Optimisation de l'extraction de l'alignement des ontologies avec la contrainte de différence

Moussa Benaissa, Yahia Lebbah

Université d'Oran.Faculté des sciences. Laboratoire LITIO
moussabenaissa@yahoo.fr, ylebbah@yahoo.fr

Résumé. Dans ce papier, nous proposons une approche basée sur la programmation par contraintes pour aborder efficacement le problème de l'alignement des ontologies, et plus particulièrement l'extraction des correspondances à partir des mesures de similarités. La complexité de ce problème est accentuée dans les applications à caractère dynamique où l'aspect performance est capital. Plus précisément, nous exploitons la contrainte globale de différence développée dans le domaine de la programmation par contraintes pour extraire un alignement total et injectif. Nous montrons que cette approche est efficace et se prête à une mise en œuvre à la fois interactive et automatique.

1 Introduction

Si beaucoup de recherche a été dédiée aux calculs des similarités et à leur affinement, peu d'attention a été accordée au problème de l'extraction des alignements [Meilicke et Stuckenshmidt (2007)]. D'autre part, les méthodes d'extraction des alignements existantes peuvent être classées en deux catégories [Euzénat et valtchev (2004)] : les méthodes interactives et les méthodes automatiques. Il a été constaté [Shvaiko et Euzénat (2008)] que l'approche interactive est meilleure que l'approche automatique dans les applications traditionnelles où de larges ensembles de données sont mis en jeu. En outre les alignements à extraire doivent, dans certaines applications telles que la transcription automatique des ontologies, vérifier certaines propriétés telles que la complétude et l'injectivité. Sur un autre plan, un problème de performance se pose dans les applications industrielles à caractère dynamique où l'utilisateur ne dispose pas de beaucoup de temps pour obtenir une réponse du système [Shvaiko et Euzénat (2008)]. Ainsi il est nécessaire de disposer d'un schéma algorithmique efficace pour optimiser l'extraction de l'alignement des ontologies et qui permet de capter les préférences des utilisateurs quand le processus interactif est déclenché par l'utilisateur. Ces deux critères (efficacité et interactivité) sont capitaux pour le problème de l'alignement des ontologies.

L'idée de ce papier consiste à exploiter la contrainte globale de différence [Régin (1994)] et son algorithme développés dans le domaine de la programmation par contraintes, qui sont bien adaptés pour aborder l'extraction de l'alignement des ontologies. L'algorithme de filtrage permet d'éliminer les concepts qui ne peuvent pas appartenir à un alignement.

Nous commençons par introduire un exemple pour présenter l'approche. La section 3 présente des notions préliminaires sur les ontologies et la PPC. Dans la section 4 nous introduisons notre contribution. Nous finissons ce papier avec une conclusion et des perspectives.

2 Exemple illustratif

Soient O et O' deux ontologies. O contient les concepts $\{C_1, C_2, C_3, C_4, C_5, C_6\}$ et O' contient les concepts $\{C'_1, C'_2, C'_3, C'_4, C'_5, C'_6, C'_7\}$. Une certaine technique calculant les similarités entre les concepts de O et O' a produit la matrice de similarités suivante.

O O'	C'_1	C'_2	C'_3	C'_4	C'_5	C'_6	C'_7
C_1	0.65	0.60	0.22	0.11	0.33	0.44	0.35
C_2	0.15	0.80	0.75	0.39	0.30	0.12	0.27
C_3	0.90	0.44	0.56	0.37	0.24	0.31	0.26
C_4	0.23	0.68	0.32	0.87	0.44	0.26	0.25
C_5	0.17	0.18	0.70	0.60	0.50	0.82	0.32
C_6	0.36	0.37	0.39	0.41	0.29	0.92	0.83

Nous supposons aussi que toutes les valeurs de similarités doivent être supérieures ou égales à un certain seuil $s = 0.5$. Un filtrage de S sur le seuil s permet d'éliminer toutes les valeurs de la matrice inférieures à 0.5. Le problème consiste alors à extraire de façon interactive un alignement total et injectif entre les concepts de O et O'. Nous suggérons l'exploitation de la contrainte de différence et l'algorithme qui établit sa arc-consistance pour supprimer interactivement, toutes les correspondances qui ne peuvent pas appartenir à l'alignement cible avec les propriétés requises. Dans l'exemple ci-dessus les correspondances sont supprimées en exploitant l'algorithme basé sur le couplage [Régin (1994)]. Ceci permet d'améliorer la performance du système et de réduire la charge de l'utilisateur. Après l'application de ces suppressions et des choix de l'utilisateur, nous obtenons l'alignement suivant : $\{(C_1, C'_1), (C_2, C'_2), (C_3, C'_3), (C_4, C'_4), (C_5, C'_6), (C_6, C'_7)\}$. Dans la section 4, nous donnerons plus de détails sur le processus de suppression et sur l'approche globale.

3 Préliminaires

3.1 Alignement des ontologies

Pour plus de détails sur le contenu de cette section, nous renvoyons le lecteur à la référence [Euzenat et Shvaiko (2007)]. Soient O et O' deux ontologies. Une correspondance M entre O et O' est un quintuple $< id, e, e', R, n >$ où : id est un identificateur unique de la correspondance M ; e et e' sont des entités de O et O' ; R est une relation sémantique entre e et e' ; n est une mesure de confiance. L'alignement peut être défini comme un ensemble de correspondances. Le Processus d'alignement reçoit en entrée deux ontologies O et O' et produit en sortie un alignement A entre les entités de O et O'. Un alignement A est dit complet ou total pour l'ontologie O (resp. O') si toute entité de O (resp. de O') a une entité correspondante de O'(resp. de O). Un alignement A est dit injectif de O à O' si et seulement si pour toutes correspondances M et M', avec $M =< id_M, e_1, e'_1, R, n >\in A$ et $M' =< id_{M'}, e_2, e'_2, R, n >\in A$, nous avons $e_1 \neq e_2 \Rightarrow e'_1 \neq e'_2$. Globalement nous pouvons distinguer deux approches pour l'extraction des correspondances entre les entités des ontologies : l'approche interactive, et l'approche automatique. Les algorithmes d'extraction des correspondances entre les entités des ontologies dépendent des propriétés de l'alignement à extraire. Les méthodes basées sur des optimisations locales consistent à extraire l'alignement final en itérant sur les correspondances formant l'alignement initial, en maximisant localement la similarité à chaque paire d'entités. Les méthodes basées sur une optimisation globale, contrairement aux méthodes locales, consistent à optimiser un critère global.

3.2 La programmation par contraintes

Dans le cadre de la PPC est né un cadre restreint appelé Problèmes de Satisfaction de Contraintes (CSPs) que nous définissons ci-dessous et nous recommandons les références spécialisées suivantes pour plus de détails [Apt (2003); Macworth (1977); Fages (1996)]. Les CSPs forment un cadre formel simple pour la résolution des problèmes combinatoires. Un CSP P est un triplet $\langle X, D, C\rangle$, où : X un ensemble de n variables $x_1, \ldots, x_n$; D un ensemble de domaines finis $D_1, \ldots, D_n$ où D_i est le domaine de la variable x_i ; $C = \{C_1, \ldots, C_m\}$ un ensemble de contraintes sur les variables. Une contrainte C est définie entre un ensemble de variables $X(C) = \{x_i, \ldots, x_k\}$, comme un sous-ensemble du produit cartésien $D_i \times \ldots \times D_k$. On note $D(X(C)) = \cup_{x \in X(C)} D_x$. Une solution est une affectation de valeurs à toutes les variables telles que toutes les contraintes soient vérifiées.

3.2.1 La consistance d'arc et la contrainte de différence

Soit P un CSP, C une contrainte, x une variable appartenant à l'ensemble $X(C)$ et v une valeur qui appartient au domaine de x. Nous appelons support pour l'affectation $x = v$ pour C toute extension de cette affectation à toutes les variables de $X(C)$ qui vérifie C. La valeur v est dite arc-consistante pour la contrainte C si elle a un support pour C. Dans le cas contraire elle n'est pas arc-consistante. C est arc-consistante si et seulement si toutes les valeurs de toutes les variables de C sont arc-consistantes pour C. D_x est arc-consistant si et seulement si pour chaque contrainte C, toute valeur v appartenant à D_x est arc-consistante pour C. P est arc-consistant si et seulement si tous ses domaines sont arc-consistants. La contrainte de différence Alldiff [Régin (1994)] impose que toutes les valeurs des variables du même tuple soient différentes. L'algorithme établissant l'arc-consistance de Alldiff en un temps polynomial, a été développé par [Régin (1994)] et est basé sur la théorie des couplages. La complexité de cet algorithme est $O(p^2d^2)$, où $p = Card(X(C))$, $d = max_{x \in X(C)} Card(D_x)$.

4 Un modèle CSP pour l'extraction de l'alignement

4.1 Le modèle de l'alignement

Soient O et O' deux ontologies et A l'alignement à extraire entre les concepts de O et O' (L'approche est présentée ici pour les concepts mais elle reste valable pour les rôles et les instances). Nous supposons que A doit être total et injectif. On note e et e' les concepts de O et O' respectivement. Le problème de l'extraction de A peut être modélisé avec le CSP suivant : $P = (X, D, C)$ où :

- X un ensemble de n variables $x_1, ..., x_n$ où x_i correspond au concept e_i de O.
- D un ensemble de domaines finis $\{D_1, ..., D_n\}$ où D_i désigne l'ensemble des valeurs possibles de x_i et est défini par $D_i = \{e'_j \in O' / conf\langle id, e, e', R, n\rangle = n \geq s\}$. D_i est obtenu à partir de la matrice de similarités S en retenant seulement les concepts e'_i de O' qui correspondent au concept e_i et qui ont un niveau de similarité supérieur ou égal à un seuil donné s.
- $C = \{Alldiff(x_1, ..., x_n)\}$

Proposition 1 *L'ensemble de tous les alignements totaux et injectifs est égal à l'ensemble des CSP solutions de la contrainte de différence.*

Preuve. Soit S une solution de la contrainte de différence C. Donc S couvre tous les concepts de l'ontologie O. Par conséquent l'alignement A est total. D'autre part les valeurs affectées aux variables apparaissent au plus une fois, donc l'alignement A est injectif. Inversement, soit A un alignement total et injectif. Puisque A est total, chaque variable est affectée d'une valeur et puisque A est injectif alors la même valeur ne peut pas être affectée à deux variables distinctes. □

Exemple. Dans l'exemple de la section 2, le CSP qui modélise le problème de l'extraction de l'alignement est le suivant :

- $X = \{C_1, C_2, C_3, C_4, C_5, C_6\}$
- $D = D_1 \bigcup D_2 \bigcup D_3 \bigcup D_4 \bigcup D_5 \bigcup D_6$ avec : $D_1 = \{C'_1, C'_2\}$, $D_2 = \{C'_2, C'_3\}$, $D_3 = \{C'_1, C'_3\}$, $D_4 = \{C'_2, C'_4\}$, $D_5 = \{C'_3, C'_4, C'_5, C'_6\}$, $D_6 = \{C'_6, C'_7\}$,
  $D = \{C'_1, C'_2, C'_3, C'_4, C'_5, C'_6, C'_7\}$.
- $C = \{Alldiff(C_1, C_2, C_3, C_4, C_5, C_6)\}$

4.2 Extraction de l'alignement

L'extraction peut être effectuée selon deux approches : l'approche interactive et l'approche automatique.

Approche interactive. Nous proposons l'algorithme 1.

Algorithm 1 Approche interactive pour l'extraction de l'alignement

1: {On note O et O' les deux ontologies à aligner, A_I l'alignement initial, A_R l'alignement résultat et C une correspondance.}
2: Filtrer toutes les valeurs des domaines qui sont inconsistantes avec la contrainte Alldiff.
3: **if** il existe un domaine D_i tel que $D_i = \emptyset$ **then**
4: **return** "il n'existe pas d'alignement total.".
5: **end if**
6: $A_R = \phi$
7: **repeat**
8: Proposer à l'utilisateur de sélectionner une correspondance. Soit $G =< id_G, e_1, e'_m, n, R >$ cette correspondance.
9: $A_R = A_R \cup \{G\}$
10: Instancier le domaine de e_1 comme suit $D_{e_1} = \{e'_m\}$
11: Filtrer toutes les valeurs du D_i qui sont inconsistantes avec la contrainte Alldiff.
12: **until** (toutes les variables sont affectées)
13: **return** "Un alignement total et injectif est trouvé.".

La correction de l'algorithme peut être facilement prouvée. En effet, il consiste à appeler itérativement l'algorithme qui établit la arc-consistance de la contrainte de différence pour éliminer toutes les correspondances qui ne peuvent pas faire partie d'un alignement total et injectif et demande à l'utilisateur de sélectionner une correspondance. De ce fait et conformément à l'algorithme de filtrage, l'alignement produit est nécessairement total et injectif. Nous remarquons aussi que la boucle termine toujours avec un alignement total et injectif étant donné que

l'algorithme de filtrage de la contrainte AllDiff assure la consistance des entités restantes. On peut aussi démontrer que la terminaison de l'algorithme est assurée, car le filtrage est global.

L'approche automatique. Dans cette approche, l'alignement est extrait sans l'intervention de l'utilisateur. L'algorithme qui établit la arc-consistance de la contrainte Alldiff est combiné avec une méthode d'optimisation qui maximise la similarité globale entre les concepts de O et O'. Ceci peut être réalisé simplement avec la procédure à deux étapes suivante : (1) Filtrer les domaines avec l'algorithme de Alldiff ; (2) Enumérer les couplages existants. Cette procédure peut être améliorée en adoptant dans la seconde étape des schémas algorithmiques plus raffinés. L'algorithme de la méthode hongroise peut être appliqué si le nombre de concepts dans O est le même que dans O', et la fonction objectif reste la maximisation de la similarité. Toutefois si la fonction objectif est plus complexe nous pouvons adopter un schéma algorithmique plus général (branch and bound).

4.3 Illustration de l'approche

Considérons l'exemple de la section 2 et supposons que nous voulons extraire interactivement un alignement total et injectif entre les concepts de O et O'. Le CSP correspondant est donné dans la section 4.1. Nous commençons par filtrer tous les domaines. D'après l'algorithme de la arc-consistance, le concept C'_2 est supprimé du domaine D_4 et les concepts C'_3 et C'_4 sont supprimés du domaine D_5. Les nouveaux domaines sont : $D_1 = \{C'_1, C'_2\}$, $D_2 = \{C'_2, C'_3\}$, $D_3 = \{C'_1, C'_3\}$, $D_4 = \{C'_4\}$, $D_5 = \{C'_5, C'_6\}$, $D_6 = \{C'_6, C'_7\}$.

On note qu'il n'existe pas de domaine vide. Ceci assure la consistance de la contrainte. On continue en appliquant les deux itérations suivantes. A la première itération, on demande à l'utilisateur de choisir une correspondance (C_5, C'_6). Les nouveaux domaines sont : $D_1 = \{C'_1, C'_2\}$, $D_2 = \{C'_2, C'_3\}$, $D_3 = \{C'_1, C'_3\}$, $D_4 = \{C'_4\}$, $D_5 = \{C'_6\}$, $D_6 = \{C'_6, C'_7\}$. Ces domaines sont filtrés. Le concept C'_6 est supprimé du domaine D_6. Les nouveaux domaines sont : $D_1 = \{C'_1, C'_2\}$, $D_2 = \{C'_2, C'_3\}$, $D_3 = \{C'_1, C'_3\}$, $D_4 = \{C'_4\}$, $D_5 = \{C'_6\}$, $D_6 = \{C'_7\}$. Dans la seconde itération, on demande à l'utilisateur de choisir une correspondance (C_1, C'_1). Les nouveaux domaines sont : $D_1 = \{C'_1\}$, $D_2 = \{C'_2, C'_3\}$, $D_3 = \{C'_1, C'_3\}$, $D_4 = \{C'_4\}$, $D_5 = \{C'_6\}$, $D_6 = \{C'_7\}$. Ces domaines sont filtrés. Le concept C'_3 est supprimé du domaine D_2 et le concept C'_1 est supprimé du domaine D_3. Les nouveaux domaines sont : $D_1 = \{C'_1\}$, $D_2 = \{C'_2\}$, $D_3 = \{C'_3\}$, $D_4 = \{C'_4\}$, $D_5 = \{C'_6\}$, $D_6 = \{C'_7\}$. On note que toutes les variables sont affectées et l'algorithme termine. L'alignement résultat est le suivant $\{(C_1, C'_1)$,(C_2, C'_2),(C_3, C'_3),(C_4, C'_4),(C_5, C'_6),$(C_6, C'_7)\}$.

5 Conclusion et Perspectives

Dans ce papier nous avons montré que le problème de la performance des algorithmes de l'alignement des ontologies peut être abordé efficacement par les techniques algorithmiques développées dans le cadre de la programmation par contraintes. Plus précisément nous avons modélisé le problème de l'extraction d'un alignement total et injectif par un problème de satisfaction de contraintes. Afin d'extraire un tel alignement nous avons exploité la contrainte de différence Alldiff et un algorithme qui établit sa arc-consistance. Nous avons montré que l'approche présentée ici aborde efficacement le problème et se prête à une mise en œuvre à

la fois interactive et automatique. Nous avons mentionné que la complexité théorique de l'approche est faible. Nous envisageons de combiner l'approche présentée ici avec des aspects appropriées des interfaces homme/machine pour améliorer l'interaction de l'utilisateur avec le système. Nous projetons aussi d'étendre l'approche pour prendre en compte efficacement des critères plus généraux.

Références

Apt, K. (2003). *Principles of Constraint Programming*. Cambridge University Press.

Euzenat, J. et P. Shvaiko (2007). *Ontology matching*. Heidelberg (DE) : Springer-Verlag.

Euzénat, J. et P. valtchev (2004). Similarity-based ontology alignment in owl-lite. In *The proceedings of the 15th ECAI*, pp. 333–337.

Fages, J. (1996). *Programmation logique et par contraintes*. Ellipses.

Macworth, A. (1977). Consistency in networks of relations. *Journal of Artificial Intelligence 8(1)*, 99–118.

Meilicke, C. et H. Stuckenshmidt (2007). Analysing mapping extraction approaches. In *the proceedings of the second International workshop on Ontology Matching (OM-2007)*, Busan,Corea.

Régin, J. (1994). A filtering algorithm for constraints of difference in csps. In *AAAI*, pp. 362–367.

Shvaiko, P. et J. Euzénat (2008). Ten challenges for ontology matching. In *The proceedings of 7th ODBASE*, Monterry, Mexico.

Summary

In ontology matching, few research has been devoted to mappings extraction from similarity measures. Ontology alignment extraction complexity is particularly pointed out in dynamic applications where performance aspect is crucial. In this paper, we exploit difference constraints developed within constraint programming to model the problem of extracting an alignment which is both total and injective. We show that this approach is efficient and can easily be implemented both in interactive and automatic ways.