Open Archive TOULOUSE Archive Ouverte (OATAO)
OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in : http://oatao.univ-toulouse.fr/
Eprints ID : 16989

To link to this article : DOI : 10.3166/isi.20.3.37-61
URL : http://dx.doi.org/10.3166/isi.20.3.37-61

To cite this version : Amarger, Fabien and Chanet, Jean-Pierre and Haemmerlé, Ollivier and Hernandez, Nathalie and Roussey, Catherine *Construction d'une ontologie par transformation de systèmes d'organisation des connaissances et évaluation de la confiance*. (2015) Ingénierie des Systèmes d'Information, vol. 20 (n° 3). pp. 37-61. ISSN 1633-1311

Any correspondence concerning this service should be sent to the repository administrator: staff-oatao@listes-diff.inp-toulouse.fr

Construction d'une ontologie par transformation de systèmes d'organisation des connaissances et évaluation de la confiance

Fabien Amarger[1, 2], **Jean-Pierre Chanet**[2], **Ollivier Haemmerlé**[1], **Nathalie Hernandez**[1], **Catherine Roussey**[2]

1. IRIT, UMR 5505, UT2J, 5 allées Antonio Machado, 31058 Toulouse, France
firstname.lastname@univ-tlse2.fr

2. UR TSCF, Irstea, 9 av. Blaise Pascal CS 20085, 63172 Aubière, France
firstname.lastname@irstea.fr

RÉSUMÉ. *Cet article présente une méthode originale de création d'une base de connaissances (ontologie) à partir de transformations de plusieurs sources de connaissances. Ces sources sont issues des systèmes d'organisation des connaissances (thésaurus, taxonomies, etc.). Les sources sont pondérées par un score de confiance. La méthode s'appuie sur un module défini à l'aide de patrons de conception ontologiques. Nous avons mis en œuvre cette méthode dans un contexte agronomique. Nous proposons dans cet article plusieurs fonctions de pondération que nous avons évaluées à partir de trois sources : Agrovoc, TaxRef et NCBI.*

ABSTRACT. *This paper presents an original method of knowledge base (ontology) creation from transformations of several sources of knowledge. These sources are derived from the knowledge organization systems (thesauri, taxonomies, etc.). The sources are weighted by a confidence score. The method is based on a module aggregating ontological design patterns. We have implemented this method in an agronomical context. We propose in this paper several weighted functions that we have evaluated from three sources: Agrovoc, TaxRef and NCBI.*

MOTS-CLÉS : *développement d'ontologies, patrons de conception d'ontologies, sources non ontologiques, SKOS, système d'organisation des connaissances, confiance, agriculture.*

KEYWORDS: *ontology development, ontology design patterns, non ontological sources, SKOS, trust, agriculture.*

1. Introduction

En agriculture, de nombreuses données concernant les cultures sont disponibles dans différents formats électroniques : thésaurus, bases de données... Un défi pour les années à venir est de rendre ces données accessibles à tous les acteurs (agriculteurs, chercheurs en agronomie) afin qu'ils puissent les réutiliser dans des Bases de Connaissances (BC) représentées au format OWL [1]. Dans notre cas, une BC est une ontologie peuplée d'individus, sur laquelle il est possible d'effectuer des raisonnements logiques.

Dans cet article, nous détaillons une méthode permettant de construire une base de connaissances à partir de systèmes d'organisation des connaissances comme les thésaurus ou les taxonomies. Les principes généraux de cette méthode ont déjà été publiés dans (Amarger *et al*., 2014). Notre présent travail détaille différents types de connaissances extraites (candidats sommets et candidats arcs). Pour ce faire, nous proposons une formalisation des bases de connaissances sous forme de graphes. Comme les sources peuvent parfois contenir des erreurs (Soergel *et al*., 2006), nous évaluons le degré de confiance des connaissances extraites. Nous partons de l'hypothèse selon laquelle si une connaissance est issue de plusieurs sources, alors son score de confiance augmente. Cet article propose plusieurs fonctions de confiance qui sont évaluées sur un cas d'usage de construction d'une taxonomie des blés. Cet article est organisé comme suit : la section 2 est un état de l'art concernant l'ingénierie d'ontologies et la confiance. Notre méthode générale est ensuite présentée en section 3. La section 4 utilise le thésaurus Agrovoc comme cas d'étude pour dérouler notre méthode de construction d'ontologies. La section 5 décrit l'implémentation de notre proposition. La section 6 est consacrée aux expérimentations. Nous concluons et traçons quelques perspectives en section 7.

2. État de l'art

Pour clarifier notre apport présenté en 2.3, nous rappelons en quelques grandes lignes les conclusions des états de l'art déjà publié dans (Amarger *et al*., 2013) et (Amarger *et al*., 2014).

2.1. Réutilisation de sources non ontologiques et méthodologie Neon

La plus grande partie des méthodes d'ingénierie d'ontologies utilisent des sources non ontologiques au cours de leur processus d'extraction. Nous pouvons citer par exemple MethOntology (Fernandez-Lopez *et al*., 1997), la méthode NOR2O (Villazon-Terrazas *et al*., 2010) de la méthodologie Neon (Suárez-Figueroa *et al*., 2012) ou la methodologie SMOL (Gil, Martin-Bautista, 2014). Dans notre travail, nous nous focalisons sur des méthodes d'ingénierie utilisant un système d'organisation de connais-

1. http://www.w3.org/TR/owl2-overview/

sances comme des thésaurus, des taxonomies et des schémas de classification, parce qu'ils sont les plus couramment utilisés pour la description et la classification d'organismes vivants. De nombreux systèmes d'organisation des connaissances partagent un modèle similaire et sont utilisés dans des applications identiques. Un système d'organisation de connaissances peut être défini comme une organisation hiérarchique de termes normalisés utilisée pour la classification d'entités réelles. Certains systèmes d'organisation de connaissances sont disponibles sur le web de données liées (ou LOD pour Linked Open Data) en utilisant la norme SKOS (Simple Knowledge Organisation System) [2]. Nous avons étudié dix méthodes permettant de créer une base de connaissances en OWL utilisant des systèmes d'organisation de connaissances (Amarger *et al*., 2013). Ces méthodes peuvent être catégorisées en manuelles, semi-automatiques ou automatiques. Notre étude est détaillée dans (Amarger *et al*., 2013). Toutes ces méthodes montrent que la transformation d'un système d'organisation de connaissances doit être guidée si l'on veut construire une base de connaissances exploitable.

En raison de sa généricité, nous avons choisi de travailler avec la méthodologie NeOn. NeOn se décline en neuf scénarios, chacun proposant une méthode de construction d'ontologies différentes. Chaque scénario implique différents processus pour la construction collaborative d'ontologies et de réseaux d'ontologies. Nous avons tout d'abord étudié le scénario 2 de Neon « réutilisation et réingénierie de sources non ontologiques » (Villazon-Terrazas *et al*., 2010). Ce scénario permet le développement d'ontologies à partir de sources non ontologiques comme des systèmes d'organisation des connaissances, des bases de données ou d'autres types de sources. Cette méthode prend en compte les choix de modélisation et d'implémentation et applique le même ensemble de règles de transformation sur la source. Chaque ensemble de règles de transformation constitue un patron de transformation. Comme la transformation d'un thésaurus doit être guidée, nous avons également étudié le scénario 7 « réutilisation de patrons de conception d'ontologies ». Un patron de conception d'ontologies (ou ODP, pour Ontology Design Pattern) (Gangemi, Presutti, 2009) est défini en tant que solution de modélisation pour un problème récurrent de conception d'ontologies (Gangemi, Presutti, 2009). Les ODP sont normalement générés par l'expérience des ontologues, qui les soumettent en ligne dans des répertoires de patrons de conception [3]. Ces patrons sont évalués par la communauté des ontologues et sont en général adoptés en tant que bonne pratique. Le scénario 7 sélectionne des patrons pour construire des ontologies.

2.2. Confiance dans des éléments ontologiques

L'extraction d'éléments ontologiques à partir de sources non ontologiques de différentes qualités implique de prendre en considération la confiance dans ces éléments. Plusieurs définitions de la notion de confiance en informatique et sur le web séman-

2. http://www.w3.org/TR/2009/REC-skos-reference-20090818/
3. Par exemple sur le site web http://ontologydesignpatterns.org

tique sont présentées dans (Artz, Gil, 2007). Celle qui correspond le mieux à notre propos est :

« La confiance d'une partie A à une partie B pour un service X est la croyance mesurable de A dans B pour une période déterminée dans un contexte spécifique (par rapport au service X). »

Considérons A comme étant l'utilisateur qui veut créer une base de connaissances, B étant une source et X le processus d'extraction. Dans cette définition, la confiance concerne la source B, avec le processus d'extraction X, qui génère des éléments ontologiques avec un score de confiance associé. Cette définition est bien adaptée à notre propos puisqu'on considère qu'un score de confiance est spécifique à une période, un contexte et un service. Cela correspond au fait que la confiance en une source est variable selon les objectifs du projet, le temps et la source elle-même. L'utilisation de sources multiples pour extraire des éléments ontologiques conduit à une agrégation de scores de confiance : trouver le même élément ontologique dans plusieurs sources va augmenter le score de confiance de cet élément. Comme montré dans (Downey *et al.*, 2005), l'agrégation de scores de confiance est plus appropriée que les systèmes d'apprentissage ou les modèles « noisy-or » (Pearl, 1988). Les systèmes d'apprentissage nécessitent la création d'une base d'apprentissage qui peut être coûteuse à obtenir sans être nécessairement représentative du domaine étudié. Les modèles de type « noisy-or » ignorent la répétition de l'extraction, ce qui est contre-intuitif pour notre problème.

2.3. Synthèse

La méthode que nous proposons peut être vue comme une combinaison de deux méthodes d'ingénierie d'ontologies de NeOn (Suárez-Figueroa *et al.*, 2012) : l'une fondée sur des patrons de conception d'ontologies (Presutti *et al.*, 2012) et l'autre fondée sur des transformations de sources non ontologiques avec NOR2O (Villazon-Terrazas *et al.*, 2010). Les patrons de transformation des sources utilisés dans notre méthode sont fondés sur le même ensemble de patrons de conception ontologique.

Notons que dans la méthodologie SMOL, les auteurs incluent dans leur chaîne de traitements un processus de « construction de structure de connaissances » afin de réorganiser et harmoniser les modèles conceptuels hérités de différentes sources. Malheureusement, ce processus n'est pas détaillé même si les auteurs indiquent que l'éditeur d'ontologies Protégé peut être utilisé. Dans ces travaux, les patrons ne sont pas mentionnés. Au contraire, la méthode Hepp (Hepp, Bruijn, 2007) inclut un patron de conception afin de transformer un système d'organisation des connaissances au format SKOS : deux termes liés par la relation hiérarchique $skos : broader$ génèrent 4 $owl : classes$ reliées par 3 relations $owl : subClasOf$. Notre méthode peut être vue comme une généralisation de (Hepp, Bruijn, 2007) dans laquelle différents patrons de conception peuvent être utilisés.

De plus, nous prenons en considération le consensus à propos de chaque élément ontologique, afin de déterminer si nous souhaitons le conserver ou pas. Pour ce faire, nous utilisons un score de confiance calculé à partir de toutes les sources utilisées pour extraire les éléments ontologiques. À notre connaissance, il n'y a pas de méthode d'ingénierie d'ontologies capable de transformer un système d'organisation de connaissances en utilisant le consensus et des patrons de conception ontologique. Concernant le consensus, nous avons à définir des fonctions pour calculer les scores de confiance et les agréger.

3. Approche globale

Notre méthode se compose de trois processus déjà présentés dans (Amarger *et al.*, 2014). Par rapport à l'article précédent, nous avons détaillé les critères sur l'évaluation des sources.

– **1 Analyse de sources :** Pendant ce processus, l'expert du domaine et l'ontologue travaillent ensemble pour sélectionner les sources les plus appropriées pour la construction de leur base de connaissances. Ils inspectent chaque source afin d'évaluer leur couverture mais également afin d'évaluer la faisabilité de la transformation automatique en base de connaissances.

– **2 Transformation de sources :** Ce processus transforme chaque source en une base de connaissances au format OWL. Il est fondé sur les deux méthodes de NeOn: la méthode qui fusionne des patrons de conception pour construire un module ontologique et la méthode qui utilise des patrons de transformation.

– **3 Fusion de bases de connaissances :** Ce processus construit la base de connaissances finale en se fondant sur toutes les bases de connaissances extraites à partir des sources. À notre connaissance, ce processus n'est proposé dans aucune des méthodes d'ingénierie d'ontologies. Généralement, les méthodes d'ingénierie d'ontologies utilisent plusieurs sources séparément afin d'enrichir la base de connaissances de manière incrémentale. Le processus de fusion utilise plusieurs bases de connaissances au même moment, afin d'extraire des connaissances consensuelles.

Figure 1. Vue d'ensemble de la méthode proposée

3.1. Analyse de sources

Nombre de critères ont été définis afin d'évaluer la qualité des sources. Dans l'état actuel des choses, nous utilisons quatre critères afin de sélectionner des sources :

– **La réputation de la source** est généralement fondée sur la réputation de ses auteurs. Si l'auteur est une institution gouvernementale ou internationale, la source sera plus probablement acceptée par une large communauté d'utilisateurs et sera réutilisée. La réputation de la source peut également être fondée sur le nombre de ses utilisateurs. Ce critère est évalué manuellement par l'expert du domaine.

– **La fraîcheur de la source** se fonde sur la dernière date de mise à jour de la source, ainsi que sur le fait que la source est souvent mise à jour ou pas. Ce critère est évalué manuellement par l'expert du domaine.

– **L'adéquation de la source à la cible** représente la similarité entre la source et la base de connaissances souhaitée. Elle prend en compte la couverture de la source par rapport à la base de connaissances finale. Ce critère peut également être estimé par le nombre de liens établis entre le module et la base de connaissances extraite de la source.

– **La clarté du modèle de la source** évalue le fait que le modèle conceptuel de la source peut être facilement trouvé. Par exemple si différents patrons sont utilisés pour stocker le même type d'informations, alors le modèle est ambigu. Ceci est évalué manuellement par l'expert de connaissances quand il navigue dans la source pour trouver une information spécifique. Si la source a un modèle clair, elle sera facilement transformée en utilisant des patrons de détection et de transformation.

Les deux premiers critères sont issus de (Jayawardene *et al.*, 2013). Dans notre méthode, un module guide le processus de transformation de la source. Nous définissons une source de bonne qualité comme étant une source adaptée à notre module, ce qui nous amène à définir les deux derniers critères : si la source n'est pas adaptée au module ou si elle ne peut pas être transformée en utilisant les patrons, alors cette source sera sans intérêt dans notre approche.

3.2. Transformation de sources

La figure 2 (Amarger *et al.*, 2014) présente notre processus de « transformation de sources », qui fait appel à plusieurs autres activités. La « construction de module » est le scénario 7 de NeOn (Gangemi, Presutti, 2009). Ce scénario propose de construire un module réutilisant des patrons de conception d'ontologies et des questions de compétences. Nous appliquons complètement cette méthode pour construire un module. Le module est construit une fois pour toutes et est utilisé pour tous les processus de « transformation automatique de sources ». La « transformation syntaxique » est une adaptation du scénario 2 de NeOn (Villazon-Terrazas *et al.*, 2010). Le scénario 2 propose de construire une base de connaissances en réutilisant une source non-ontologique. Nous adaptons cette méthode pour enrichir le module préalablement construit par le scénario 7.

3.2.1. Transformation automatique de sources

Chaque source non ontologique suit généralement certains principes de modélisation et de mise en œuvre dans un format spécifique. Les méthodes de (Villazon-

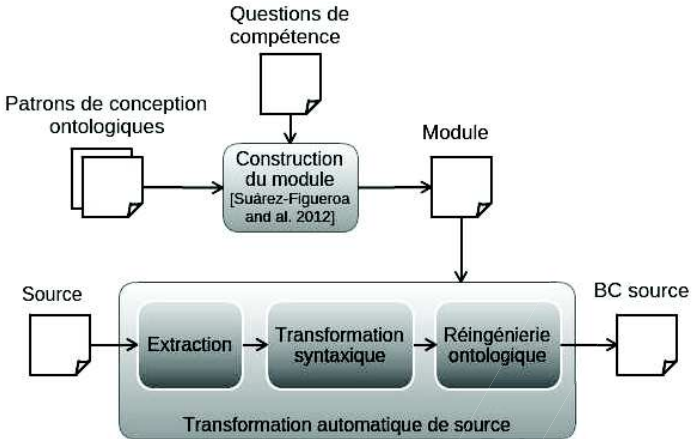

Figure 2. Processus de transformation de sources

Terrazas *et al*., 2010) (Soergel *et al*., 2006) proposent des transformations reposant sur des patrons. La méthode (Villazon-Terrazas *et al*., 2010) prend en compte les choix de modélisation et d'implémentation et applique le même patron de transformation sur la source. La méthode (Soergel *et al*., 2006) prend en compte le fait que les choix de modélisation peuvent changer sur la même source et que le même patron ne peut pas être appliqué sur la totalité de la source. Nous avons tiré profit de ces deux méthodes et appliquons une transformation fondée sur des patrons. Comme montré en figure 2, nous extrayons d'abord de la source les parties qui semblent suivre les mêmes principes de modélisation et qui répondent à nos exigences. Le processus précédent d'analyse de sources a mis en évidence le fait que ces parties existaient dans la source. Nous appliquons ensuite une transformation syntaxique utilisant la méthode NOR2O (Villazon-Terrazas *et al*., 2010) et les outils associés afin d'avoir un fichier respectant la syntaxe OWL. L'activité "réingénierie ontologique" produit un nouveau fichier OWL qui est un enrichissement d'un module, autrement dit une base de connaissances. Pour ce faire, le module est aligné avec le premier fichier OWL. La sortie est un ensemble de correspondances. Le module est alors étendu en utilisant le fichier OWL et en suivant le nouveau patron qui effectue la réingénierie du fichier OWL. Les nouveaux axiomes OWL sont donc compatibles avec le module. À la fin de l'application du patron de réingénierie, la base de connaissances source contient le module, complété entre autre par de nouveaux individus. Nous obtenons donc à la fin de cette étape une base de connaissances source, qui est une base de connaissances obtenue à partir d'une transformation automatique, comme présenté précédemment.

Une base de connaissances : $BC = (V_{bc}, E_{bc}, \Sigma V_{bc}, \Sigma E_{bc}, etiqV_{bc}, etiqE_{bc}, sourceE_{bc}, cibleE_{bc})$ est un multigraphe orienté étiqueté. La décision de formaliser une base de connaissances sous forme d'un graphe étiqueté est fondée sur la volonté de clarifier le lien entre une propriété définie dans l'ontologie et son instanciation dans la base de connaissances. En RDF, le même URI est utilisé comme étiquette du sommet

lors de sa définition et comme étiquette d'arc lors de son instanciation. De plus, les connaissances que nous allons extraire des bases de connaissances sont également formalisables sous forme de graphes. Les éléments de la base de connaissances BC sont définis de la manière suivante :

Base de connaissances

$V_{bc} = C \cup PropO \cup PropDT \cup I \cup L$ est un ensemble fini de sommets qui sont répartis dans des sous-ensembles disjoints tels que:

– C est l'ensemble des classes.

– $PropO$ est l'ensemble des propriétés d'objets.

– $PropDT$ est l'ensemble des propriétés de types de données.

– I est l'ensemble des individus.

– L est l'ensemble des sommets représentant les littéraux, dont les labels. Nous définissons L_{label} comme l'ensemble des sommets représentant les labels. $L_{label} \subset L$.

ΣV_{bc} est l'ensemble des étiquettes des sommets de la base de connaissances. Ces identifiants sont des chaînes de caractères alphanumériques pour les littéraux ou des URI pour les autres sommets.

$etiqV_{bc} : V_{bc} \rightarrow \Sigma V_{bc}$ est une application qui, à tout sommet v de V_{bc}, associe une seule étiquette $l \in \Sigma V_{bc}$.

$\Sigma E_{bc} = \{subClassOf, subPropertyOf, instanceOf, rdfs : label\} \cup \Sigma_{propO} \cup \Sigma_{propDT}$ est un ensemble fini d'étiquettes d'arcs.

– $subPropertyOf$ est la relation de spécialisation entre propriétés (propriété d'objet ou de type de données),

– $rdfs : label$ est la propriété d'annotation associant un label à une classe, à une propriété ou à un individu.

– $instanceOf$ exprime l'appartenance d'un individu à une classe.

– Σ_{propO} est l'ensemble des étiquettes des sommets $v \in PropO$ correspondant aux propriétés d'objet qui vont être utilisées comme étiquettes d'arcs dans BC. $\forall l \in \Sigma_{propO}$ $\exists v \in PropO$ tel que $etiqV_{bc}(v) = l$. En effet, pour qu'une propriété d'objet soit utilisée dans une base de connaissances, elle doit d'abord être définie par un sommet dans l'ontologie associée.

– Σ_{propDT} est l'ensemble des étiquettes des sommets $v \in PropDT$ correspondant aux propriétés de types de données qui vont être utilisées comme étiquettes d'arcs dans BC. $\forall l \in \Sigma_{propDT}$ $\exists v \in PropDT$ tel que $etiqV_{bc}(v) = l$. En effet pour qu'une propriété de type de données soit utilisée dans une base de connaissances, elle doit d'abord être définie par un sommet.

$E_{bc} \subset \Sigma E_{bc} \times V_{bc} \times V_{bc}$ est l'ensemble fini d'arcs étiquetés de la forme $p(v_1, v_2)$ ayant une étiquette $p \in \Sigma E_{bc}$, un sommet initial $v_1 \in V_{bc}$ et un sommet terminal $v_2 \in V_{bc}$.

Les arcs de BC remplissent l'une des conditions suivantes :

– $\forall p(v_1, v_2)$ un arc étiqueté tel que $p = rdfs : label$ ssi $v_1 \in C \cup PropO \cup PropDT \cup I$ et $v_2 \in L_{label}$

– $\forall p(v_1, v_2)$ un arc étiqueté tel que $p = subClassOf$ ssi $v_1 \in C$ et $v_2 \in C$

– $\forall p(v_1, v_2)$ un arc étiqueté tel que $p = subPropertyOf$ ssi $v_1 \in PropO \cup PropDT$ et $v_2 \in PropO \cup PropDT$,

– $\forall p(v_1, v_2)$ un arc étiqueté tel que $p \in \Sigma_{propO}$ ssi $v_1 \in I$ et $v_2 \in I$,

– $\forall p(v_1, v_2)$ un arc étiqueté tel que $p \in \Sigma_{propDT}$ ssi $v_1 \in I$ et $v_2 \in L$,

– $\forall p(v_1, v_2)$ un arc étiqueté tel que $p = instanceOf$ ssi $v_1 \in I$ et $v_2 \in C$.

Une base de connaissances est aussi un multi-graphe tel qu'il n'existe qu'un seul arc étiqueté par une étiquette reliant un sommet initial à un sommet terminal. Nous définissons trois applications sur les arcs de la base de connaissances :
$etiqE_{bc} : E_{bc} \rightarrow \Sigma E_{bc}$ est une application qui associe à chaque arc son étiquette.
$sourceE_{bc} : E_{bc} \rightarrow V_{bc}$ est une application qui associe à chaque arc son sommet initial.
$cibleE_{bc} : E_{bc} \rightarrow V_{bc}$ est une application qui associe à chaque arc son sommet terminal.

Lors de la fusion des bases de connaissances, nous nous concentrons sur un sous ensemble des éléments ontologiques constituant une base de connaissances source, que nous intitulons élément ontologique source. Dans ce cadre, un élément ontologique source oe est un sommet d'une base de connaissances source qui peut être aligné par un outil d'alignement automatique. Actuellement, au vu des résultats des outils d'alignement que nous avons pu tester, un élément ontologique source est soit une classe, soit un individu, soit un label. Par contre, la base de connaissances finale contient toute sorte d'éléments ontologiques, sommets et arcs.

Elément ontologique source

$oe \in C \cup I \cup L_{label}$
On définit également une fonction $nature(oe)$ qui permet d'obtenir le type de l'élément ontologique associé :

– $nature(oe) =$ "classe" si $oe \in C$,

– $nature(oe) =$ "label" si $oe \in L_{label}$,

– $nature(oe) =$ "individu" si $oe \in I$,

– $nature(oe) =$ "null" sinon.

3.3. Fusion de bases de connaissances

La figure 3 montre que la fusion de bases de connaissances préalablement construites, appelées *Bases de Connaissances Sources* (BCS), se compose de trois activités :

– **Génération de correspondances :** cette activité calcule des alignements entre toutes les bases de connaissances sources grâce à des outils d'alignement. L'activité d'alignement identifie les éléments ontologiques sources similaires contenus dans des bases de connaissances sources distinctes.

– **Génération de candidats :** Cette activité identifie les candidats. Un candidat contient un ensemble d'éléments ontologiques sources qui ont été considérés comme similaires par l'activité de génération des correspondances.

– **Calcul de confiance :** Cette activité calcule les scores de confiance des candidats et crée de nouveaux éléments ontologiques fondés sur les candidats. La sortie est une nouvelle base de connaissances d'éléments ontologiques pondérés.

– **Filtrage :** Cette activité filtre les éléments ontologiques en fonction de leurs scores de confiance et les insère dans la base de connaissances finale.

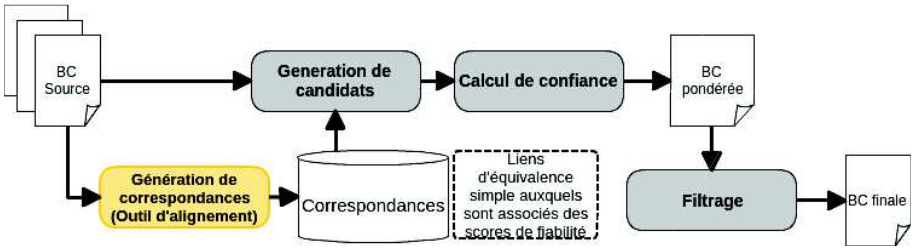

Figure 3. Processus de fusion de bases de connaissances

3.3.1. Génération des alignements

Aligner BC_1 et BC_2 consiste à calculer toutes les correspondances entre les éléments de BC_1 et les éléments de BC_2. C'est un sujet de recherche actif (Euzenat, Shvaiko, 2013) et de nombreuses méthodes ont été proposées et implémentées. Notre travail réutilise les outils qui implémentent diverses méthodes.

Soit deux bases de connaissances sources BCS_i et BCS_j construites grâce à la transformation de sources :

$BCS_i = (V_{bcsi}, E_{bcsi}, \Sigma V_{bcsi}, \Sigma E_{bcsi}, etiqV_{bcsi}, etiqE_{bcsi}, sourceE_{bcsi}, cibleE_{bcsi})$ et

$BCS_j = (V_{bcsj}, E_{bcsj}, \Sigma V_{bcsj}, \Sigma E_{bcsj}, etiqV_{bcsj}, etiqE_{bcsj}, sourceE_{bcsj}, cibleE_{bcsj})$,

nous définissons une correspondance m comme étant une arête entre un couple de sommets $\{oe_i, oe_j\}$, avec $oe_i \in V_{bcsi}, oe_j \in V_{bcsj}$. Une correspondance représente une équivalence entre deux éléments ontologiques sources, détectée à l'aide d'outils d'alignement. Les correspondances sont définies de la manière suivante :

Correspondance

– $V_{bcsi} \neq V_{bcsj}$ une correspondance est toujours établie entre deux sommets appartenant à des ensembles de sommets de BCS différentes. $\nexists m = \{oe_i, oe_k\}$ tel que $oe_i \in V_{bcsi}$ et $oe_k \in V_{bcsi}$.

– $nature(oe_i) = nature(oe_j) \neq$"null". Une correspondance est toujours établie entre deux éléments ontologiques sources de même nature.

– $valueE : (C \cup I) \times (C \cup I) \rightarrow]0, 1]$ est une application qui, à toute arête définie comme correspondance, associe un unique degré de correspondance 0 et 1 tel que $valueE(oe_i, oe_j) = valueE(oe_j, oe_i)$.

Les correspondances générées par des outils d'alignement nous permettent de construire un nouveau graphe composé des bases de connaissances sources alignées, intitulé SA. Dans la suite de cet article, nous considérons que N BCS ont été alignées. Rappelons que ces N BCS ont toutes en commun le module, noté BC_m. Ce module n'apparaît qu'une fois dans SA.

3.3.2. Génération des candidats

Un candidat $Cand$ est un sous-graphe de SA qui génèrera peut être un élément (sommet ou arc) dans la base de connaissances finale. Cet élément possible est donc le résultat de la fusion d'éléments issus de plusieurs BCS, considérés comme équivalents. Les éléments des BCS jugés équivalents sont des composants du sous-graphe formant le candidat. Nous considérons deux types de $Cand$, les candidats sommets $CandS$, qui produiront un sommet de la base finale, et les candidats arcs $CandA$ qui génèreront un arc dans la base finale. Chaque candidat arc est donc associé à au moins un candidat sommet.

3.3.2.1. Candidat sommet

$CandS$ = (V_{CandS}, E_{CandS}, ΣV_{CandS}, $etiqV_{CandS}$, $valueE$) un candidat sommet, est un graphe non orienté connexe dont les sommets sont des éléments ontologiques sources provenant de BCS différentes et les arêtes sont les correspondances issues des alignements entre les N BCS. Nous définissons les composants d'un candidat sommet de la manière suivante :

Candidat sommet

V_{candS} est l'ensemble des sommets de $CandS$. Tous les sommets d'un candidat sommet appartiennent à des BCS différentes. Par conséquent $\|V_{CandS}\| \leq N$.
E_{CandS} est l'ensemble des arêtes de $CandS$. Les arêtes de $CandS$ sont des correspondances.
$valueE$: $E_{CandS} \rightarrow]0, 1]$ est une application qui associe à chaque arête son degré de correspondance.
Un candidat sommet est un graphe connexe. Par conséquent, tous les sommets de $CandS$ sont liés à au moins un autre sommet de $CandS$ par une correspondance, ce qui implique que tous les sommets de $CandS$ sont de même nature.

Les candidats sommets peuvent être classés suivant la nature de leurs sommets : candidat sommet individu ic, candidat sommet classe cc, candidat sommet label lc.

3.3.2.2. Candidat arc

Un candidat arc $CandA = (V_{CandA}, E_{CandA}, \Sigma V_{CandA}, \Sigma E_{CandA}, etiqV_{CandA}, sourceE_{CandA}, cibleE_{CandA})$ est un graphe biparti orienté étiqueté. Un candidat arc se construit à partir d'au moins un candidat sommet. Les sous-ensembles de sommets de $CandA$ sont soit un ensemble de sommets d'un candidat sommet, soit un singleton composé d'un sommet du module BC_m. Tous les arcs de $CandA$ sont étiquetés par

la même étiquette d'arc. Les candidats arcs sont des sous-graphes du multigraphe SA construit à partir des N bases de connaissances alignées.

Considérons deux candidats sommets distincts :

$CandS_1 = (V_{CandS1}, E_{CandS1}, \Sigma V_{CandS1}, etiqV_{CandS1}, valueE)$,

$CandS_2 = (V_{CandS2}, E_{CandS2}, \Sigma V_{CandS2}, etiqV_{CandS2}, valueE)$

Candidat arc entre 2 candidats sommets

$V_{CandA} = V_{CandS1} \cup V_{CandS2}$. L'ensemble des sommets de $CandA$ se compose de deux ensembles de sommets disjoints V_{CandS1} et V_{CandS2}.
$\Sigma V_{CandA} = \Sigma V_{CandS1} \cup \Sigma V_{CandS2}$. L'ensemble des étiquettes de sommet de $CandA$ est l'union des ensembles d'etiquettes de sommets de $CandS_1$ et $CandS_2$.
$etiqV_{CandA} : V_{CandS1} \cup V_{CandS2} \rightarrow \Sigma V_{CandS1} \cup \Sigma V_{CandS2}$ est l'appplication qui associe à tout sommet $v \in V_{CandA}$ son étiquette $l \in \Sigma V_{CandA}$ tel que $etiqV_{CandA}(v) = l$. Cette application est le prolongement des applications $etiqV_{CandS1}$ et $etiqV_{CandS2}$.
$\Sigma E_{CandA} = \{l_{CandA}\}$ l'ensemble des étiquettes d'arcs est composé d'une seule étiquette l_{CandA} tel que l_{CandA} est défini dans le module BC_m.
$E_{CandA} \subset \Sigma E_{CandA} \times V_{CandS1} \times V_{CandS2}$ est l'ensemble des arcs de $CandA$. Ces arcs sont tous étiquetés par la même étiquette l_{CandA} et tous orientés dans le même sens. Les arcs de $CandA$ vont tous d'un sommet de $CandS_1$ vers un sommet de $CandS_2$.
$sourceE_{CandA} : E_{CandA} \rightarrow V_{CandS1}$ est l'application qui à tout arc de $CandA$ associe son sommet initial pris dans V_{CandS1}
$cibleE_{CandA} : E_{CandA} \rightarrow V_{CandS2}$ est l'application qui à tout arc de $CandA$ associe son sommet terminal pris dans V_{CandS2}

Considérons un candidat sommet

$CandS = (V_{CandS}, E_{CandS}, \Sigma V_{CandS}, etiqV_{CandS}, valueE)$,

et un module $BC_m = (V_m, E_m, \Sigma V_m, \Sigma E_m, etiqV_m, etiqE_m, sourceE_m, cibleE_m)$.

Candidat arc entre 1 candidat sommet 1 un sommet du module

$V_{CandA} = V_{CandS} \cup \{v_m\}$. L'ensemble des sommets de $CandA$ se compose de deux ensembles de sommets disjoints V_{CandS} et $\{v_m\}$ tel que $v_m \in V_m$.
$\Sigma V_{CandA} \subset \Sigma V_{candS1} \cup \{etiqV_m(v_m)\}$. L'ensemble des étiquettes de sommets de $CandA$ est l'ensemble d'etiquettes de sommets de $CandS$ augmenté de l'étiquette du sommet du module v_m.
$etiqV_{CandA} : V_{CandA} \rightarrow \Sigma V_{CandA}$ est l'application qui associe à tout sommet $v \in V_{CandA}$ son étiquette $l \in \Sigma V_{candA}$ tel que $etiqV_{CandA}(v) = l$. Cette application est le prolongement des applications $etiqV_{CandS}$ et $etiqV_m$.
$\Sigma E_{CandA} = \{l_{CandA}\}$ l'ensemble des étiquettes d'arcs est composé d'une seule étiquette l_{CandA} tel que $l_{CandA} \in \Sigma E_m$.
$E_{CandA} \subset \Sigma E_{CandA} \times V_{CandA} \times V_{CandA}$ est l'ensemble des arcs de $CandA$. Ces arcs sont tous étiquetés par la même étiquette l_{CandA} et tous orientés dans le même sens : soit les arcs de $CandA$ vont d'un sommet de $CandS$ vers v_m, soit les arcs de $CandA$ vont de v_m vers un sommet de $CandS$.

$sourceE_{CandA}$: $E_{CandA} \rightarrow V_{CandA}$ est l'application qui à tout arc de $CandA$ associe son sommet initial. $cibleE_{CandA}$: $E_{CandA} \rightarrow V_{CandA}$ est l'application qui à tout arc de $CandA$ associe son sommet terminal.

3.3.3. Processus de génération de candidats

Pour générer ces candidats, deux étapes importantes apparaissent. La première est la génération de candidats sommets, qui se fait à partir des correspondances générées par les outils d'alignement. L'idée de la génération de ces candidats sommets est de trouver les composantes connexes dans le graphe multiparti SA composé des N BCS alignées. Une condition particulière est qu'il ne peut pas y avoir plusieurs éléments ontologiques sources dans un candidat sommet provenant de la même source. Nous ne considérons que les candidats sommets principaux. En effet, un candidat sommet ne peut être inclus dans un autre candidat sommet, ce qui définit une relation d'ordre partiel sur les candidats sommets. Nous ne prenons pas en considération les candidats sommets qui sont inclus dans un autre candidat sommet. Une fois ces candidats sommets générés, il est possible de déterminer les candidats arcs liés à ces candidats sommets. Pour chaque couple de candidats sommets, si le même arc est présent dans au moins deux sources, un candidat arc est généré, un cas particulier étant le candidat sommet label. Un candidat sommet label est composé des labels alignés, et de ce candidat sommet doivent arriver des arcs étiquetés $rdfs : label$ qui associent les labels à un autre candidat sommet. Ainsi, un candidat sommet label doit nécessairement appartenir à un candidat arc. Si un tel candidat arc n'existe pas, alors le candidat sommet label est ignoré. Ce regroupement de candidats est fondé sur l'intuition selon laquelle un label rattaché à aucun élément ontologique n'est pas pertinent.

3.3.4. Proposition de fonction de confiance

Chaque candidat a un score de confiance permettant de définir à quel point nous pouvons croire en la pertinence de ce candidat. Il y a plusieurs moyens de calculer ce score. Nous définissons plusieurs fonctions de confiance qui seront testées dans nos expérimentations.

Fonction de confiance : $trust_{simple}$

La façon simple d'obtenir des éléments ontologiques consensuels est de déterminer combien de bases de connaissances sont impliquées dans le candidat. Nous considérons qu'un candidat est consensuel si la moitié au moins des bases de connaissances sont impliquées dans son extraction. Dans le cas contraire, le candidat ne devrait pas appartenir à la base de connaissances finale. Nous avons défini une fonction de confiance appelée $trust_{simple}$ afin d'implémenter le consensus. Cette fonction est différente suivant le type de candidats (candidats sommets et candidats arcs). $trust_{simple}$ est définie par la formule suivante :

$$trust_{simple}(CandS) = \frac{|V_{CandS}|}{N}$$
$$trust_{simple}(CandA) = \frac{|E_{CandA}|}{N} \quad (1)$$

Rappelons que N est le nombre de BCS alignées et correspond donc au nombre de sources. En ne considérant que les candidats pour lesquels la valeur de $trust_{simple}$ est supérieure ou égale à 0,5 nous pouvons obtenir les candidats consensuels.

Fonction de confiance : $trust_{degree}$

Nous pouvons également utiliser le degré de correspondance afin de calculer le score de confiance d'un candidat. Nous considérons qu'un candidat dont la somme des degrés de correspondance est élevée implique plus de confiance. Pour implémenter cette fonction, nous utilisons une fonction différente pour chaque type de candidat.

– **Fonction de confiance** $trust_{degree}$ **pour les candidats sommets**

Cette fonction de confiance s'applique sur les candidats sommets individus et classes. Elle est définie par la formule :

$$trust_{degree}(CandS) = \frac{\sum_{\forall e_i \in E_{CandS}}^{|E_{CandS}|} valueE(e_i)}{\frac{N(N-1)}{2}} \quad (2)$$

Cette fonction additionne tous les degrés de correspondances impliqués dans le candidat sommet. Nous normalisons le résultat avec le nombre maximum de correspondances possibles dans un candidat sommet.

– **Fonction de confiance** $trust_{degree}$ **pour les candidats arcs**

Cette fonction de confiance s'applique à tous les candidats arcs ayant une étiquette d'arc autre que $rdfs : label$. La fonction est définie par la formule :

$$trust_{degree}(CandA) = \frac{|E_{CandA}| + \frac{trust_{degree}(CandS1)+trust_{degree}(CandS2)}{2}}{N+1} \quad (3)$$

tel que $CandS1$ et $CandS2$ sont lié par $CandA$

Cette formule prend en compte $|E_{CandA}|$ (le nombre d'arcs présents dans le candidat arc) et la moyenne des scores de confiance des candidats sommets liés par le candidat arc. Nous faisons cela afin de simuler un degré de correspondance entre arcs, car les outils d'alignement n'alignent pas les arcs. Nous avons normalisé ce résultat avec le N, qui est la valeur maximale de nombre d'arcs, plus 1, qui est la valeur maximum que la moyenne de deux scores de confiance des candidats sommets peut valoir. Si un candidat arc relie un candidat sommet à un sommet du module, alors le $trust_{degree}$ du sommet du module est égal à 1.

– **Fonction de confiance** $trust_{degree}$ **de candidat label**

Cette fonction de confiance est définie par la formule :

$$trust_{degree}(lc) = \frac{trust_{degree}(CandS) + \frac{\sum_{\forall e_i \in E_{lc}}^{|E_{lc}|} valueE(e_i)}{\frac{N(N-1)}{2}}}{2} \quad (4)$$

tel que lc est lié à $CandS$ par $CandA$

Le score de confiance du candidat sommet label lc est la moyenne entre le score confiance du candidat sommet $CandS$ (auquel lc est lié par un candidat arc) et la somme normalisée de ses degrés de correspondances. Le score de confiance du candidat arc $CandA$ ayant comme étiquette d'arc $rdfs : label$ liant lc à $CandS$ est égal au score de confiance de lc.

3.3.5. Filtrage

L'étape de filtrage utilise le score de confiance associé à chaque candidat extrait lors de l'étape précédente. Dans ce cas, nous pouvons définir un seuil minimum et un seuil maximum, pris entre 0 et 1. Le seuil minimum est utilisé comme limite en-dessous de laquelle le candidat est automatiquement rejeté, alors que le seuil maximum est utilisé comme limite au-dessus de laquelle le candidat est automatiquement accepté. Entre ces deux limites, l'expert du domaine doit valider manuellement chaque candidat. Cette étape de filtrage réduit le nombre de candidats que l'expert du domaine doit valider manuellement.

4. Application de la méthode sur le cas d'étude du thésaurus Agrovoc

Pour illustrer notre méthode, nous avons construit un cas d'utilisation à propos des cultures de blé. Nous voulons construire une base de connaissances finale décrivant la taxonomie des blés.

4.1. Processus d'analyse de source

Pour le projet de taxonomie du blé, nos experts ont sélectionné les sources suivantes : **Agrovoc**[4]**:** un thésaurus multilingue contenant plus de 40 000 termes maintenu par la FAO, **TaxRef**[5]**:** un référentiel taxonomique français contenant 80 000 taxons créé par le museum national d'histoire naturelle , **NCBI Taxonomy**[6]**:** une taxonomie créée par le National Center for Biotechnology Information (NCBI) des Etats-Unis, contenant 1 million de taxons. Ces trois sources ont été choisies parce que toutes sont bien connues en agriculture mais également parce qu'elles sont complémentaires. NCBI est la source qui contient le plus de taxons. Elle est considérée par les experts comme ayant les noms de taxons les plus récents, mais elle contient des incohérences et il y a peu de labels. Parallèlement, Agrovoc contient des labels

4. http://aims.fao.org/standards/agrovoc/about
5. http://inpn.mnhn.fr/programme/referentiel-taxonomique-taxref
6. http://www.ncbi.nlm.nih.gov/

en plusieurs langues et distingue les noms scientifiques des noms vernaculaires, mais elle contient moins de taxons que NCBI et sa qualité est souvent critiquée (Soergel *et al*., 2006). TaxRef surmonte cet inconvénient. Elle est la référence nationale française dans la classification du monde vivant ; par conséquent, sa cohérence est assurée, mais le nombre de taxons est limité par le processus de vérification de la cohérence. La combinaison de ces trois sources nous semble appropriée parce que nous combinons la quantité de taxons (NCBI), avec des labels présents en quantité (Agrovoc) et l'assurance de la qualité (TaxRef). Agrovoc est publié sur le LOD selon la norme SKOS [7]. La nouvelle norme des thésaurus se fonde sur le format SKOS (Miles, Bechhofer, 2009). Elle inclut plus de types de relations et donne la possibilité d'en définir de nouvelles. Par exemple, dans la figure 4, le lien $hasTaxonomicLevel$ a été défini par la FAO dans le thésaurus Agrovoc.

4.2. Processus de transformation de sources

Nous commençons le processus de transformation de sources en construisant un module concernant la classification des plantes : *Agronomic Taxon* (Roussey *et al*., 2013) (https://sites.google.com/site/agriontology/home/irstea/agronomictaxon).
Ce module a été manuellement créé pour une tâche spécifique (représenter le nom scientifique des organismes en utilisant une taxonomie). Le module est composé de $owl : Classes$ et définit l'ensemble de $owl : Properties$ qui peuvent exister entre-elles. Il réutilise certains ODP provenant du projet NeOn et deux vocabulaires déjà publiés sur le LOD (Roussey *et al*., 2013). Les types de taxons principaux souhaités dans notre base de connaissances sont définis dans le module en tant que $owl : class$, enfants de la classe $neon : Taxon$. Nous nous focalisons par exemple sur les sept types de taxons les plus connus : règne, phylum, classe, ordre, famille, genre et espèce. La figure 4 présente une partie du module *Agronomic Taxon*. À partir de chaque source, nous voulons extraire automatiquement des sous-parties de la classification taxonomique du blé. Nous nous focalisons sur l'extraction du taxon Triticum. Nous appliquons un patron de réingénierie présenté dans (Amarger *et al*., 2014) inspiré de (Villazon-Terrazas *et al*., 2010). Pour ce faire, nous définissons des algorithmes pour extraire des instances et des classes enfants de la classe $neon : Taxon$ à partir de nos trois sources. Pour chaque individu, nous le typons et le lions en utilisant la propriété d'objet $neon : hasHigherRank$. Rappelons que cette propriété est la relation hiérarchique utilisée pour décrire la hiérarchie taxonomique dans le module. Nous avons publié ces algorithmes implémentés sur github :

- **Agrovoc :** https://github.com/Murloc6/T2RKB,
- **TaxRef :** https://github.com/Murloc6/TaxRef2RKB,
- **NCBI :** https://github.com/Murloc6/NCBI2RKB.

La figure 4 présente un exemple de transformation du thésaurus Agrovoc. Le processus enrichit automatiquement le module. La base de connaissances finale contient

7. http://www.w3.org/2004/02/skos/

plusieurs taxons, individus de la classe $neon : Taxon$ ou de l'une de ses classes filles. Ces taxons sont utilisés pour décrire les organismes apparaissant dans les champs. À la fin du processus, nous obtenons une base de connaissances source compatible avec le module.

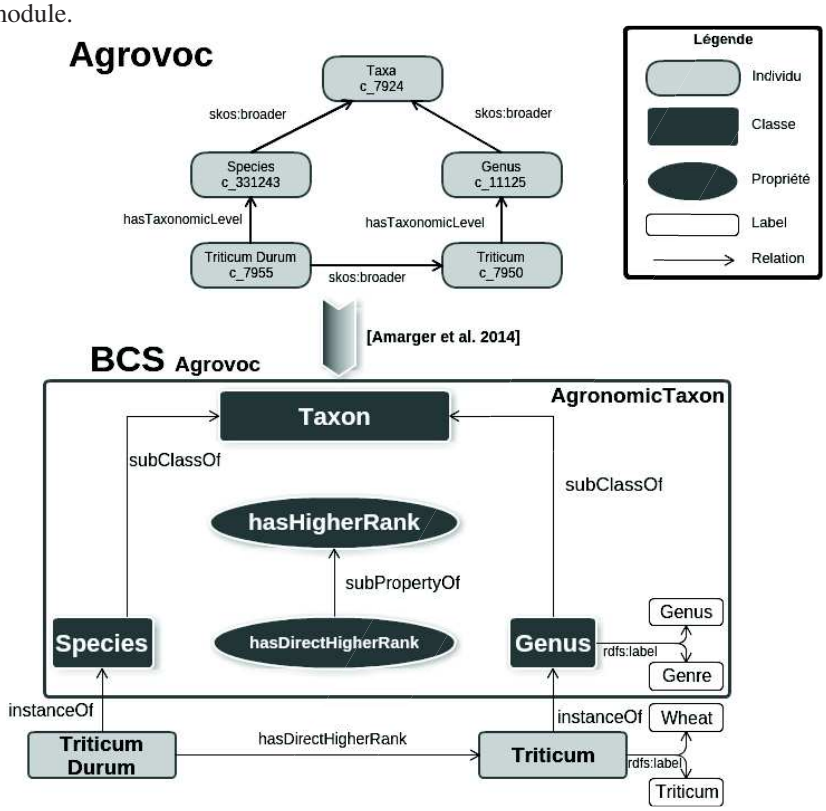

Figure 4. Un exemple de transformation d'une sous partie d'Agrovoc

4.3. Génération de candidats sommets

Nous avons 3 bases de connaissances sources. Nous recherchons des alignements entre ces bases. Nous appliquons ensuite le processus de génération de candidats sommets. La figure 5 présente deux candidats générés à partir de ces 3 sources : $Cand1$ qui correspond au taxon « Triticum » et $Cand2$ qui correspond à « Triticum Durum ».

4.4. Génération de candidats arcs

Une fois les candidats sommets générés, nous pouvons générer les candidats arcs. La figure 6 montre la création de deux candidats arcs suivant l'exemple précédent. Le premier candidat généré est l'arc « hasHigherRank » entre le candidat $Cand2$ et $Cand1$. Cet arc étant présent dans Agrovoc et NCBI, nous pouvons donc générer le

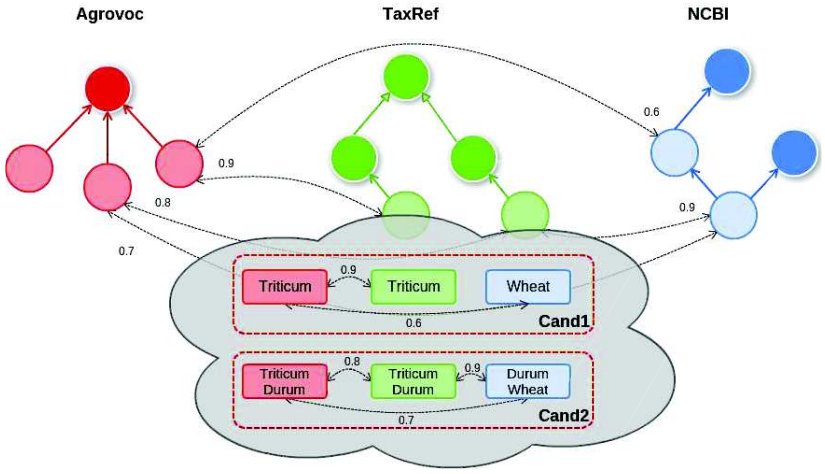

Figure 5. Exemple de génération de candidats sommets

candidat définissant que « Triticum Durum » a pour rang supérieur « Triticum ». Le second candidat arc généré ici lie le candidat label « Wheat » au candidat « Triticum ». Comme décrit précédemment, l'existence de candidats labels est un peu spécifique et dépend de l'existence d'un candidat arc associé. Le label « Wheat » apparaissant dans Agrovoc et TaxRef, un candidat label associé est généré. Les labels sont considérés comme des éléments ontologiques, une correspondance apparaît donc ici (valuée suivant la distance de Jaro-Winkler). Comme le candidat arc nommé $Cand\ arc\ 2$ lie le candidat label au candidat $Cand1$ par deux arcs étiquetés $rdfs : label$, le candidat label est pris en considération.

4.5. Calculs des scores de confiance

Comme défini précédemment, il existe deux méthodes de calcul de la confiance. Le tableau 1 présente les scores des différents candidats générés précédemment avec la fonction de calcul $trust_{simple}$. Nous pouvons observer que le score du candidat

Tableau 1. Exemple de calculs avec la fonction $trust_{simple}$

Cand1	Cand2	Cand arc 1	Cand arc 2	Cand label
1	1	0,66	0,66	0,66

$Cand1$ (Triticum) et celui du candidat $Cand2$ (Triticum Durum) est le même, alors que leur nombre de correspondances est différent.

Le tableau 2 présente les calculs avec la fonction $trust_{degree}$. Nous observons ici une plus grande différence de score entre les candidats $Cand1$ et $Cand2$. Nous observons aussi la différence sur le candidat arc $Cand\ arc\ 2$ qui est rattaché à $Cand1$. Le candidat $Cand1$ n'ayant que 0,5 en confiance, cela influe sur les candidats $Cand\ label$ et $Cand\ arc\ 2$ associés en diminuant leur score.

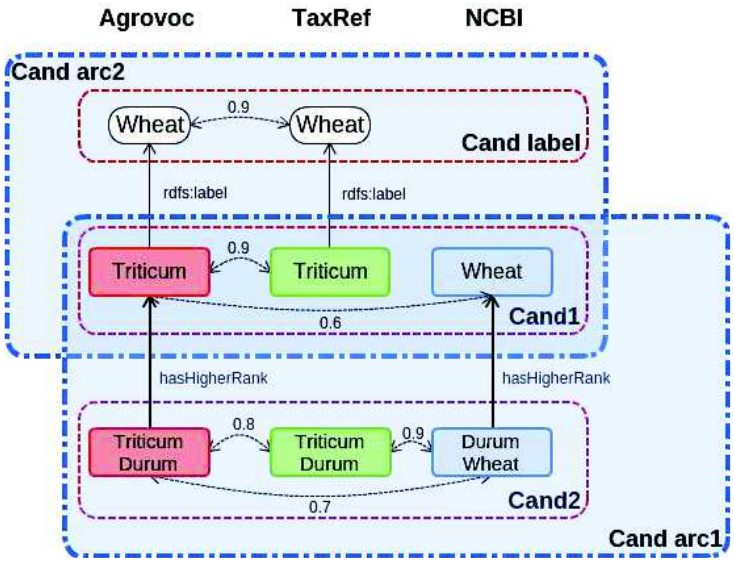

Figure 6. Exemple de génération de candidats arcs

Tableau 2. Exemple de calculs avec la fonction $trust_{degree}$

Cand1	Cand2	Cand arc 1	Cand arc 2	Cand label
0,5	0,8	0,66	0,4	0,4

5. Implémentation

Notre prototype appelé Muskca (MUlti Sources Knowledge CAndidates) a été développé en Java et est disponible sur github (https://github.com/Murloc6/Muskca).

5.1. Outil de transformation de sources

Pour chaque source en entrée, une méthode de transformation spécifique est appliquée afin d'obtenir la base de connaissances utilisée dans Muskca. Comme décrit précédemment, cette méthode est fonction du module utilisé et du format initial de la source. Nous pourrions utiliser un système comme NOR2O [8] mais nous avons décidé d'implémenter notre propre système afin de transformer les sources sélectionnées. Nous avons procédé de la sorte parce que nous avions des algorithmes spécifiques déjà écrits pour chaque source et qu'il nous semblait plus efficace de développer notre système en le fondant sur ces algorithmes. Nos algorithmes peuvent être vus comme une combinaison de patrons de transformation syntaxique de NOR2O et de patrons de réingénierie de connaissances.

8. http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/downloads/57-nor2o

5.2. Outils d'alignement

Afin de sélectionner l'outil d'alignement le plus approprié, nous avons étudié le dernier challenge OAEI [9] et tout particulièrement les résultats concernant l'alignement d'instances [10] (Dragisic *et al*., 2014), parce que nous voulons pouvoir aligner toutes les sortes d'éléments ontologiques. Nous avons choisi d'utiliser LogMap (Jiménez-Ruiz *et al*., 2012) parce qu'il permet d'aligner des éléments ontologiques quelconques, qu'il obtient de bons résultats dans le challenge OAEI [11] et que son code source est disponible en ligne [12]. LogMap est réellement facile à utiliser, tout particulièrement grâce à l'API SEALS [13] que tous les participants à OAEI doivent utiliser. Cela signifie que si, dans l'avenir, un système obtient de meilleurs résultats que celui-là, il sera simple de changer l'outil d'alignement.

Pour l'alignement des labels, nous utilisons la distance de Jaro Winkler (Winkler, 1990) comme mesure de similarité entre chaînes de caractères. Les correspondances entre sommets labels sont étiquetés par cette mesure.

5.3. Format de sortie

La sortie du processus de génération de candidats est stockée dans une nouvelle base de connaissances utilisant l'ontologie PROV. Cette base de connaissances est mise à jour par le processus de calcul des scores de confiance afin de stocker le score de confiance de chaque candidat. Le modèle de sortie qui était utilisé pour cette extraction est présenté en figure 7. Un exemple de ce genre d'extraction est disponible à l'adresse :
https://github.com/Murloc6/Muskca/blob/master/out/mini_Triticum_CandProvo.owl
Ce fichier est l'ontologie de provenance fondée sur l'extraction des expériences présentées ci-dessus.

6. Expérimentations

Le but de nos expérimentations est de tester différents paramètres de notre méthode générique capable de construire des bases de connaissances finales. Nous voulons déterminer quelles fonctions de confiance et quels seuils de filtrage permettent d'obtenir les meilleurs résultats. Ces expérimentations ont été menées avec le cas d'étude sur la taxonomie des blés.

9. Ontology Alignment Evaluation Initiative - http://oaei.ontologymatching.org/2013/
10. http://www.instancematching.org/oaei/imei2013/results.html
11. Ontology Alignment Evaluation Initiative - http://oaei.ontologymatching.org/2013/
12. https://code.google.com/p/logmap-matcher/
13. http://www.seals-project.eu/

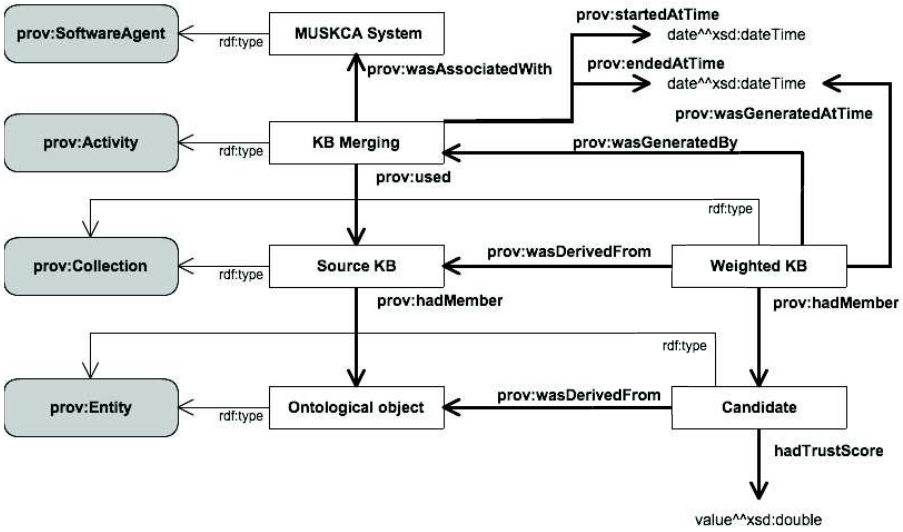

Figure 7. Utilisation de l'ontologie PROV

6.1. Configuration des expérimentations

Tout d'abord nous avons construit une référence sur la taxonomie des blés avec l'aide de trois experts (Amarger *et al.*, 2014). La précision, le rappel et la f-mesure sont calculés afin d'évaluer la qualité de la base de connaissances finale :

– La précision est le rapport entre le nombre d'éléments ontologiques de la référence qui apparaissent dans la base de connaissances finale et le nombre total d'éléments ontologiques de la base de connaissances finale.

– Le rappel est le rapport entre le nombre d'éléments ontologiques de la référence qui apparaissent dans la base de connaissances finale et le nombre total d'éléments ontologiques de la référence.

– La f-mesure est une combinaison des mesures de précision et rappel.

Nous avons effectué trois expérimentations :

1. La première expérimentation utilise la fonction $trust_{simple}$ afin de calculer le score de confiance de tout candidat et le seuil du processus de filtrage est fixé à 0,6. Un candidat devient donc un composant de la base de connaissances finale si son score est supérieur à 0,6, sinon, il est rejeté.

2. La deuxième expérimentation utilise la fonction $trust_{degree}$ afin de calculer le score de confiance de tout candidat et le seuil du processus de filtrage est fixé à 0,6.

3. La troisième expérimentation utilise la fonction $trust_{degree}$ afin de calculer le score de confiance de tout candidat et le seuil du processus de filtrage est fixé à 0,9.

Nous avons comparé les résultats des trois expérimentations avec notre référence.

6.2. Résultats et analyse

Le tableau 3 présente les résultats de la première expérimentation. La première ligne présente les résultats des candidats sommets individus, la deuxième ligne présente les résultats des candidats arcs $instanceOf$, la troisième ligne concerne les résultats des candidats arcs (autres que $intanceOf$ et $rdfs : label$) et la dernière ligne présente les résultats des candidats sommets labels. Le tableau 4 présente les résultats de la deuxième expérimentation.

Tableau 3. Résultats de la première expérimentation : $trust_{simple}$

Type de candidat	Précision	Rappel	F-Measure
Sommet individu	0,92	0,66	0,77
Arc $instanceOf$	0,70	0,43	0,54
Arc	0,65	0,51	0,57
Sommet label	0,32	0,35	0,34

Tableau 4. Résultats de la deuxième expérimentation : $trust_{degree}$ *et seuil fixé à 0,6*

Type de candidat	Precision	Rappel	F-Measure
Sommet individu	0,97	0,63	0,76
Arc $instanceOf$	0,77	0,39	0,52
Arc	0,68	0,51	0,58
Sommet label	0,39	0,24	0,30

Nous pouvons observer dans les tableaux 3 et 4 que les résultats sont encourageants. Tous les individus que notre méthode est capable d'extraire sont valables (les résultats de nos candidats sommets individus présentent une précision élevée). En revanche, notre méthode n'est pas encore en mesure d'extraire tous les éléments ontologique valables. La fonction $trust_{degree}$ donne des résultats légèrement meilleurs que ceux de la fonction $trust_{simple}$. De plus, notre méthode est capable d'extraire des individus plus que des liens entre individus. L'évaluation des candidats sommets individus a par exemple une précision de 0,92 pour la première expérimentation et nous atteignons 0,97 pour la deuxième expérimentation. De manière générale, la fonction $trust_{degree}$ améliore la précision et diminue le rappel. Seulement quelques candidats sont alors générés (le rappel diminue) mais les candidats générés sont de bonne qualité (la précision augmente).

Les faibles résultats obtenus pour les candidats sommets labels peuvent être expliqués par le fait que les labels ne sont présents, la plupart du temps, que dans Agrovoc, et que les experts ont observé beaucoup d'erreurs sur les labels lors de l'évaluation. Pour améliorer les résultats sur les labels, nous pouvons tenir compte des préférences des sources pour un type spécifique d'élément ontologique : Agrovoc est par exemple une bonne source pour les labels.

Jusqu'à présent, nous avons utilisé 0,6 comme seuil pour obtenir nos résultats. C'est un seuil qui a été fixé empiriquement afin de comparer sur la même base les

fonctions de confiance. Nous avons également évalué nos résultats avec un seuil à 0,9. Nous obtenons alors les résultats présentés dans le tableau 5.

Tableau 5. Résultats de la troisième expérimentation :$trust_{degree}$ et seuil fixé à 0,9

Type de candidat	Précision	Rappel	F-Measure
Sommet individu	0,78	0,28	0,41
Arc $instanceOf$	0,81	0,39	0,53
Arc	0,93	0,32	0,47
Sommet label	0	0	0

En fixant un seuil à 0,9, nous observons une augmentation significative de la précision (en particulier pour les candidats arcs) alors que le rappel décroît. Ceci peut s'expliquer par le fait qu'il y a moins de candidats acceptés, mais ils sont plus vraisemblables. La précision des candidats sommets individus décroît parce qu'il y a certains candidats ayant un grand score de confiance qui ne sont pas validés par les experts. Cela signifie qu'il y a les mêmes erreurs dans les trois sources. Le rappel et la précision pour les labels sont à zéro parce que les labels proviennent principalement d'une seule source (Agrovoc). Donc les candidats labels obtiennent de faibles scores de confiance et sont éliminés par le filtrage. Une idée pour améliorer les résutats des candidats labels pourrait être de différencier les scores de confiance par type de candidat en fonction de la source. Par exemple, dans notre cas, si les candidats labels viennent d'Agrovoc, nous pouvons associer un bon score de confiance, contrairement aux candidats sommets individus de cette source.

Ces expérimentations ont pu mettre en évidence l'influence du seuil : plus le seuil est haut, plus les éléments ontologiques extraits sont de bonne qualité mais moins ils sont nombreux. En fonction de nos besoins, nous pouvons choisir un seuil bas ou, au contraire, un seuil élevé.

7. Conclusion et perspectives

Dans cet article, nous proposons une méthode de transformation de plusieurs systèmes d'organisation de connaissances en une base de connaissances, en nous fondant sur des patrons de conception d'ontologies ainsi que sur des calculs de scores de confiance. Notre méthode calcule un score de confiance pour chaque candidat extrait. Nous déterminons quelle formule de confiance est la plus adaptée à notre cas d'utilisation. Cette méthode aide la validation à la fin du processus parce que certains candidats peuvent être validés ou rejetés automatiquement, grâce à un filtrage par seuil.

Nous allons focaliser nos travaux futurs sur le filtrage des résultats afin de résoudre plusieurs problèmes que nous rencontrons. Nous devrons faire face au problème de contradictions entre candidats. Actuellement, tous les candidats sont considérés et peuvent être acceptés, même s'ils sont en conflit. Pour sélectionner les candidats qui ne sont pas en conflits, il serait possible d'utiliser la théorie de l'argumentation (Schneider *et al.*, 2013) associée au score de confiance. Une autre perspective est d'intégrer une valeur de qualité de chaque source à l'intérieur de la fonction de confiance.

Cela pourrait aider à promouvoir un type spécifique d'élément ontologique par source (par exemple promouvoir les labels d'Agrovoc). Nous voulons aussi travailler sur un autre sous-domaine que la taxonomie des plantes. Nous envisageons de travailler sur les attaques de bio-agreseurs utilisant le module sur les plantes cultivées [14] avec une base de données d'Arvalis [15]. *In fine*, le but de cette génération d'une base de connaissances est de pouvoir l'interfacer avec un système d'interrogation intelligente. Le projet SWIP (Pradel, 2013) semble parfaitement adapté à cette tâche puisqu'il permet l'interrogation d'une base de connaissances à partir de requêtes exprimées en langage naturel. Il serait particulièrement intéressant de proposer ce système pour interagir avec des agriculteurs et observer s'ils arrivent à obtenir les connaissances dont ils ont besoin.

Remerciements

Nous voulons particulièrement remercier les trois experts qui nous ont permis la validation de nos résultats : Franck Jabot d'Irstea Clermont-Ferrand, Jacques Le Gouis de l' INRA Clermont-Ferrand et Vincent Soulignac d'Irstea Clermont-Ferrand.

Bibliographie

Amarger F., Chanet J.-P., Haemmerlé O., Hernandez N., Roussey C. (2013). Etat de l'art : Extraction d'information à partir de thésaurus pour générer une ontologie. *INFormatique des Organisations et Systemes d'Information et de Decision (INFORSID), Paris, 29/05/2013-31/05/2013*, p. 29–44.

Amarger F., Chanet J.-P., Haemmerlé O., Hernandez N., Roussey C. (2014). SKOS Sources Transformations for Ontology Engineering: Agronomical Taxonomy Use Case. *Metadata and Semantics Research - 8th Research Conference, MTSR 2014, Karlsruhe, Germany, November 27-29, 2014. Proceedings*, p. 314–328.

Artz D., Gil Y. (2007, décembre). A survey of trust in computer science and the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, n° 2, p. 58–71.

Downey D., Etzioni O., Soderland S. (2005). A probabilistic model of redundancy in information extraction. *Proceedings of the 19th International Joint Conference on Artificial Intelligence, IJCAI 2005, Edinburgh, Scotland, 30/07/2005-05/08/2005*, p. 1034–1041.

Dragisic Z., Eckert K., Euzenat J., Faria D., Ferrara A., Granada R. *et al*. (2014). Results of the ontology alignment evaluation initiative 2014. *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.*, p. 61–104.

Euzenat J., Shvaiko P. (2013). *Ontology matching* (2nd éd.). Heidelberg (DE), Springer-Verlag, p. 520.

14. https://sites.google.com/site/agriontology/home/irstea/cultivatedplant
15. http://www.arvalis-infos.fr

Fernandez-Lopez M., Gomez-Perez A., Juristo N. (1997). Methontology: From ontological art towards ontological engineering. *Proceedings of the AAAI Spring Symposium on Ontological Engineering, March 24–25, 1997, Palo Alto, California*, p. 33–40.

Gangemi A., Presutti V. (2009). Ontology design patterns. In *Handbook on ontologies.* Springer-Verlag Berlin Heidelberg, p. 221–243.

Gil R., Martin-Bautista M. J. (2014). SMOL: a systemic methodology for ontology learning from heterogeneous sources. *Journal of Intelligent Information Systems*, vol. 42, n° 3, p. 415–455.

Hepp M., Bruijn J. de. (2007). GenTax: A generic methodology for deriving OWL and RDF-S ontologies from hierarchical classifications, thesauri, and inconsistent taxonomies. *The semantic web: research and applications, ESWC 2007, Innsbruck, Austria, June 3-7, 2007, Proceedings*, p. 129–144.

Jayawardene V., Sadiq S., Indulska M. (2013). *An analysis of data quality dimensions*. Rapport technique n° 2013-01. School of Information Technology and Electrical Engineering, The University of Queensland, p. 1–32.

Jiménez-Ruiz E., Grau B. C., Zhou Y., Horrocks I. (2012). Large-scale Interactive Ontology Matching: Algorithms and Implementation. *20th European Conference on Artificial Intelligence, ECAI 2012, Montpellier, France, August 27-31 2012*, vol. 242, p. 444–449.

Miles A., Bechhofer S. (2009). *SKOS simple knowledge organization system reference*. Working Draft. W3C.

Pearl J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.

Pradel C. (2013). *D'un langage de haut niveau à des requêtes graphes permettant d'interroger le web sémantique*. Thèse de doctorat, Université de Toulouse, Toulouse, France.

Presutti V., Blomqvist E., Daga E., Gangemi A. (2012). Pattern-based ontology design. In *Ontology engineering in a networked world.* Springer Science & Business Media, Berlin, Germany, p. 35–64.

Roussey C., Chanet J.-P., Cellier V., Amarger F. (2013). Agronomic taxon. *Proceedings of the 2nd International Workshop on Open Data, WOD 2013, Paris, France, June 3, 2013*, p. 5.

Schneider J., Groza T., Passant A. (2013). A review of argumentation for the social semantic web. *Semantic Web – Interoperability, Usability, Applicability, IOS Press Journal*, vol. 4, n° 2, p. 159–218.

Soergel D., Lauser B., Liang A., Fisseha F., Keizer J., Katz S. (2006). Reengineering thesauri for new applications: the AGROVOC example. *Journal of digital information, Article No. 257*, vol. 4, n° 4.

Suárez-Figueroa M. C., Gómez-Pérez A., Motta E., Gangemi A. (2012). *Ontology engineering in a networked world*. Springer Science & Business Media, Berlin, Germany.

Villazon-Terrazas B., Carmen Suarez-Figueroa M., Gomez-Perez A. (2010). A Pattern-Based Method for Re-Engineering Non-Ontological Resources into Ontologies. *International Journal on Semantic Web and Information Systems*, vol. 6, n° 4, p. 27-63.

Winkler W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. *Proceedings of the Section on Survey Research, American Statistical Association, Wachington, DC*, p. 354–359.