

Enrichir une ontologie pour améliorer la recherche d'information approximative

Valmi Dufour-Lussier, Jean Lieber,
Emmanuel Nauer, and Yannick Toussaint

LORIA (CNRS, INRIA, Nancy-Université)
BP 239, 54506 Vandœuvre-lès-Nancy, France
{prénom.nom}@loria.fr

Résumé : Une façon de traiter la recherche d'information approximative consiste à exploiter une ontologie pour généraliser la requête initiale. Cet article montre comment enrichir une ontologie pour rendre la généralisation plus fine. L'ontologie existante est enrichie en ajoutant automatiquement de nouveaux concepts qui vont affiner l'organisation initiale des concepts. Les nouveaux concepts introduits résultent d'un processus de fouille de données — l'analyse de concepts formels — sur de nouvelles propriétés des concepts collectées dans des données textuelles. Les concepts formels créés par la fouille sont intégrés à l'ontologie, permettant ainsi de généraliser une requête de façon plus fine qu'avant cet ajout.

Mots-clés : raffinement d'ontologie, analyse de concepts formels, recherche d'information approximative, raisonnement à partir de cas.

1 Introduction

Nous proposons dans cet article une approche pour mieux organiser les concepts d'une ontologie existante par un processus de fouille de données. De nouveaux concepts, créés par la fouille, sont intégrés à l'ontologie pour affiner l'organisation des concepts initiaux. Nous montrons comment l'ontologie raffinée rend plus fine la généralisation d'une requête d'un système de recherche d'information (RI) en vue d'une recherche approximative. Le cadre d'application est le système TAAABLE¹ (Badra *et al.*, 2009), un système de raisonnement à partir de cas (RÀPC) sur des recettes de cuisine. Le système TAAABLE recherche dans une base des recettes vérifiant les contraintes énoncées par l'utilisateur (présence/absence d'ingrédients). S'il en existe, elles sont proposées à l'utilisateur, sinon le système recherche des recettes satisfaisant *approximativement* les contraintes, et les adapte aux contraintes en proposant de substituer certains ingrédients par d'autres. La recherche de recettes similaires est guidée par une ontologie d'ingrédients qui permet de relâcher les contraintes de recherche en les généralisant. La

¹<http://taaable.fr>

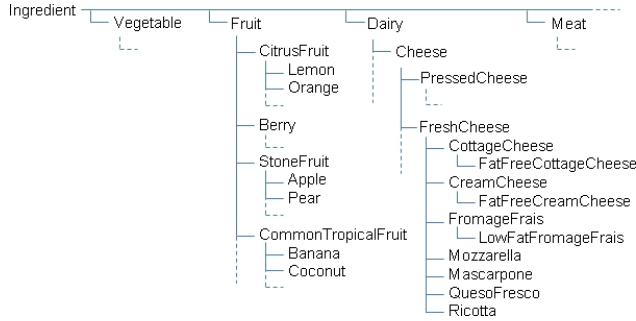


FIG. 1 – Un extrait de la hiérarchie des ingrédients.

relaxation est itérative et progressive : elle conduit à trouver la généralisation la plus spécifique de la requête pour laquelle il existe des recettes dans la base de recettes.

Les ontologies qui guident ce type de recherche approximative sont rarement établies spécifiquement pour le processus de RI lui-même : soit elles existent déjà et sont injectées dans le système (éventuellement avec certains ajustements), soit elles sont établies pour l’occasion, mais en tant que modélisation des connaissances consensuelles et partageables du domaine d’application (ce qui est le cas de la première version d’ontologie de TAAABLE). Sauf erreur, elles ne sont, en aucun cas, établies ou adaptées spécifiquement pour la recherche approximative. Aussi, lors de la généralisation d’un concept C_1 en un concept C , les concepts C_i subsumés par C peuvent être très *similaires* entre eux, ou très *différents*. Notre objectif est de mieux contrôler la généralisation en la rendant plus progressive. L’idée que nous défendons est que l’introduction de concepts intermédiaires dans l’ontologie rend cette généralisation plus fine. Un processus de fouille de données – l’analyse de concepts formels (ACF) – est employé pour dégager de nouveaux concepts qui vont enrichir l’ontologie initiale en affinant l’organisation des concepts. Pour ce faire, nous utilisons des sources d’informations complémentaires pour collecter de nouvelles propriétés sur les concepts de l’ontologie initiale.

L’organisation de ce document est la suivante : la section 2 présente le système TAAABLE, la section 3 donne un état de l’art sur les approches existantes pour la construction et le raffinement d’ontologie par ACF, la section 4 détaille le processus de raffinement de l’ontologie, la section 5 montre l’impact de l’ontologie raffinée dans TAAABLE.

2 Le système TAAABLE

Dans TAAABLE, la recherche de cas similaires est guidée par une ontologie du domaine. Cette ontologie est un ensemble de concepts atomiques organisé dans une hiérarchie dont le sommet est le concept `Ingredient`. Tous les recettes sont indexées par des instances de ce concept. La relation entre deux concepts A et B de cette ontologie tels que A est « en dessous » de B dans la hiérarchie est une relation de subsumption : l’ensemble des recettes indexées par A est inclus dans l’ensemble des recettes indexées par B. Par exemple, le concept `Ricotta` désigne les recettes dont un ingrédient est

de la ricotta (un fromage frais italien). *Riccotta* est subsumé par *FreshCheese*, lui-même subsumé par *Cheese*, lui-même subsumé par *Dairy*. Un extrait de la hiérarchie des ingrédients est donné en FIGURE 1.

TAAABLE implémente un raisonnement à partir de cas, dont l'étape de remémoration consistant à rechercher des cas satisfaisant *approximativement* une requête, se fait en généralisation celle-ci. L'adaptation se fait en proposant de substituer les ingrédients généralisés par les ingrédients souhaités par l'utilisateur. Faute de place, nous renvoyons à (Lieber, 2009) pour les explications détaillées du processus de raisonnement.

Dans le cadre du système TAAABLE, ce travail est motivé par le fait que certaines adaptations échouent : la recette n'est pas *cuisinable* car certaines actions, appliquées initialement à un ingrédient substitué, ne sont plus applicables à l'ingrédient substituant. Par exemple, lors d'une substitution de la *mozzarella* (fromage relativement ferme) par du *mascarpone* (fromage de consistance crémeuse), il ne sera plus possible de réaliser l'action *couper en tranches* sur l'ingrédient substituant, alors que cette action est courante sur l'ingrédient substitué. Pour réduire cet effet, nous avons travaillé sur l'enrichissement de la hiérarchie des ingrédients, à partir des actions culinaires qui leurs sont applicables. Ainsi, la nouvelle proximité des ingrédients dans la hiérarchie prendra en compte les actions qui leurs seront potentiellement applicables : la *mozzarella* et le *mascarpone* qui étaient très proches dans l'ontologie initiale seront plus éloignées eu égard aux autres *fromages frais* dont ils font partie.

Une requête Q du système TAAABLE est une conjonction de concepts atomiques de l'ontologie. À titre d'exemple, la requête « Je veux une recette avec de la tomate et du fromage frais » s'écrit $Q = \text{Tomato} \sqcap \text{FreshCheese}$. Les recettes sont représentées dans le même langage que celui des requêtes : une conjonction des concepts correspondant aux ingrédients qu'elles nécessitent. Par exemple, une recette avec de la ricotta, de la tomate, de l'huile d'olive, et du sel sera représentée par

$$R = \text{Ricotta} \sqcap \text{Tomato} \sqcap \text{OliveOil} \sqcap \text{Salt} \quad (1)$$

Étant donné une recette R et une requête Q , on dit que R résout Q si l'ensemble des recettes ayant la représentation R est contenu dans l'ensemble des recettes représentée par Q . Ainsi, la recette définie par (1) résout la requête $Q = \text{OliveOil} \sqcap \text{Tomato}$. Cela se calcule en testant si R est plus spécifique que Q : soient Q et R deux conjonctions de concepts atomiques, $R = \prod_i A_i$ est plus spécifique que $Q = \prod_j B_j$ si, pour tout j il existe au moins un i tel que A_i est subsumé par B_j dans l'ontologie.

Le principe de recherche approximative consiste à généraliser de façon minimale la requête, jusqu'à ce qu'au moins une recette résolve cette requête modifiée. On note $\Gamma_t(Q)$ la requête modifiée à l'instant (l'itération) t et Γ_0 est l'identité. Chaque généralisation Γ_t s'écrit comme une composition de substitutions $A \rightsquigarrow B$ telles que (A, B) est un arc de la hiérarchie représentant l'ontologie. Par ailleurs, cette composition doit être applicable sur Q : si $\Gamma = (A_p \rightsquigarrow B_p) \circ \dots \circ (A_2 \rightsquigarrow B_2) \circ (A_1 \rightsquigarrow B_1)$ alors A_1 est un élément de la conjonction Q , A_2 est un élément de la conjonction $(A_1 \rightsquigarrow B_1)(Q)$, etc. L'algorithme de parcours de l'espace des généralisations Γ de Q est un algorithme A^* paramétré par une fonction de coût. Il s'arrête quand on trouve au moins une recette R telle que R résout $\Gamma_t(Q)$. Par exemple, soit $Q = \text{Tomato} \sqcap \text{Mascarpone} \sqcap \text{Oil}$. Parmi les généralisations de cette requête, supposons que celle de moindre coût

soit $\Gamma = \text{Mascarpone} \rightsquigarrow \text{FreshCheese}$. Comme Ricotta est subsumé par FreshCheese dans l'ontologie, TAAABLE retrouve la recette R de (1). L'adaptation consiste à substituer Ricotta par Mascarpone (généralisation-spécialisation passant par FreshCheese). Nous renvoyons à (Lieber, 2009) pour les détails relatifs à la fonction de coût et à l'adaptation.

3 Construction d'ontologie par ACF

	FreshCheese	CottageCheese	FatFreeCottageCheese	CreamCheese	FatFreeCreamCheese	FromageFrais	LowFatFromageFrais	Mozzarella	Mascarpone	QuesoFresco	Ricotta	mashAble	meltable	beatAble	mixable	sliceable	cutAble	crumbleable	
CottageCheese	x	x										x	x						
FatFreeCottageCheese	x	x	x									x	x						
CreamCheese	x			x								x	x	x	x				
FatFreeCreamCheese	x			x	x							x	x	x	x				
FromageFrais	x					x						x	x	x					
LowFatFromageFrais	x					x	x					x	x	x					
Mozzarella	x							x								x	x		
Mascarpone	x								x					x	x	x			
QuesoFresco	x									x								x	x
Ricotta	x										x			x	x	x			

TAB. 1 – Un contexte formel de 10 objets décrits par 18 propriétés.

L'ACF est une approche pour l'analyse de données qui exploite la théorie des treillis. Un *contexte formel* est un triplet $\mathcal{K} = (G, M, I)$, où G est un ensemble d'objets, M un ensemble de propriétés et I la relation sur $G \times M$ exprimant qu'un objet possède une propriété (Ganter & Wille, 1999). TABLE 1 donne un exemple de contexte : G est un ensemble d'objets (qui sont des fromages frais plus spécifique que le concept FreshCheese), M un ensemble de propriétés relatives aux 6 fromages : les 11 premières propriétés viennent de la hiérarchie initiale, les 7 suivantes sont des actions qui leur sont potentiellement applicables : mashAble (peut être écrasé), sliceAble (peut être coupé en tranches), beatAble (peut être battu) cutAble (peut être coupé), etc.

Un *concept formel* est un couple (I, E) , où E est l'ensemble maximal d'objets (appelé *extension*) possédant toutes les propriétés de I , et I est l'ensemble maximal de propriétés (appelé *intension*) partagées par tous les objets de E . L'extension réduite (Reduced E, dans FIGURE 2) d'un concept C est l'ensemble des objets qui ne possèdent pas d'autres propriétés que celle contenues dans l'intension de C ; l'extension complète d'un concept C est l'union des extensions réduites des concepts formels subsumés par C . Par exemple, $(\{\text{FreshCheese}, \text{cutAble}\}, \{\text{QuesoFresco}, \text{Mozzarella}\})$ est un concept formel (cf. nœud numéroté 3, FIGURE 2).

L'ensemble $\mathcal{C}_{\mathcal{K}}$ de tous les concepts formels du contexte $\mathcal{K} = (G, M, I)$ est partiellement ordonné par l'inclusion des extensions. Cette relation dite de *spécialisation* entre concepts est notée $\leq_{\mathcal{K}}$. $\mathcal{L} = \langle \mathcal{C}_{\mathcal{K}}, \leq_{\mathcal{K}} \rangle$ est un treillis complet, appelé *treillis de concepts*. Le treillis \mathcal{L} peut être représenté par un diagramme de Hasse dans lequel les nœuds sont les concepts et les arêtes, les liens de spécialisation/généralisation. La FIGURE 2 illustre

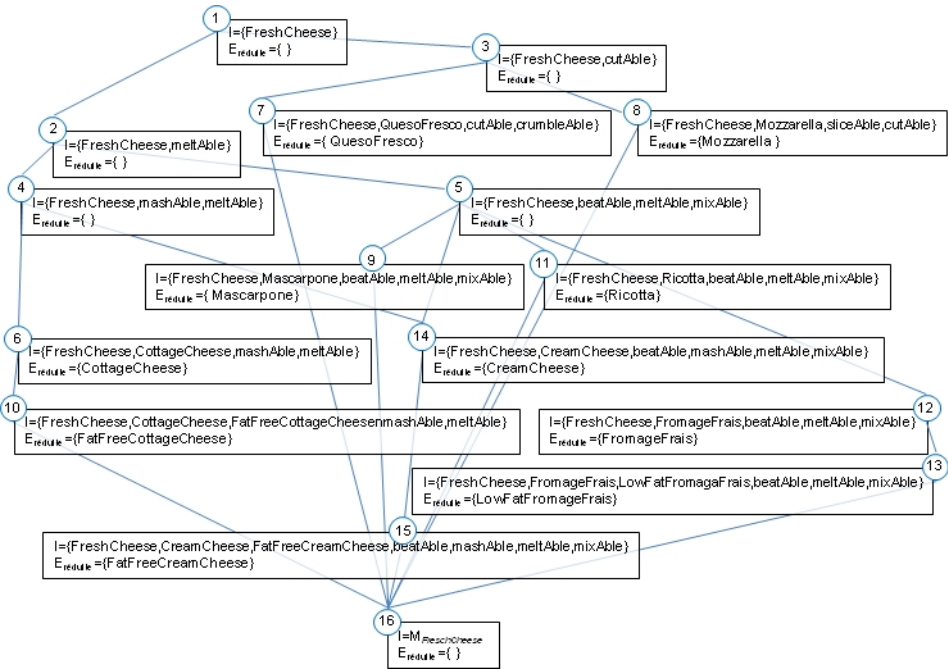


FIG. 2 – Le treillis de concepts correspondant au contexte formel donné en TABLE 1.

le treillis de concepts correspondant au contexte formel donné en TABLE 1.

De nombreux algorithmes ont été proposés pour la construction de treillis de concepts (cf. (Ganter & Wille, 1999)). Pour notre application, nous utilisons la plate-forme CORON (<http://coron.loria.fr/>), qui implémente une large collection d’algorithmes pour la fouille de données symboliques, incluant des algorithmes de construction de treillis de concepts (Szathmary & Napoli, 2005).

Plusieurs travaux reposent sur l’ACF pour construire des ontologies. L’approche de Cimiano (Cimiano *et al.*, 2005) vise à construire une hiérarchie de concepts à partir de textes dans le domaine du tourisme. Les objets (hôtels, voitures, vélos, excursions...) sont caractérisés par les actions qui peuvent leur être appliquées (pouvant être réservé, pouvant être conduit, pouvant être loué...). Afin de réduire la taille du treillis, une première classification numérique est appliquée avant la construction du contexte formel.

Stumme *et al.* (Stumme & Maedche, 2001) discute de la fusion de deux ontologies. Les classes des ontologies sont repérées dans les textes, et un contexte pour chacune des ontologies est créé dans lequel les classes sont caractérisées par les documents dans lesquels elles apparaissent. Ces contextes sont fusionnés pour regrouper en fonction de leur citation dans les textes les différentes classes de chacune des deux ontologies.

(Bendaoud *et al.*, 2008) présente l’ACF comme un cadre unifié pour la construction et l’enrichissement d’ontologies. Il part du constat que, selon les types de ressources textuelles considérées, les informations qui en sont extraites diffèrent. Ainsi, un article

scientifique permettra d'identifier des objets et de les associer à des propriétés comme le fait (Cimiano *et al.*, 2005). En revanche, une ontologie organise les concepts d'un domaine en hiérarchie. Cette information d'organisation par spécialisation/généralisation est peu présente dans les articles scientifiques. Dans Bendaoud *et al.* (Bendaoud *et al.*, 2008), ces deux types d'information sont traités séparément puis sont combinés pour ne produire qu'un seul et même treillis. Nous utilisons cette méthode pour construire notre ontologie sur les ingrédients en cuisine, exploitant les recettes de cuisine pour caractériser les ingrédients par la façon dont ils sont préparés et exploitant l'ontologie initiale pour introduire les relations de spécialisation/généralisation.

4 Processus de raffinement de l'ontologie

Le processus de raffinement vise à introduire des concepts intermédiaires dans la hiérarchie initiale du système. Pour ce faire, les concepts de la hiérarchie initiale sont caractérisés par des propriétés complémentaires. Ces propriétés permettent de créer un contexte formel qui servira à construire un treillis de concepts par utilisation de l'ACF. Les concepts formels du treillis sont insérés dans la hiérarchie initiale pour enrichir son organisation. Nous détaillons maintenant ces étapes.

4.1 Présentation de la hiérarchie des ingrédients

La hiérarchie initiale des ingrédients comprend actuellement 1450 concepts ; une illustration est donnée en FIGURE 1. La partie *haute* de la hiérarchie contient des concepts très généraux (Vegetable, Fruit, Dairy, Meat, etc.), la partie *basse*, des concepts qui sont des ingrédients *concrets*, présents dans les recettes.

Dans cette hiérarchie, la *ressemblance* entre *ingrédients-frères* (sous-concepts directs d'un même concept) n'est pas contrôlée. Par exemple, le mascarpone est aussi proche de la mozzarella que du fromage blanc (FromageFrais). Lors d'une généralisation de requête pour la remémoration, si Mascarpone est généralisé en FreshCheese, l'ensemble des cas remémorés contiendra indifféremment des recettes avec de la mozzarella et de des recettes avec du fromage blanc. C'est pourquoi, nous voulons affiner l'organisation de la hiérarchie pour regrouper entre eux les concepts initiaux se ressemblant (i.e. concepts qui auraient au moins une propriété en commun). La généralisation d'un concept C à un concept parent garantirait que les concepts-frères de C posséderait au moins les mêmes propriétés que C . La généralisation serait alors plus fine.

4.2 Raffinement local

Nous appelons *raffinement local* le fait de raffiner une sous-partie limitée de l'ontologie et non l'ontologie complète : par exemple, la sous-partie relative aux FreshCheese. Cette approche a été mise en œuvre pour plusieurs raisons :

- le fait de se focaliser sur une sous-partie de la hiérarchie (i.e. un sous-ensemble de concepts) permet de limiter la taille du contexte en nombre d'objets et en nombre de propriétés associés aux objets. Un contexte limité produit un treillis de moindre complexité : celui-ci est visualisable et interprétable.

- nous ne souhaitons pas construire de concepts qui regroupent des ingrédients n’appartenant pas à une même sous-partie de hiérarchie. Par exemple, nous ne souhaitons pas regrouper les kiwis et les oignons, du simple fait qu’on puisse leur appliquer une action culinaire commune, comme *peler*.

L’idée est donc de se focaliser sur le raffinement de sous-parties (*basses*) de la hiérarchie, en conservant les structurations du *haut*. Les différents raffinements locaux viendront se greffer sur l’ontologie initiale (cf. 4.4).

Formellement, nous notons R le concept racine de la sous-partie de hiérarchie à raffiner. Le contexte $\mathcal{K}_R = (G_R, M_R, I_R)$ dénote le contexte binaire constitué de :

- l’ensemble des objets (des ingrédients) plus spécifiques que R : $G_R = \{x/R \sqsubseteq x\}$,
- l’ensemble de propriétés (des actions culinaires) M_R sélectionnées pour discriminer entre eux les objets de G_R ,
- la relation I_R qui explicite qu’un ingrédient $i \in G_R$ peut subir l’action $a \in M_R$.

Les sections suivantes illustrent le processus complet appliqué aux fromages frais.

4.3 Construction du contexte formel pour le raffinement.

Le raffinement d’ontologie consiste à modifier une ontologie existante. Pour ce faire, il faut garder des propriétés liées à l’ontologie initiale, et introduire de nouvelles propriétés pour mieux positionner les concepts (ingrédients) les uns par rapport aux autres. Regrouper le fromage blanc avec le mascarpone et éloigner ces 2 fromages frais de la mozzarella est un résultat attendu. Il s’agit ici de construire un contexte formel à partir des propriétés provenant de l’ontologie (dans lequel chaque objet-ingrédient est relié à ses concepts plus génériques) et des propriétés intrinsèques aux objets-ingrédients. Nous avons choisi de collecter des propriétés relatives aux actions culinaires que les ingrédients peuvent subir : par exemple, le fromage blanc et le mascarpone peuvent être battus tandis que la mozzarella non ; par contre, la mozzarella peut être coupée en tranches, ce qui n’est pas le cas des fromages de consistance crémeuse.

Les propriétés des ingrédients ont été extraits à partir de 73795 recettes de cuisine en provenance de la base de données *Recipe Source*². L’extraction des propriétés est actuellement réalisée depuis la liste des ingrédients qui inclut certaines transformations (exemple : *2 oz Queso fresco, crumbled, 1 cup chopped onion*) et du texte de la préparation (exemple : *Slice the mozzarella.*). Les couples (ingrédient, action culinaire) sont extraits de la liste des ingrédients par la reconnaissance dans le texte de termes présents dans des bases terminologiques (d’ingrédients et d’actions culinaires). Dans le texte de la préparation, l’extraction, discutée dans (Dufour-Lussier *et al.*, 2010), repose d’une part sur des méthodes classiques d’analyse morpho-syntaxique, et d’autre part sur un type de représentation du discours spécifiquement conçu pour les textes dits procéduraux (textes d’instructions, incluant les recettes) et particulièrement adapté à la résolution de certains phénomènes anaphoriques compliqués trouvés dans ces textes.

Les couples (ingrédient, action culinaire) extraits servent à construire le contexte formel. Nous contrôlons l’ensemble des propriétés M_R de \mathcal{K}_R en ne gardant que certaines actions culinaires. Pour $R = FreshCheese$ et l’illustration dans cet article, le contexte formel $\mathcal{K}_{FreshCheese}$ est celui donné en TABLE 1.

²<http://www.recipe-source.com/>

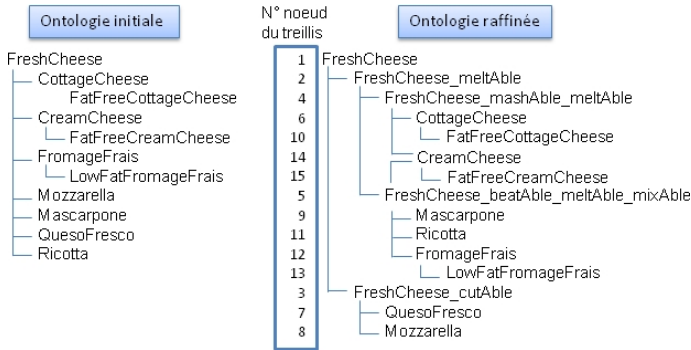


FIG. 3 – Les fromages frais dans l’ontologie initiale et dans l’ontologie raffinée.

4.4 Intégration d’un treillis local dans la hiérarchie initiale

Les treillis locaux enrichissent la hiérarchie initiale. Le processus d’intégration de chacun d’eux dans la hiérarchie initiale consiste à substituer la sous-partie de hiérarchie enracinée en R par le treillis local construit à partir de \mathcal{K}_R , modifié comme suit :

- le concept le plus spécifique est supprimé car son extension est vide (nœud numéro 16 de FIGURE 2) ;
- les concepts dont l’extension réduite ne contient qu’un élément sont, par construction du contexte formel, des concepts qui représentent les objets de M_R , l’ensemble des concepts de l’ontologie initiale (nœuds 6, 7, 8, 9, 10, 11, 12, 13, 14, 15). Ces concepts formels produisent dans l’ontologie un concept dénommé par l’unique élément de l’extension réduite, qui désigne l’objet lui-même. Par exemple, le nœud 8 du treillis produira le concept `Mozzarella` dans l’ontologie raffinée.
- les concepts formels dont l’extension réduite est vide sont les concepts structurants que l’on cherchait à construire. Chacun d’eux donne naissance à un concept de l’ontologie, dénommé par une concaténation des propriétés de l’intension (nœuds 1, 2, 3, 4, 5). Par exemple, le nœud 5 du treillis donnera naissance au concept `FreshCheese_beatAble_meltAble_mixAble` dans l’ontologie.

La partie de l’ontologie raffinée sur les fromages frais apparaît en FIGURE 3, à droite.

5 Apport de l’ontologie raffinée dans TAAABLE

Nous illustrons sur un exemple l’apport du raffinement de l’ontologie pour la RI. Pour juger cela, nous avons instancié deux systèmes TAAABLE : $TAAABLE_0$ utilisant l’ontologie initiale et $TAAABLE_1$ l’ontologie raffinée. FIGURE 4 donne une illustration de l’interrogation des systèmes sur des recettes avec de la tomate et du mascarpone. Comme dans la base de cas, aucune recette ne contient de la tomate et de la mozzarella, les deux systèmes déclenchent la recherche de cas similaires.

$TAAABLE_0$ arrête la recherche après la généralisation $Mascarpone \rightsquigarrow FreshCheese$. Pour $TAAABLE_1$, la recherche approximative de recettes est raffinée et la géné-

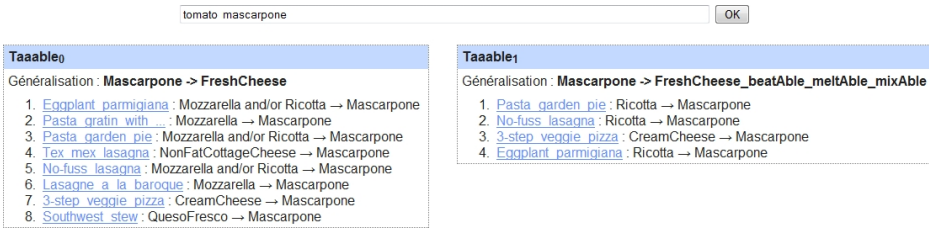


FIG. 4 – Réponses données par les systèmes TAAABLE₀ et TAAABLE₁ sur la requête : tomato \sqcap mascarpone.

ralisation s’arrête à un concept plus spécifique, à savoir `FreshCheese_beatAble_meltAble_mixAble`, correspondant au concept des fromages frais qu’on peut battre, mélanger et mixer, actions également applicables au mascarpone.

Les substitutions proposées par TAAABLE₀ sont plus variées que celles effectuées dans TAAABLE₁ : dans TAAABLE₁, le mascarpone vient remplacer de la ricotta ou du *cream cheese* (un fromage crémeux à tartiner), alors que dans TAAABLE₀, le mascarpone peut se substituer, en plus, à de la mozzarella, du *non-fat cottage cheese* ou encore du *queso fresco*, des fromages frais d’une moindre *ressemblance*.

Une analyse plus profonde des substitutions, recette par recette, permet de juger du succès ou de l’échec de l’adaptation. Dans les réponses retournées par TAAABLE₀, il y a 3 cas d’échecs nets, pour les recettes 2, 6 et 8, dans lesquelles il faudrait respectivement râper, couper en tranches et émietter le mascarpone. Trois *demi*-échecs sont également à considérer dans les cas où TAAABLE propose un choix dans l’ingrédient à substituer. Ainsi, l’adaptation de la recette 3 est un succès si on choisit de remplacer la ricotta par le mascarpone, mais est un échec si on remplace la mozzarella, car cela implique que le mascarpone doit être mis en lambeaux. Il en est de même pour la recette 5 et 1, pour lesquels une des substitutions proposées n’est pas possible.

Au final, l’ontologie raffinée améliore significativement le résultat global du système puisque toutes les adaptations de TAAABLE₀ qui échouaient ne sont plus proposés par TAAABLE₁. Cependant, une des recettes pour laquelle l’adaptation réussissait dans TAAABLE₀ (*Tex-mex lasagna*) n’est plus proposé dans TAAABLE₁, en raison du nouvel éloignement de `NonFatCottageCheese` par rapport à `Mascarpone` dans l’ontologie de TAAABLE₁. Ceci est inhérent à la condition d’arrêt de l’algorithme de parcours de l’espace des généralisations de TAAABLE. Ce problème est résolu par la possibilité de demander, dans l’interface du système, de relancer la recherche des recettes similaires, au-delà de la généralisation courante, au risque, à nouveau, d’introduire des adaptations impossibles.

6 Conclusion

Nous avons montré, dans cet article, comment raffiner une ontologie du domaine pour une recherche d’information approximative, comme dans le cas d’un processus de re-

mémorisation d'un système de RÀPC, et comment ce raffinement améliore le résultat de la recherche. Notre approche permet de remémorer plus progressivement les cas et, dans le cadre de TAAABLE, les cas d'adaptation problématiques sont éliminés. Ce travail est transposable dans tout système de RI qui utilise une hiérarchie pour accéder aux données. pour affiner des recherches approximatives, ou encore établir un classement plus fin des réponses à partir de distances prenant en compte l'organisation hiérarchique.

Certains points comme, par exemple, la façon de choisir les propriétés à prendre en compte pour le raffinement, nécessitent d'être étudiés et améliorés. Choisir, a priori, des propriétés (actions) qui discriminent entre eux les ingrédients restreint certes les risques de regroupements indésirables. Cependant, comme les contextes d'utilisation d'un même ingrédient peuvent changer, il ne serait pas impossible, étant donné un ensemble d'actions A_1 , qu'un ingrédient I soit plus proche d'un ingrédient I_1 et qu'étant donné un autre ensemble d'actions $A_2 \neq A_1$, I soit plus proche de I_2 (avec $I_1 \neq I_2$). Ainsi, dans un contexte, la ricotta pourrait être proche du mascarpone et dans un autre plus proche du cottage cheese.

Références

- BADRA F., COJAN J., CORDIER A., LIEBER J., MEILENDER T., MILLE A., MOLLI P., NAUER E., NAPOLI A., SKAF-MOLLI H. & TOUSSAINT Y. (2009). Knowledge Acquisition and Discovery for the Textual Case-Based Cooking system WikiTaaable. In *Workshops of the 8th International Conference on Case-Based Reasoning*.
- BENDAOU D., NAPOLI A. & TOUSSAINT Y. (2008). Formal Concept Analysis : A unified framework for building and refining ontologies. In ALDO GANGEMI & JÉRÔME EUZENAT, Eds., *16th International Conference on Knowledge Engineering and Knowledge Management - EKAW 2008*, volume 5268, p. 156–171, Acitrezza, Catania Italie : Springer Berlin / Heidelberg.
- CIMIANO P., HOTHO A. & STAAB S. (2005). Learning concept hierarchies from text corpora using formal concept analysis. In *Journal of Artificial Intelligence Research (JAIR'05)*, volume Volume 24, p. 305–339 : AAAI Press.
- DUFOUR-LUSSIER V., LIEBER J., NAUER E. & TOUSSAINT Y. (2010). Text adaptation using formal concept analysis. In *Case-Based Reasoning Research and Development (proceedings of ICCBR-2010, to be published)*, Lecture Notes in Artificial Intelligence : Springer.
- GANTER B. & WILLE R. (1999). *Formal Concept Analysis : Mathematical Foundations*. Berlin : Springer.
- LIEBER J. (2009). Le moteur de raisonnement à partir de cas de WIKITAAABLE. In *Actes du 17^{ème} atelier raisonnement à partir de cas (RàPC-09)*.
- STUMME G. & MAEDCHE A. (2001). Fca-merge : Bottom-up merging of ontologies. In *17th International Joint Conferences on Artificial Intelligence (IJCAI'01)*, p. 225–234, San Francisco, CA : Morgan Kaufmann Publishers, Inc.
- SZATHMARY L. & NAPOLI A. (2005). CORON : A Framework for Levelwise Itemset Mining Algorithms. *Supplementary Proceedings of The Third International Conference on Formal Concept Analysis (ICFCA '05)*, Lens, France, p. 110–113.