Some approaches of ontology Decomposing in Description Logics

Thi Anh Le PHAM ??,1 and Nhan LE-THANH ??

[a] *Laboratoire I3S, Université de Nice Sophia-Antipolis, France*

[b] *Laboratoire I3S, Université de Nice Sophia-Antipolis, France ; E-mail:Nhan.Le-Thanh@unice.fr*

Abstract. In this paper, we investigate the problem of decomposing an ontology in Description Logics (DLs) based on graph partitioning algorithms. Also, we focus on syntax features of axioms in given ontology. Our approach aims at decomposing the ontology into many of sub-ontologies that are the most distinct as possible. We analyze the algorithms and exploit parameters of partitioning that influence the efficiency of computation and reasoning. These parameters are the number of concepts and roles shared by a pair of sub-ontologies, the size (the number of axioms) of each sub-ontology, and the topology of decomposing. We provide two concrete approaches for automatically decomposing the ontology, one is called partitioning based on minimal separator, and the other is segmenting based on eigenvectors and eigenvalues.

Keywords. graph partitioning, ontology decomposing, image segmentation.

Introduction

The previous studies about DL-based ontologies focus on the tasks as ontology design, ontology integration, ontology deployment,... Starting from the fact that one wants to effectively solve with a large ontology, instead of the ontology integrating we examine the ontology decomposing. As we proposed in the previous paper, some algorithms for reasoning on the system of decomposed ontologies and the criteria for decomposing. In this paper, we resolve the problem of decomposing, more concretely, we delve the techniques of decomposing an ontology into several sub-ontologies. Our principal goal is how do select a "good" one? A decomposing is called "good" if it improves the efficiency of reasoning and guarantees the properties proposed in [4]. Our computational analysis for reasoning algorithms guides us to suggest the parameters of a such decomposing: the number of concepts and roles included in the semantic mappings between partitions, the size of each partition (the number of axioms in each partition) and the topology of the decomposing graph. There are two approaches to be considered here. They are based on two presentation ways of the ontology. First, we present the ontology by a symbols graph, and implement decomposing based on minimal separator method. Second, the ontology is presented by an axiom graph, corresponding to the image segmentation method.

The paper is organized as follows. Section 1 describes two ways for transforming of an ontology by an undirected graph (symbol graph and weighted graph). Section 2 defines an overlap decomposing of a TBox and the criteria for a good decomposing. In sections 3 and 4, we discus the methods and two partitioning algorithms of the graph represented an ontology corresponding to the above graphs. Section 5 presents some evaluations of effect of the decomposition algorithms. Section 6 provides some conclusions and future works.

[1]Corresponding Author: Laboratoire I3S, UMR 6070, CNRS Université de Nice Sophia-Antipolis, France ; E-mail: tpham@i3s.unice.fr.

1. Representing ontology as the graph

As we knew, a knowledge base (ontology) built by a concept language is generally composed of two component levels: intensional (called TBox) and extensional (called ABox). Our attention is the reasoning problem in a large TBox. A TBox is a finite set of concept inclusions (axioms). Therefore decomposing ontology is reduced to examining the set of axioms, we split the axioms into small groups that follows the proposed criteria in [4].
In this section, we present ontology as an undirected adjacency graph. In particular, we examine two approaches for representing a TBox by graph that conduct two ways dealing with the problem of decomposing, hence the axioms are coded as follows. Given an original TBox $\mathcal{T}$ with N axioms.
- The first, unfolding (expanding) all the axioms in original TBox $\mathcal{T}$ into the expressions of primitive concepts and primitive roles (i.e. computing $Ex(A_i)$ as in [4], we recall that $Ex(A_i)$ is a set of primitive concepts and roles which occurs in A_i).
- Let $|Ex(A_i)| = n_i$ (the number of concepts and roles in A_i), $|Ex(A_i) \cap Ex(A_j)| = n_{ij}$ (the number of common concepts and roles in two axioms A_i and A_j), wand $i, j = \overline{1, m}, i \neq j$:
Example 1: We have a TBox $\mathcal{T}$:

(A_1) $: C_1 \sqcap C_2 \sqsubseteq X$	(A_6) $: Y \sqcap T \sqsubseteq H$
(A_2) $: C_3 \sqsubseteq \neg C_2$	(A_7) $: C_3 \sqsubseteq X$
(A_3) $: X \sqcap C_4 \sqcap C_5 \sqsubseteq Y$	(A_8) $: \neg C_3 \sqsubseteq C_2$
(A_4) $: \neg C_4 \sqsubseteq \neg Y$	(A_9) $: \neg X \sqsubseteq \neg Y$
(A_5) $: Y \sqcap C_6 \sqsubseteq H$	(A_{10}) $: \neg C_5 \sqsubseteq \neg Y$

Figure 1: *TBox $\mathcal{T}$*

Thus, there is ten axioms in $\mathcal{T}$, i.e $N = 10$ and they are unfolded as follows:
$Ex(\mathcal{T}) = \{C_1, C_2, C_3, C_4, C_5, C_6, X, Y, H, T\}$
$|\mathcal{T}| = 10$

1.1. Symbol graph

Definition 1 *A graph $G = (V, E)$ (with V is a set of nodes and E is a set of edges) is called symbol graph if each node $v \in V$ is a symbol, an edge $e_{ij} = (v_i, v_j) \in E$ if v_i and v_j occur in the same of axiom.*

In our case, a symbol is either a primitive concept or role in TBox $\mathcal{T}$. Construct a symbol graph $G = (V, E)$ from TBox $\mathcal{T}$, by taking each concept (role) in every axiom in $\mathcal{T}$ as a node and connecting each pair of concepts (roles) by an edge if this pair co-occurs in the same axiom. According this method, each axiom is presented as a "clique" (see definition 7) in the graph.
Example 2: The above TBox $\mathcal{T}$ can be presented by a symbol graph as follows:

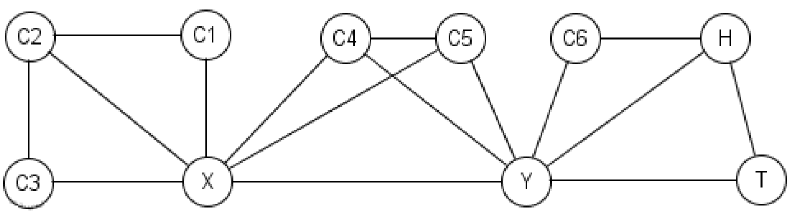

Figure 2: *Symbol graph of the TBox $\mathcal{T}$*

1.2. Axiom graph

Definition 2 *A graph $G = (V, E)$ (with V is a set of nodes and E is a set of edges) is called an axiom graph if each node $v \in V$ is an axiom in TBox $\mathcal{T}$, and an edge $e_{ij} = (v_i, v_j)$ if v_i and v_j have some symbols shared.*

From a point of view of graph, we present a TBox $\mathcal{T}$ by a weighted graph $G = (V, E)$, denoted the "connected axiom graph": each vertex v of V corresponds to an axiom in $\mathcal{T}$, and connecting each pair of vertices in V by an edge if they have some shared concepts and roles. A weight is then assigned to each edge in E. This weight should be chosen such that it reflects the association degree between the pair of linked vertices. Using just the common concepts and roles between every pair of axioms, we can define the graph edge weight connecting the two nodes i and j (w_{ij})that are corresponding to two axioms A_i and A_j as: $w_{ij} = \frac{n_{ij}}{n_i+n_j}$.
Example 3: An presentation of axiom graph for the above TBox $\mathcal{T}$:

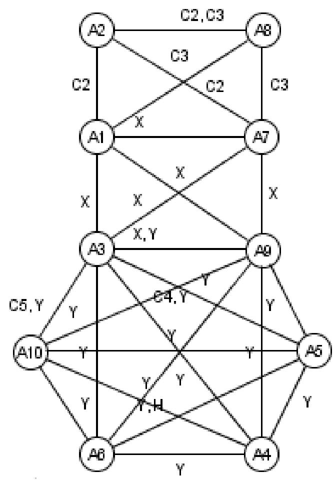

Figure 3: *Axiom graph of the TBox $\mathcal{T}$*

2. Ontology Decomposing

2.1. *Definition*

Because we examine an ontology as a graph, in this paper, we define the notion of ontology decomposing based on graph partitioning.

Definition 3 *A graph partitioning $G_p = V_p, E_p$ is a connected graph (see definition 9) of the individual partitions G_i. Each node $v \in V_p$ is a partition (sub-graph) G_i, each edge $e_{ij} = (v_i, v_j) \in E_p$ is the set of shi ared symbols between G_i and G_j.*

This definition is corresponding to the overlap decomposing in [4]. We see that G_i corresponds to component TBox $\mathcal{T}_i$, the edges e_{ij} corresponds to the set of identical bridge rules $\mathcal{B}_{ij}$.

2.2. *What is a good decomposing*

Our goal is eliminating GCIs as much as possible from a general ontology (presented by a TBox) by decomposing an ontology into several sub-ontologies (presented by a distributed TBox). In this paper, we are only considered the syntax approach that are based on the structures of GCIs. As we mentioned in the previous paper, the decomposing only implements with the set of axioms (GCIs). All the concepts and roles of the original ontology will be kept through the decomposing. As a result, we propose two techniques for decomposing based on graph. For legibility reason, in this paper, we examine only the simplest case, decomposing a TBox into two smaller TBoxes. The general case is presented in [11].
We defined the decomposing of a TBox as a distributed TBox consisting of sub-TBoxes and the semantic mappings between each pair of sub-TBoxes. Our decomposing approach is most related to the graph formulation and partitioning. A delicate aspect of decomposing-based logical reasoning is the selection of a good partitioning of the ontology, we need to also ask the following questions:
1. What is the precise criterion for a good decomposing ?

2. How can such a decomposing be computed efficiently ?
These questions have been also proposed in the image segmentation and data clustering. In the general DL case, our purpose is not reducing of computational complexity, but the results for reasoning in decomposing suggest similar relationship between the decomposed TBoxes and original TBox. The computation analysis of our decomposing-based reasoning algorithms provides a metric for identifying parameters of decomposing that influence our computation: the size of the communication part between component TBoxes, the size of each component, and the topology of the decomposing graph. Our goal is minimizing the disassociation between the component TBoxes and maximizing the association within the components. Moreover these component TBoxes must preserve the properties of decomposing that we proposed in the previous paper. These parameters guide us to propose two greedy algorithms for decomposing ontology into sub-ontologies, trying to optimize these parameters. The algorithm depends on the presenting case of ontology. The detail of these algorithms will be presented in the following sections.

3. Partitioning based on the minimal separator

Starting from the symbol graph, we examine a decomposing method based on the minimal separators. The first of all, we recall some notions that can be found in [5].

3.1. Definitions

Definition 4 *Let $G = (V, E)$ be an undirected graph, with $|V| = n, |E| = m$. For $x, y \in V$, $N(x) = \{y \neq x | xy \in E\}$. For $X \subseteq V$, $N(X) = \bigcup_{x \in X} N(x)$.*

Definition 5 *Minimum Vertex Separators*
We recall the notion of a vertex separator that was defined in [2]. A set of vertices S is called an (a, b)-vertex-separator if $\{a, b\} \subset V \setminus S$ and every path connecting a and b in G passes through at least one vertex contained in S. A set S of vertices is called an (a, b)-vertex-separator if $\{a, b\} \subset V \setminus S$ and every path connecting a and b in G passes through at least one vertex contained in S.
Two vertices a, b are said to be adjacent if there is an edge connecting a and b in G. Let a, b are not adjacent vertices. If S is a minimal a, b-separator which contains only neighbors of a the S is called close to a.

Definition 6 *Connectivity*
Let $N(a, b)$ be the least cardinality of an (a, b)-vertex-separator. The connectivity of the graph G is the minimum $N(a, b)$ for any $a, b \in V$ that are not adjacent.

Definition 7 *Cliques*
A clique (complete graph) is a graph with edge set consisting of all possible edges between the vertices of the graph. We denote by $K[S]$ the clique built on vertices from S.

Definition 8 *Maximum Cliques*
A maximal clique in a graph G is a clique $G[S]$ such that there exists no vertex set $S' \supset S$ for which $G[S']$ is a clique. The set of minimal cliques of G is denoted by K_G.

Definition 9 *Connected graph*
A graph G is connected if for each pair u, v of its vertices there exists a path from u to v. If G is not connected, then a connected subgraph C of G is called a (maximal) connected component if there exists no connected subgraph C' of G such that C is a proper subgraph of C'.

We have another definition of minimal vertex separator: S is a minimal separator of the graph $G = (V, E)$ if and only if there are two different connected components of $G[V - S]$ such that every vertex of S has a neighbor in both of these components. This definition as a lemma that have been proved in [6]
If S is a minimal a, b-separator which contains only neighbors of a then S is called *close* to a.

3.2. Algorithm

We present a recursive algorithm that uses Even's algorithm [2] to find sets of vertices that together separate a graph into partitions. The algorithm returns a set of vertices that determine the separate subgraphs. Implement this algorithm by taking as input its symbol graph $G = (V, E)$ that is transformed from a TBox $\mathcal{T}$. We briefly describe the algorithm as follows:
Input: $G = (V, E)$
Output: connected graph $G_p = (V_p, E_p)$

- Finding the set of minimal vertex separators of graph G:
  + Choosing an arbitrary pair a, b of non-adjacent vertices, and compute the set of minimal ab-separators.
  + Repeat this process on every possible pair x,y of non-adjacent vertices.
- Finding the overall minimum vertex separator S^* between all vertices in G.
- Split G into the two subgraphs G_1, G_2 that are separated by S^*, with S^* included in both subgraphs.
- Create an undirected graph $G_p = (V_p, E_p)$ with $V_p = \{G_1, G_2\}$ and $E_p = S^*$

Figure 4 illustrates the connected graph of TBox $\mathcal{T}$ in the example of Figure 2

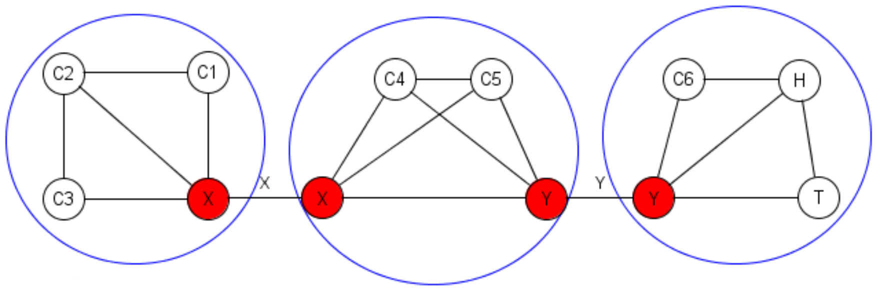

Figure 4: *Decomposing $\mathcal{T}$'s symbol graph*

In this example, we have two minimum vertex separators $\{X\}$ and $\{Y\}$. If chose $\{X\}$ to split $\mathcal{T}$, then we collect two groups of symbols $\{C_1, C_2, C_3, X\}$ and $\{X, C_4, C_5, C_6, Y, H, T\}$. Hence, we obtain two sub-TBoxes respectively : $\mathcal{T}_1 = \{A_1, A_2, A_7, A_8\}$ and $\mathcal{T}_2 = \{A_3, A_4, A_5, A_6, A_9, A_{10}\}$. Similarly, if chose $\{Y\}$, we obtain two sub-TBoxes $\mathcal{T}_1 = \{A_1, A_2, A_3, A_4, A_7, A_8, A_9, A_{10}\}$ and $\mathcal{T}_2 = \{A_5, A_6\}$.
We will take the first result, because there is a balance of the number of axioms of two sub-TBoxes in the first. Thus, the axioms in original TBox is distributed into $\mathcal{T}_1$ and $\mathcal{T}_2$ with the number of axioms respectively $N_1 = 4$, $N_2 = 6$.
The number of symbols in the minimum vertex separator is $|S^*| = |\{X\}| = 1$

4. Decomposing based on image segmentation

This section describe a grouping method as image segmentation using eigenvectors. First of all, we assume that a graph $G = (V, E)$ could be splited into two disjoint parts. The degree of association between these two parts can be computed as total weight of edges that have been removed from the original graph. In the fact, these edges will be transformed into "bridge rules". The optimal partitioning of a graph is not only minimize this degree but also maximize the degree of association within parts. In graph theory, these degrees are selected as [3]:
$Ncut(A, B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$
$Nassoc(A, B) = \frac{assoc(A,A)}{assoc(A,V)} + \frac{assoc(B,B)}{assoc(B,V)}$
where $cut(A, B) = \sum_{u \in A, v \in B} w(u, v)$, the total connection from nodes in A to nodes in B.
and $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$, the total connection from nodes in A to all nodes in the graph.

We can simply see that, $Ncut(A, B) = 2 - Nassoc(A, B)$.
We recall some notations in [3]: Assume that a graph $G = (V, E)$ is partitioned into two parts A and B, let x be an $N = |V|$ dimensional indicator vector with $x_i = 1$ if node i is in A, and $x_i = -1$, otherwise. Let $d(i) = \sum_j w(i, j)$ be the total connection from node i to all other nodes. Let D be an NxN diagonal matrix with d on its diagonal, W be an NxN symmetrical matrix with $W(i, j) = w_{ij}$.
$k = \frac{\sum_{x_i>0} d_i}{\sum_i d_i}$, $b = \frac{k}{1-k}$, $y = (1 + x) - b(1 - x)$
They are used for computing the optimal partition by solving $(D - W)x = \lambda Dx$.

We implement the algorithm according to the following principal steps [3]:
1. Coding a given ontology by an undirected graph (as the above section).
2. Finding minimum value of NCut by solving the eigenvectors with the smallest eigenvalues of the system
$(D - W)y = \lambda Dy$ (*)
As we saw above, the generalized eigensystem in (*) can be transformed into a standard eigenvalue problem of
$D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}x = \lambda x$ (**)
3. Once the eigenvector are computed, we can decompose the graph into two partitions using the second smallest eigenvector. In the idea case, the eigenvector should only take on two discrete values and the signs of the values can tell us exactly how to decompose the graph. However, our eigenvectors can take on continuous values and we need to choose a "good splitting point" to decompose it into two parts.
4. After the graph is broken into two partitions, we can recursively implement our algorithm on two these decomposed partitions.
Figure 5a illustrates an obtained NCut value (denoted by the green line) of the example in Figure 2, and result graph of decomposing is showed in Figure 5b. This result is the same in the method based on minimal separator.

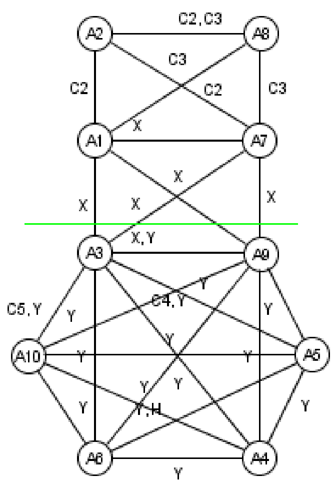

Figure 5a: *Axiom graph of $\mathcal{T}$ and its NCut*

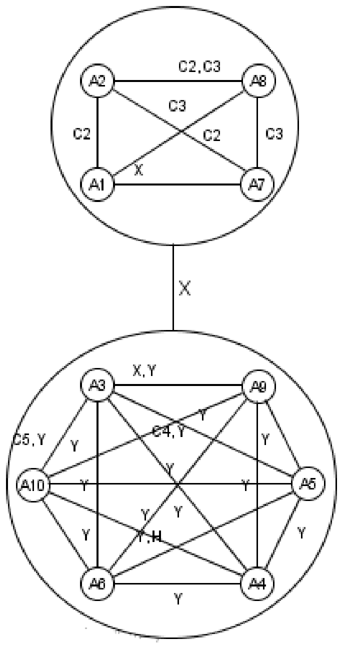

Figure 5b: *Decomposing $\mathcal{T}$'s axiom graph*

5. Effect of the decomposition algorithms

We have applied two partitioning algorithms to graph decomposition based on minimal separator and image segmentation. Two these methods return the same result that satisfies the proposed de-

composing properties. All the concepts, roles and axioms are preserved over decomposition implementing. The relations between them are expressed by the edges in symbol, axiom and connected graphs. In the first place, the set of axioms in the original TBox is reduced by regular distributing into sub-TBoxes. These decomposition techniques focus on finding a good decomposing. The approach based on minimal separator minimizes the number of symbols shared between partitions and tries balancing the number of axioms in the partitions. However, one have to regroup the axioms based on the cliques in the symbol graph after decomposing. Also, in reality there can be some cliques in symbol graph which does not present axioms.
The possible advantage of approach based on image segmentation is that it conserves the axioms. Furthermore, the NCut measure is normalized, it expresses the disassociation between partitions and the association within partitions. However, to implement effectively this technique, one must propose a proper weight function for edges between nodes of axiom graph.

6. Conclusion and future works

In this paper, we presented two algorithms for ontology decomposing of the DL language $\mathcal{SHIQ}$ that have done by using the graph partitioning. By treating the grouping problem in graph theory, we proposed two methods for representing the ontology by symbol graph and weighted graph. Each graph type is applied corresponding to a decomposing algorithm. Two graph partitioning algorithms are independently implemented. We see that two approaches are sufficient for our purpose in requirements of decomposed ontologies. However, the choice of decomposing method depends on the structure of original ontology. For example, with ontology that have the large number of symbols then one can chose the method of weighted graph. And for ontology have many of axioms then one can use the method of symbol graph. These proposals only are suggestions for future works. We propose besides some essential properties of a good decomposing that influence the reasoning performance as have also been provided in [4]. The graph transformations of ontology again depends on having an effective (and cheap) method for analyzing the likely characteristics of a given test ontology. We are also performing more experiments with very large KBs (as UMLS,...) for decomposing. Preliminary results in the decomposed ontologies is likely similarity in the both of methods. We are embarking in the optimizing of these decomposition algorithms and treating effectively in more large ontologies.

References

[1] Dieter Jungnickel, Graphs,Networks and Algorithms. Springer 1999.

[2] Eyal Amir and Sheila McIlraith, Partition-Based Logical Reasoning for First-Order and Propositional Theories. *Artificial Intelligence, Volume 162, February 2005, pp. 49-88.*

[3] Jianbo Shi and Jitendra Malik, Normalized cuts and Image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8), 888-905, August 2000.*

[4] Thi Anh Le Pham and Nhan Le Thanh, Decomposition-Based Reasoning for Large Knowledge Bases in Description Logics. *In Proceedings of the 13th ISPE International Conference on Concurrent Engineering: Research and Applications, p. 288-295, Antibles, France, September 2006*

[5] Kirill Shoikhet and Dan Geiger, Finding optimal triangulations via minimal vertex separators. *In Proceedings of the 3rd International Conference, p. 270-281, Cambridge, MA, October 1992.*

[6] T.Kloks and D.Kratsch, Listing all minimal separators of a graph. *In Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science, Spinger, Lecture Notes in Computer Science, 775, pp.759-768.*

[7] Borgida A., Serafini L., Distributed Description Logics: Assimilating information from peer sources. *Journal of Data Semantics, 1:153-184, 2003.*

[8] Serafini L.and Tamilin A., DRAGO: Distributed Reasoning Architecture for the Semantic Web. *Technical Report T04-12-05, ITC-irst, December 2004.*

[9] Jon William Freeman, Improvements to Propositional Satisfiability Search Algorithms. *Dissertation in Computer and Information Science, 1995.*

[10] Francesco M. Donini, Complexity of Reasoning. *Description Logic Handbook 2003: 96-136*

[11] Thi Anh Le Pham, Raisonnement sur des ontologies décomposées. *Rapport de recherche, Lab I3S, Université de Nice-Sophia Antipolis, 2006.*