	10 / 23	11 / 20	0 / 0

removed: ⟨ *conference:Poster* , *ekaw:Poster_Paper* , ≡ ⟩ and ⟨ *conference:Poster* , *confOf:Poster* , ≡ ⟩ because we considered the poster object in *conference* and the paper associated with a poster in *ekaw* and *confOf*. Then, the instances of the source and target member of each correspondence of the modified *ra1* were compared. None were disjoint.

5.3.4 Selection of CQAs and translation into SPARQL SELECT queries for evaluation

In step 7 of the population process, the CQAs as SPARQL INSERT queries are selected and transformed into SPARQL SELECT queries. Indeed, the CQA Coverage (Section 5.2.2) is about CQAs which can actually be covered by two or more ontologies. The list of CQAs used in the ontology population was trimmed as follows:

- the CQAs which were only covered by one ontology
- some CQAs which were not considered relevant such as "*What is the name of a reception?*", the answer being an *rdfs:label* "Reception" for all reception instances.

The remaining CQAs were then written as SPARQL SELECT queries by adapting the SPARQL INSERT queries. Table 5.8 shows the number of CQAs which were covered by the pivot format by each ontology (*i.e.*, in the SPARQL INSERT queries

of the ontology population phase) and which were transformed into SPARQL SELECT queries for the evaluation dataset. 278 SPARQL SELECT queries result from this process which correspond to 100 CQAs. Table 5.9 shows the number of SPARQL SELECT queries which represent a simple or a complex expression for each ontology. Out of the 278 SPARQL SELECT CQAs, 133 represent simple expressions and 145 complex expressions.

Table 5.8: Number of initial (pivot) CQAs and number of evaluation (eval) CQAs created and selected and covered by each ontology. The global column shows the total number of CQAs.

	cmt	conference	confOf	edas	ekaw	global
pivot CQAs	46	90	67	60	84	152
eval unary CQAs	20	36	21	19	36	45
eval binary CQAs	14	37	33	33	29	55
eval CQAs	34	73	54	52	65	100

Table 5.9: Number of SPARQL CQAs which represent a simple or a complex expression.

	cmt	conference	confOf	edas	ekaw
simple expression SPARQL CQAs	21	28	24	29	31
complex expression SPARQL CQAs	13	45	30	23	34
SPARQL CQAs	34	73	54	52	65

We counted the number of pairs of equivalent SPARQL CQAs and classified them as *(s:s)* if the two represent simple expressions, *(s:c)* if one of the queries represents a simple expression and the other a complex expression and *(c:c)* if both queries represent complex expressions. There are 111 *(s:s)* pairs, 86 *(s:c)* pairs and 98 *(c:c)* pairs in the dataset for a total of 295 equivalent query pairs (or 590 oriented pairs).

5.4 Positioning and conclusion

In this section we have proposed a complex alignment evaluation benchmark which includes i) an evaluation system implementing instance-based comparison, ii) a dataset with controlled instances. A CQA-based methodology was proposed and applied to five ontologies of the well-known Conference dataset [Šváb Zamazal & Svátek 2017] to create the dataset. Two complementary evaluation approaches (CQA Coverage and Intrinsic Precision) have been proposed.

From the analysis of the existing work on complex ontology alignment evaluation, there is no automatic system to evaluate complex alignment. In the evaluation workflow, the comparison and anchoring steps are the hardest to automate. Instance-based comparison (of correspondences/queries, *etc.*) is, so far, the easiest

comparison method to automate. However, this comparison must be done over controlled instances, and a complex alignment dataset fulfilling such requirements does not exist.

Table 5.10 shows an analysis of the existing complex ontology alignment evaluation benchmarks made on the workflow steps (Section 5.1). The proposition of this paper is the **Populated conference** benchmark, presented in the last row of Table 5.10.

Even if the benchmark we propose bridges a gap in complex alignment evaluation, it has limitations. The query rewriting systems are not perfect, *(c:c)* correspondences can only be treated in the specific case where the source member of correspondence describes the same instances as the query. The relation and confidence value of the correspondences are not taken into account in the evaluation process.

Table 5.10: Comparison of alignment evaluation benchmarks over the workflow steps. The benchmarks presented here deal with complex alignment or with queries as reference. The *Corr.* column represents the form of the most complex correspondences dealt with by the benchmarks (*(c:c)* is more complex than *(s:c)*, which is more complex than *(s:s)*). *Comp. val.* stands for comparison value for the scoring function.

Benchmark	Type of evaluation	Reference	Anchoring	Comparison	Scoring	Corr.
OAEI simple tracks	Automatic	Alignment	URI	Syntactic	Classical	*(s:s)*
OA4QA [Solimando *et al.* 2014b]	Automatic	Alignment	Source Query	Instance-based	Comp. val.	*(s:s)*
[Hollink *et al.* 2008]	Manual	Query	Source Query	Instance-based	Classical/ Comp. val.	*(s:s)*
[Walshe *et al.* 2016]	Manual	Alignment	Source URI	Syntactic	Classical	*(s:c)*
[Parundekar *et al.* 2012]	Manual	Alignment	Source URI	Syntactic	Classical	*(s:c)*
[Thiéblin *et al.* 2018b]	Manual	Alignment	Source URI	Semantic	Classical	*(s:c)*
Complex conf. [Thiéblin *et al.* 2018a]	Manual	Alignment	Source URI	Semantic	Classical	*(s:c)*
GeoLink [Thiéblin *et al.* 2018a]	Automatic	Alignment	Source URI	Syntactic	Classical	*(c:c)*
Hydrography [Thiéblin *et al.* 2018a]	Automatic	Alignment	Source URI	Syntactic	Classical	*(c:c)*
Taxon [Thiéblin *et al.* 2018a]	Manual	Query	Source Query	Semantic	Classical	*(c:c)*
Populated conf.	Automatic	Query	Source Query	Instance-based	Many	*(c:c)*

The proposed benchmark is used for the evaluation of CANARD in the following Chapter.

Chapter 6

Experimentation

Content

Evaluating a matcher allows for its characterisation and for comparing it with existing ones. An automatic evaluation was performed on the Populated Conference evaluation benchmark of Chapter 5. This evaluation measures the impact of various parameters on the approach.

The approach was also evaluated on LOD repositories about plant taxonomy. This scenario shows how the approach performs when faced with LOD challenges such as large ontologies and millions of triples. Some of the knowledge bases chosen for this experiment are irregularly populated. This means that the same piece of knowledge can be represented in various ways in the same ontology and that all instances are not described identically. As discussed in Section 5.1.3, this can entail errors in the instance-based comparison. All the different ways a piece of knowledge can be represented in DBpedia are hard to number. Moreover, the ontologies are not populated with exactly the same instances. For these reasons, the evaluation could not be automated as on the Populated Conference benchmark and was manually performed.

After detailing the implementation parameters of the matching approach in Section 6.1 and the evaluation settings in Section 6.2, we give the results of the

experiments over the Populated Conference dataset in Section 6.3. We then present the Taxon dataset and its associated experiments in Section 6.4. Finally, we discuss the overall results in Section 6.5.

6.1 Matching approach set-up

The matching approach presented in Chapter 4 has been implemented. We describe here the choices that were made in terms of similarity metrics (label and structural), threshold values, *etc.* The implementation is available at `https://framagit.org/IRIT_UT2J/ComplexAlignmentGenerator`.

Label similarity In step (7) of the approach, a label similarity metric is needed to compare two sets of labels L_s and L_t. In this implementation, a Levenshtein distance-based similarity metric was chosen. The similarity between two sets of labels is the cartesian product of the string similarities between the labels of L_s and L_t (equation 6.1). $strSim$ is the string similarity of two labels l_s and l_t (equation 6.2).

$$sim(L_s, L_t) = \sum_{l_s \in L_s} \sum_{l_t \in L_t} strSim(l_s, l_t) \tag{6.1}$$

$$strSim(l_s, l_t) = \begin{cases} \sigma \text{ if } \sigma > \tau, \text{ where } \sigma = 1 - \dfrac{levenshteinDist(l_s, l_t)}{\max(|l_s|, |l_t|)} \\ 0 \text{ otherwise} \end{cases} \tag{6.2}$$

The Levenshtein distance measures the minimum number of single-character edits (insertions, deletions or substitutions) between two strings [Levenshtein 1966]. In the formula, the Levenshtein distance is divided by the length of the longest term. This way, a Levenshtein distance of 3 on a string of 3 characters gives a higher distance (lower similarity) than a Levenshtein distance of 3 on a string of 10 characters. Finally, a filter is applied to the similarity measure obtained: if the similarity between two labels is below a threshold τ, this similarity is considered noisy and is set to zero. As the string similarity in equation 6.1 is a cartesian product, it does not have a maximal limit.

Path length threshold The maximum path length sought is 3. We consider that paths longer than three properties may bring noise in the correspondences. Moreover, the path-finding algorithm searches for all combinations of properties and the longer the path sought is, the longer and harder its search is.

Structural similarity The structural similarity is a constant which is 0 for a triple (in the case of a unary CQA) and 0.5 for a path found between two matched entities (in the case of a binary CQA). As discussed in Section 4.3.7, finding a path

between two instances (the matched answers of a binary CQA) is a hint that this subgraph can be correct. In opposition, the structure subgraphs for unary CQAs are not that informative.

DL formula filtering threshold The DL formulae with a similarity of 0.6 or above are put into a correspondence with their source CQA. If a CQA has no DL formula with a similarity higher than 0.6, the best formulae are put in a correspondence (the formulae having the best similarity, if this similarity is above 0.01). The 0.6 threshold was chosen to be above the structural similarity threshold (0.5) for a path subgraph. Indeed, if two paths are found but only one has a label similarity above 0, then its associated DL formula will be the only one output. These thresholds were empirically chosen.

Approach variants The other parameters such as the number of support answers (*i.e.*, CQA answers with a match in the target knowledge base which are used to find subgraphs), the Levenshtein threshold of the similarity metric, the instance matching strategy, *etc.* are changed to create variants of the approach. The parameter values of the variants are details in Table 6.1 in Section 6.3. The values of the baseline approach were empirically chosen: a Levenshtein distance threshold of 0.4, 10 support answers and no similarity value reassessment based on counter-examples.

6.2 Evaluation settings

The evaluation system presented in Chapter 5 has been implemented. The implementation is detailed in this section. The environment on which the matcher and evaluation system were run are also described.

Evaluation system The workflow presented in Chapter 5 has been implemented to compute the **CQA Coverage** (Section 5.2.2). The scoring metrics used in the scoring step of the workflow are classical, recall-oriented, precision-oriented, query f-measure, overlap. A best-match (query f-measure) aggregation over the reference CQAs is performed. An average of the best-match scores gives the **CQA Coverage**.

The **Precision** (intrinsic precision, see Section 5.2.3) is calculated by comparing the source and target members of the correspondences. The scoring metrics used for the Precision are those used in the CQA Coverage.

The Precision and CQA Coverage with the same scoring metric are finally aggregated in a **Harmonic Mean** (see Section 5.2.4).

The implementation in Java of the evaluation system as well as the Populated Conference dataset is available at `https://framagit.org/IRIT_UT2J/conference-dataset-population`.

Environment The approach and evaluation system have been executed on a Ubuntu 16.04 machine configured with 16GB of RAM running under an i7-4790K CPU 4.00GHz × 8 processors. The runtimes are given for a single run. The local SPARQL endpoints were run on the same machine with Fuseki 2[1].

The approach has been automatically evaluated on the populated conference dataset (Section 5.3). The results are described in Section 6.3.

The approach has been manually evaluated on the Taxon dataset as discussed before. The results are described in Section 6.4.

6.3 Evaluation results on populated Conference

Dataset The approach has been run and evaluated on the populated conference 100% dataset presented in Chapter 5. The populated ontologies of the 100% dataset were run on a local Fuseki 2 server. The approach would work on the 20%, 40%, 60% and 80 % dataset described in Section 5.3 because it only needs one common instance per CQA. However, we chose the 100% dataset because the search for a common instance is faster when the proportion of common instances in the source answers is higher.

Table 6.1: Parameters of the evaluated variants of the approach: number of support answers (Nb. ans.), Levenshtein threshold in the similarity metric (Lev. thr.), type of instance matching strategy (Inst. match), computation of counter-examples (Co.-ex.), CQAs input

Evaluated variant	Nb ans.	Lev. thr.	Inst. match	Co.-ex.	CQAs
baseline	10	0.4	links		√
Levenshtein (§6.3.1)	10	0.0–1.0	links		√
Support answers (§6.3.2)	1-100	0.4	links		√
exact label match (§6.3.3)	10	0.4	labels		√
query (§6.3.4)	10	0.4	links		
query+reassess (§6.3.5)	10	0.4	links	√	
cqa+reassess (§6.3.5)	10	0.4	links	√	√

Evaluation strategy On this dataset, we compare the variants of the approach to its baseline. Each variant is described in Table 6.1. The parameters which are not described in this table such as path length threshold (3), DL formula filtering threshold (0.6) and structural similarity constants (0.5 for a path, 0 for a class expression) are set-up as presented in Section 6.1 for every variant. This evaluation strategy allows to isolate the parameters and measure their impact. In Section 6.3.1, the Levenshtein threshold varies. In Section 6.3.2, the number of support answers changes. In Section 6.3.3, the baseline approach which relies on *owl:sameAs* links between instances is compared to a variant where the instance links are found with

[1] `https://jena.apache.org/documentation/fuseki2/`

an exact label match. In Section 6.3.4, a variant of the approach which generates queries instead of using CQAs is compared to the baseline. In Section 6.3.5, the impact of the similarity reassessment based on counter-examples is studied. We give a qualitative evaluation of the alignments in Section 6.3.6 and compare the results with existing complex alignment generation approaches in Section 6.3.7.

6.3.1 Impact of the threshold in the string similarity metric

In order to evaluate the impact of the string similarity metric when comparing the labels of the entities in the approach, the threshold in the Levenshtein distance-based metric is changed. An evaluation was performed for each version of the baseline approach with a threshold set between 0.0 and 1.0. Figure 6.1 shows the number of correspondence per type (*i.e.*, *(s:s)*,*(s:c)*,*(c:s)* and *(c:c)*) which were found by the variants of the approach. The evaluation results are shown in Figure 6.2.

The number of correspondences decreases when the Levenshtein threshold increases. In the evaluation results, we see that numerous correspondences obtained with a low Levenshtein threshold cover a lot of CQAs (high CQA Coverage) but contain a lot of errors (low Precision). The lower the threshold, the best the CQA Coverage and the lowest the Precision. The Harmonic Mean is the highest for a threshold of 0.4 in the similarity metric. The Classical Harmonic Mean scores are rather low overall but the query f-measure Harmonic Mean scores show that even though the correspondences output are not exactly equivalences, they are quite relevant overall. The baseline approach Levenshtein threshold (0.4) was chosen based on this experiment.

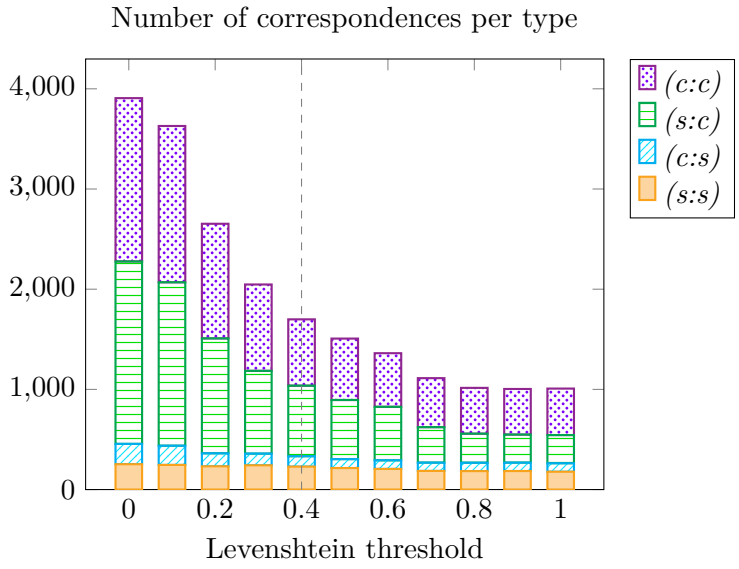


Figure 6.1: Number of correspondences per type for each variant with a different Levenshtein threshold

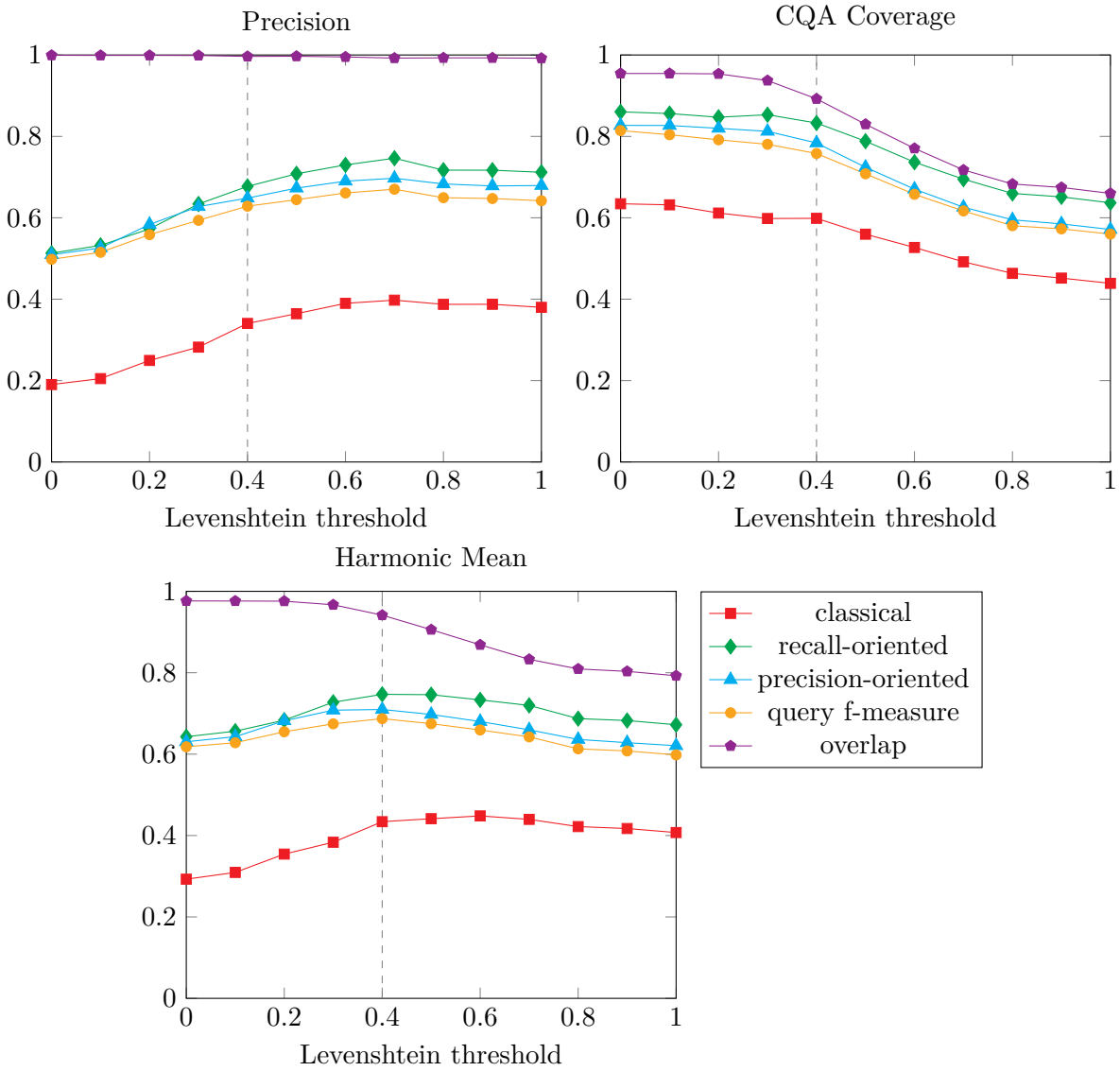


Figure 6.2: Results of the evaluation with 10 support answers and variable levenshtein threshold in the string similarity measure. The baseline results are highlighted by a vertical dashed line.

6.3.2 Impact of the number of support answers

In order to evaluate the impact of the number of support answers, different versions of the baseline approach were evaluated with a number of support answers between 1 and 100. The runtime of the approach over the 20 oriented pairs of ontologies is displayed in Figure 6.3 and Figure 6.4 shows the number of correspondences per type output by the variants. The evaluation results are shown in Figure 6.5.

It could be observed that even with 1 answer as support, the CQA Coverage and Precision scores are high, which shows that the approach can make a generalisation from few examples. As expected, the bigger the number of support answers, the longest the process is to run. Some CQAs have only 5 answers (only 5 conference instances in the population of the ontologies), that explains why the time rises linearly between 1 support answer and 5 support answer and has a lower linear coefficient for support instance over 5.

The Precision scores gets lower with more support answers. The main reason is that particular answer cases which are lexically similar to the CQA labels can be discovered when a lot of instances are considered. For example, the correspondence $\langle$ *cmt:Person* , $\exists$ *conference:has_the_last_name.*{*"Benson"*} , $\equiv$ $\rangle$ was discovered by the variant with 100 support answers. Indeed, *"Benson"* is lexically similar to *"Person"*. The increase of the number of correspondences with the number of support answers shows that incorrect correspondences have been introduced.

The same problem occurs in the CQA Coverage: with 100 support answers, special cases having a higher similarity to the CQA than the expected formula can be found. As in the approach, the formulae are filtered, and when the similarity of the best formula is below a threshold (0.6), only the best one is kept. For example, with 10 support answers, the correspondence $\langle$ *conference:Rejected_contribution* , $\exists$ *cmt:hasDecision.cmt:Rejection* , $\equiv$ $\rangle$ was found for the *"rejected paper"* CQA, the similarity of the target DL formula (0.43) was below the threshold (0.6) but it was the best formula so it was kept. For the 100 support answers, the correspondence $\langle$ *conference:Rejected_contribution* , $\exists$ *cmt:hasSubjectArea.*{*"entity consolidation","distribution", "categorization"*} , $\equiv$ $\rangle$ and had a DL formula similarity similarity (0.44) higher than the expected formula, so only this correspondence was output. Therefore the *conference:Rejected_contribution* CQA could not be covered with this alignment.

However, the overlap CQA Coverage gets slightly higher for a high number of support answers because accidental correspondences have been introduced. For example, the correspondence $\langle$ *conference:Topic* , $\exists$ *rdfs:label.{"compliance"}* , $\equiv$ $\rangle$ was found with 100 support answers because "topic" and "compliance" have a 0.4 label similarity score. The *Topic* CQA over the *conference-cmt* pair was not covered by the variants of the approach with less than 100 support answers because no DL formula with a similarity above 0 was found.

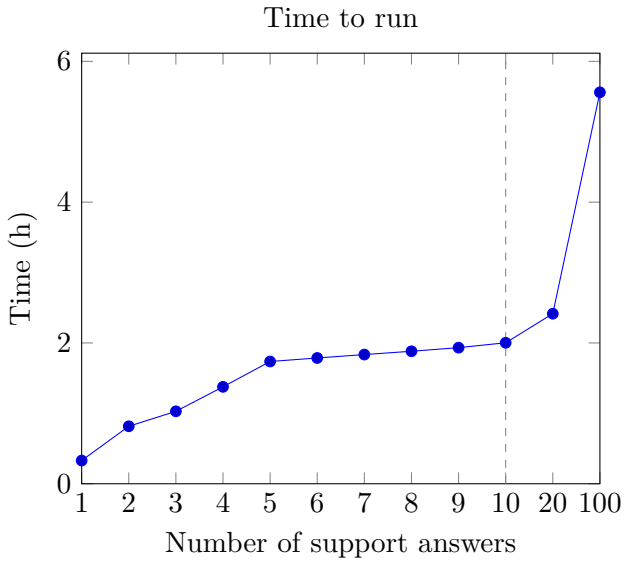


Figure 6.3: Time taken by the approach to run for the 20 oriented pairs of ontologies with a different number of support answers

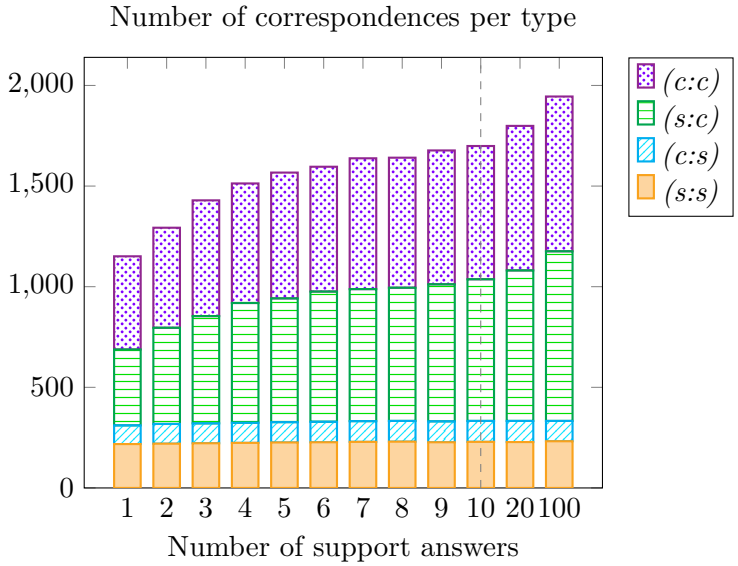


Figure 6.4: Number of correspondence per type for each variant with a different number of support answers

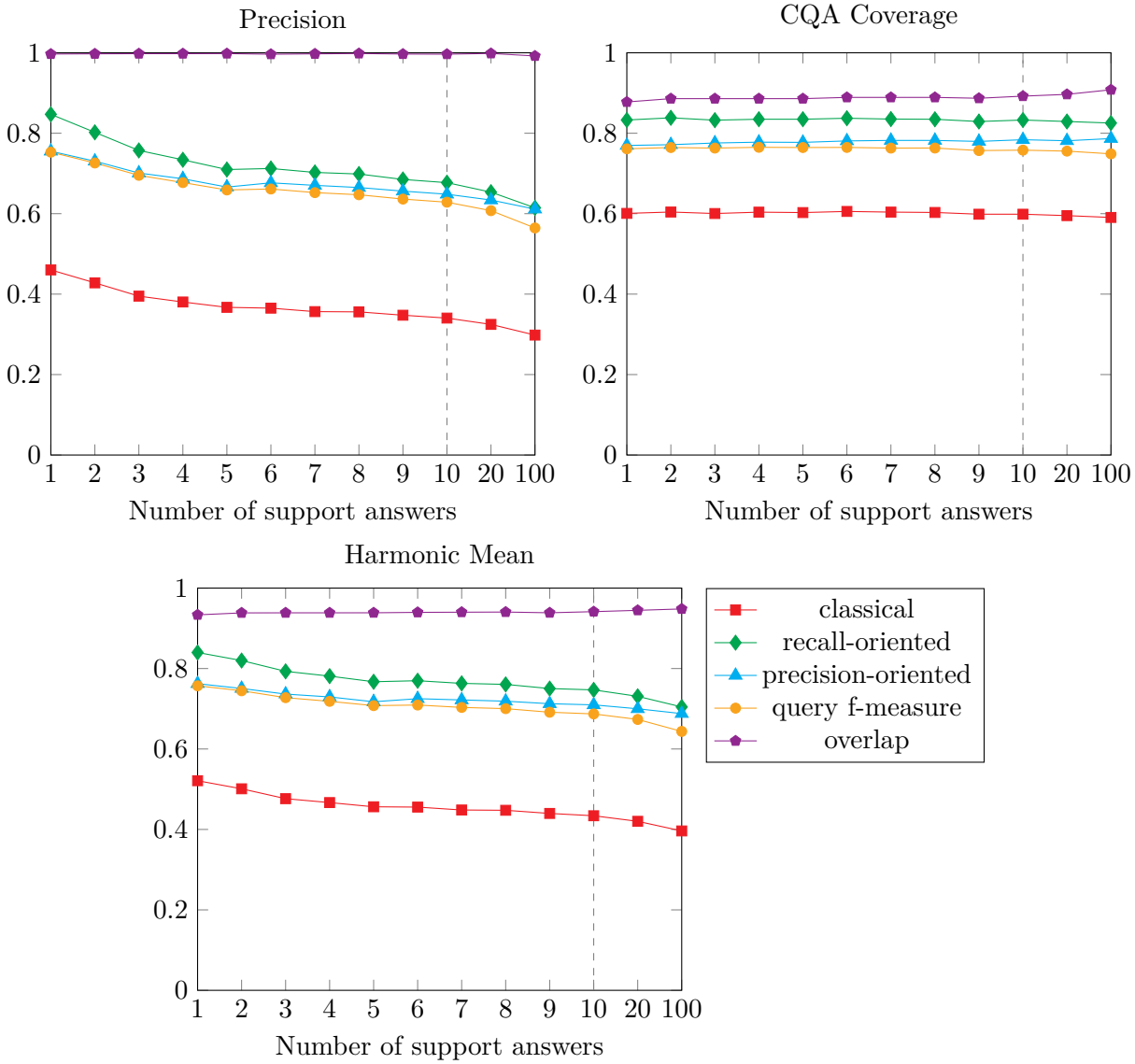


Figure 6.5: Results of the evaluation with a 0.4 Levenshtein similarity threshold and variable number of support answers. The baseline results are highlighted by a vertical dashed line.

6.3.3 Similar instances based on exact label match or existing links

A variant of the approach does not use existing links between instances, instead it performs an exact label match between instances. Figure 6.6 shows the number of correspondences per type of the baseline and its variant. Figure 6.7 shows the results of the baseline and its *exact label match* variant.

The use of exact label match for the instance matching phase brings noise to the correspondences and lowers the Precision. The overlap Precision also decreases because the correspondence are not ensured to share a common instance. In the baseline approach, which uses *owl:sameAs* links, the support answers were by definition common instance and outputting correspondences with no overlap was not possible (except when dealing CQAs with literal values). For example, the paper submissions and their abstract share the same title. Therefore, a rejected paper instance can be matched with its abstract in the target knowledge base. The following correspondence results from this wrong instance match: $\langle$ *ekaw:RejectedPaper* , $\exists$ *conference:is_the_first_part_of.conference:Rejected_contribution* , $\equiv \rangle$. This impacts the number of *(c:c)* correspondences which increases significantly when using the *exact label match*.

Some ontologies use two data properties to link a person to its first and last name. The first and last name are then considered as independent labels of the person instance. This induces a confusion between two people sharing a first or a last name. The following correspondence was obtained by matching a person to another sharing the same first name: $\langle$ *conference:has_the_first_name* , *edas:isReviewedBy*$^{-}\circ$ *edas:isWrittenBy* $\circ$ *edas:hasFirstName* , $\equiv \rangle$

The baseline approach (existing *owl:sameAs* links) takes **2.0 hours** to run over the 20 pairs of ontologies whereas the exact label match approach takes **59.2 hours**. The long runtime for the exact label match approach can be explained by the necessary steps to find the exact label match answers.

First, the labels of each source answer to the CQA must be retrieved. This query takes about 64ms. Then, for each label of the source answer, a match is sought. The runtime of the query to retrieve all instances annotated by a given label is about 2s. The reason is that this query contains a tautology. The choice of this query was made because some ontologies define their own labelling properties instead of using *rdfs:label* or other widely used properties. The following query is used to retrieve the instances having `labelValue` among their associated literals:

```
SELECT DISTINCT ?x WHERE {
   {?x ?z ?label.}
   UNION {?x skos-xl:prefLabel ?z.
      ?z skos-xl:literalForm ?label.}
   filter (regex(?label, "^labelValue$","i")).
}
```

When using direct links, these steps are replaced by directly retrieving *owl:sameAs* links, which takes about 20ms per source instance.

If the number of common support answers between the source and target ontology is reached (in the baseline, when 10 support answers are found), the approach stops looking for new matches. However, when no common instance can be found, the approach looks for a match for every answer of the CQA. This fact coupled with the slow label queries results in such a long time. When common instances exist but do not share the same exact labels, the approach also looks for matches for every source answer, without success. For example, *cmt* represents the full name of a person and *conference* represents its first name and its last name in two different labels. For the CQA retrieving all the Person instances, the approach goes through the 4351 instances without finding any match.

This test shows the impact of the quality of instance matching links on the approach. The best-case scenario is of course when relevant *owl:sameAs* links are available.

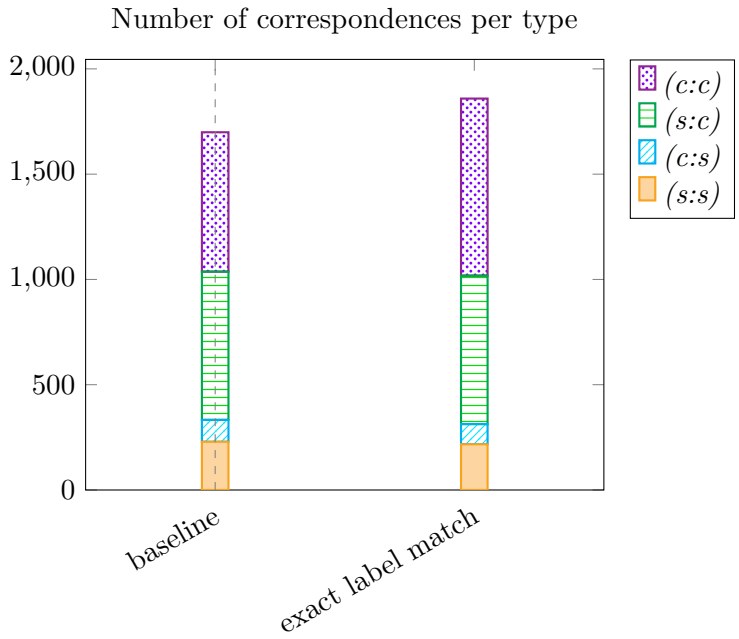


Figure 6.6: Number of correspondence per type for the baseline and the variant based on exact label match

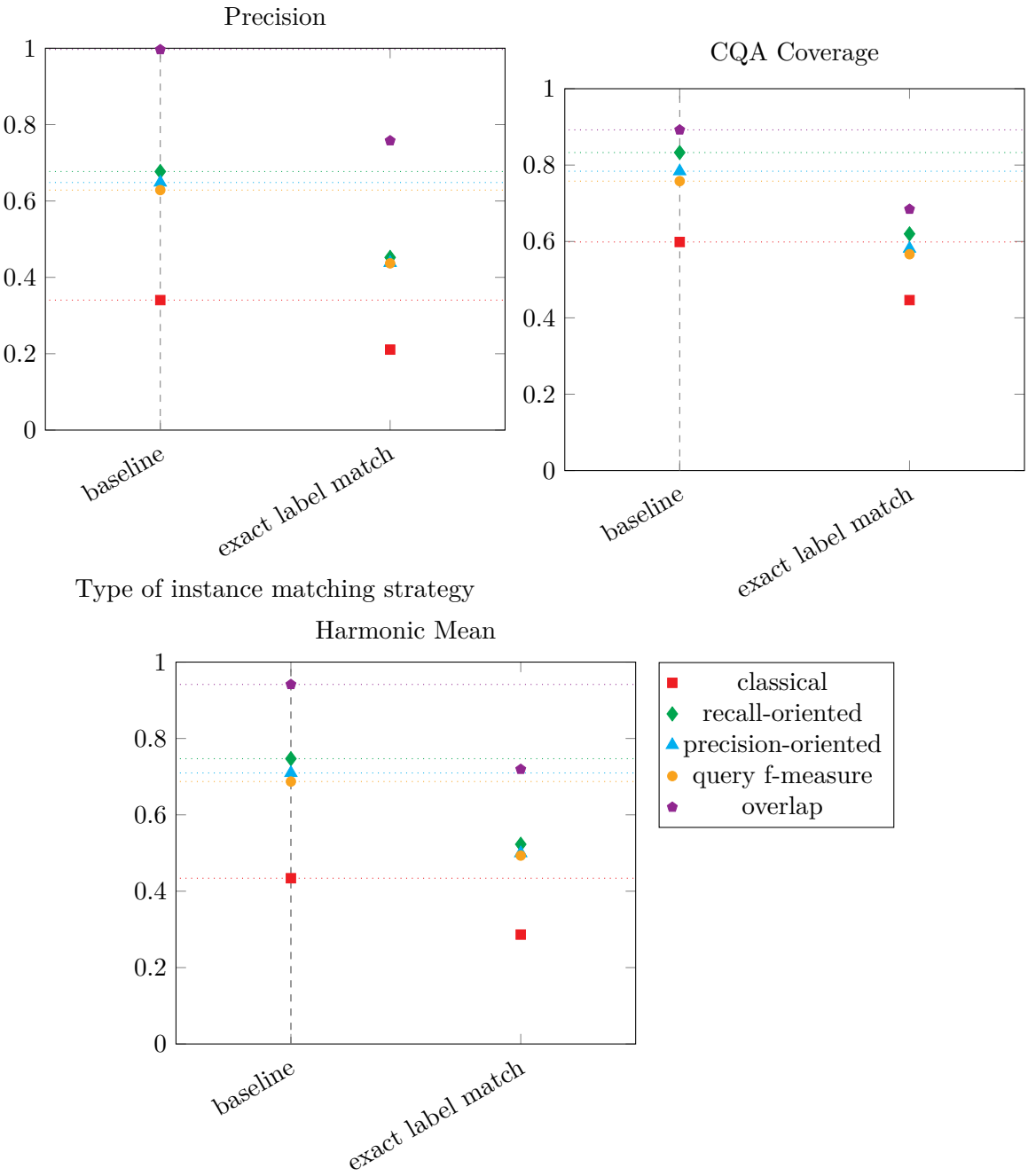


Figure 6.7: Comparison of the approach results when relying on existing *owl:sameAs* links or on an exact label-based instance matching. The baseline results are highlighted by a vertical dashed line.

6.3.4 CQAs or generated queries

In order to measure how the CQAs impact the results of the approach, the baseline approach is compared to a variant which does not rely on input CQAs but automatically generates queries.

Three types of SPARQL queries are generated for a given source ontology: *Classes*, *Properties* and *Property-Value pairs*.

Classes For each *owl:Class* populated with at least one instance, a SPARQL query is created to retrieve all the instances of this class. For instance, if `o1:class1` is a populated class of the source ontology, the following query is created:
`SELECT DISTINCT ?x WHERE {?x a o1:class1.}`

Properties For each *owl:ObjetProperty* or *owl:DatatypeProperty* with at least one instantiation, a SPARQL query is created to retrieve all the pairs of instances of this class. For instance, if `o1:property1` is a populated class of the source ontology, the following query is created:
`SELECT DISTINCT ?x ?y WHERE {?x o1:property1 ?y.}`

Property-Value pairs Inspired by the approaches of [Parundekar *et al.* 2010, Parundekar *et al.* 2012, Walshe *et al.* 2016], SPARQL queries of the following form are created:

- `SELECT DISTINCT ?x WHERE {?x o1:property1 o1:Entity1.}`
- `SELECT DISTINCT ?x WHERE {o1:Entity1 o1:property1 ?x.}`
- `SELECT DISTINCT ?x WHERE {?x o1:property1 "Value".}`

These property-value pairs are computed as follow: for each property (object or data property), the number of distinct object and subject values are retrieved. If the ratio of these two numbers is over a threshold (arbitrarily set to 30) and the smallest number is smaller than a threshold (arbitrarily set to 20), a query is created for each of the less than 20 values. For example, if the property `o1:property1` has 300 different subject values and 3 different object values (`"Value1"`, `"Value2"`, `"Value3"`), the ratio $|subject|/|object| = 300/3 > 30$ and $|object| = 3 < 20$. The 3 following queries are created as CQAs:

- `SELECT DISTINCT ?x WHERE {?x o1:property1 "Value1".}`
- `SELECT DISTINCT ?x WHERE {?x o1:property1 "Value2".}`
- `SELECT DISTINCT ?x WHERE {?x o1:property1 "Value3".}`

The threshold on the smallest number ensures that the property-value pairs represent a category. The threshold on the ratio ensures that properties represent categories and not properties with few instantiations.

Table 6.2: Number of generated queries and CQAs per source ontology

Nb of queries	cmt	conference	confOf	edas	ekaw
classes	26	51	29	43	57
properties	50	50	20	28	26
Attribute-value	30	20	0	5	15
TOTAL	106	121	49	76	98
CQAs	34	73	54	52	65

Table 6.2 shows the number of generated queries per source ontology of the evaluation set.

The approach based on generated queries will not output a correspondence for each CQA in the evaluation. Therefore, the rewriting systems in the evaluation process will bring in noise. The CQA Coverage scores are comparable as only the best result is kept. The Precision of the alignment output by the generated query-based approach is computed as presented in Chapter 5: by comparing the instances of the source and target members in their respective ontologies. These Precision scores give an indicator of the actual precision of these approaches for the problems mentioned in Chapter 5.

The results of the evaluation of the baseline (based on CQAs) and the query variant are presented in Figure 6.8. Figure 6.9 shows the number of correspondence per type.

The CQA Coverage scores when the approach is based on generated queries are between 10% and 20% lower than those obtained with CQAs. Indeed, the *(c:c)* correspondences it retrieves are limited to the Class-by-Attribute-Value pattern on their source member. The Precision scores are not really comparable because the ontologies were populated based on CQAs and not on entities: a *Document* class may be populated with more or less instances given its subclasses. As the approach relies on common instances, the overlap Precision (percentage of correspondences whose member's instances overlap) is around 1.0. The classical Precision (percentage of correspondences whose members are strictly equivalent) is however rather low overall.

The baseline and the *query* variant both take **2.0 hours** to run on the 20 pairs of ontologies. Even if there are more queries to cover than CQAs, the runtime of the *query* variant is compensated by the "difficulty" of the CQAs: some CQAs contain unions or property paths and therefore take more time to be answered by the Fuseki server than the generated queries.

The number of *(s:s)* and *(s:c)* correspondences is much higher for the *query* variant. This approach generates 380 queries which express simple expressions (lines classes and properties of Table 6.2) and therefore, will give *(s:s)* or *(s:c)* correspondences if a match is found. In comparison, the baseline approach relies on 133 SPARQL CQAs which represent a simple expression and 145 which represent a complex expression.

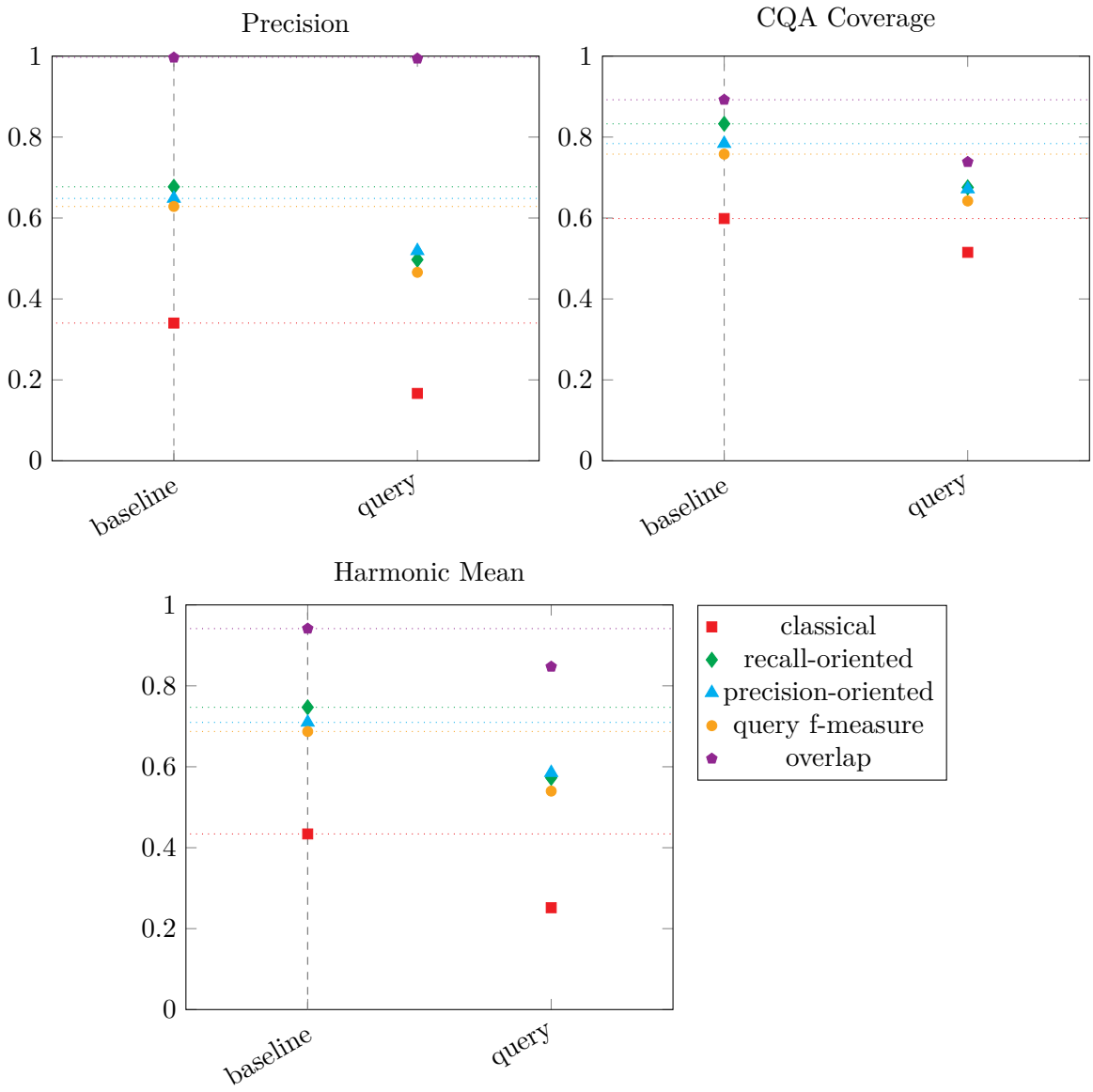


Figure 6.8: Results for the baseline and the variant which generates queries (query)

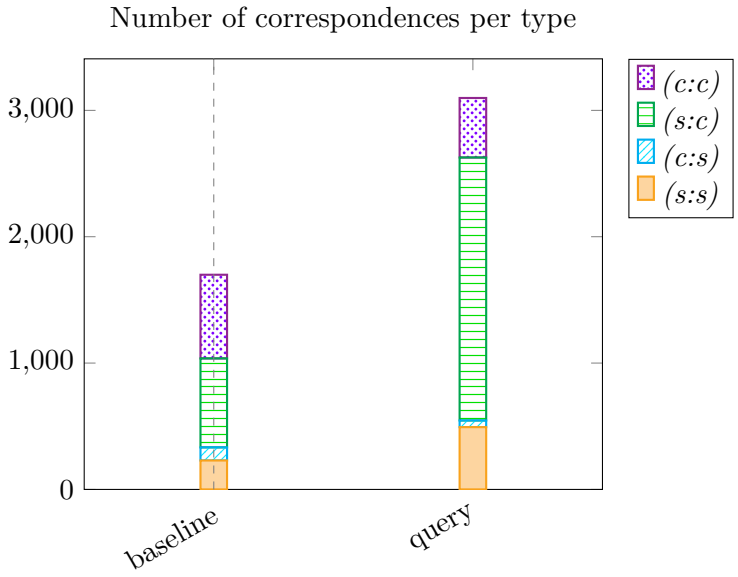


Figure 6.9: Number of correspondence per type for the baseline and the variant which generates queries

6.3.5 Similarity reassessment with counter-examples

In order to evaluate the impact of the similarity reassessment phase based on counter-examples (Section 4.3.6), we run the baseline, the *query* variant and their equivalent with a similarity reassessment phase. The runtime of the variants is presented in Figure 6.10. Figure 6.11 shows the number of correspondences per type output by the baseline and its variants. The results of this evaluation are presented in Figure 6.12.

The reassessment phase (finding counter examples) increases the runtime by far, especially when running queries. It took **46.4 hours** to run the *cqa+reassess* approach and **99.9 hours** to run the *query+reassess* over the 20 pairs of ontologies, when it only took **2.0 hours** for the baseline or *query* versions. The baseline approach and the generated queries variants have approximately the same runtime over the 20 pairs of ontologies. However, for a similar runtime, the results of the approach with the CQAs are better than those with the generated queries.

As expected, the reassessment phase decreases the number of correspondences as they are filtered. It entails an increase of the Precision. The Precision of *cqa+reassess* is between 8% and 15% higher than that of the baseline. The Precision of *query+reassess* is between 6% and 17% higher than that of the *query* variant.

The CQA Coverage remains the same for the baseline and *cqa+reassess*. The CQA Coverage score of *query+reassess* is about 3% lower than that of *query*. As more specific correspondences are preferred over more general ones during the similarity reassessment phase, it leaves less possibilities during the rewriting phase. For example, in the *cmt-conference* pair, the CQA representing accepted papers was:

```
SELECT DISTINCT ?s WHERE{
  {?s cmt:hasDecision ?o. ?o a cmt:Acceptance. }
  UNION  { ?s cmt:acceptedBy ?o. }
}
```

For this example, no correspondence for *cmt:hasDecision* or *cmt:Acceptance* could be found by the approach because the population of *conference* does not contain any *Decision* instance. Therefore, only the correspondences with the *cmt:acceptedBy* object property as source member could be used to rewrite the CQA. To find a correspondence for the generated query about *cmt:acceptedBy*, the approach looked for paths between an administrator and an accepted paper. This resulted in 8 correspondences such as $\langle$ *cmt:acceptedBy , conference:is_submitted_at* $\circ$ *conference:has_contributions* $\circ$ *conference:has_authors* , $\equiv \rangle$ which links every paper of a conference to every paper author of the same conference. Using this correspondence to rewrite the CQA gives (the parts of the query which cannot be rewritten are left identical by the rewriting system):

```
SELECT DISTINCT ?s WHERE{
  {?s cmt:hasDecision ?o. ?o a cmt:Acceptance. }
  UNION { ?s conference:is_submitted_at ?v1.
  ?v1 conference:has_contributions ?v2.
  ?v2 conference:has_authors ?o.  }
}
```

This query returns all the papers submitted to a conference having submissions which themselves have authors, *i.e.*, all the paper instances of the ontology. This gives a query precision of 0.4 and a query recall of 1.0. In the query reassessment phase, the 8 correspondences were filtered because their similarity was lower than 0.01 (there were a lot of counter examples to these correspondences). Therefore, the CQA could not be rewritten with *query+reassess*: the query precision and query recall of are then 0.0.

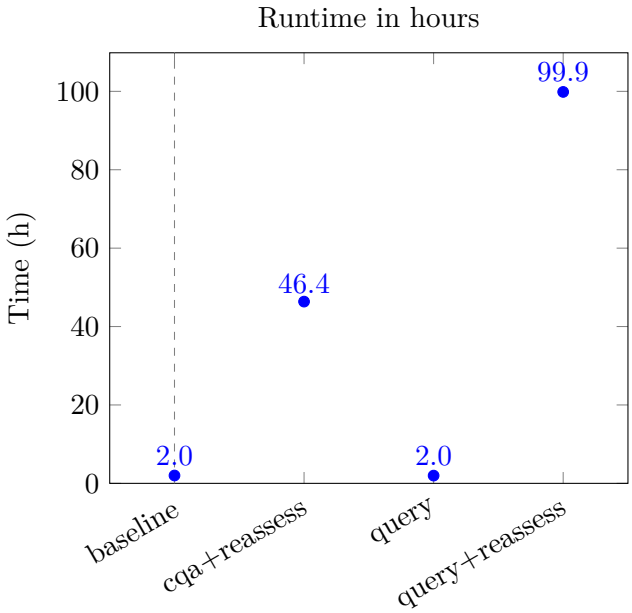


Figure 6.10: Runtime of the baseline and its variants over the 20 oriented pairs of ontologies.

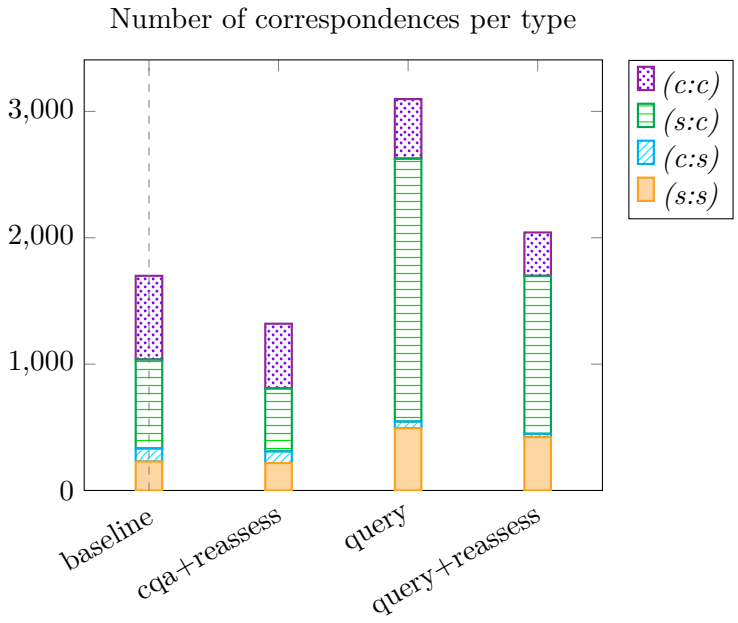


Figure 6.11: Number of correspondence per type for the baseline, the variant which generates queries (*query*) and their equivalent variants with similarity reassessment based on counter-examples

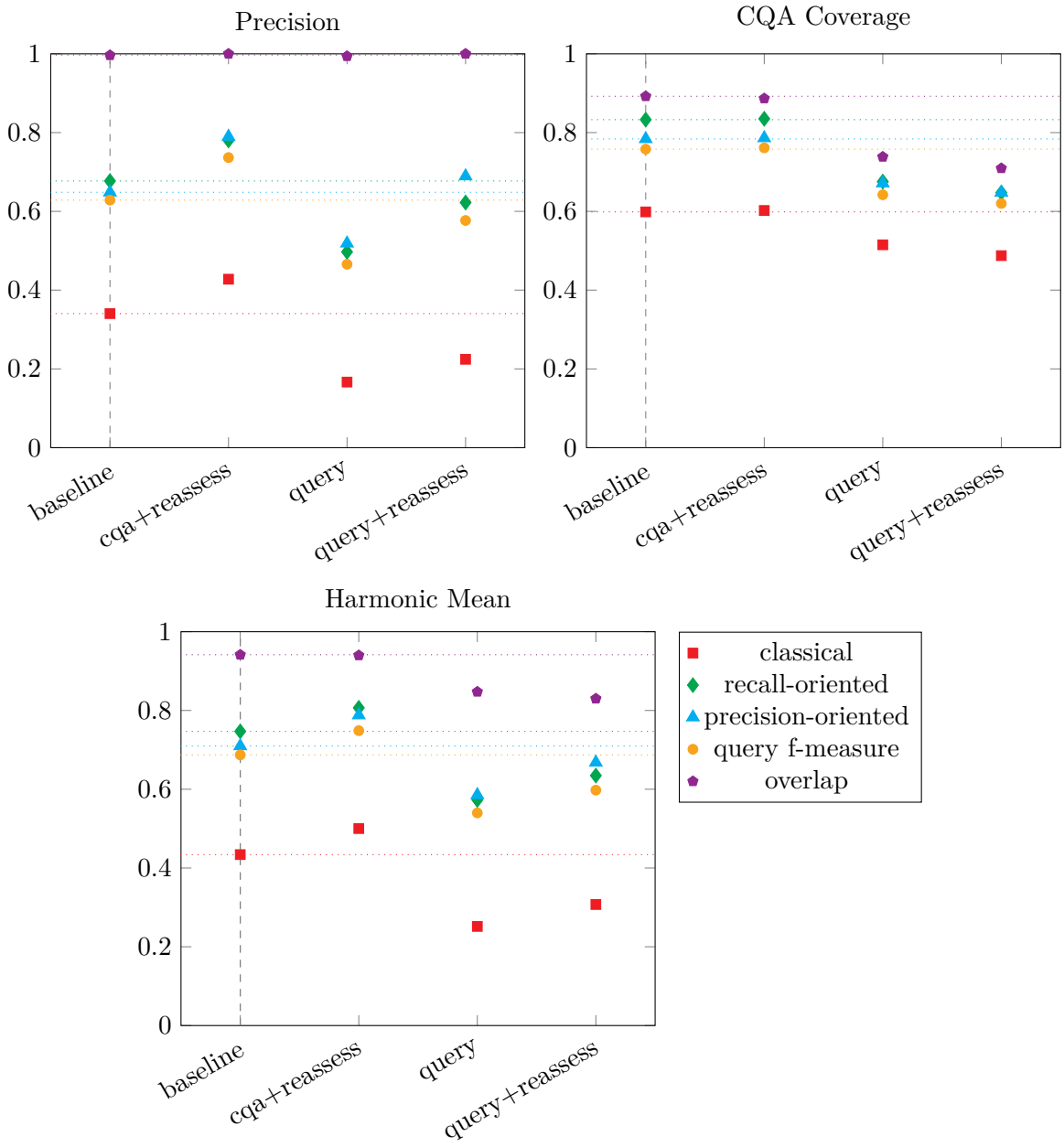


Figure 6.12: Results for the baseline, the variant which generates queries (query) and their equivalent with a counter-example-based similarity reassessment

6.3.6 Qualitative evaluation

In this section, we analyse the cases where the baseline approach did not succeed in finding a correct correspondence.

First, some CQAs were outside the scope of the approach. The CQA *What are the conference websites URL?* expects a set of literal values (not instances). The approach was not set to deal with such CQAs. Some CQAs such as *What is the full name of a person?* needed transformation function (string concatenation) to be matched for some pairs of ontologies. When the transformation was included in the source member (in the CQA) and the target member was a simple data-property URI, the approach could find a correspondence. For example, $\langle$ *concatenation(confOf:hasFirstName,“ ”, confOf:hasSurname) , cmt:name ,* $\equiv \rangle$ was found by the approach. In the other cases, *i.e.*, when a transformation function was needed in the target member, the approach failed to output such a correspondence.

Then, the string similarity metric used in the implementation of the approach induced some errors. Concepts with synonym labels having different strings could not be matched. For example, *Written contribution* could not be matched with *Paper*; *Topic* could not be matched with *Subject Area*, *etc.* Composed nouns and abbreviations were not dealt with successfully for the same reason. For example, *Regular Paper* could not be matched with *Paper* because their Levenshtein distance is 8, which gives a *strSim* score of 0.39 before the filter and 0 after (Equation 6.2). *PC Member* could not be matched with *Program Commitee Member* because their Levenshtein distance was too high.

Finally, the aggregation choice made in the approach (see Section 4.3.5) misses a few correspondences. For instance, the CQA *What are the early-registered participants ?* has a correct formula in *confOf*: $\exists$ *confOf:earlyRegistration.{true}*. However, the predicate (*confOf:earlyRegistration*) has the highest label similarity with the CQA. Therefore, the formula is aggregated into: $\exists$ *confOf:earlyRegistration.*$\top$.

We also analysed how CQAs induce overly complex correspondences in our approach, *i.e.*, complex correspondences which can be decomposed in simple correspondences. For example $\langle$ *dom(confOf:Conference)* $\sqcap$ *confOf:starts_on , dom(edas:Conference)* $\sqcap$ *edas:startDate ,* $\equiv \rangle$ can be decomposed into $\langle$ *confOf:Conference , edas:Conference ,* $\equiv \rangle$ and $\langle$ *confOf:starts_on , edas:startDate ,* $\sqsupseteq \rangle$. To do so, we analysed each *(c:c)* correspondence output by the baseline approach and analysed their content independently from the alignment. We counted that about 14% of *(c:c)* correspondences are overly complex: (92/662). This can be explained by the fact that we directly translate the CQA into a SPARQL query without separating its entities. Moreover, the number of *(c:s)* and *(s:s)* correspondences is lower than that of *(s:c)* and *(c:c)* for every variant of the approach. This shows that the approach has the tendency to generate more complex than simple expressions.

6.3.7 Comparison with existing approaches

We compare the alignment of our approach with three reference alignments and two complex alignment generation approaches:

Query rewriting the query rewriting oriented alignment set[2] from [Thiéblin *et al.* 2018b] - 10 pairs of ontologies

Ontology merging the ontology merging oriented alignment set[2] from [Thiéblin *et al.* 2018b] - 10 pairs of ontologies

ra1 the reference simple alignment[3] from the OAEI conference dataset [Šváb Zamazal & Svátek 2017] - 10 pairs of ontologies

Ritze 2010 the output alignment[4] from [Ritze *et al.* 2010] - complex correspondences found on 4 pairs of ontologies

AMLC the output alignment[5] from [Faria *et al.* 2018] - output alignments between 10 pairs of ontologies

We chose these two matching approaches because their implementations are available online and they output alignments in EDOAL. Ritze 2010 [Ritze *et al.* 2010] and AMLC [Faria *et al.* 2018] both require simple alignments as input. They were run with ra1 as input. ra1 has then been added to Ritze 2010 and AMLC for the CQA Coverage evaluation. The Precision evaluation was made only on their output (ra1 correspondences excluded). Ritze 2010 took **58 minutes** while AMLC took about **3 minutes** to run over the 20 pairs of ontologies. Even though these two approaches are similar, this difference of runtime can be explained by the fact that Ritze 2010 loads the ontologies and parses their labels for each pattern while AMLC only loads the ontologies once. Moreover, Ritze 2010 covers 5 patterns while Faria only covers 2.

Some refactoring was necessary so that the alignments could be automatically processed by the evaluation system. The ra1 dataset had to be transformed into EDOAL, instead of the basic alignment format. The Alignment API could not be used to perform this transformation as the type of entity (class, object property, data property) must be specified in EDOAL. The Ritze 2010 alignments used the wrong EDOAL syntax to describe some constructions (*AttributeTypeRestriction* was used instead of *AttributeDomainRestriction*). The AMLC alignments were not parsable because of RDF/XML syntax errors. The entities in the correspondences were referred to by their URI suffix instead of their full URI (*e.g.*, *Accepted_Paper* instead of *http://ekaw#Accepted_Paper*). Some correspondences were written in the wrong way: the source member was made out of entities from the target ontology and the target member made out of entities from the source ontology. As the

[2] `https://doi.org/10.6084/m9.figshare.4986368.v7`
[3] `http://oaei.ontologymatching.org/2018/conference/`
[4] `https://code.google.com/archive/p/generatingcomplexalignments/downloads/`
[5] `http://oaei.ontologymatching.org/2018/results/complex/conference/`

evaluation of these alignments was manual so far in the OAEI complex track, these errors had not been detected. The alignments' syntax have been manually fixed so that they could be automatically evaluated.

Figure 6.13 shows the number of correspondences per type over the 20 pairs of ontologies. These alignments were not directional so their number of *(s:c)* and *(c:s)* correspondences are identical.

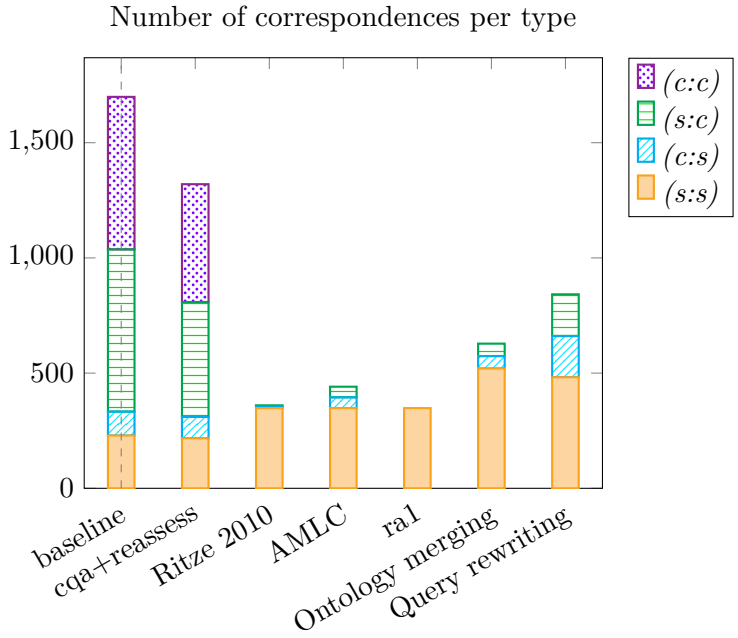


Figure 6.13: Number of correspondence per type for the proposed approach, reference alignments and complex alignment generation approaches. The alignments of Ritze 2010 and AMLC include ra1.

Figure 6.14 shows the results of the baseline approach (*baseline*), the baseline approach with counter-example-based similarity reassessment (*cqa + reassess*) and the compared alignments. The Precision results should be considered carefully. First of all, the relation of the correspondence is not considered in this score: all correspondences are compared as if they were equivalences. The Ontology merging and Query rewriting alignments contain a lot of correspondences with a subsumption relations so their classical Precision score is lower than the percentage of correct correspondences it contains. Second, the precision of the alignments is considered to be between the classical Precision and the percentage of correspondences whose members are either overlapping or both empty (*not disjoint*) due to the way the ontologies were populated.

For example, for the pair *cmt-edas*, the ra1 correspondence $\langle$ *cmt:Document* , *edas:Document* , $\equiv$ $\rangle$ is not interpreted as an equivalence for the Precision score. *cmt:Document* has for subclasses *cmt:Paper* and *cmt:Review*, whereas *edas:Document* has for subclasses *edas:Paper*, *edas:Review*, *edas:Programme* and

edas:SlideSet which were all populated. Therefore, even if the correspondence is correct with an equivalence relation, the instance sets of its members are not equivalent but one subsumes the other. Note that the instance sets of the correspondence members could also be overlapping if *cmt* had another subclass (*e.g.*, Website) which did not appear in *edas*.

Another limitation of the Precision score is related to the correspondences whose members are not populated in the dataset. For instance, $\langle$ *cmt:Preference , conference:Review_preference ,* $\equiv \rangle$ is a correct correspondence which was not detected as such in the Precision evaluation. The review preference of a reviewer for a paper was not part of the CQAs for the population process. There is therefore no instance for either member of the correspondence.

To compensate these errors, we use the *not disjoint* scoring metric in the Precision evaluation (Section 5.2.3). The score for a correspondence is 1 when the members are overlapping or both empty, and 0 otherwise.

This metric gives the upper bound of the precision of an alignment. When calculating the Harmonic Mean of CQA Coverage and Precision, the *overlap* CQA Coverage was used with the *not disjoint* Precision score to give an upper bound. Indeed, in the CQA Coverage, the source query will never return empty results.

The CQA Coverage of Ritze 2010 and AMLC is higher than thats of ra1 which they include. Overall, the CQA Coverage of the other alignments (Ontology Merging, Query Rewriting, ra1, Ritze 2010 and AMLC) is lower than the score of our approach. Indeed, ra1 only contains simple equivalence correspondences, Ritze 2010 and AMLC are mostly restrained to finding *(s:c)* class expressions correspondences (and therefore do not cover binary CQAs). The Ontology merging and query rewriting alignments are limited to *(s:c)*, *(c:s)* correspondences.

Globally, the Query rewriting alignment outperforms the Ontology merging in terms of CQA Coverage except for the *edas-confOf* pair. In the Ontology merging alignments, unions of properties were separated into individual subsumptions which were usable by the rewriting system. In the Query rewriting alignment, the subsumptions are unions. For example:

Query rewriting correspondence:

$\langle$ *confOf:starts_on , edas:startDate* $\sqcup$ *edas:hasStartDate ,* $\sqsupseteq \rangle$

$\langle$ *dom(confOf:Conference)* $\sqcap$ *confOf:starts_on , edas:startDate ,* $\equiv \rangle$

Ontology merging correspondences:

$\langle$ *confOf:starts_on , edas:startDate ,* $\sqsupseteq \rangle$

$\langle$ *confOf:starts_on , edas:hasStartDate ,* $\sqsupseteq \rangle$

Therefore, when a query contained the *edas:hasStartDate* relation, the Ontology merging correspondence could be used, but the Query rewriting ones could not.

Our approach obtains the best CQA Coverage scores except for the classical CQA Coverage where the Query rewriting alignment is slightly better (0.62 vs.

0.60). Our approach can generate *(c:c)* correspondences which cover more CQAs than the other alignments limited to *(s:s), (s:c)* and *(c:s)*.

The Precision of our approach is overall lower than the Precision of reference alignments (considering that their Precision score is between the classical and not disjoint score). Ritze 2010 only outputs equivalent or disjoint correspondences. Its Precision score is therefore the same (0.75) for all metrics. AMLC achieves a better classical Precision than our baseline approach but contains a high number of disjoint correspondences (37% of all the output correspondences had members whose instance sets were disjoint).

Overall, as expected, the Precision scores of the reference alignments are higher than those output by the matchers. Our approach relies on CQAs and for this reason, it gets higher CQA Coverage scores than Ritze 2010 and AMLC. More over, these two matchers both rely on correspondence patterns which limit the types of correspondences they can generate.

6.4 Evaluation on Taxon

6.4.1 The Taxon dataset: a LOD use case

The Taxon dataset is composed of 4 ontologies which describe the classification of species: AgronomicTaxon [Roussey *et al.* 2013], AgroVoc [Caracciolo *et al.* 2012], DBpedia [Auer *et al.* 2007] and TaxRef-LD [Michel *et al.* 2017]. All the ontologies are populated. The common scope of these ontologies is plant taxonomy. This dataset is the one from [Thiéblin *et al.* 2018d]. These ontologies share a common scope but their ontologies and instances differ.

Moreover, in DBpedia, the same information can be represented in various ways but irregularly across instances. For this reason, creating a set of reference and exhaustive CQAs is not easily feasible. The knowledge bases described by these ontologies are large. The English version of DBpedia describes more than 6.6 million entities alone and over 18 million entities[6]. The TaxRef-LD endpoint contains 2,117,434 instances[7] and the AgroVoc endpoint 754,874[7]. AgronomicTaxon has only been populated with the wheat taxonomy and only describes 32 instances.

All repositories except AgronomicTaxon have an available SPARQL endpoint:

- AgroVoc: `http://agrovoc.uniroma2.it:3030/agrovoc/sparql/`
- DBpedia: `http://www.dbpedia.org/sparql/`
- TaxRef-LD: `http://taxref.mnhn.fr/sparql/`

Linking the $\mathcal{T}box$ of these knowledge ontologies is challenging because of their size.

[6]Statistics from the 2016-10 release `https://wiki.dbpedia.org/develop/datasets/dbpedia-version-2016-10`.

[7]Tested on 2019/04/12.

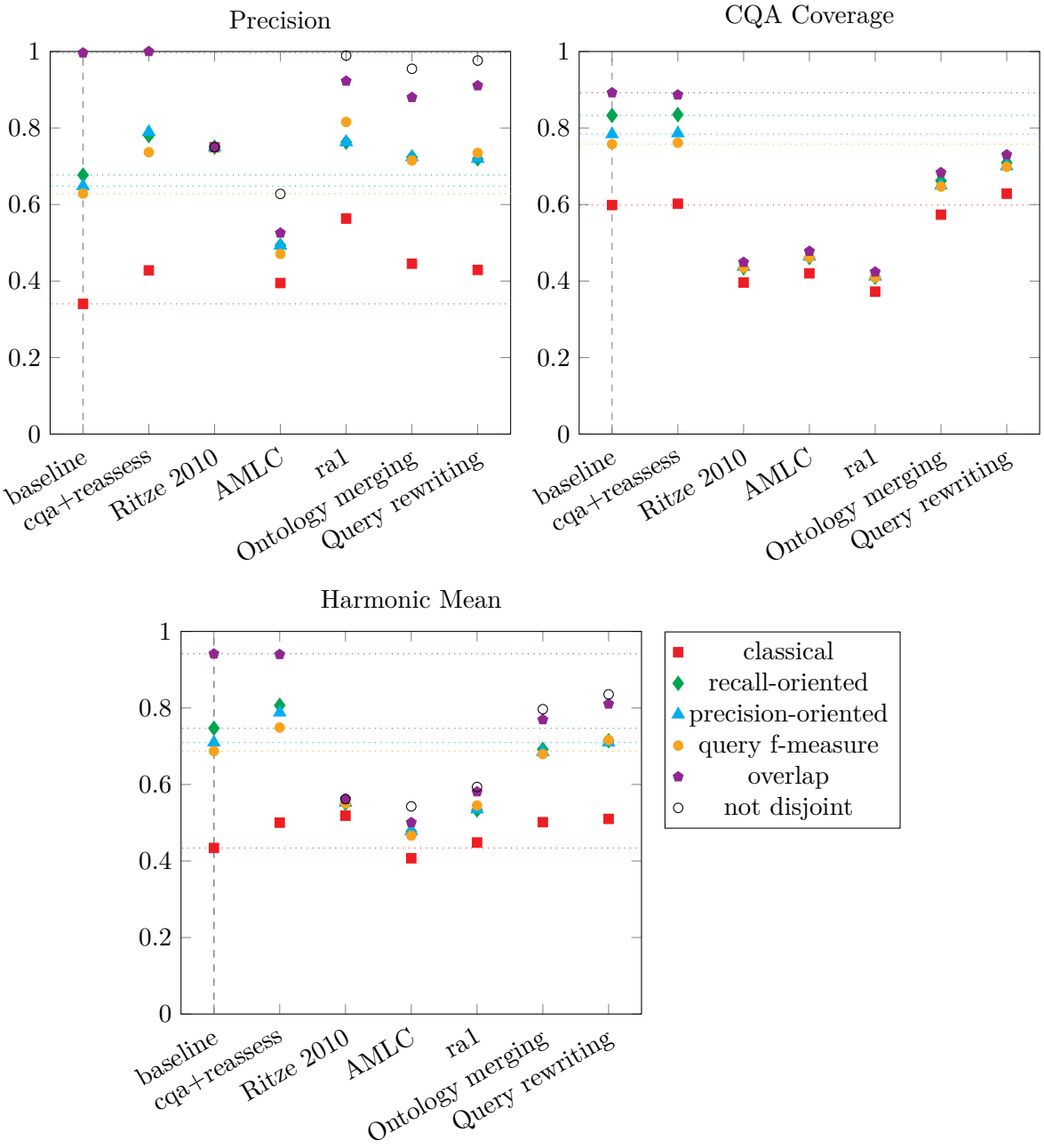


Figure 6.14: Results of the proposed approach, reference alignments and complex alignment generation approaches.

The CQAs used in this evaluation are the one presented in [Thiéblin *et al.* 2018d] which were manually written from AgronomicTaxon CQs [Roussey *et al.* 2013]. The CQA *What is the vernacular name of a taxon?* is not exactly covered by AgroVoc and DBpedia. These two ontologies represent the labels (in AgroVoc preferred or alternative labels) without specifying if they are scientific or vernacular names.

These ontologies have more or less overlapping taxa instances. For example, the taxon *Triticum monococcum* is a wheat species which is represented in AgronomicTaxon, AgroVoc, TaxRef-LD but not in DBpedia. The plant classification (plant taxonomy) is not exactly the same between two knowledge bases. For example, *Triticum spelta* is a subspecies of *Triticum aestivum* in AgronomicTaxon and TaxRef-LD while it is a separate species in AgroVoc and DBpedia.

AgronomicTaxon, AgroVoc and TaxRef-LD have well defined ontologies and focus on describing taxa. DBpedia is not specialised and extracted from Wikipedia[8] infoboxes and articles. This results in an uneven description of the instances. The SPARQL endpoint of DBpedia also contains a part of Wikidata ontology and instances. The SPARQL endpoint of TaxRef-LD includes a part of AgroVoc and Geospecies.

We tried to run our approach on the distant SPARQL endpoints but we encountered many server exceptions probably due to an unstable network connection on our side or an overload of the servers. We then decided to host a reduced version of the datasets on a local machine to avoid these problems. The reduced datasets contain all the plant taxa and their information (surrounding triples, annotations, *etc.*) from the SPARQL endpoint of the knowledge bases. Table 6.3 shows the number of plant taxa in each knowledge base. Even though the number of instances was reduced, the knowledge bases are still large-scale.

Table 6.3: Number of taxa and plant taxa in each knowledge base of the track, in its original and reduced version.

Version	AgronomicTaxon	AgroVoc	DBpedia	TaxRef-LD
Taxa (original)	32	8,077	306,833	570,531
Plant taxa (reduced)	32	4,563	58,257	47,058

6.4.2 Approach settings

The approach is run with the following settings on top of those from Section 6.1.

- Levenshtein threshold: 0.4
- Number of support answers: 1 and 10 (two runs)
- Instance matching: First, look for existing links (*owl:sameAs*, *skos:closeMatch*, *skos:exactMatch*) and if no target answer is found like

[8] `https://www.wikipedia.org/`

that, perform an exact label match. We consider a target answer found for a unary query when a URI which appears in the target knowledge-base is found. A target answer for a binary query is found when both URIs appear in the target knowledge base.

- No counter-example reassessment. Computing the percentage of counter-examples would last too long on this dataset. Moreover, as discussed in Section 6.4.1, in DBpedia at least, a DL formula with counter-examples is not necessarily a bad formula because the instances are not regularly populated.

6.4.3 Evaluation results

The number of correspondences per type is shown in Figure 6.15. The correspondences have been manually classified as equivalent, more general, more specific or overlapping. The classical, recall-oriented, precision-oriented and overlap scores have been calculated based on this classification. The results are shown in Figure 6.16.

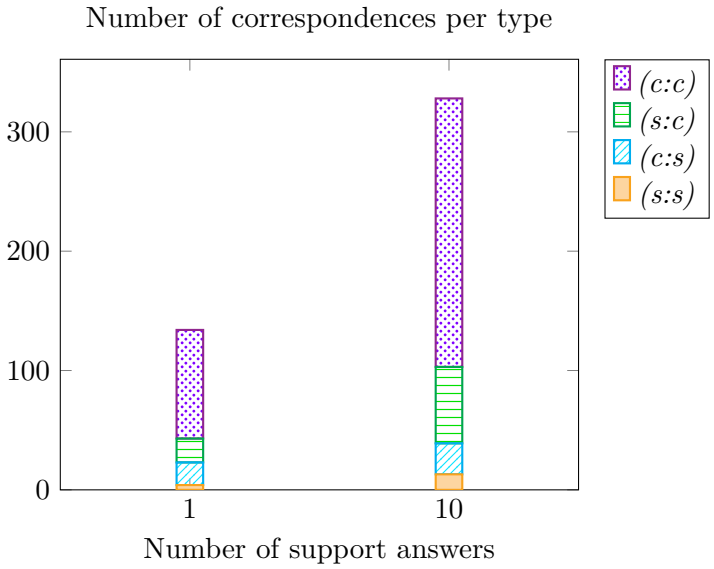


Figure 6.15: Number of correspondence per type for the approach with 1 and 10 support answers on the Taxon dataset.

Overall, the classical Precision and CQA Coverage scores are rather low. The Precision of the approach with 1 or 10 support answers is approximately the same. However, the CQA Coverage is higher with 10 instances. In comparison with the Conference dataset, this can be explained by the differences of population between the knowledge bases and the uneven population of a knowledge base in itself. We guess that the more support answers the approach takes, the best its CQA Coverage will be when dealing with unevenly populated ontologies.

The formula *skos:broader*$^-$ ∘ *skos:broader*$^-$ ∘ *agronto:hasTaxonomicRank* was found in the AgronomicTaxon-Agrovoc pair for the CQA *What is the taxonomic rank of which taxon?*. This correspondence shows that the plant taxonomies are not the same in every ontology. There seems to be no uniform and consensual classification of all the living beings in the knowledge bases of this dataset. In AgronomicTaxon, there is no distinction between *Tracheophyta* and *Magnolyphyta*. They are represented by the same taxon instance. In Agrovoc however, the classification is more fine-grained and they are represented by distinct instances. This difference of classification also occurs in the other pairs of ontologies, *Magnoliophyta* is considered a division rank in TaxRef-LD but a *phylum* rank in Agrovoc and AgronomicTaxon.

The uneven population of some knowledge bases leads to missing correspondences. For example, *agronto:hasTaxonomicRank* is not represented for every instance of Agrovoc. *agrovoc:c_35661* which is the *Asplenium* genus taxon has no *agronto:hasTaxonomicRank* property. When this instance was used as support instance, it could not lead to the detection of a correspondence involving its rank. When running our matching approach with only 1 support instance, using this instance would result in an empty set of correspondences for some CQAs. Consequently, the CQA Coverage is globally higher for the approach with 10 support answers.

The particularity of a dataset about species taxonomy is that two taxa are likely to share the same scientific name. Our exact label match strategy is therefore rather suited for such a dataset. In some cases however, it introduced noise. For example, a confusion was made between *wheat* the plant taxon and *wheat* the consumable good, or between a *division*, part of an administrative structure and the taxonomic rank *division*.

The Levenshtein-based string similarity brings noise. For example, the correspondence ⟨ *agrotaxon:GenusRank* , ∃*agronto:produces.*{*agrovoc:c_8373*} , ≡ ⟩ whose target member represents all the agronomic taxa which produce wheat has been output. This is due to the string similarity between the Malay label of wheat *"Gandum"* and the English label *"Genus"* of the *agrotaxon:GenusRank* class. We could have chosen to compare labels in the same language together but sometimes, the language of a label was missing, sometimes the scientific name was either tagged as English or Latin.

The total runtime over the 12 pairs of ontologies was **99,297s (27.6h)** for the approach with 1 support instance and **113,474s (31.5h)** for the approach with 10 support answers. The runtime per pair of ontologies is detailed in Table 6.4. Three factors explain the different runtime over the pairs of ontologies in Table 6.4.

Query difficulty Some SPARQL CQAs were really long to run on large knowledge bases. For example, the CQA *What is the higher rank taxon of this taxon ?* is a union of seven properties in DBpedia (*dbo:genus*, *dbo:kingdom*, *dbo:order*, *dbo:family*, *etc.*) the query execution takes between 1 and 10 hours. This query is run as many times as no match is found for its instances or all its

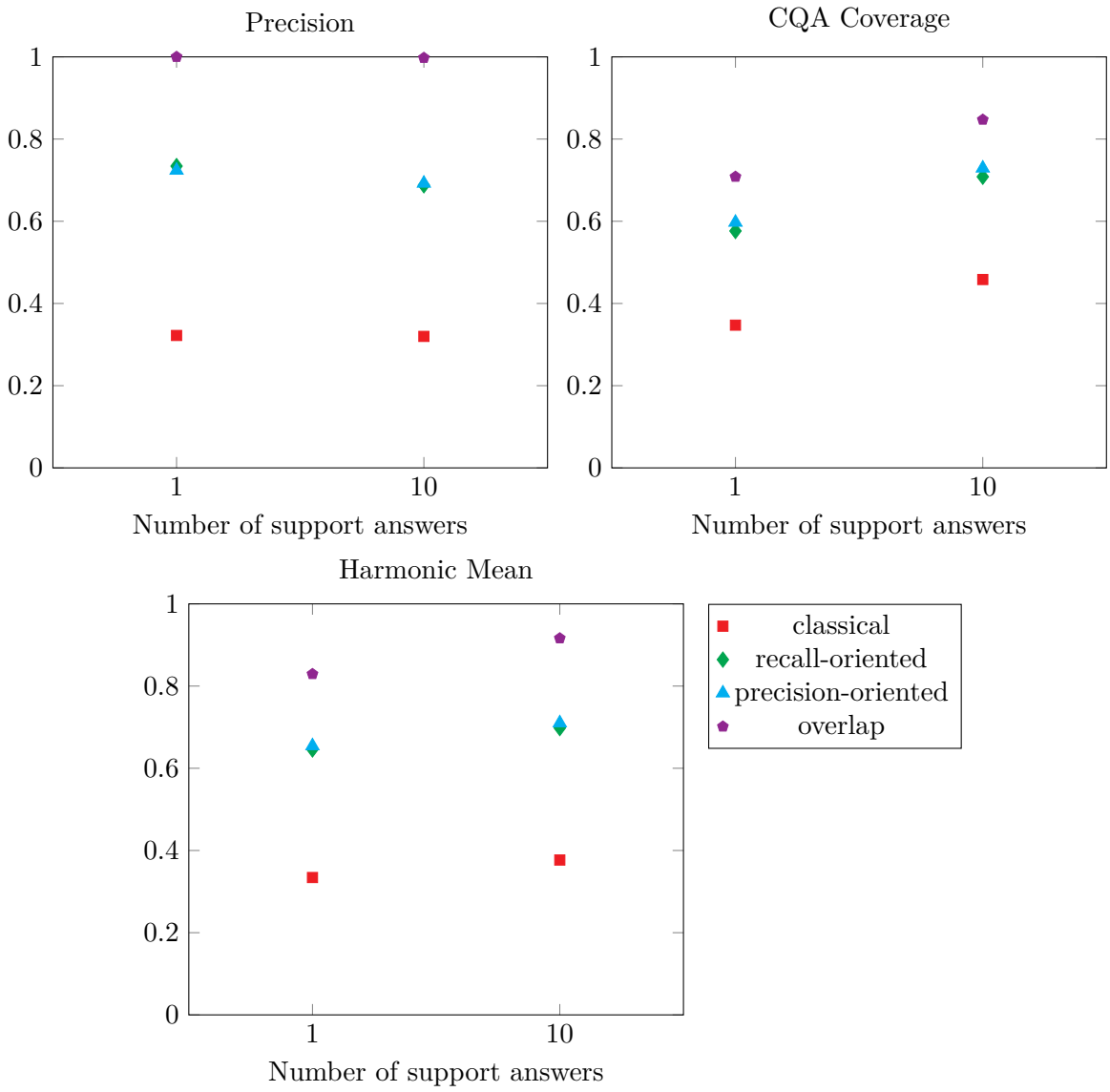


Figure 6.16: Results of the approach with 1 and 10 support answers on the Taxon dataset.

Table 6.4: Runtime (s) of our approach on each pair of ontologies. These measures are based on a single run.

1 sup. inst.	source \ target	AgronomicTaxon	AgroVoc	DBpedia	TaxRef-LD
	AgronomicTaxon	–	67	**4**	421
	AgroVoc	747	–	238	27,776
	DBpedia	**50,542**	2,733	–	2,477
	TaxRef-LD	4,517	5,758	4,017	–

10 sup. inst	source \ target	AgronomicTaxon	AgroVoc	DBpedia	TaxRef-LD
	AgronomicTaxon	–	1,084	1,019	753
	AgroVoc	1,173	–	**220**	29,351
	DBpedia	**52,214**	4,813	–	5,062
	TaxRef-LD	4,718	8,005	5,062	–

results have been retrieved. It resulted in much higher execution time for the ontology pairs when DBpedia was the source ontology.

Percentage of source common instances As shown in Table 6.3, the number of taxa instances can be different between knowledge-bases. AgronomicTaxon and DBpedia share 22 instances. When AgronomicTaxon, which has only 32 instances is matched to DBpedia, finding a common instance between the two is rather easy because about 68% of its instances have an equivalent in DBpedia. The other way around, is way harder because only 0.04% of DBpedia taxa instances have an equivalent in AgronomicTaxon.

Existence of instance links When no explicit instance links exist between two knowledge bases, all the source instances are explored (to find hypothetical links) then, the exact label match is performed. This can take a lot of time according to the size of the target knowledge base.

6.5 Discussion

Even though the similarity metric in this implementation of the approach is naive, the results of the approach are quite high (the query f-measure Harmonic Mean score of the baseline approach is 0.70). The approach is rather CQA Coverage-oriented, as it will try to output a correspondence for each source CQA. The values of the CQA Coverage are overall higher than the Precision values. The baseline achieves a classical CQA Coverage of 0.60 which means that 60% of the CQA have been covered with a strictly equivalent match by our approach while its classical Precision score is only 0.34. Using existing identity links gives better results than exact label matches. The use of CQAs improves the Precision and CQA Coverage performance of the approach. The counter-example exploration (similarity reassessment phase) increases significantly the Precision but extends the runtime drastically.

In comparison with the other matching approaches evaluated, our approach has high CQA Coverage scores. It would be interesting to compare our approach with extensional approaches such as [Hu *et al.* 2011, Walshe *et al.* 2016, Parundekar *et al.* 2012, Parundekar *et al.* 2010] (whose implementation was not available) even though all of them are limited to *(s:c)* and *(c:s)* correspondences.

The experiment on the Taxon dataset showed that our approach can perform on large knowledge bases but its runtime can be very long. It also highlighted the need for regularly populated knowledge bases and quality instance links.

The experiments described in this Chapter have helped answer the research questions of this thesis:

Is one common instance per Competency Question for Alignment enough evidence to generate complex correspondences? In the experiments on the Populated Conference benchmark and on the Taxon dataset, the approach based on only one common instance could generate complex correspondences. While in the Populated Conference dataset, the results with one support answer are slightly higher than with more support answers, in the Taxon dataset, they are lower. This can be explained by the irregular population of some Taxon dataset ontologies as well as the existence of inaccurate instance links. These aspects are also discussed in the next research question.

What is the impact of the number of support answers on the alignment quality? The impact of the number of support answers depends on the ontology population. In the experiment on the Taxon dataset, using 10 support answers instead of 1 improved the quality of the alignment. The reason is because the ontologies are not all regularly populated. The Precision score was about the same for 1 or 10 support answers while the CQA Coverage scores are about 12% higher with 10 support answers than with 1. In the Conference dataset which is regularly populated, using more support answers reduced the Precision score because noise was introduced. When dealing with many support answers, the noisy correspondences could be filtered out based on their frequency. For example, the formula $\exists$ *conference:has_the_last_name.*{*"Benson"*} only appears for one support instance of Person whereas *conference:Person* appears for all support answers. However, it was a choice in the approach design to not disregard "accidental" formulae (those which only appear for 1 answer and not in the other answers) because unevenly populated datasets may be faced with this problem. For example, in DBpedia, the taxonomic rank of a taxon can be represented in different ways: the label of a property (*e.g.*, a taxon is the *dbo:genus* of another taxon or has a *dbp:genus* literal), a link to the rank instance (*e.g.*, link to *dbr:Genus*), or the presence of a rank authority (*e.g.*, *dbp:genusAuthority*). The problem is that all the genus instances do not share the same representation. It is possible that among the genus rank instances, only one is represented as a genus rank thanks to the *dbp:genusAuthority*. This may seem statistically accidental but it is relevant to our problem.

What is the impact of the quality of the instance links on the generated alignments quality? If the links are expressed and not erroneous, the generated alignment will have a better Precision and CQA Coverage. If wrong links are used, as in the experiment with exact label matches, a lot of noise is introduced and the Precision of the alignment decreases. The CQA Coverage score also decreases because the noise can prevent correct support answers to be found and all the output correspondences for a given correspondence can be erroneous. The quality of the instance links impacts the runtime, Precision and CQA Coverage scores of our approach. This highlights the need for effective instance matching systems and the disambiguation of existing links.

Can Competency Questions for Alignment improve the Precision of generated alignments? Both the Precision and CQA Coverage scores are higher when the approach relies on CQAs. The baseline and the *cqa+reassess* variants obtain a Precision score in average 15% above that of their generated query variants (*query* and *query+reassess*). The CQA Coverage also increases in average of 14% because the CQAs help generate *(c:c)* correspondences which are relevant to the user (and to the evaluation). However, as part of the input CQA is used for the calculation of the CQA Coverage score, the evaluation is somewhat biased. In a user's need-oriented scenario, nonetheless, this evaluation makes sense: if users input their needs into a matcher, they may expect an output alignment which covers them well.

Does similarity reassessment based on counter-examples improve the quality of the generated alignments? When comparing the results of the baseline approach with the *cqa+reassess* variant which reassesses the similarity based on counter-examples, the CQA Coverage remains the same while the Precision is improved. The Precision of the *cqa+reassess* variant is between 8% and 15% higher than that of the baseline. The Precision of the *query+reassess* variant is between 6% and 17% higher than that of the *query* variant while its CQA Coverage is 3% lower.

Can CQAs improve the runtime performance of complex ontology matching? Comparing the *cqa+reassess* and *query+reassess* variants, the runtime is improved thanks to the use of CQAs. However, when comparing the baseline *cqa* to the *query* variants, the runtime is the same. This depends on the complexity of the CQAs. Our baseline approach took about 2 hours to align the 20 pairs of ontologies of the Populated Conference dataset. AMLC is much faster than our baseline approach as it took only 3 minutes. Ritze 2010 is also faster than our baseline approach. However, when running with only 1 support answer, our approach takes 33 minutes which make it faster than Ritze 2010.

What is the impact of the CQA on the type of output correspondence? Overly complex correspondences can be introduced in the alignment because of the way the approach uses the input CQAs. We counted that about 14% of the

(c:c) correspondences output by the baseline approach are overly complex, which means that they could be decomposed into simple correspondences. This comes from the translation of the input CQA into a DL formula without any analysis or decomposition of its elements. Moreover, the approach outputs more *(s:c)* and *(c:c)* correspondences than *(s:s)* and *(c:s)* which shows a tendency to output more complex than simple correspondences.

Chapter 7

Conclusion and future work

While most works focus on finding simple equivalences between ontologies, more expressiveness is sometimes needed to express links such as $\langle$ o_1*:AcceptedPaper* , $\exists o_2$*:hasDecision.*o_2*:Acceptance* , $\equiv \rangle$. These expressive correspondences are qualified as "complex", and an alignment containing such correspondences is a **complex alignment**.

This thesis studied the problematic of complex alignment generation. Our main contributions are:

- a classification of complex alignment generation approaches and a study of their existing implementations,
- a complex alignment generation approach based on Competency Questions for Alignment (CQAs),
- a complex alignment benchmark.

These contributions and their perspectives are discussed in this section.

Complex alignment generation approach classification The classification proposed in Chapter 3 analyses the specificities of complex alignment generation approaches. It is based on the structure guiding the construction of the correspondences, and on their type. However, this classification is not absolute. Some approaches can be classified in more than one category, and a new approach (*i.e.*, one that we did not foresee) could fit in none of the categories. This being said, the proposed classification complements the generic ontology matching classifications and helped us analyse the complex alignment generation approaches. The state of the art of complex alignment generation approaches proposed in Chapter 3 is intended to those who are faced with a complex alignment generation problem. It would be interesting to further specify the classification of [Euzenat & Shvaiko 2013] on their *instance-based* category, as these approaches all contain a generalisation phase because they infer general rules from instances (facts). In the [Euzenat & Shvaiko 2013] classification, they are all classified in the same category. We think that refining this category by investigating generalisation techniques [Bishop 2007, Ganter *et al.* 2005, Agrawal *et al.* 1993] could highlight which have been applied to ontology matching, and which have not. This would give pointers for new systems on the pros and cons of each technique, and how their cons could be overcome.

Complex alignment generation based on CQAs The novelty of the complex alignment generation approach we propose is that it relies on user's needs, and for that we introduce the concept of CQAs.

For each CQA expressed as a SPARQL query over the source knowledge base, the approach retrieves its answers and finds equivalent answers (instances or pairs of instances) in the target knowledge base. The surroundings of the target answers are then lexically compared to the entities of the CQA. The most similar surroundings are kept and transformed into a DL formula which will become the target member of the correspondence. The source member of the correspondence is the CQA transformed into a DL formula.

This approach (detailed in Chapter 4) uses the CQAs and common instances to reduce the correspondence search space. Thanks to that, it can produce *(c:c)* correspondences whose members are not restricted to a pattern. The approach can find correspondences even if there is only one common answer between the two knowledge bases.

The evaluation of this approach in Chapter 6 shows that it is a promising complex alignment generation approach as it covers the CQAs rather well. The Precision of the approach can be improved, for instance by using alternative similarity metrics. As short term perspective, we plan on lemmatising the labels, then disambiguating the lemmas by matching them with WordNet [Miller 1992] synsets using word embeddings. The bag of disambiguated lemmas would then be compared using a synset distance on a linguistic resource such as WordNet.

One can state that the scope of the output alignment is limited to that of the CQAs. However, we argue that the correspondences which were found could be saved in an alignment repository. Therefore, the alignment between knowledge bases would be incrementally created. Each new knowledge need of a user would add correspondences to this alignment. This is inline with **community-driven ontology matching** as in [Zhdanova & Shvaiko 2006]. However, such a repository would require that the alignments are expressed in the same language or interoperable languages. Even if various alignment languages have been proposed (Section 2.2.5 in Chapter 2), the EDOAL vocabulary implemented in the Alignment API can be seen as an up-coming standard given its use in the complex OAEI track. However, EDOAL has limited expressiveness, as discussed by [Zhou *et al.* 2018]. Extending the pattern library of EDOAL could be a solution so that it could express more correspondences. Furthermore, an ontology alignment repository would require correspondence validation or edition by a user, especially if the Precision of the alignments is low. This entails that the correspondences should also be visualised in a user-friendly layout. As far as we know, there are few edition or visualisation tools for EDOAL. Ontology alignment visualisation and edition field are still focused on simple correspondences [Severo *et al.* 2017]. Some tools such as Protégé [Musen 2015] or [Hassanpour *et al.* 2010] allow for axiom (OWL) or rules (SWRL) visualisation and edition, while [Silva & Rocha 2003] and [Chondrogiannis *et al.* 2014] are dedicated to simple and complex ontology alignment. Only [Chondrogiannis *et al.* 2014], which is not available online, can deal

with EDOAL correspondences. This makes, however, this format only usable or browse-able by experts.

The approach relies on the assumption that a user is able to express the CQAs as SPARQL queries over one of the ontologies. This is a limitation of the approach. However, natural language to SPARQL query systems exist [Pradel *et al.* 2014], in particular in the field of Question Answering over Linked Data (QALD) [Unger *et al.* 2014]. Such systems mitigate the issue, by hiding the underlying technical complexity to the end user, who therefore does not need to be an expert in Semantic Web technologies.

The approach mostly relies on existing instance links or exact label instance matching process. However, when the links are not missing, they may be erroneous [Halpin *et al.* 2015]. Instance-matching techniques could be investigated for this issue. We are especially interested in using keys and linkkeys which express conditions under which two instances should be considered equivalent [Atencia *et al.* 2014]. Linkkeys may require complex correspondences; in [Symeonidou *et al.* 2017], conditional keys which may be abstract classes (*i.e.*, complex class expressions) are sought between two knowledge bases. Moreover, the object properties in the linkkeys could need complex expressions including for instance property paths. It would be interesting to consider an iterative approach which relies on few existing instance links to find ontology correspondences and vice-versa until no more instance link or correspondence is found.

Another limitation is the need of a common instance between the aligned knowledge bases. In future works, an approach which would not need such information could be studied. We could, for example, rely on instances which share some properties instead of equivalent ones. We could also consider a strategy based on both linguistic and semantic properties of the ontologies and CQAs, which would not rely on instances.

Finally, the semantics of correspondence confidence could be investigated. In our approach, it is a simple aggregation of similarity metrics, while in [Cheatham & Hitzler 2014] it represents the consensus of experts or "Turker" (people paid to execute micro-tasks on Amazon Mechanical Turk) on simple correspondences. [Amini *et al.* 2016] studied the impact of the type of question, of how much context of the entities is presented to the Turker and how is this context represented. Different level of confidence for correspondences implies some uncertainty in ontology matching. This problem identified as a challenge by [Shvaiko & Euzenat 2008] has been recently studied by [Essaid *et al.* 2014].

Complex alignment benchmark Few works studied the automatic evaluation of complex alignments. In Chapter 5, we have proposed a first benchmark made of an instance and CQA-based evaluation system and an associated dataset.

This benchmark is a first proposition. One of its strengths is the automatisation of the testing. However, the query rewriting system on which it relies could be improved, in particular for dealing with *(c:c)* correspondences. For instance, we

could propose a system which would first retrieve the instances of the source query. It would then explore the correspondences of the alignment and try combinations of their source members until the results of the source query are met. The rewritten query would in this case be the combination of the target members. This approach does not take semantics into account at all. We could also go in the opposite direction and extract a logic formula out of the initial query. Then, the source members of the correspondences could be semantically compared to the formula replacing some of its parts if semantically equivalent. This requires semantic comparison as for complex alignment evaluation and reasoning.

The workflow on which this benchmark is based could be used as the foundation of different approaches. For example, a semantic comparison of correspondences or queries (*i.e.*, without instance comparison) would be an interesting lead for future research.

Comparing the evaluation system with others is also important. In future works, a comparative study of evaluation strategies could be run. The challenges of such as strategy is to list exhaustively the types of expression, how constructors can be factorised and how what relation exists between two constructions given the axioms of the ontology. For example, the expression $\exists$ *o2:hasDecision.o2:Acceptance* is equivalent to $\geq$ 1 *o2:hasDecision.o2:Acceptance*, and more specific than $\exists$ *o2:hasDecision.o2:Decision.*

There may however be problems where there are two possible correspondences. For example, the correspondences c_1=$\langle$ *o1:AcceptedPaper* , $\exists$ *o2:hasDecision.o2:Acceptance* , $\equiv$ $\rangle$ and c_2= $\langle$ *o1:AcceptedPaper* , $\exists$ *o2:acceptedBy.*$\top$, $\equiv$ $\rangle$ could both be in a reference alignment A_1 = $\{c_1, c_2\}$. They could also be joint in a union like: c_3=$\langle$ *o1:AcceptedPaper* , $\exists$ *o2:hasDecision.o2:Acceptance* $\sqcup$ $\exists$ *o2:acceptedBy.*$\top$, $\equiv$ $\rangle$, A_2 = $\{c_3\}$. We consider in this example that there is no axiom in o_2 stating that $\exists$ *o2:hasDecision.o2:Acceptance* $\equiv$ $\exists$ *o2:acceptedBy.*$\top$. If an evaluated alignment contains only one of the two correspondences, *e.g.*, c_1, the instance-based comparison with A_1 or A_2 makes no difference over a consistently populated dataset as the one proposed in this thesis. However, for semantic comparison, the result could be different with A_1 or A_2. The evaluation system with A_1 as reference could consider that the alignment missed a correspondence. With A_2 as correspondence, it could consider that since only a part of the union has been found, the evaluated correspondence is more specific than the reference one instead of equivalent. With this example, we only discuss a small part of the problems that such a comparison would have to tackle.

To conclude this manuscript, we discuss the ethical perspective of interoperability, as complex alignments aim at making ontologies interoperable.

Ethical interoperability "*Data is the oil, some say the gold, of the 21st century — the raw material that our economies, societies and democracies are increasingly being built on,*" Joe Kaeser the CEO of Siemens said[1].

This metaphor, although not exactly accurate (data is not a finite resource), can be extended with unwanted side effects. Data, like oil, has impacted the daily life of many making information accessible from anywhere, simplifying bureaucracy, *etc.* However, there is a price to pay (for oil, it was the ruin of the environment but we will stop the metaphor here). The personal data either published by people or retrieved from the people is mined and exploited, and its applications are sometimes ethically questionable.

Interoperability, the very objective of ontology matching, aims at linking the data from diverse sources. With such technology, refined profiles of individuals can be produced. And this has tangible consequences.

First, profiling can be dangerous for the integrity of the individuals themselves. The American Civil Liberties Union warns about racial profiling, which may lead to discrimination and persecution[2]. Personal data such as medical records, beliefs, sexual orientation, ethnicity, genome, *etc.*, is extremely sensitive and its access should be regulated. The French commission for data protection and liberties (CNIL), stood up against a unique identification number in order to keep the data about individuals separated and ensure its use in the sole purpose of which it was collected[3].

Then, profiling can be used to analyse and predict individual behaviours. If behaviour prediction is an option, then why not influence it? Recently the Cambridge Analytica scandal was revealed showing how profiling can serve political interests [Rosenberg *et al.* 2018]. With such power, the data holders can shape the world as they please [Zuboff 2019].

In general, those who produce the data (willingly or not) get less benefits than those who exploit it [Pasquale 2018]. Regulating the access and application of personal data is a political matter [Rimbert 2016]. Indeed, the creation of extremely detailed profiles of individuals jeopardises fundamental freedoms.

The purpose of this thesis is to enable interoperability between knowledge repositories. However, linking data should only be done after weighing its possible impact on society and the life of people.

[1] https://www.linkedin.com/pulse/technology-society-digital-transformation-joe-kaeser

[2] https://www.aclu.org/issues/racial-justice/race-and-criminal-justice/racial-profiling

[3] https://www.senat.fr/lc/lc181/lc181_mono.html

Appendix A

Appendix

A.1 Acronyms

CANARD Complex Alignment Need-based Abox-based Relation Discovery.

CML Conceptual Modelling Language.

CQ Competency Question.

CQA Competency Question for Alignment.

DL Description Logic.

EDOAL Expressive and Declarative Ontology Alignment Language.

FOL First-Order Logic.

HTML HyperText Markup Language.

HTTP HyperText Transfer Protocol.

IRI Internationalised Resource Identifier.

JSON JavaScript Object Notation.

LD Linked Data.

LOD Linked Open Data.

OA4QA Ontology Alignment for Query Answering.

OAEI Ontology Alignment Evaluation Initiative.

OWL Web Ontology Language.

QALD Question Answering over Linked Data.

R2RML RDB to RDF Mapping Language.

RDF Ressource Description Framework.

RDFS RDF Schema.

SPARQL SPARQL Protocol And RDF Query Language.

SQL Structured Query Language.

SWRL Semantic Web Rule Language.

UMLS Unified Medical Language System.

URI Uniform Resource Identifier.

W3C World Wide Web Consortium.

WWW World Wide Web.

XML Extensible Markup Language.

A.2 Namespaces and prefixes

All along the manuscript, the following prefixes have been used to represent namespaces:

Prefix	Namespace
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl	http://www.w3.org/2002/07/owl#
skos	http://www.w3.org/2004/02/skos/core#
dc	http://rs.tdwg.org/dwc/terms/
cmt	http://cmt#
conference	http://conference#
confOf	http://confOf#
edas	http://edas#
ekaw	http://ekaw#
agrotaxon	http://ontology.irstea.fr/agronomictaxon/core#
agronto	http://aims.fao.org/aos/agrontology#
agrovoc	http://aims.fao.org/aos/agrovoc/
dbo	http://dbpedia.org/ontology/
dbp	http://dbpedia.org/property/
dbr	http://dbpedia.org/resource/
txrfp	http://taxref.mnhn.fr/lod/property/
txrfrk	http://taxref.mnhn.fr/lod/taxrank/

A.3 Schemata legend

There are two types of schemata to represent knowledge in this manuscript. There are a few differences between their legends

- **Ontology schemata** These schemata represent the axioms of the ontologies. The formalism used is that of [Stapleton *et al.* 2014] which is a set-oriented representation. Blue circles represent classes, green arrows properties and red dots instances or values. This formalisation is well suited to represent axioms such as class subsumption, class disjunction or overlap, property domain and range, *etc.* However, it lacks flexibility to easily represent knowledge graphs.

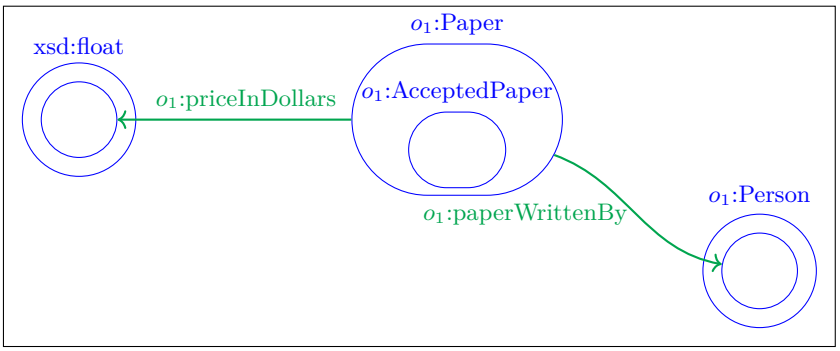


Figure A.1: Ontology schema

- **Knowledge graph schemata** This schemata represent the sets of triples which as in a knowledge graph. The classes are blue ellipses, the instances black rectangles and the object properties green arrows. The domain and range of a properties are not represented.

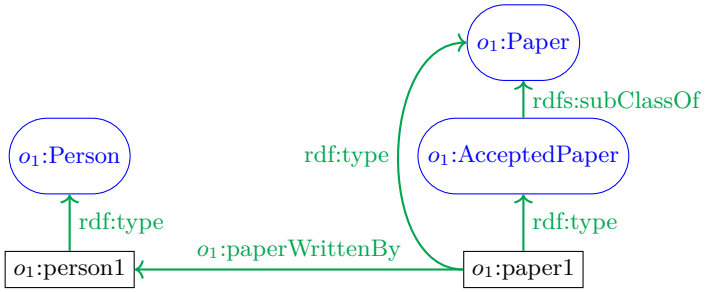


Figure A.2: Knowledge graph schema

Bibliography

[Agrawal *et al.* 1993] Rakesh Agrawal, Tomasz Imieliński and Arun Swami. *Mining association rules between sets of items in large databases.* In Sushil Jajodia and Peter Buneman, editors, Proceedings of the 1993 ACM SIGMOD Conference, Washington DC, USA, May 26-28 1993, volume 22, pages 207–216. ACM, 1993. (Cited in pages 86 and 149.)

[Algergawy *et al.* 2018] Alsayed Algergawy, Michelle Cheatham, Daniel Faria, Alfio Ferrara, Irini Fundulaki, Ian Harrow, Sven Hertling, Ernesto Jiménez-Ruiz, Naouel Karam, Abderrahmane Khiat, Patrick Lambrix, Huanyu Li, Stefano Montanelli, Heiko Paulheim, Catia Pesquita, Tzanina Saveta, Daniela Schmidt, Pavel Shvaiko, Andrea Splendiani, Elodie Thiéblin, Cássia Trojahn, Jana Vatascinová, Ondrej Šváb Zamazal and Lu Zhou. *Results of the Ontology Alignment Evaluation Initiative 2018.* In Proceedings of the 13th International Workshop on Ontology Matching co-located with the 17th International Semantic Web Conference, OM@ISWC 2018, Monterey, CA, USA, October 8, 2018., pages 76–116, 2018. (Cited in pages 64 and 102.)

[Amarger 2015] Fabien Amarger. *Vers un système intelligent de capitalisation de connaissances pour l'agriculture durable : construction d'ontologies agricoles par transformation de sources existantes. (Towards an intelligent knowledge capitalization for sustainable agriculture : agricultural building ontologies by transforming existing sources).* PhD thesis, University of Toulouse II, France, 2015. (Cited in page 6.)

[Amini *et al.* 2016] Reihaneh Amini, Michelle Cheatham, Pawel Grzebala and Helena B. McCurdy. *Towards best practices for crowdsourcing ontology alignment benchmarks.* In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Michelle Cheatham, Oktie Hassanzadeh and Ryutaro Ichise, editors, Proceedings of the 11th International Workshop on Ontology Matching co-located with the 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 18, 2016., volume 1766 of *CEUR Workshop Proceedings*, pages 1–12. CEUR-WS.org, 2016. (Cited in page 151.)

[An & Song 2008] Yuan An and Il-Yeol Song. *Discovering semantically similar associations (SeSA) for complex mappings between conceptual models.* In Qing Li, Stefano Spaccapietra, Eric S. K. Yu and Antoni Olivé, editors, Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings, volume 5231 of *Lecture Notes in Computer Science*, pages 369–382. Springer, 2008. (Cited in pages 47, 49, 56, 57, 58 and 59.)

[An *et al.* 2005a] Yuan An, Alexander Borgida and John Mylopoulos. *Constructing Complex Semantic Mappings Between XML Data and Ontologies.* In

Yolanda Gil, Enrico Motta, V. Richard Benjamins and Mark A. Musen, editors, The Semantic Web - ISWC 2005, 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005, Proceedings, volume 3729 of *Lecture Notes in Computer Science*, pages 6–20. Springer, 2005. (Cited in pages 50, 51, 56, 57, 58 and 59.)

[An *et al.* 2005b] Yuan An, Alexander Borgida and John Mylopoulos. *Inferring Complex Semantic Mappings Between Relational Tables and Ontologies from Simple Correspondences.* In Robert Meersman, Zahir Tari, Mohand-Said Hacid, John Mylopoulos, Barbara Pernici, Özalp Babaoglu, Hans-Arno Jacobsen, Joseph P. Loyall, Michael Kifer and Stefano Spaccapietra, editors, On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2005, Agia Napa, Cyprus, October 31 - November 4, 2005, Proceedings, Part II, volume 3761 of *Lecture Notes in Computer Science*, pages 1152–1169. Springer, 2005. (Cited in pages 50, 51, 56, 57, 58 and 59.)

[An *et al.* 2012] Yuan An, Xiaohua Hu and Il-Yeol Song. *Learning to discover complex mappings from web forms to ontologies.* In Xue-wen Chen, Guy Lebanon, Haixun Wang and Mohammed J. Zaki, editors, 21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012, pages 1253–1262. ACM, 2012. (Cited in pages 47, 56, 57, 58 and 59.)

[Arnold 2013] Patrick Arnold. *Semantic Enrichment of Ontology Mappings: Detecting Relation Types and Complex Correspondences.* In Kai-Uwe Sattler, Stephan Baumann, Felix Beier, Heiko Betz, Francis Gropengießer and Stefan Hagedorn, editors, Proceedings of the 25th GI-Workshop "Grundlagen von Datenbanken 2013", Ilmenau, Germany, May 28 - 31, 2013, volume 1020 of *CEUR Workshop Proceedings*, pages 34–39. CEUR-WS.org, 2013. (Cited in pages 38, 39, 42, 56, 57, 58 and 59.)

[Atencia *et al.* 2014] Manuel Atencia, Jérôme David and Jérôme Euzenat. *Data interlinking through robust linkkey extraction.* In Torsten Schaub, Gerhard Friedrich and Barry O'Sullivan, editors, ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014), volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 15–20. IOS Press, 2014. (Cited in page 151.)

[Auer *et al.* 2007] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak and Zachary Ives. *DBpedia: A Nucleus for a Web of Open Data.* In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber and Philippe Cudré-Mauroux, editors, The Semantic Web: The 6th International Semantic Web Conference ISWC

and the 2nd Asian Semantic Web Conference ASWC, volume 4825 of *LNCS*, pages 722–735, Busan, Korea, November 2007. Springer Berlin Heidelberg. (Cited in page 138.)

[Baader *et al.* 2003] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi and Peter F. Patel-Schneider, editors. The description logic handbook: Theory, implementation, and applications. Cambridge University Press, 2003. (Cited in page 10.)

[Baader *et al.* 2009] Franz Baader, Ian Horrocks and Ulrike Sattler. *Description Logics*. In Steffen Staab and Rudi Studer, editors, Handbook on Ontologies, Second edition, International Handbooks on Information Systems, pages 21–43. Springer-Verlag Berlin Heidelberg, 2009. (Cited in page 8.)

[Berners-Lee & Fischetti 2000] Tim Berners-Lee and Mark Fischetti. Weaving the web - the original design and ultimate destiny of the world wide web by its inventor. HarperBusiness, 2000. (Cited in page 1.)

[Berners-Lee *et al.* 2001] Tim Berners-Lee, James Hendler and Ora Lassila. *The semantic web*. Scientific american, vol. 284, no. 5, pages 28–37, 2001. (Cited in pages 1 and 5.)

[Bishop 2007] Christopher M. Bishop. Pattern recognition and machine learning, 5th edition. Information science and statistics. Springer, 2007. (Cited in page 149.)

[Bodenreider *et al.* 2005] Olivier Bodenreider, Terry F. Hayamizu, Martin Ringwald, Sherri de Coronado and Songmao Zhang. *Of Mice and Men: Aligning Mouse and Human Anatomies*. In AMIA 2005, American Medical Informatics Association Annual Symposium, Washington, DC, USA, October 22-26, 2005. AMIA, 2005. (Cited in page 62.)

[Borgida 1996] Alex Borgida. *On the relative expressiveness of description logics and predicate logics*. Artificial intelligence, vol. 82, no. 1-2, pages 353–367, 1996. (Cited in page 80.)

[Boroditsky 2011] Lera Boroditsky. *How language shapes thought*. Scientific American, vol. 304, no. 2, pages 62–65, 2011. (Cited in pages 1 and 15.)

[Borst 1997] Willem Nico Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. PhD thesis, Institute for Telematica and Information Technology, University of Twente, Enschede, The Netherlands, 1997. (Cited in page 7.)

[Boukottaya & Vanoirbeek 2005] Aida Boukottaya and Christine Vanoirbeek. *Schema matching for transforming structured documents*. In Anthony Wiley and Peter R. King, editors, Proceedings of the 2005 ACM Symposium on

Document Engineering, Bristol, UK, November 2-4, 2005, pages 101–110. ACM, 2005. (Cited in pages 39, 41, 56, 57, 58 and 59.)

[Bouquet *et al.* 2003] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini and Heiner Stuckenschmidt. *C-OWL: Contextualizing Ontologies.* In Dieter Fensel, Katia P. Sycara and John Mylopoulos, editors, The Semantic Web - ISWC 2003, Second International Semantic Web Conference, Sanibel Island, FL, USA, October 20-23, 2003, Proceedings, volume 2870 of *Lecture Notes in Computer Science*, pages 164–179. Springer, 2003. (Cited in page 70.)

[Bouquet *et al.* 2004] Paolo Bouquet, Jérôme Euzenat, Enrico Franconi, Luciano Serafini, Giorgos Stamou and Sergio Tessaris. *D2. 2.1 Specification of a common framework for characterizing alignment.* Project deliverable, Knowledge Web, 2004. (Cited in page 16.)

[Calvanese *et al.* 2017] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro and Guohui Xiao. *Ontop: Answering SPARQL queries over relational databases.* Semantic Web, vol. 8, no. 3, pages 471–487, 2017. (Cited in page 38.)

[Caracciolo *et al.* 2012] Caterina Caracciolo, Armando Stellato, Sachit Rajbahndari, Ahsan Morshed, Gudrun Johannsen, Johannes Keizer and Yves Jaques. *Thesaurus maintenance, alignment and publication as linked data: the AGROVOC use case.* International Journal of Metadata, Semantics and Ontologies, vol. 7, no. 1, page 65, 2012. (Cited in page 138.)

[Chang *et al.* 2003] Kevin Chen-Chuan Chang, Bin He, Chengkai Li and Zhen Zhang. *The UIUC Web Integration Repository*, 2003. (Cited in page 61.)

[Cheatham & Hitzler 2013] Michelle Cheatham and Pascal Hitzler. *String Similarity Metrics for Ontology Alignment.* In Harith Alani, Lalana Kagal, Achille Fokoue, Paul T. Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha F. Noy, Chris Welty and Krzysztof Janowicz, editors, The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II, volume 8219 of *Lecture Notes in Computer Science*, pages 294–309. Springer, 2013. (Cited in page 82.)

[Cheatham & Hitzler 2014] Michelle Cheatham and Pascal Hitzler. *Conference v2.0: An Uncertain Version of the OAEI Conference Benchmark.* In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz and Carole A. Goble, editors, The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II, volume 8797 of *Lecture Notes in*

Computer Science, pages 33–48. Springer, 2014. (Cited in pages x, 70, 107 and 151.)

[Chondrogiannis *et al.* 2014] Efthymios Chondrogiannis, Vassiliki Andronikou, Efstathios Karanastasis and Theodora A. Varvarigou. *An Intelligent Ontology Alignment Tool Dealing with Complicated Mismatches.* In Adrian Paschke, Albert Burger, Paolo Romano, M. Scott Marshall and Andrea Splendiani, editors, Proceedings of the 7th International Workshop on Semantic Web Applications and Tools for Life Sciences, Berlin, Germany, December 9-11, 2014, volume 1320 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014. (Cited in pages 33, 34, 36, 56, 57, 58, 59 and 150.)

[Chortaras & Stamou 2018] Alexandros Chortaras and Giorgos Stamou. *Mapping Diverse Data to RDF in Practice.* In Denny Vrandečić, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee and Elena Simperl, editors, The Semantic Web – ISWC 2018, volume 11136, pages 441–457. Springer International Publishing, Cham, 2018. (Cited in page 24.)

[Correndo & Shadbolt 2011] Gianluca Correndo and Nigel Shadbolt. *Translating expressive ontology mappings into rewriting rules to implement query rewriting.* In Pavel Shvaiko, Jérôme Euzenat, Tom Heath, Christoph Quix, Ming Mao and Isabel F. Cruz, editors, Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011, volume 814 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011. (Cited in pages 68 and 94.)

[Das *et al.* 2012] Souripriya Das, Seema Sundara and Richard Cyganiak. *R2RML: RDB to RDF Mapping Language*, 2012. (Cited in page 23.)

[David *et al.* 2011] Jérôme David, Jérôme Euzenat, François Scharffe and Cássia Trojahn. *The Alignment API 4.0.* Semantic Web, vol. 2, no. 1, pages 3–10, 2011. (Cited in pages 19, 22 and 91.)

[David *et al.* 2018] Jérôme David, Jérôme Euzenat, Pierre Genevès and Nabil Layaïda. *Evaluation of Query Transformations without Data: Short paper.* In Companion of the The Web Conference 2018 on The Web Conference 2018, pages 1599–1602. International World Wide Web Conferences Steering Committee, 2018. (Cited in pages 66, 93 and 95.)

[de Boer *et al.* 2012] Victor de Boer, Jan Wielemaker, Judith van Gent, Michiel Hildebrand, Antoine Isaac, Jacco van Ossenbruggen and Guus Schreiber. *Supporting Linked Data Production for Cultural Heritage Institutes: The Amsterdam Museum Case Study.* In Elena Simperl, Philipp Cimiano, Axel Polleres, Óscar Corcho and Valentina Presutti, editors, The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC

2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings, volume 7295 of *Lecture Notes in Computer Science*, pages 733–747. Springer, 2012. (Cited in page 2.)

[De Bruijn *et al.* 2006] Jos De Bruijn, Marc Ehrig, Cristina Feier, Francisco Martín-Recuerda, François Scharffe and Moritz Weiten. *Ontology mediation, merging and aligning.* In John Davies, Rudi Studer and Paul Warren, editors, Semantic Web Technologies: Trends and Research in Ontology-Based Systems, pages 95–113. John Wiley and Sons, 2006. (Cited in page 28.)

[de Carvalho *et al.* 2013] Moisés Gomes de Carvalho, Alberto H. F. Laender, Marcos André Gonçalves and Altigran S. da Silva. *An evolutionary approach to complex schema matching.* Information Systems, vol. 38, no. 3, pages 302–316, May 2013. (Cited in pages 50, 52, 54, 56, 57, 58 and 59.)

[de Medeiros *et al.* 2015] Luciano Frontino de Medeiros, Freddy Priyatna and Óscar Corcho. *MIRROR: Automatic R2RML Mapping Generation from Relational Databases.* In Philipp Cimiano, Flavius Frasincar, Geert-Jan Houben and Daniel Schwabe, editors, Engineering the Web in the Big Data Era - 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26, 2015, Proceedings, volume 9114 of *Lecture Notes in Computer Science*, pages 326–343. Springer, 2015. (Cited in page 38.)

[De Meester *et al.* 2016] Ben De Meester, Anastasia Dimou, Ruben Verborgh and Erik Mannens. *An Ontology to Semantically Declare and Describe Functions.* In Harald Sack, Giuseppe Rizzo, Nadine Steinmetz, Dunja Mladenić, Sören Auer and Christoph Lange, editors, The Semantic Web, volume 9989, pages 46–49. Springer International Publishing, Cham, 2016. (Cited in page 23.)

[Dennis *et al.* 2017] Matt Dennis, Kees van Deemter, Daniele Dell'Aglio and Jeff Z. Pan. *Computing Authoring Tests from Competency Questions: Experimental Validation.* In Claudia d'Amato, Miriam Fernández, Valentina A. M. Tamma, Freddy Lécué, Philippe Cudré-Mauroux, Juan F. Sequeda, Christoph Lange and Jeff Heflin, editors, The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I, volume 10587 of *Lecture Notes in Computer Science*, pages 243–259. Springer, 2017. (Cited in pages iv and 14.)

[Dhamankar *et al.* 2004] Robin Dhamankar, Yoonkyong Lee, AnHai Doan, Alon Y. Halevy and Pedro M. Domingos. *iMAP: Discovering Complex Mappings between Database Schemas.* In Gerhard Weikum, Arnd Christian König and Stefan Deßloch, editors, Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18, 2004, pages 383–394. ACM, 2004. (Cited in pages 33, 34, 37, 38, 39, 42, 52, 53, 56, 57, 58 and 59.)

[Dimou *et al.* 2014] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens and Rik Van de Walle. *RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data.* In Christian Bizer, Tom Heath, Sören Auer and Tim Berners-Lee, editors, Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, April 8, 2014, volume 1184 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014. (Cited in page 23.)

[Doan & Halevy 2005] Anhai Doan and Alon Y Halevy. *Semantic integration research in the database community: A brief survey.* AI Magazine, vol. 26, no. 1, pages 83–94, 2005. (Cited in page 28.)

[Doan *et al.* 2003] AnHai Doan, Jayant Madhavan, Robin Dhamankar, Pedro Domingos and Alon Halevy. *Learning to match ontologies on the semantic web.* The VLDB Journal—The International Journal on Very Large Data Bases, vol. 12, no. 4, pages 303–319, 2003. (Cited in pages 38, 39, 40, 41, 56, 57, 58 and 59.)

[Doan 2005] Anhai Doan. *The Illinois Semantic Integration Archive*, 2005. (Cited in page 61.)

[Dou *et al.* 2010] Dejing Dou, Han Qin and Paea Lependu. *Ontograte: towards Automatic Integration for Relational Databases and the Semantic Web through an Ontology-Based Framework.* International Journal of Semantic Computing, vol. 04, no. 01, pages 123–151, March 2010. (Cited in pages 47, 48, 56, 57, 58 and 59.)

[Dou 2008] Dejing Dou. *The Formal Syntax and Semantics of Web-PDDL.* Technical Report, Technical report, University of Oregon, 2008. (Cited in page 20.)

[Dragisic *et al.* 2016] Zlatan Dragisic, Valentina Ivanova, Patrick Lambrix, Daniel Faria, Ernesto Jiménez-Ruiz and Catia Pesquita. *User Validation in Ontology Alignment.* In Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck and Yolanda Gil, editors, The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I, volume 9981 of *Lecture Notes in Computer Science*, pages 200–217, 2016. (Cited in page 70.)

[Duchateau *et al.* 2007] Fabien Duchateau, Zohra Bellahsene and Ela Hunt. *XBenchMatch: a benchmark for XML schema matching tools.* In Christoph Koch, Johannes Gehrke, Minos N. Garofalakis, Divesh Srivastava, Karl Aberer, Anand Deshpande, Daniela Florescu, Chee Yong Chan, Venkatesh Ganti, Carl-Christian Kanne, Wolfgang Klas and Erich J. Neuhold, editors, Proceedings of the 33rd International Conference on Very Large Data

Bases, University of Vienna, Austria, September 23-27, 2007, pages 1318–1321. ACM, 2007. (Cited in page 61.)

[Ehrig & Euzenat 2005] Marc Ehrig and Jérôme Euzenat. *Relaxed precision and recall for ontology matching.* In Integrating Ontologies '05, Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, Banff, Canada, October 2, 2005, volume 156 of *CEUR Workshop Proceedings*, pages 25–32. CEUR-WS.org, 2005. (Cited in pages xi, 91, 92, 93 and 98.)

[Ermilov *et al.* 2016] Ivan Ermilov, Jens Lehmann, Michael Martin and Sören Auer. *LODStats: The Data Web Census Dataset.* In Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck and Yolanda Gil, editors, The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II, volume 9982 of *Lecture Notes in Computer Science*, pages 38–46, 2016. (Cited in page 14.)

[Essaid *et al.* 2014] Amira Essaid, Arnaud Martin, Grégory Smits and Boutheina Ben Yaghlane. *Uncertainty in Ontology Matching: A Decision Rule-Based Approach.* In Anne Laurent, Olivier Strauss, Bernadette Bouchon-Meunier and Ronald R. Yager, editors, Information Processing and Management of Uncertainty in Knowledge-Based Systems - 15th International Conference, IPMU 2014, Montpellier, France, July 15-19, 2014, Proceedings, Part I, volume 442 of *Communications in Computer and Information Science*, pages 46–55. Springer, 2014. (Cited in page 151.)

[Euzenat & Shvaiko 2013] Jérôme Euzenat and Pavel Shvaiko. Ontology Matching, second edition. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. (Cited in pages vi, 1, 6, 7, 15, 16, 18, 24, 25, 28, 29, 32, 59, 60 and 149.)

[Euzenat *et al.* 2007] Jérôme Euzenat, François Scharffe and Antoine Zimmermann. *Expressive alignment language and implementation.* Project deliverable, Knowledge Web, 2007. (Cited in pages 19 and 21.)

[Euzenat 2003] Jérôme Euzenat. *Towards composing and benchmarking ontology alignments.* In Proceedings of the Semantic Integration Workshop Collocated with the Second International Semantic Web Conference (ISWC-03), Sanibel Island, Florida, USA, 2003, volume 82, pages 165–166. CEUR-WS.org, 2003. (Cited in page 17.)

[Euzenat 2007] Jérôme Euzenat. *Semantic Precision and Recall for Ontology Alignment Evaluation.* In Manuela M. Veloso, editor, IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007, pages 348–353, 2007. (Cited in page 66.)

[Faria *et al.* 2018] Daniel Faria, Catia Pesquita, Booma Sowkarthiga Balasubramani, Teemu Tervo, David Carriço, Rodrigo Garrilha, Francisco M. Couto

and Isabel F. Cruz. *Results of AML participation in OAEI 2018.* In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Michelle Cheatham and Oktie Hassanzadeh, editors, Proceedings of the 13th International Workshop on Ontology Matching co-located with the 17th International Semantic Web Conference, OM@ISWC 2018, Monterey, CA, USA, October 8, 2018, volume 2288 of *CEUR Workshop Proceedings*, pages 125–131. CEUR-WS.org, 2018. (Cited in pages xi, 33, 34, 35, 56, 57, 58, 59, 60 and 135.)

[Fletcher & Wyss 2009] George HL Fletcher and Catharine M Wyss. *Towards a general framework for effective solutions to the data mapping problem.* In Stefano Spaccapietra and Lois Delcambre, editors, Journal on Data Semantics XIV, volume 14, pages 37–73. Springer, 2009. (Cited in page 50.)

[Fung & Xu 2012] Kin Wah Fung and Junchuan Xu. *Synergism between the Mapping Projects from SNOMED CT to ICD-10 and ICD-10-CM.* In AMIA 2012, American Medical Informatics Association Annual Symposium, Chicago, Illinois, USA, November 3-7, 2012. American Medical Informatics Association, 2012. (Cited in page 2.)

[Ganter *et al.* 2005] Bernhard Ganter, Gerd Stumme and Rudolf Wille, editors. Formal concept analysis, foundations and applications, volume 3626 of *Lecture Notes in Computer Science*. Springer, 2005. (Cited in pages 86 and 149.)

[Giannangelo & Millar 2012] Kathy Giannangelo and Jane Millar. *Mapping SNOMED CT to ICD-10.* In John Mantas, Stig Kjær Andersen, Maria Christina Mazzoleni, Bernd Blobel, Silvana Quaglini and Anne Moen, editors, Quality of Life through Quality of Information - Proceedings of MIE2012, The XXIVth International Congress of the European Federation for Medical Informatics, Pisa, Italy, August 26-29, 2012, volume 180 of *Studies in Health Technology and Informatics*, pages 83–87. IOS Press, 2012. (Cited in page 2.)

[Gruber 1993] Thomas R Gruber. *A translation approach to portable ontology specifications.* Knowledge acquisition, vol. 5, no. 2, pages 199–220, 1993. (Cited in page 7.)

[Grüninger & Fox 1995] Michael Grüninger and Mark S Fox. *Methodology for the Design and Evaluation of Ontologies. International Joint Conference on Artificial Inteligence.* In Workshop on Basic Ontological Issues in Knowledge Sharing, volume 15, 1995. (Cited in pages iv, 14 and 72.)

[Grütze *et al.* 2012] Toni Grütze, Christoph Böhm and Felix Naumann. *Holistic and Scalable Ontology Alignment for Linked Open Data.* In Christian Bizer, Tom Heath, Tim Berners-Lee and Michael Hausenblas, editors, WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012, volume 937 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012. (Cited in page 18.)

[Guarino *et al.* 2009] Nicola Guarino, Daniel Oberle and Steffen Staab. *What Is an Ontology?* In Steffen Staab and Rudi Studer, editors, Handbook on Ontologies, International Handbooks on Information Systems, pages 1–17. Springer-Verlag Berlin Heidelberg, 2009. (Cited in page 7.)

[Guarino 1998] Nicola Guarino. *Formal Ontology and Information Systems.* In Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy, volume 46, pages 3–15. IOS Press, 1998. (Cited in page 8.)

[Halpin *et al.* 2015] Harry Halpin, Patrick J. Hayes and Henry S. Thompson. *When owl:sameAs isn't the Same Redux: Towards a Theory of Identity, Context, and Inference on the Semantic Web.* In Henning Christiansen, Isidora Stojanovic and George A. Papadopoulos, editors, Modeling and Using Context - 9th International and Interdisciplinary Conference, CONTEXT 2015, Lanarca, Cyprus, November 2-6, 2015. Proceedings, volume 9405 of *Lecture Notes in Computer Science*, pages 47–60. Springer, 2015. (Cited in pages 14 and 151.)

[Harrow *et al.* 2017] Ian Harrow, Ernesto Jiménez-Ruiz, Andrea Splendiani, Martin Romacker, Peter Woollard, Scott Markel, Yasmin Alam-Faruque, Martin Koch, James Malone and Arild Waaler. *Matching disease and phenotype ontologies in the ontology alignment evaluation initiative.* Journal of Biomedical Semantics, vol. 8, no. 1, pages 55:1–55:13, 2017. (Cited in page 62.)

[Hartung *et al.* 2013] Michael Hartung, Anika Groß and Erhard Rahm. *COnto-Diff: generation of complex evolution mappings for life science ontologies.* Journal of Biomedical Informatics, vol. 46, no. 1, pages 15–32, 2013. (Cited in page 50.)

[Hassanpour *et al.* 2010] Saeed Hassanpour, Martin J. O'Connor and Amar K. Das. *A Software Tool for Visualizing, Managing and Eliciting SWRL Rules.* In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Lora Aroyo, Grigoris Antoniou, Eero Hyvönen, Annette ten Teije, Heiner Stuckenschmidt, Liliana Cabral and Tania Tudorache, editors, The Semantic Web: Research and Applications, volume 6089, pages 381–385. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. (Cited in page 150.)

[He *et al.* 2004] Bin He, Kevin Chen-Chuan Chang and Jiawei Han. *Discovering complex matchings across web query interfaces: a correlation mining approach.* In Won Kim, Ron Kohavi, Johannes Gehrke and William DuMouchel, editors, Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004, pages 148–157. ACM, 2004. (Cited in pages 39, 45, 46, 56, 57, 58 and 59.)

[Hert *et al.* 2011] Matthias Hert, Gerald Reif and Harald C. Gall. *A comparison of RDB-to-RDF mapping languages.* In Proceedings of the 7th International Conference on Semantic Systems - I-Semantics '11, pages 25–32, Graz, Austria, 2011. ACM Press. (Cited in page 23.)

[Hollink *et al.* 2008] L. Hollink, M. Van Assem, S. Wang, A. Isaac and G. Schreiber. *Two Variations on Ontology Alignment Evaluation: Methodological Issues.* In 5th European Semantic Web Conference, pages 388–401, 2008. (Cited in pages 66, 91 and 114.)

[Horrocks *et al.* 2004] Ian Horrocks, Peter F Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof and Mike Dean. *SWRL: A semantic web rule language combining OWL and RuleML.* W3C Member submission 21, W3C, 2004. (Cited in page 20.)

[Horrocks *et al.* 2006] Ian Horrocks, Oliver Kutz and Ulrike Sattler. *The Even More Irresistible SROIQ.* In Patrick Doherty, John Mylopoulos and Christopher A. Welty, editors, Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom, June 2-5, 2006, pages 57–67. AAAI Press, 2006. (Cited in pages 67 and 95.)

[Horrocks 2002] Ian Horrocks. *DAML+OIL: A Description Logic for the Semantic Web.* IEEE Data Eng. Bull., vol. 25, no. 1, pages 4–9, 2002. (Cited in page 8.)

[Hu & Qu 2006] Wei Hu and Yuzhong Qu. *Block Matching for Ontologies.* In Isabel F. Cruz, Stefan Decker, Dean Allemang, Chris Preist, Daniel Schwabe, Peter Mika, Michael Uschold and Lora Aroyo, editors, The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings, volume 4273 of *Lecture Notes in Computer Science*, pages 300–313. Springer, 2006. (Cited in pages 51, 55, 56, 57, 58 and 59.)

[Hu *et al.* 2011] Wei Hu, Jianfeng Chen, Hang Zhang and Yuzhong Qu. *Learning complex mappings between ontologies.* In Jeff Z. Pan, Huajun Chen, Hong-Gee Kim, Juanzi Li, Zhe Wu, Ian Horrocks, Riichiro Mizoguchi and Zhaohui Wu, editors, The Semantic Web - Joint International Semantic Technology Conference, JIST 2011, Hangzhou, China, December 4-7, 2011. Proceedings, volume 7185 of *Lecture Notes in Computer Science*, pages 350–357. Springer, 2011. (Cited in pages viii, 51, 52, 56, 57, 58, 59, 86 and 145.)

[Jiang *et al.* 2016] Shangpu Jiang, Daniel Lowd, Sabin Kafle and Dejing Dou. *Ontology matching with knowledge rules.* In Transactions on Large-Scale Data-and Knowledge-Centered Systems XXVIII, volume 28, pages 75–95. Springer, 2016. (Cited in pages 33, 34, 38, 51, 53, 56, 57, 58 and 59.)

[Jiménez-Ruiz & Grau 2011] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. *LogMap: Logic-Based and Scalable Ontology Matching.* In Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy and Eva Blomqvist, editors, The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I, volume 7031 of *Lecture Notes in Computer Science*, pages 273–288. Springer, 2011. (Cited in page 38.)

[Jiménez-Ruiz *et al.* 2015] Ernesto Jiménez-Ruiz, Evgeny Kharlamov, Dmitriy Zheleznyakov, Ian Horrocks, Christoph Pinkel, Martin G Skjæveland, Evgenij Thorstensen and Jose Mora. *BootOX: Practical mapping of RDBs to OWL 2.* In Marcelo Arenas, Óscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul T. Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan and Steffen Staab, editors, The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II, volume 9367 of *Lecture Notes in Computer Science*, pages 113–132. Springer, Springer, 2015. (Cited in pages 33, 34, 38, 56, 57, 58 and 59.)

[Jouhet *et al.* 2017] Vianney Jouhet, Fleur Mougin, Bérénice Bréchat and Frantz Thiessard. *Building a model for disease classification integration in oncology, an approach based on the national cancer institute thesaurus.* Journal of Biomedical Semantics, vol. 8, no. 1, pages 6:1–6:12, 2017. (Cited in page 2.)

[Kaabi & Gargouri 2012] Fatma Kaabi and Faiez Gargouri. *A new approach to discover the complex mappings between ontologies.* International Journal of Web Science, vol. 1, no. 3, pages 242–256, 2012. (Cited in pages 38, 39, 41, 56, 57, 58 and 59.)

[Kakali *et al.* 2007] Constantia Kakali, Irene Lourdi, Thomais Stasinopoulou, Lina Bountouri, Christos Papatheodorou, Martin Doerr and Manolis Gergatsoulis. *Integrating Dublin Core Metadata for Cultural Heritage Collections Using Ontologies.* In Stuart A. Sutton, Abdus Sattar Chaudhry and Christopher S. G. Khoo, editors, Proceedings of the 2007 International Conference on Dublin Core and Metadata Applications, DC 2007, Singapore, August 27-31, 2007, pages 128–139. Dublin Core Metadata Initiative, 2007. (Cited in page 2.)

[Kalfoglou & Schorlemmer 2003] Yannis Kalfoglou and Marco Schorlemmer. *Ontology mapping: the state of the art.* The Knowledge Engineering Review, vol. 18, no. 1, pages 1–31, January 2003. (Cited in page 28.)

[Kay 2017] Michael Kay. *XSL Transformations (XSLT) Version 3.0.* W3c recommandation, W3C, 2017. (Cited in page 21.)

[Klein 2001] Michel Klein. *Combining and relating ontologies: an analysis of problems and solutions.* In Asunción Gómez-Pérez, Michael Gruninger, Heiner

Stuckenschmidt and Michael Uschold, editors, Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing Seattle, USA, August 4-5, 2001., volume 47 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2001. (Cited in pages 15 and 16.)

[Knoblock *et al.* 2012] Craig A. Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan and Parag Mallick. *Semi-automatically Mapping Structured Sources into the Semantic Web*. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Elena Simperl, Philipp Cimiano, Axel Polleres, Oscar Corcho and Valentina Presutti, editors, The Semantic Web: Research and Applications, volume 7295, pages 375–390. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. (Cited in pages 50, 51, 54, 56, 57, 58 and 59.)

[Levenshtein 1966] Vladimir I Levenshtein. *Binary codes capable of correcting deletions, insertions, and reversals*. Soviet physics doklady, vol. 10, no. 8, pages 707–710, 1966. (Cited in page 116.)

[Maedche *et al.* 2002] Alexander Maedche, Boris Motik, Nuno Silva and Raphael Volz. *MAFRA - A MApping FRAmework for Distributed Ontologies*. In Asunción Gómez-Pérez and V. Richard Benjamins, editors, Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings, volume 2473 of *Lecture Notes in Computer Science*, pages 235–250. Springer, 2002. (Cited in page 22.)

[Makris *et al.* 2012] Konstantinos Makris, Nikos Bikakis, Nektarios Gioldasis and Stavros Christodoulakis. *SPARQL-RW: transparent query access over mapped RDF data sources*. In Proceedings of the 15th International Conference on Extending Database Technology, pages 610–613. ACM, 2012. (Cited in pages 68 and 94.)

[Maßmann *et al.* 2011] Sabine Maßmann, Salvatore Raunich, David Aumüller, Patrick Arnold and Erhard Rahm. *Evolution of the COMA match system*. In Pavel Shvaiko, Jérôme Euzenat, Tom Heath, Christoph Quix, Ming Mao and Isabel F. Cruz, editors, Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011, volume 814 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011. (Cited in page 42.)

[McGuinness & Van Harmelen 2004] Deborah L McGuinness and Frank Van Harmelen. *OWL web ontology language overview*. W3c recommendation, W3C, 2004. (Cited in pages 8 and 19.)

[Megdiche *et al.* 2016] Imen Megdiche, Olivier Teste and Cássia Trojahn dos Santos. *An Extensible Linear Approach for Holistic Ontology Matching.* In Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck and Yolanda Gil, editors, The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I, volume 9981 of *Lecture Notes in Computer Science*, pages 393–410, 2016. (Cited in page 18.)

[Meilicke & Stuckenschmidt 2008] Christian Meilicke and Heiner Stuckenschmidt. *Incoherence as a basis for measuring the quality of ontology mappings.* In Proceedings of the 3rd International Conference on Ontology Matching-Volume 431, pages 1–12. CEUR-WS. org, 2008. (Cited in pages 25 and 65.)

[Meilicke *et al.* 2012] Christian Meilicke, Raul Garcia-Castro, Fred Freitas, Willem Robert van Hage, Elena Montiel-Ponsoda, Ryan Ribeiro de Azevedo, Heiner Stuckenschmidt, Ondrej Šváb Zamazal, Vojtech Svátek, Andrei Tamilin, Cássia Trojahn dos Santos and Shenghui Wang. *MultiFarm: A benchmark for multilingual ontology matching.* J. Web Semant., vol. 15, pages 62–68, 2012. (Cited in page 62.)

[Michel *et al.* 2015] Franck Michel, Loïc Djimenou, Catherine Faron-Zucker and Johan Montagnat. *Translation of Relational and Non-relational Databases into RDF with xR2RML.* In Valérie Monfort, Karl-Heinz Krempels, Tim A. Majchrzak and Ziga Turk, editors, WEBIST 2015 - Proceedings of the 11th International Conference on Web Information Systems and Technologies, Lisbon, Portugal, 20-22 May, 2015, pages 443–454. SciTePress, 2015. (Cited in page 24.)

[Michel *et al.* 2017] Franck Michel, Olivier Gargominy, Sandrine Tercerie and Catherine Faron-Zucker. *A Model to Represent Nomenclatural and Taxonomic Information as Linked Data.Application to the French Taxonomic Register, TAXREF.* In Alsayed Algergawy, Naouel Karam, Friederike Klan and Clément Jonquet, editors, Proceedings of the 2nd International Workshop on Semantics for Biodiversity (S4BioDiv 2017) co-located with 16th International Semantic Web Conference (ISWC 2017), volume 1933, Vienna, Austria, October 2017. CEUR-WS.org. (Cited in page 138.)

[Miller *et al.* 2000] Renée J. Miller, Laura M. Haas and Mauricio A. Hernández. *Schema Mapping as Query Discovery.* In Amr El Abbadi, Michael L. Brodie, Sharma Chakravarthy, Umeshwar Dayal, Nabil Kamel, Gunter Schlageter and Kyu-Young Whang, editors, VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt, pages 77–88. Morgan Kaufmann, 2000. (Cited in pages 47, 56, 57, 58 and 59.)

[Miller 1992] George A. Miller. *WORDNET: a Lexical Database for English.* In

Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, USA, February 23-26, 1992. Morgan Kaufmann, 1992. (Cited in page 150.)

[Musen 2015] Mark A Musen. *The Protégé project: a look back and a look forward.* AI matters, vol. 1, no. 4, pages 4–12, 2015. (Cited in page 150.)

[Noy & Musen 2003] Natalya F Noy and Mark A Musen. *The PROMPT suite: interactive tools for ontology merging and mapping.* International Journal of Human-Computer Studies, vol. 59, no. 6, pages 983–1024, 2003. (Cited in page 70.)

[Noy 2004] Natalya F Noy. *Semantic integration: a survey of ontology-based approaches.* ACM Sigmod Record, vol. 33, no. 4, pages 65–70, 2004. (Cited in page 28.)

[Nunes *et al.* 2011] Bernardo Pereira Nunes, Alexander Arturo Mera Caraballo, Marco A. Casanova, Karin K. Breitman and Luiz André P. Paes Leme. *Complex matching of RDF datatype properties.* In Pavel Shvaiko, Jérôme Euzenat, Tom Heath, Christoph Quix, Ming Mao and Isabel F. Cruz, editors, Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011, volume 814 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011. (Cited in pages 50, 52, 53, 54, 56, 57, 58 and 59.)

[Nurmikko-Fuller *et al.* 2015] Terhi Nurmikko-Fuller, Kevin R. Page, Pip Willcox, Jacob Jett, Chris Maden, Timothy W. Cole, Colleen Fallaw, Megan Senseney and J. Stephen Downie. *Building Complex Research Collections in Digital Libraries: A Survey of Ontology Implications.* In Paul Logasa Bogen II, Suzie Allard, Holly Mercer, Micah Beck, Sally Jo Cunningham, Dion Hoe-Lian Goh and Geneva Henry, editors, Proceedings of the 15th ACM/IEEE-CE Joint Conference on Digital Libraries, Knoxville, TN, USA, June 21-25, 2015, pages 169–172. ACM, 2015. (Cited in page 2.)

[Nuzzolese *et al.* 2016] Andrea Giovanni Nuzzolese, Anna Lisa Gentile, Valentina Presutti and Aldo Gangemi. *Conference Linked Data: The ScholarlyData Project.* In Paul T. Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck and Yolanda Gil, editors, The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II, volume 9982 of *Lecture Notes in Computer Science*, pages 150–158, 2016. (Cited in page 108.)

[Oliveira & Pesquita 2015] Daniela Oliveira and Catia Pesquita. *Compound matching of biomedical ontologies.* In Francisco M. Couto and Janna Hastings, editors, Proceedings of the International Conference on Biomedical Ontology, ICBO 2015, Lisbon, Portugal, July 27-30, 2015, volume 1515 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015. (Cited in page 69.)

[Oliveira & Pesquita 2018] Daniela Oliveira and Catia Pesquita. *Improving the interoperability of biomedical ontologies with compound alignments.* Journal of Biomedical Semantics, vol. 9, no. 1, December 2018. (Cited in pages 18, 33, 35, 56, 57, 58, 59 and 64.)

[Otero-Cerdeira *et al.* 2015] Lorena Otero-Cerdeira, Francisco J. Rodríguez-Martínez and Alma Gómez-Rodríguez. *Ontology matching: A literature review.* Expert Systems with Applications, vol. 42, no. 2, pages 949–971, February 2015. (Cited in page 28.)

[Parundekar *et al.* 2010] Rahul Parundekar, Craig A. Knoblock and José Luis Ambite. *Linking and Building Ontologies of Linked Data.* In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks and Birte Glimm, editors, The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part I, volume 6496 of *Lecture Notes in Computer Science*, pages 598–614. Springer, 2010. (Cited in pages viii, 38, 39, 40, 56, 57, 58, 59, 86, 127 and 145.)

[Parundekar *et al.* 2012] Rahul Parundekar, Craig A. Knoblock and José Luis Ambite. *Discovering Concept Coverings in Ontologies of Linked Data Sources.* In Philippe Cudré-Mauroux, Jeff Heflin, Evren Sirin, Tania Tudorache, Jérôme Euzenat, Manfred Hauswirth, Josiane Xavier Parreira, Jim Hendler, Guus Schreiber, Abraham Bernstein and Eva Blomqvist, editors, The Semantic Web - ISWC 2012 - 11th International Semantic Web Conference, Boston, MA, USA, November 11-15, 2012, Proceedings, Part I, volume 7649 of *Lecture Notes in Computer Science*, pages 427–443. Springer, 2012. (Cited in pages viii, 38, 39, 40, 56, 57, 58, 59, 64, 65, 67, 68, 86, 114, 127 and 145.)

[Pasquale 2018] Frank Pasquale. *Mettre fin au trafic des données personnelles.* Monde diplomatique, no. 770, pages 16–17, May 2018. (Cited in page 153.)

[Pazienza *et al.* 2004] Maria Teresa Pazienza, Armando Stellato, Michele Vindigni and Fabio Massimo Zanzotto. *XeOML: An XML-based extensible ontology mapping language.* In Paolo Bouquet and Luciano Serafini, editors, Working notes of the ISWC-04 Workshop on Meaning Coordination and Negotiation(MCN-04) held in conjunction with3rd International Semantic Web Conference (ISWC-2004), Hiroshima, Japan, pages 83–94, 2004. (Cited in page 22.)

[Pesquita *et al.* 2014] Catia Pesquita, Michelle Cheatham, Daniel Faria, Joana Barros, Emanuel Santos and Francisco M. Couto. *Building reference alignments for compound matching of multiple ontologies using OBO cross-products.* In Pavel Shvaiko, Jérôme Euzenat, Ming Mao, Ernesto Jiménez-Ruiz, Juanzi Li and Axel Ngonga, editors, Proceedings of the 9th International Workshop on Ontology Matching collocated with the 13th International Semantic

Web Conference (ISWC 2014), Riva del Garda, Trentino, Italy, October 20, 2014, volume 1317 of *CEUR Workshop Proceedings*, pages 172–173. CEUR-WS.org, 2014. (Cited in pages 64 and 69.)

[Pinkel *et al.* 2017] Christoph Pinkel, Carsten Binnig, Ernesto Jiménez-Ruiz, Evgeny Kharlamov, Wolfgang May, Andriy Nikolov, Ana Sasa Bastinos, Martin G. Skjæveland, Alessandro Solimando, Mohsen Taheriyan, Christian Heupel and Ian Horrocks. *RODI: Benchmarking relational-to-ontology mapping generation quality.* Semantic Web, vol. 9, no. 1, pages 25–52, November 2017. (Cited in page 61.)

[Pradel *et al.* 2014] Camille Pradel, Ollivier Haemmerlé and Nathalie Hernandez. *Swip: A Natural Language to SPARQL Interface Implemented with SPARQL.* In Nathalie Hernandez, Robert Jäschke and Madalina Croitoru, editors, Graph-Based Representation and Reasoning - 21st International Conference on Conceptual Structures, ICCS 2014, Iaşi, Romania, July 27-30, 2014, Proceedings, volume 8577 of *Lecture Notes in Computer Science*, pages 260–274. Springer, 2014. (Cited in page 151.)

[Qin *et al.* 2007] Han Qin, Dejing Dou and Paea LePendu. *Discovering Executable Semantic Mappings Between Ontologies.* In Robert Meersman and Zahir Tari, editors, On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS, OTM Confederated International Conferences CoopIS, DOA, ODBASE, GADA, and IS 2007, Vilamoura, Portugal, November 25-30, 2007, Proceedings, Part I, volume 4803 of *Lecture Notes in Computer Science*, pages 832–849. Springer, 2007. (Cited in pages 47, 48, 56, 57, 58 and 59.)

[Rahm & Bernstein 2001] Erhard Rahm and Philip A. Bernstein. *A survey of approaches to automatic schema matching.* The VLDB Journal, vol. 10, no. 4, pages 334–350, December 2001. (Cited in pages 17 and 28.)

[Ren *et al.* 2014] Yuan Ren, Artemis Parvizi, Chris Mellish, Jeff Z. Pan, Kees van Deemter and Robert Stevens. *Towards Competency Question-Driven Ontology Authoring.* In The Semantic Web: Trends and Challenges, volume 8465, pages 752–767. Springer, 2014. (Cited in pages iv, 14 and 72.)

[Rimbert 2016] Pierre Rimbert. *Données personnelles, une affaire politique.* Monde diplomatique, no. 750, page 3, September 2016. (Cited in page 153.)

[Ritze *et al.* 2009] Dominique Ritze, Christian Meilicke, Ondrej Šváb Zamazal and Heiner Stuckenschmidt. *A Pattern-based Ontology Matching Approach for Detecting Complex Correspondences.* In Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Natalya Fridman Noy and Arnon Rosenthal, editors, Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009,

volume 551 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009. (Cited in pages 33, 34, 35, 53, 56, 57, 58, 59 and 67.)

[Ritze *et al.* 2010] Dominique Ritze, Johanna Völker, Christian Meilicke and Ondrej Šváb Zamazal. *Linguistic analysis for complex ontology matching*. In Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Ming Mao and Isabel F. Cruz, editors, Proceedings of the 5th International Workshop on Ontology Matching (OM-2010), Shanghai, China, November 7, 2010, volume 689 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2010. (Cited in pages xi, 33, 34, 53, 56, 57, 58, 59, 67 and 135.)

[Rosenberg *et al.* 2018] Matthew Rosenberg, Nicholas Confessore and Carole Cadwalladr. *How Trump Consultants Exploited the Facebook Data of Millions*. The New York Times, March 2018. (Cited in page 153.)

[Rouces *et al.* 2016] Jacobo Rouces, Gerard de Melo and Katja Hose. *Complex Schema Mapping and Linking Data: Beyond Binary Predicates*. In Sören Auer, Tim Berners-Lee, Christian Bizer and Tom Heath, editors, Proceedings of the Workshop on Linked Data on the Web, LDOW 2016, co-located with 25th International World Wide Web Conference (WWW 2016), volume 1593 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016. (Cited in pages 33, 34, 35, 56, 57, 58 and 59.)

[Rouces *et al.* 2018] Jacobo Rouces, Gerard de Melo and Katja Hose. *Addressing structural and linguistic heterogeneity in the Web1*. AI Communications, vol. 31, no. 1, pages 3–18, February 2018. (Cited in page 36.)

[Roussey *et al.* 2013] Catherine Roussey, Jean-Pierre Chanet, Vincent Cellier and Fabien Amarger. *Agronomic taxon*. In Vassilis Christophides and Dan Vodislav, editors, Proceedings of the 2nd International Workshop on Open Data, WOD 2013, Paris, France, June 3, 2013, pages 5:1–5:4. ACM, 2013. (Cited in pages 138 and 140.)

[Rowley 2007] Jennifer Rowley. *The wisdom hierarchy: representations of the DIKW hierarchy*. Journal of Information Science, vol. 33, no. 2, pages 163–180, 2007. (Cited in page 6.)

[Saleem *et al.* 2008] Khalid Saleem, Zohra Bellahsene and Ela Hunt. *Porsche: Performance oriented schema mediation*. Information Systems, vol. 33, no. 7, pages 637–657, 2008. (Cited in pages 39, 45, 56, 57, 58 and 59.)

[Saveta *et al.* 2015] Tzanina Saveta, Evangelia Daskalaki, Giorgos Flouris, Irini Fundulaki, Melanie Herschel and Axel-Cyrille Ngonga Ngomo. *Pushing the Limits of Instance Matching Systems: A Semantics-Aware Benchmark for Linked Data*. In Aldo Gangemi, Stefano Leonardi and Alessandro Panconesi, editors, Proceedings of the 24th International Conference on World

Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume, pages 105–106. ACM, 2015. (Cited in page 22.)

[Scharffe 2009] François Scharffe. *Correspondence Patterns Representation.* PhD thesis, Faculty of Mathematics, Computer Science and University of Innsbruck, 2009. (Cited in pages 21, 22, 30 and 34.)

[Schreiber *et al.* 1994] Guus Schreiber, Bob J. Wielinga, Hans Akkermans, Walter Van de Velde and Anjo Anjewierden. *CML: The CommonKADS Conceptual Modelling Language.* In Luc Steels, Guus Schreiber and Walter Van de Velde, editors, A Future for Knowledge Acquisition, 8th European Knowledge Acquisition Workshop, EKAW'94, Hoegaarden, Belgium, September 26-29, 1994, Proceedings, volume 867 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 1994. (Cited in page 8.)

[Semy *et al.* 2004] Salim K. Semy, Mary K. Pulvermacher and Leo J. Obrst. *Toward the use of an upper ontology for U.S. government and U.S. military domains: An evaluation.* Technical Report, The MITRE Corporation, 2004. (Cited in page 8.)

[Severo *et al.* 2017] Bernardo Severo, Cássia Trojahn and Renata Vieira. *VOAR 3.0 : a Configurable Environment for Manipulating Multiple Ontology Alignments.* In Nadeschda Nikitina, Dezhao Song, Achille Fokoue and Peter Haase, editors, Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017., volume 1963 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017. (Cited in page 150.)

[Shearer *et al.* 2008] Rob Shearer, Boris Motik and Ian Horrocks. *HermiT: A Highly-Efficient OWL Reasoner.* In Catherine Dolbear, Alan Ruttenberg and Ulrike Sattler, editors, Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26-27, 2008, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008. (Cited in page 108.)

[Shvaiko & Euzenat 2005] Pavel Shvaiko and Jérôme Euzenat. *A Survey of Schema-Based Matching Approaches.* In Stefano Spaccapietra, editor, Journal on Data Semantics IV, number 3730 in Lecture Notes in Computer Science, pages 146–171. Springer Berlin Heidelberg, 2005. DOI: 10.1007/116034125. (Cited in pages 3 and 28.)

[Shvaiko & Euzenat 2008] Pavel Shvaiko and Jérôme Euzenat. *Ten Challenges for Ontology Matching.* In Robert Meersman and Zahir Tari, editors, On the Move to Meaningful Internet Systems: OTM 2008, OTM 2008 Confederated

International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008, Monterrey, Mexico, November 9-14, 2008, Proceedings, Part II, volume 5332 of *Lecture Notes in Computer Science*, pages 1164–1182. Springer, 2008. (Cited in page 151.)

[Silva & Rocha 2003] Nuno Silva and João Rocha. *Semantic Web Complex Ontology Mapping.* In 2003 IEEE / WIC International Conference on Web Intelligence, (WI 2003), 13-17 October 2003, Halifax, Canada, pages 82–88. IEEE Computer Society, 2003. (Cited in pages 22 and 150.)

[Solimando *et al.* 2014a] Alessandro Solimando, Ernesto Jiménez-Ruiz and Giovanna Guerrini. *Detecting and Correcting Conservativity Principle Violations in Ontology-to-Ontology Mappings.* In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz and Carole A. Goble, editors, The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II, volume 8797 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2014. (Cited in pages 25 and 65.)

[Solimando *et al.* 2014b] Alessandro Solimando, Ernesto Jiménez-Ruiz and Christoph Pinkel. *Evaluating ontology alignment systems in query answering tasks.* In Matthew Horridge, Marco Rospocher and Jacco van Ossenbruggen, editors, Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014, volume 1272 of *CEUR Workshop Proceedings*, pages 301–304. CEUR-WS.org, 2014. (Cited in pages 25, 62, 65, 68, 107 and 114.)

[Solimando *et al.* 2017] Alessandro Solimando, Ernesto Jiménez-Ruiz and Giovanna Guerrini. *Minimizing conservativity violations in ontology alignments: algorithms and evaluation.* Knowledge Information Systems, vol. 51, no. 3, pages 775–819, 2017. (Cited in page 65.)

[Stapleton *et al.* 2014] Gem Stapleton, John Howse, Adrienne Bonnington and Jim Burton. *A Vision for Diagrammatic Ontology Engineering.* In Valentina Ivanova, Tomi Kauppinen, Steffen Lohmann, Suvodeep Mazumdar, Catia Pesquita and Kai Xu, editors, Proceedings of the International Workshop on Visualizations and User Interfaces for Knowledge Engineering and Linked Data Analytics co-located with 19th International Conference on Knowledge Engineering and Knowledge Management, VISUAL@EKAW 2014, Linköping, Sweden, November 24, 2014, volume 1299 of *CEUR Workshop Proceedings*, pages 1–13. CEUR-WS.org, 2014. (Cited in page 157.)

[Stuckenschmidt *et al.* 2008] Heiner Stuckenschmidt, Livia Predoiu and Christian Meilicke. *Learning Complex Ontology Alignments A Challenge for ILP*

Research. In Filip Železný and Nada Lavrač, editors, Proceedings of the 18th International Conference on Inductive Logic Programming, ILP 2008 Prague, Czech Republic, September 10-12, 2008, Lecture Notes in Computer Science, pages 105–111. Springer Berlin Heidelberg, 2008. (Cited in page 52.)

[Studer *et al.* 1998] Rudi Studer, V Richard Benjamins and Dieter Fensel. *Knowledge engineering: principles and methods*. Data & knowledge engineering, vol. 25, no. 1-2, pages 161–197, 1998. (Cited in page 7.)

[Su *et al.* 2006] Weifeng Su, Jiying Wang and Frederick H. Lochovsky. *Holistic Schema Matching for Web Query Interfaces*. In Yannis E. Ioannidis, Marc H. Scholl, Joachim W. Schmidt, Florian Matthes, Michael Hatzopoulos, Klemens Böhm, Alfons Kemper, Torsten Grust and Christian Böhm, editors, Advances in Database Technology - EDBT 2006, 10th International Conference on Extending Database Technology, Munich, Germany, March 26-31, 2006, Proceedings, volume 3896 of *Lecture Notes in Computer Science*, pages 77–94. Springer, 2006. (Cited in pages 39, 46, 56, 57, 58 and 59.)

[Suárez-Figueroa *et al.* 2012] Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez and Mariano Fernández-López. *The NeOn Methodology for Ontology Engineering*. In Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, Enrico Motta and Aldo Gangemi, editors, Ontology Engineering in a Networked World., pages 9–34. Springer, 2012. (Cited in pages iv, 5, 14, 71 and 107.)

[Symeonidou *et al.* 2017] Danai Symeonidou, Luis Galárraga, Nathalie Pernelle, Fatiha Saïs and Fabian M. Suchanek. *VICKEY: Mining Conditional Keys on Knowledge Bases*. In Claudia d'Amato, Miriam Fernández, Valentina A. M. Tamma, Freddy Lécué, Philippe Cudré-Mauroux, Juan F. Sequeda, Christoph Lange and Jeff Heflin, editors, The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I, volume 10587 of *Lecture Notes in Computer Science*, pages 661–677. Springer, 2017. (Cited in page 151.)

[Szekely *et al.* 2013] Pedro Szekely, Craig A. Knoblock, Fengyu Yang, Xuming Zhu, Eleanor E. Fink, Rachel Allen and Georgina Goodlander. *Connecting the Smithsonian American Art Museum to the Linked Data Cloud*. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Philipp Cimiano, Oscar Corcho, Valentina Presutti, Laura Hollink and Sebastian Rudolph, editors, The Semantic Web: Semantics and Big Data, volume 7882, pages 593–607. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. (Cited in page 2.)

[Thiéblin *et al.* 2016] Elodie Thiéblin, Fabien Amarger, Ollivier Haemmerlé, Nathalie Hernandez and Cássia Trojahn dos Santos. *Rewriting SELECT SPARQL queries from 1:n complex correspondences.* In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Michelle Cheatham, Oktie Hassanzadeh and Ryutaro Ichise, editors, Proceedings of the 11th International Workshop on Ontology Matching co-located with the 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 18, 2016, volume 1766 of *CEUR Workshop Proceedings*, pages 49–60. CEUR-WS.org, 2016. (Cited in pages x, 68, 94 and 100.)

[Thiéblin *et al.* 2017] Elodie Thiéblin, Fabien Amarger, Nathalie Hernandez, Catherine Roussey and Cássia Trojahn dos Santos. *Cross-Querying LOD Datasets Using Complex Alignments: An Application to Agronomic Taxa.* In Emmanouel Garoufallou, Sirje Virkus, Rania Siatri and Damiana Koutsomiha, editors, Metadata and Semantic Research - 11th International Conference, MTSR 2017 Tallinn, Estonia, November 28 - December 1, 2017, Proceedings, volume 755 of *Communications in Computer and Information Science*, pages 25–37. Springer, 2017. (Cited in pages 2 and 63.)

[Thiéblin *et al.* 2018a] Elodie Thiéblin, Michelle Cheatham, Cassia Trojahn, Ondrej Šváb Zamazal and Lu Zhou. *The First Version of the OAEI Complex Alignment Benchmark.* In Marieke van Erp, Medha Atre, Vanessa López, Kavitha Srinivas and Carolina Fortuna, editors, Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8-12, 2018, volume 2180 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018. (Cited in pages vi, 25, 61, 63, 67, 68, 90, 102, 107 and 114.)

[Thiéblin *et al.* 2018b] Elodie Thiéblin, Ollivier Haemmerlé, Nathalie Hernandez and Cássia Trojahn. *Task-Oriented Complex Ontology Alignment: Two Alignment Evaluation Sets.* In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai and Mehwish Alam, editors, The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings, volume 10843 of *Lecture Notes in Computer Science*, pages 655–670. Springer, 2018. (Cited in pages x, xi, 61, 63, 94, 107, 114 and 135.)

[Thiéblin *et al.* 2018c] Elodie Thiéblin, Ollivier Haemmerlé and Cassia Trojahn. *Complex matching based on competency questions for alignment: a first sketch.* In Proceedings of the 13th International Workshop on Ontology Matching co-located with the 17th International Semantic Web Conference, OM@ISWC 2018, Monterey, CA, USA, October 8, 2018, volume 2288 of *CEUR Workshop Proceedings*, pages 66–70. CEUR-WS.org, 2018. (Cited in pages 60 and 72.)

[Thiéblin *et al.* 2018d] Elodie Thiéblin, Nathalie Hernandez, Catherine Roussey and Cássia Trojahn. *Cross-querying LOD data sets using complex alignments: an experiment using AgronomicTaxon, Agrovoc, DBpedia and TAXREF-LD.* IJMSO, vol. 13, no. 2, pages 104–119, 2018. (Cited in pages 138 and 140.)

[Thiéblin *et al.* 2019] Elodie Thiéblin, Ollivier Haemmerlé, Nathalie Hernandez and Cassia Trojahn. *Survey on complex ontology matching.* Semantic Web Journal, 2019. (to appear). (Cited in page 27.)

[Unger *et al.* 2014] Christina Unger, Corina Forascu, Vanessa López, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano and Sebastian Walter. *Question Answering over Linked Data (QALD-4).* In Linda Cappellato, Nicola Ferro, Martin Halvey and Wessel Kraaij, editors, Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014., volume 1180 of *CEUR Workshop Proceedings*, pages 1172–1180. CEUR-WS.org, 2014. (Cited in pages 86 and 151.)

[Uschold & Gruninger 2004] Michael Uschold and Michael Gruninger. *Ontologies and semantics for seamless connectivity.* ACM SIGMod Record, vol. 33, no. 4, pages 58–64, 2004. (Cited in page 7.)

[Visser *et al.* 1997] Pepijn RS Visser, Dean M Jones, Trevor JM Bench-Capon and MJR Shave. *An analysis of ontology mismatches: heterogeneity versus interoperability.* In AAAI 1997 Spring Symposium on Ontological Engineering, Stanford CA., USA, pages 164–72, 1997. (Cited in pages 15 and 16.)

[Walshe *et al.* 2016] Brian Walshe, Rob Brennan and Declan O'Sullivan. *BayesReCCE: A Bayesian Model for Detecting Restriction Class Correspondences in Linked Open Data Knowledge Bases.* Int. J. Semant. Web Inf. Syst., vol. 12, no. 2, pages 25–52, April 2016. (Cited in pages viii, 33, 34, 36, 56, 57, 58, 59, 64, 65, 67, 68, 86, 114, 127 and 145.)

[Warren & Tompa 2006] Robert H. Warren and Frank Wm. Tompa. *Multi-column Substring Matching for Database Schema Translation.* In Umeshwar Dayal, Kyu-Young Whang, David B. Lomet, Gustavo Alonso, Guy M. Lohman, Martin L. Kersten, Sang Kyun Cha and Young-Kuk Kim, editors, Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006, pages 331–342. ACM, 2006. (Cited in pages 38, 39, 43, 56, 57, 58 and 59.)

[Wu & Knoblock 2015] Bo Wu and Craig A. Knoblock. *An Iterative Approach to Synthesize Data Transformation Programs.* In Qiang Yang and Michael J. Wooldridge, editors, Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, pages 1726–1732. AAAI Press, 2015. (Cited in pages 51, 52, 54, 56, 57, 58, 59, 69 and 86.)

[Wu *et al.* 2004] Wensheng Wu, Clement T. Yu, AnHai Doan and Weiyi Meng. *An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web.* In Gerhard Weikum, Arnd Christian König and Stefan Deßloch, editors, Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18, 2004, pages 95–106. ACM, 2004. (Cited in pages 39, 46, 56, 57, 58 and 59.)

[Xiao *et al.* 2018] Guohui Xiao, Diego Calvanese, Roman Kontchakov, Domenico Lembo, Antonella Poggi, Riccardo Rosati and Michael Zakharyaschev. *Ontology-Based Data Access: A Survey.* In Jérôme Lang, editor, Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden, pages 5511–5519. ijcai.org, 2018. (Cited in page 23.)

[Xu & Embley 2003] Li Xu and David W Embley. *Using domain ontologies to discover direct and indirect matches for schema elements.* In AnHai Doan, Alon Halevy and Natasha Noy, editors, Semantic Integration Workshop (SI-2003) collocated with the Second International Semantic Web Conference, October 20, 2003, Sanibel Island, Florida, USA, 2003. (Cited in pages 39, 43, 56, 57, 58 and 59.)

[Xu & Embley 2006] Li Xu and David W. Embley. *A composite approach to automating direct and indirect schema mappings.* Information Systems, vol. 31, no. 8, pages 697–732, December 2006. (Cited in pages 39, 43, 56, 57, 58 and 59.)

[Yan *et al.* 2001] Ling-Ling Yan, Renée J. Miller, Laura M. Haas and Ronald Fagin. *Data-Driven Understanding and Refinement of Schema Mappings.* In Sharad Mehrotra and Timos K. Sellis, editors, Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Santa Barbara, CA, USA, May 21-24, 2001, pages 485–496. ACM, 2001. (Cited in pages 47, 56, 57, 58 and 59.)

[Zhdanova & Shvaiko 2006] Anna V. Zhdanova and Pavel Shvaiko. *Community-Driven Ontology Matching.* In York Sure and John Domingue, editors, The Semantic Web: Research and Applications, 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11-14, 2006, Proceedings, volume 4011 of *Lecture Notes in Computer Science*, pages 34–49. Springer, 2006. (Cited in pages xii and 150.)

[Zheng *et al.* 2016] Weiguo Zheng, Lei Zou, Wei Peng, Xifeng Yan, Shaoxu Song and Dongyan Zhao. *Semantic SPARQL Similarity Search over RDF Knowledge Graphs.* Proceedings of the VLDB Endowment, vol. 9, no. 11, pages 840–851, July 2016. (Cited in page 81.)

[Zhou *et al.* 2018] Lu Zhou, Michelle Cheatham, Adila Krisnadhi and Pascal Hitzler. *A Complex Alignment Benchmark: Geolink Dataset.* In Denny Vran-

decic, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee and Elena Simperl, editors, The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II, volume 11137 of *Lecture Notes in Computer Science*, pages 273–288. Springer, 2018. (Cited in pages x, 17, 61, 63 and 150.)

[Zins 2007] Chaim Zins. *Conceptual approaches for defining data, information, and knowledge*. JASIST, vol. 58, no. 4, pages 479–493, 2007. (Cited in page 6.)

[Zuboff 2019] Shoshana Zuboff. The age of surveillance capitalism: the fight for the future at the new frontier of power. Profile Books, 2019. (Cited in page 153.)

[Šváb Zamazal & Svátek 2009] Ondrej Šváb Zamazal and Vojtech Svátek. *Towards ontology matching via pattern-based detection of semantic structures in OWL ontologies*. In Proceedings of the Znalosti Czecho-Slovak Knowledge Technology conference, 2009. (Cited in pages 39, 44, 56, 57, 58 and 59.)

[Šváb Zamazal & Svátek 2017] Ondřej Šváb Zamazal and Vojtěch Svátek. *The Ten-Year OntoFarm and its Fertilization within the Onto-Sphere*. Web Semantics: Science, Services and Agents on the World Wide Web, vol. 43, pages 46–53, March 2017. (Cited in pages x, 62, 107, 113 and 135.)

[Šváb Zamazal *et al.* 2005] Ondrej Šváb Zamazal, Vojtech Svátek, Petr Berka, Dušan Rak and Petr Tomášek. *Ontofarm: Towards an experimental collection of parallel ontologies*. Poster Track of ISWC, vol. 2005, 2005. (Cited in pages x, xi, 62 and 107.)

[Šváb Zamazal 2010] Ondrej Šváb Zamazal. *Pattern-based ontology matching and ontology alignment evaluation*. PhD thesis, University of Economics, Prague, 2010. (Cited in page 34.)

Abstract: The Linked Open Data (LOD) cloud is composed of data repositories. The data in the repositories are described by vocabularies also called ontologies. Each ontology has its own terminology and model. This leads to heterogeneity between them. To make the ontologies and the data they describe interoperable, ontology alignments establish correspondences, or links between their entities. There are many ontology matching systems which generate simple alignments, i.e., they link an entity to another. However, to overcome the ontology heterogeneity, more expressive correspondences are sometimes needed. Finding this kind of correspondence is a fastidious task that can be automated. In this thesis, an automatic complex matching approach based on a user's knowledge needs and common instances is proposed. The complex alignment field is still growing and little work address the evaluation of such alignments. To palliate this lack, we propose an automatic complex alignment evaluation system. This system is based on instances. A famous alignment evaluation dataset has been extended for this evaluation.
Keywords: Complex ontology matching, Complex alignment evaluation, Semantic web, Knowledge engineering

Résumé : Le web de données liées (LOD) est composé de nombreux entrepôts de données. Ces données sont décrites par différents vocabulaires (ou ontologies). Chaque ontologie a une terminologie et une modélisation propre ce qui les rend hétérogènes. Pour lier et rendre les données du web de données liées interopérables, les alignements d'ontologies établissent des correspondances entre les entités desdites ontologies. Il existe de nombreux systèmes d'alignement qui génèrent des correspondances simples, *i.e.*, ils lient une entité à une autre entité. Toutefois, pour surmonter l'hétérogénéité des ontologies, des correspondances plus expressives sont parfois nécessaires. Trouver ce genre de correspondances est un travail fastidieux qu'il convient d'automatiser. Dans le cadre de cette thèse, une approche d'alignement complexe basée sur des besoins utilisateurs et des instances communes est proposée. Le domaine des alignements complexes est relativement récent et peu de travaux adressent la problématique de leur évaluation. Pour pallier ce manque, un système d'évaluation automatique basé sur de la comparaison d'instances est proposé. Ce système est complété par un jeu de données artificiel sur le domaine des conférences.
Mots clés : Alignement complexe d'ontologies, Evaluation d'alignements complexes, Web sémantique, Ingénierie des connaissances