Decomposing ontology in Description Logics by graph partitioning

Thi Anh Le PHAM

Faculty of Information Technology, Hanoi National University of Education, Hanoi, Vietnam
Email address: lepta@hnue.edu.vn

Nhan LE-THANH

Laboratory I3S, Nice Sophia-Antipolis University, Nice, France
Email address: nhan.le-thanh@unice.fr

Minh Quang NGUYEN

Faculty of Information Technology, Hanoi National University of Education, Hanoi, Vietnam
Email address: quangnm@hnue.edu.vn

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 a given ontology. Our approach aims at decomposing the ontology into many sub ontologies that are as 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 decomposition. We provide two concrete approaches for automatically decomposing the ontology, one is called minimal separator based partitioning, and the other is eigenvectors and eigenvalues based segmenting. We also tested on some parts of used TBoxes in the systems FaCT, Vedaall, tambis, ... and propose estimated results.

Keywords: *Graph partitioning; ontology decomposition; image segmentation*

1 Introduction

The previous studies about DL-based ontologies focus on the tasks such as ontology design, ontology integration and 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. There were some investigations in decomposition of DL ontologies as decomposition-based module extraction [3] or based on syntax structure of ontology [1].

The previous paper [8] shown the executions on the supposition that there exists an ontology (TBox) decomposition called overlap decomposition. This decomposition resulted in preservation of semantic and inference results with respect to original TBox. Our aim is to establish the theoretical foundations for decomposition methods that improves the efficiency of reasoning and guarantees the properties proposed in [7]. The automatic decomposition of a given ontology is an optimal step in ontology design that is supported by graph theory. The graph theory provided the “good properties” that adapt necessary requirements of our decomposition.

Our computational analysis of reasoning algorithms guides us to suggest the parameters of such decomposition: the number of concepts and roles included in the semantic mappings between partitions, the size of each component ontology (the number of axioms in each component) and the topology of the decomposition graph. There are two decomposition approaches based on two ways of presenting the ontology. One presents the ontology by a symbols graph, which implements decomposition by minimal separator and the other uses axiom graph, corresponding to the image segmentation method.

The rest of the paper is organized as follows. Section 2 proposes a definition of *G-decomposition* methodology that is based on graph and summarizes some principal steps. In this section, we also recall the criteria for a good decomposition. Sections 3 and 4 describe two ways for transforming an ontology into an undirected graph (symbol graph or weighted graph), as well as two partitioning algorithms of the obtained graph. Section 5 presents some evaluations of the effects of the decomposition algorithms and experimental results. Finally, we provide some conclusions and future work in section 6.

2 G-decomposition Methodology

In this paper, ontology decomposition will be considered only from terminological level (TBox). We research some methods that decompose a given TBox into several sub-TBoxes. For simple, now a TBox can be briefly presented by the single set of axioms **A**, so we will present the set of axioms by graph.

Our goal is to eliminate general concept inclusions (GCIs), a general type of axiom, as much as possible from a general ontology (presented by a TBox) by decomposing the set of GCIs of this ontology into several subsets of GCIs (presented by a distributed TBox). In this paper, we only consider the syntax approach based on the structures of GCIs. We recall some criteria of a good decomposition [8]:

- All the concepts, roles and axioms of the original ontology are kept through the decomposition.
- The number of axioms in the sub-TBoxes is equivalent.

As a result, we propose two techniques for decomposing based on graphs. *G-decomposition* is an ontology decomposition method that applies graph partitioning techniques. This graph decomposition is presented by an intersection graph (decomposition graph) in which each vertex is a sub-graph and the edges present the connections of each pair of vertices. In [8] we defined an *overlap decomposition* of a TBox, it is presented by a distributed TBox (decomposed TBox) that consists of a set of sub-TBoxes and a set of links between these sub-TBoxes (semantic mappings). We assume that readers are familiar with basic concepts in graph theory.

Consequently, we propose an ontology decomposition method as a process contain three principal phases (illustrated in the table 1): transform a TBox into a graph, decompose the graph into sub-graphs, transform these sub-graphs into a distributed TBox. We present a *general algorithm of G-decomposition.*

Table 1: A GENERAL ALGORITHM OF G-DECOMPOSITION

```
PROCEDURE DECOMP-TBOX (T = (C, R, A))

T = {C, R, A} is a TBox, with the set of concepts C, the set of roles R, and the set of axioms A

(1) TRANS-TBOX-GRAPH (T = (C, R, A))

Build a graph G = (V, E) of this TBox, where each vertex v ∈V is a concept in C or a role in R (or an axiom in A), each edge e = (u, v) ∈ E if u and v appear in the same axiom (or u and v have at least a common concept (role))

(2) DECOMP-GRAPH (G = (V, E ))

Decompose the obtained graph G = (V, E) in the procedure TRANS-TBOX-GRAPH into an intersection graph $G_0$ = ($V_0$, $E_0$), with each vertex v∈$V_0$ is a sub-graph, each edge e = (u, v)∈$E_0$ if u and v are linked.

(3) TRANS-GRAPH-TBOX ($G_0$ = ($V_0$, $E_0$)

Transform the graph G = ($V_0$, $E_0$) into a distributed TBox, each vertex (sub-graph) corresponds to a sub-TBox, and edges of $E_0$ correspond to semantic mappings.
```

In the next sections, we will introduce the detail techniques for the steps (1) and (2).

3 Decomposition based on minimal separator

A. Symbol graph

A set of axioms **A** of TBox T, then Ex(**A**) is the set of concepts and roles that appear in the expressions of **A**. For simple, we use the notation of symbol instead of concept (role), i.e., a symbol is a concept (role) in TBox. A graph presenting TBox will be defined as follow:

Definition 1 (*symbol graph*): *A graph G = (V, E), where V is a set of vertices et E is a set of edges, is called a symbol graph of T (**A**) if each vertex v ∈ V is a symbol of Ex(**A**), each edge e = (u, v) ∈E if u, v are in the same axiom of **A**.*

So, given a set of axioms **A**, we can build a symbol graph G = (V, E) by taking each symbol in Ex(**A**) as a vertex and connecting two vertices by an edge if its symbols are in the same axiom of **A**. Follow this method, each axiom is presented as a clique in the symbol graph.

Example 1: Given a TBox as follows (figure 1):

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

Figure 1: TBox T_{fam}

The set of primitive concepts and roles of T_{fam}: $Ex(T_{fam}) = \{C_1; C_2; C_3; C_4; C_5; C_6; X; Y; T; H\}$. The figure 2 presents the symbol graph for T_{fam}

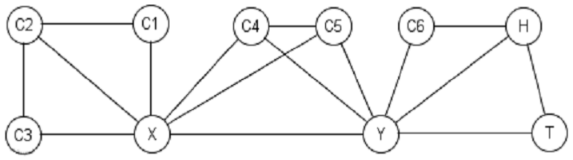

Figure 2: Symbol graph presenting TBox T_{fam}

Result of this decomposition is presented by a labeled graph (intersection graph or decomposition graph) $G_p = (V_p, E_p)$. Assume that the graph G representing a TBox T is divided by n sub-graphs $G_{i,\ i \le n}$, then a decomposition graph is defined as follows:

Definition 2 *(decomposition graph) [4]:*
Decomposition graph is a labeled graph $G_p = (V_p, E_p)$ in which each vertex $v \in V_p$ is a partition (sub-graph) G_i, each edge $e_{ij} = (v_i, v_j) \in E_p$ is marked by the set of shared symbols of G_i and G_j, where $i \neq j$, $i, j \le n$.

Definition 3 *((a,b) – minimal vertex separators)[3]:*
A set of vertices S is called *(a,b) - vertex separator if $\{a,b\} \subset V \backslash S$ and all paths connecting a and b in G pass through at least one vertex in S. If S is an (a,b) - vertex separator and not contains another (a,b) - vertex separator then S is an (a,b) - minimal vertex separator.*

B. *Algorithm*

We present a recursive algorithm using Even algorithm [4] to find the sets of vertices that divide a graph into partitions. It takes a symbol graph G = (V, E) (presenting a TBox) as input and returns a decomposition graph and the set of separate sub-graphs. The idea of algorithm is to look for a connecting part of graph (cut), compute the *minimal separator* of graph, and then the graph is carved by this separator. Initially, the TBox T is considered as a large partition, and it will be cut into two parts in each recursive iteration. The steps of algorithm are summarized as follows:

Table 2: AN ALGORITHM FOR TRANSFORMING THE TBOX INTO GRAPH

Input: TBox T (**A**), M limit number of symbols in a part (a sub-TBox T_i).

Output: $G_p = (V_p, E_p)$, and $\{T_i\}$.

```
PROCEDURE DIVISION-TBOX (A, M )
(1) Transform A into a symbol graph G (V, E) with
    $V = Ex(\mathbf{A})$, $E = \{(l_1, l_2) | \exists A \in \mathbf{A}, l_1, l_2 \in Ex(A)\}$.
    /A is an axiom in A
(2) Let $G_p = (V_p, E_p)$ a undirect graph with $V_p = \{\{V\}\}$ and
    $E_p = \emptyset$.
(3) Call DIVISION-GRAPH(G, M, nil, nil).
(4) For each $v \in V_p$, let $\{T_v = \{A \in \mathbf{A} | Ex(A) \subset v\}$. Return
    $T_v$, $v \in V_p$ and $G_p$.
```

The procedure DIVISION-GRAPH takes the input consisting of a symbol graph G = (V, E) of T, a limited parameter M and two vertices a, b that are initially assigned to nil. This procedure updates the global variable G_p for presenting the decomposition process. In each recursive call, it finds a minimal separator of vertices a, b in G. If one of a, b is nil or both are nils, it finds the global minimal separator between all vertices and the non-nil vertex (or all other vertices). This separator cuts the graph G into two parts G_1, G_2 and the process continues recursively on these parts.

Table 3: AN ALGORITHM FOR PARTITIONING SYMBOL GRAPH

```
Input: G = (V, E)

Output: connection graph G_p = (V_p, E_p)
PROCEDURE DIVISION-GRAPH (G, M, a, b)

(1) Find the set of minimal vertex separator of G:
    - Select an pair of non-adjacent arbitrary vertices
      (a, b) and compute set of (a, b) – minimal separator
    - Repeat this process on every pair of non-adjacent
      vertices x, y
(2) Find global minimal vertex separator $S^*$ between all
    vertices of G
(3) Decomposing G by $S^*$ in two sub-graphs $G_1$, $G_2$, where
    $S^*$ is in all $G_1$ and $G_2$
(4) Generate an undirect graph $G_p = (V_p, E_p)$, where $V_p = \{G_1, G_2\}$ and $E_p = S^*$.
```

The algorithm describes a method to list all of the (a, b) - vertex minimal separators from a pair of non-adjacent vertices by best-first search technique can be seen in [6].

T_{fam} (figure 1) can be presented by an undirect adjacent graph (figure 2), where the vertices of this graph correspond to the symbols, the edges of the graph connecting the two vertices corresponding to two symbols of the same axiom. Therefore each axiom would be represented as a clique.

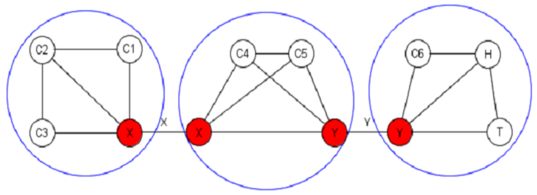

Figure 3: Decomposing result of symbol graph of T_{fam} with minimal separator S*= {X} and S*' ={Y }

If the criterion based on balance of TBox axioms number between components then $S^* = \{X\}$ and $S^{*'}=\{Y\}$. Using S^* and $S^{*'}$ to decompose the symbol graph, we obtain three symbol groups $\{C_1, C_2, C_3, X\}$, $\{C_4, C_5, X, Y\}$ and $\{C_6, H, Y, T\}$. So we get three corresponding TBoxes: $T_1 = \{A_1, A_2, A_7, A_8\}$, $T_2 = \{A_3, A_4, A_9, A_{10}\}$ and $T_3 = \{A_5, A_6\}$. The number of symbols of S^* and $S^{*'}$ is 1 ($|S^*| = |\{X\}| = 1$, $|S^{*'}| = |\{Y\}|=1$). The cardinality of three TBoxes are respectively $N_1 = 4$; $N_2 = 4$, $N_3 = 2$. In this case, the cardinality of symbols in each TBox is also equivalent.

The image of symbol graph of T_{fam} after decomposing as in the figure 3. Obtained Result TBoxes T_1, T_2 and T_3 after decomposition preserve all the concepts, roles and axioms of original T_{fam}. In addition, T_1 and T_2 satisfy the proposed criteria for decomposing.

We have executed graph partition algorithm based on minimal separator. This method returns result that satisfies the given properties. All concepts, roles, and axioms are preserved through decomposing. Relations between them are represented by the edges of symbol inter-graph. This method minimizes symbols shared between component TBoxes ensuring independency property of sub-TBoxes. However, to get the result TBoxes, requiring transfer the obtained graphs into the sets of axioms for the corresponding TBoxes.

4 Decomposition based on normalized cut

A. Axiom graph

In this section, we propose another decomposition technique based on *axiom graph* that is defined as follow:

Definition 4 *(axiom graph): A weight undirect graph G = (V, E), where V is a set of vertices and E is a set of edges with the weight values, is* called *an axiom graph if each vertex $v \in V$ is an axiom in TBox T, each edge $e = (u, v) \in E$ if $u, v \in V$ and there is at least a shared symbol between u and v, and the weight on e (u, v) is a value presenting the similarity between the vertices u and v.*

By using only the common symbols between each pair of axioms, we can simple define a weight function p: V x V → R that send a pair of vertices to a real number. In particular, each edge (i, j) is assigned a value w_{ij} describing the connection (association) between axioms A_i and A_j as: $w_{ij} = n_{ij}/(n_i + n_j)$, where $i, j = 1,..,m$, $i \neq j$, m is the number of axioms in T ($m = |A|$), n_i, n_j is the symbol number of A_i and A_j respectively, n_{ij} is the number of symbols in $A_i \cap A_j$ ($n_{ij} = |A_i \cap A_j|$).

B. Normalized cut

Ontology decomposition algorithm based on image segmentation is a *grouping* method using eigenvectors:

Assume that $G = (V, E)$ is divided into two separate sets A and B, ($A \cup B = V$ *and* $A \cap B = \varnothing$) by removing the edges connecting two parts from the original graph. The *association* between these parts is the total weight of the removed, in the language of graph theory, it is called the *cut*:

$$cut(A, B) = \sum_{u \in A, v \in B} w(u,v) \qquad (1)$$

i.e. the total number of connections between the nodes of A and the nodes of B. Optimal decomposition of graph is not only to minimize this disassociation, but also to maximize the association in every partition. In addition, *NCut* (normalized Cut) is also used to measure *disassociation*, denoted by *Ncut* as follows:

$$Ncut\ (A, B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \qquad (2)$$

where, $assoc(A, V) = \sum_{u \in A, t \in V} w(u,t)$ is the total of connections from the nodes of A to all nodes of V. Similarly, *N assoc* is denoted by:

$$Nassoc\ (A, B) = \frac{assoc(A,A)}{assoc(A,V)} + \frac{assoc(B,B)}{assoc(B,V)} \qquad (3)$$

where *assoc (A, A)* and *assoc (B, B)* are the weight totals of edges in A and in B respectively. The optimal division of graph reduce to minimize not only NCut but also to maximize Nassoc in the partitions.

It is easy to see that *Ncut(A, B)= 2 – Nassoc(A, B)*. This is an important property of decomposition. Because of two obtained criteria ((2) and (3)) from the decomposition algorithm, minimizing the dissociation between the parts and maximizing the association in each part, are identical in reality and can be satisfied simultaneously.

Unfortunately, minimizing the normalized cut is exactly complete-NP, even for the particular case of graph on grids. However, the authors in [5] also indicated that if the normalized cut problem is extended in the real value domain, then an approximate solution can be efficiently found.

C. Algorithm

Given a N-dimensional vector x, $N = |V|$, where $x_i = 1$ if the node i is in A and $x_i = -1$ if the node i is not in A. Let $d_i = \sum_j w(i, j)$ the total of connections from i to all other nodes. Let D be a diagonal matrix N × N, d the main diagonal of the matrix, W a symmetric matrix $N \times N$ with $W(i, j) = w_{ij}$.

Ontology decomposition algorithm based on image segmentation consists of the following steps:

1) Transform a set of axioms A to an axiom graph G = (V, E) with V = {v|v∈V} and E = {(u,v)|u, v∈V, $w(u,v) = \frac{|u \cap v|}{|u \cup v|}$}.
2) Find the minimum value of NCut by solving : $(D - W)x = \lambda Dx$ to find eigenvectors corresponding to the smallest eigenvalues λ.
3) Using eigenvectors with the second smallest eigenvector to decompose the graph into two parts. In the ideal case, the eigenvector only obtains two eigenvalues and the value signs propose a graph decomposition method.
4) Implementing the recursive algorithm on the two decomposed subgraphs.

The decomposition algorithm of TBox based on normalized cut [5] is illustrated by the procedure DIVISION-TBOX-NC. This procedure takes a TBox T with the set of axioms **A** as input. It transforms **A** into an axiom graph *G = (V, E)*, where each axiom A_i of **A** is a vertex $i \in V$, each edge $(i, j) \in E$ is assigned by a weight $w(i,j) = \frac{|\mathrm{Ex}(A_i) \cap \mathrm{Ex}(A_j)|}{|\mathrm{Ex}(A_i) \cup \mathrm{Ex}(A_j)|}$. Then, the process is performed as the procedure TBOX-DIVISION in the figure 3.

The DIVISION-TBOX-NC uses the DIVISION-GRAPH-A procedure for dividing the axiom graph presenting T. This procedure takes the axiom graph G as input, compute the matrices **W**, **D**. **W** is a valued matrix NxN with w(i,j) computing as below. **D** is a diagonal matrix NxN with the values $\mathbf{d}(i) = \sum_i w(i, j)$ on its diagonal. Then, we resolve the equation $(\mathbf{D}-\mathbf{W})y = \lambda \mathbf{D} y$ with the constraints $y^T \mathbf{De} = 0$ and $y_i \in \{2, -2b\}$, where e is a vector Nx1 to all ones, to find the smallest eigenvalues. The second smallest eigenvalue is chosen and it is the minimal value as NCut. We take the eigenvector that corresponds to this eigenvalue for dividing G into two parts G_1, G_2. Finally the GRAPH-DIVISION-A updates the variable G_p as in the method based on minimal separator. This procedure can be performed recursively, in each recursive call on G_i, it finds an eigenvector with the second smallest eigenvalue and the process continues on G_i.

Table 4: AN ALGORITHM FOR TRANSFORMING THE TBOX INTO AXIOM GRAPH

```
Input: the TBox T with the set of axioms A
Output: the decomposition graph $G_p = (V_p, E_p)$ and {T }.
PROCEDURE DIVISION-TBOX-NC(A)
(1) Transform a set of axioms A to an axiom graph G = (V, E)
    with $V = \{v|v \in \mathbf{A}\}$ and $E = \{(u, v) | u, v \in V, w(u,v) = \frac{|u \cap v|}{|u \cup v|}\}$
(2) Let $G_p = (V_p, E_p)$ an undirect graph with $V_p = \{\{V\}\}$ and $E_p = \emptyset$
(3) Execute DIVISION-GRAPH-A(G = (V, E))
(4) For each $v \in V_p$, take $\{T_v = \{A \in \mathbf{A} | A = v\}\}$. Return $T_v$, $v \in V$, p and $G_p$.
```

Table 5: AN ALGORITHM FOR DECOMPOSING AXIOM GRAPH

```
Input: the axiom graph G = (V, E)
Output: the decomposition graph $G_p = (V_p, E_p)$
PROCEDURE DIVISION-GRAPH-A(G (V, E))
(1) Find the minimal value of NCut et resolving the equation $(D - W)x = \lambda Dx$ for eigenvectors with the smallest eigenvalues.
(2) Use the eigenvector with the second smallest eigenvalue for decomposing the graph into two sub-graphs $G_1$, $G_2$.
(3) Let $V_p \leftarrow V_p \backslash \{\{V\}\} \cup \{\{V_1\}, \{V_2\}\}$ and $E_p \leftarrow E_p \cup \{(\{V_1\}, \{V_2\})\}$. We change the edges connecting to {V} for the links to one of $\{V_1\}$, $\{V_2\}$
(4) After the graph is divided to two parties, we can implement recursively:
DIVISION-GRAPH-A($G_1$), DIVISION-GRAPH-A($G_2$).
```

5 Experiment and evaluation

Two graph decomposition algorithms based on minimal separator and image segmentation of ontology are implemented and return the same result satisfying decomposition criteria.

The first algorithm minimizes the shared number of symbols ($|S^*| \rightarrow$ minimum) and attempts to balance the number of axioms between parts. After the decomposition, a sign to identify the axioms in obtained graph is cliques. However, some cliques do not present any axiom in the fact. Therefore we need a mechanism to determine the axioms.

The second algorithm obtains the advantage of preserving axioms. By the value of NCut, we can measure the independence between parts and the dependence between elements in every part. However, to install an efficient algorithm, a weighting function for the edges connecting the nodes of the graph axioms must be given.

Selecting of decomposition algorithm is based on structure of original ontology. For example, the second algorithm is used for ontology which has been presented with a lot of symbols while the first algorithm is more suitable to ontology which consists of many axioms.

We applied two decomposition algorithms based on the minimal separator and on the normalized cut to divide a TBox. In this section, we summarize some principal modules that are implemented in our experiments. To illustrate our results, we take a TBox extracting from the file “tambis.xml” in the system FaCT. This TBox, called Tambis1, consists of 30 axioms.

- Transform the ontology into a symbol graph: This module reads a file presenting a TBox in XML. The read file is transformed to a symbol graph. The figure 4 describes the symbol graph of Tambis1 TBox with the labeled vertices by the names of concept and role.
- Transform the ontology into an axiom graph: This module performs the same function with the above module, it results in an axiom graph. The figure 5 describes the axiom graph of tambis1 TBox with the labeled vertices by the symbols A_i ($i = 0, .., 29$).
- Decompose the ontology based on the minimal separator: decompose an axiom graph to a tree where the leaf nodes are axioms. The figure 6 presents this decomposition for Tambis1.
- Decompose the ontology based on the normalized cut: decompose a symbol graph to a tree where the leaf nodes are axioms. The figure 7 presents this decomposition for Tambis1.

These two methods return the results that satisfy the proposed properties of our decomposition. All the concepts, the roles and axioms are preserved after the decomposition execution. The axioms and their relationships are well expressed by the symbol graph and the axiom graph. The set of axioms in the original TBox decreases by regular distribution in sub-TBoxes. The decomposition techniques focus on finding a good decomposition. The method based on minimal separator minimizes the number of shared symbols between the components and tries to equalize axiom number in these parties. We need recover the axioms after decomposing. It is possible because the axioms were encoded by cliques in the symbol graph. However, in reality the difficult of this problem is that there are some cliques of symbol graph and of intersection graph that are not exactly axioms.

The possible advantage of the decomposition method based on the normalized cut is to conserver the axioms. After decomposing, we can directly to find the axioms in the components. Furthermore, the measure of NCut is normalized, it expresses the dissociation between the different parties and the association in each party of decomposing. However, the effectiveness of this method depends on the choice of appropriate parameters for calculating the relation of similarity between two axioms.

We tested on the TBoxes in the FaCT system, as Vedaall, modkit, people, platt, and tambis. The results show that for the axioms whose expressions are more complex, the application of normalized cut method is much more effective (e.g, Veda-all, modkit), while the minimal separator method is better with the simple axioms (e.g, platt, tambis).

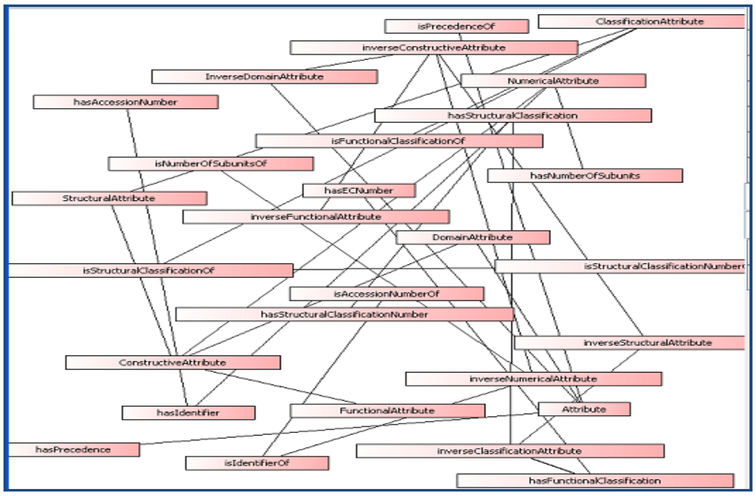
isPrecedenceOf
ClassificationAttribute
inverseConstructiveAttribute
InverseDomainAttribute
NumericalAttribute
hasAccessionNumber
hasStructuralClassification
isFunctionalClassificationOf
isNumberOfSubunitsOf
hasNumberOfSubunits
StructuralAttribute
hasECNumber
inverseFunctionalAttribute
DomainAttribute
isStructuralClassificationOf
isStructuralClassificationNumber
isAccessionNumberOf
hasStructuralClassificationNumber
inverseStructuralAttribute
ConstructiveAttribute
inverseNumericalAttribute
hasIdentifier
FunctionalAttribute
Attribute
hasPrecedence
inverseClassificationAttribute
isIdentifierOf
hasFunctionalClassification

Figure 4: Symbol graph of Tambis1

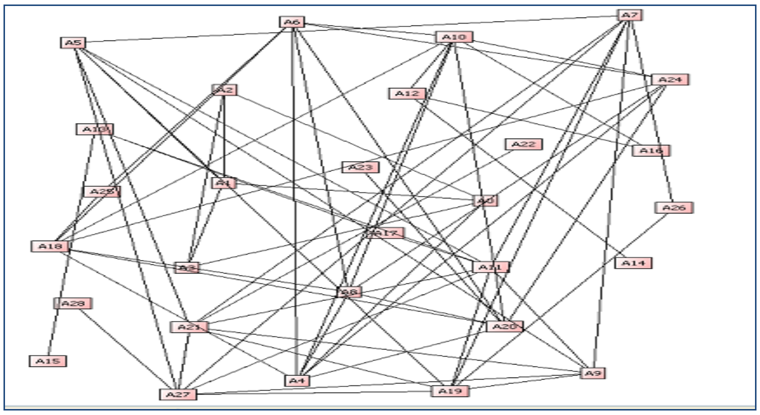

Figure 5: Axiom graph of Tambis1

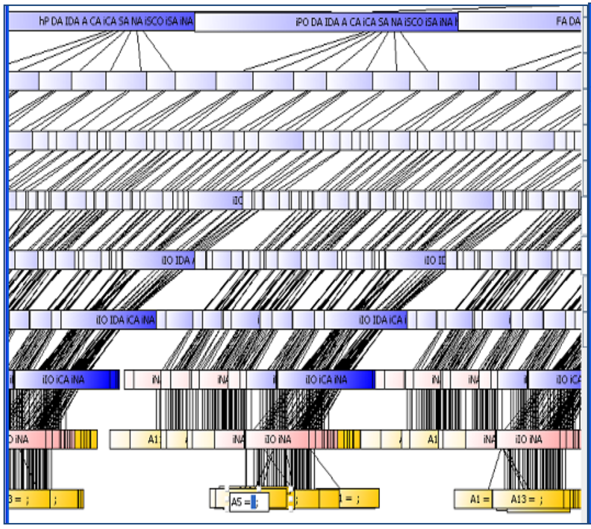

Figure 6: decomposition graph based on minimal separator of Tambis1

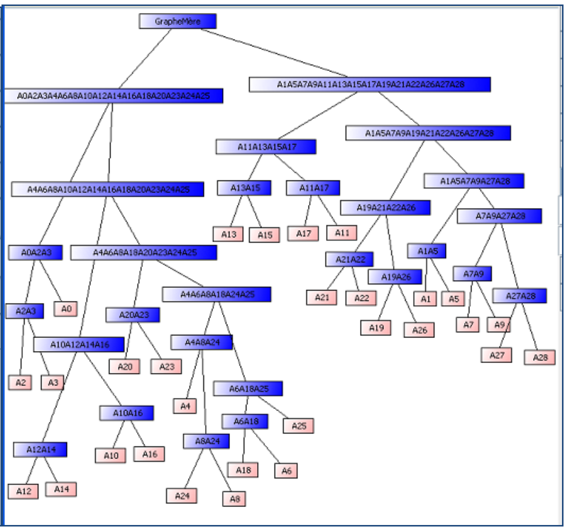

Figure 7: decomposition graph based on normalized cut of Tambis1

6 Conclusions

In this paper we have presented two techniques of decomposing from ontologies in Description Logics (TBox level). Our decomposition methods aim to reduce the number of GCIs [8], one *of* the main factors causing complexity to the algorithm argument. TBox separation method based on minimal separator only considers axioms in the aspect of syntax. We examine the simplest case, where the concept and role atoms are equivalent symbols in the axioms. However, in reality, they have different meanings. For example, the concept descriptions C † D and C ∪ D will be represented by the same symbol graph with symbols, although their meaning is different. Therefore, we will keep on developing methods of ontology separation taking into account of the dependence between symbols based on linking elements and the semantics of the axioms. Besides, we will examine the query processing issue on decomposed ontologies.

References

[1] Boris Konev, Carsten Lutz, Denis Ponomaryov, Frank Wolter, Decomposing Description Logic Ontologies, KR 2010.

[2] Chiara Del Vescovo, Damian D.G.Gessler, Pavel Klinov, Bijan Parsia, Decomposition and modular structure of BioPortal Ontologies, *In proceedings of the 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011*.

[3] Dieter Jungnickel, Graphs,Networks and Algorithms. Springer1999.

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

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

[6] 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*.

[7] Thi Anh Le PHAM and Nhan LE-THANH, Some approaches of ontology decomposing in Description Logics. *In Proceedings of the 14th ISPE International Conference on Concurrent Engineering: Research and Applications, p.543-542, Brazil, July 2007*.

[8] Thi Anh Le Pham, Nhan Le-Thanh, Peter Sander, Decomposition-based reasoning for large knowledge bases in description logics. *Integrated Computer Aided Engineering (2008), Volume: 15, Issue: 1, Pages: 53-70*.

[9] 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*.