Joining Semantic Schemas in Vision of a Distributed System Reasoning

Mihaela GHINDEANU

Department of Computer Science,
University of Craiova, Romania
mghindeanu@yahoo.com

Abstract. In this article we propose a method for joining two semantic schemas into a larger schema that incorporates and increases their reasonings. This method will lead us to a more general joining operation for n semantic schemas, $n \geq 2$. The proposed joining operation will be used in a future paper dedicated to the design of a distributed reasoning system that uses the semantic schema model for representation and processing of the knowledge.
Keywords: distributed system, semantic schema
AMS Subject Classification: 68T30, 68T45, 68T50

1 Introduction

A *distributed system* has a certain number of components by means of which it can perform complex tasks.

In a *distributed reasoning system* the composing elements are named *reasoning sources*. As we know, these sources have a representation and reasoning mechanism used to perform deductions.

Usually, in order to answer to complex interrogations, a distributed reasoning system must combine deductions obtained by more then one of its reasoning sources.

Thus, whenever we construct such a system the following problem arises:

Which is the proper manner for combining the reasonings of two (or more) different components of this kind of system such that the information can be preserved and furthermore enriched with new ones?

We intend to treat this problem for a distributed reasoning system in which all of its components use the *semantic schema* concept ([1]) as the representation and reasoning mechanism.

Thus, we have to define a method be means of which different semantic schemas generated by different reasoning components can be combined in order to increase the reasonings. Our method combines different schemas by joining them in a bigger structure which will turn out to be also a semantic schema.

In this article we present an algorithm that implements the joining operation for two semantic schemas. The presented method could be easily turn into the principal routine for a more general algorithm that joins n semantic schemas, $n \geq 2$.

This article is structured as follows:

– we start by enumerating the basic theoretical concepts used in this paper
– we define the joining operation of two semantic schemas
– then we present the algorithm that implements the joining operation of two schemas and we give the prove that the algorithm is correctly defined
– finally we present a study case in order to illustrate the presented method.

2 Prerequisites

Consider a symbol θ of arity 2 and a finite non-empty set A_0. We denote by $\overline{A}_0$ the Peano θ-algebra generated by A_0.

Definition 1. *A* **semantic θ-schema** *([1]) is a system $\mathcal{S} = (X, A_0, A, R)$ where*
• *X is a finite non-empty set of object symbols*
• *A_0 is a finite non-empty set of label symbols*
• *$A_0 \subseteq A \subseteq \overline{A}_0$, where $\overline{A}_0$ is the Peano θ-algebra generated by A_0*
• *$R \subseteq X \times A \times X$ is a non-empty set which fulfills the following conditions*
- *$(C1)$ If $(x, \theta(u, v), y) \in R$ then $\exists\ z \in X$ such that $(x, u, z) \in R$ and $(z, v, y) \in R$*
- *$(C2)$ If $\theta(u, v) \in A$, $(x, u, z) \in R$, $(z, v, y) \in R$ then $(x, \theta(u, v), y) \in R$*
- *$(C3)$ $pr_2 R = A$*

Lets consider the θ-schema $\mathcal{S} = (X, A_0, A, R)$ and a set Ob of **simple objects**. Based on this set of objects we define a bijective mapping $ob : X \to Ob$.

Definition 2. *An* **interpretation** *of $\mathcal{S}$ is a tuple $\mathcal{I} = (Ob, ob, \{Alg_u\}_{u \in A})$ where:*
• *Ob is the set of the simple objects.*
• *$ob : X \longrightarrow Ob$ is a bijective function.*
• *$\{Alg_u\}_{u \in A}$ is a set of algorithms and we note by $dom(Alg_u)$ the set of all pairs of objects such that the algorithm Alg_u can be executed.*

An interpretation (I) defines a space $Y(\mathcal{I})$ which is named the **semantic space** generated by $\mathcal{I}$:

Definition 3. *([2]) Consider the θ-schemas $\mathcal{S} = (X, A_0, A, R)$ and $\mathcal{P} = (Y, B_0, B, Q)$. We define the relation $\mathcal{S} \sqsubseteq \mathcal{P}$ if $X \subseteq Y$ and $R \subseteq Q$.*

Proposition 1. *([2]) Consider the θ-schemas $\mathcal{S} = (X, A_0, A, R)$ and $\mathcal{P} = (Y, B_0, B, Q)$. If $\mathcal{S} \sqsubseteq \mathcal{P}$ then $A_0 \subseteq B_0$ and $A \subseteq B$.*

Proposition 2. *The relation $\sqsubseteq$ is reflexive, antisymmetric and transitive, therefore it is a partial order.*

3 Joining of two semantic schemas. Theoretical aspects

Let us consider we have two semantic schemas, $\mathcal{S}_i = (X^i, A_0^i, A^i, R^i)$ and $\mathcal{S}_j = (X^j, A_0^j, A^j, R^j)$ such that $A_0^i \cap A_0^j = \emptyset$. The joining of these schemas is done based on a set $A_{joint}^{ij} \neq \emptyset$ defined in the following proposition.

Proposition 3. *The link between two schemas S_i and S_j, $\mathcal{S}_i \neq \mathcal{S}_j$ is constructed according to a set of relations' labels, noted A_{joint}^{ij}, that is defined recursively as follows:*

$$A_{joint}^{ij} \leftarrow \bigcup_{p \geq 0} (A_{joint}^{ij})_p$$

where:

- $(A_{joint}^{ij})_0$ *is a finite set of labels u of the form $u = \theta(u_1, u_2)$, $u_1 \in A^i$ and $u_2 \in A^j$ or $u_1 \in A^j$ and $u_2 \in A^i$.*
- $(A_{joint}^{ij})_p$ *is a finite set of labels u of the form $u = \theta(u_1, u_2)$, $u_1 \in (A_{joint}^{ij})_{p-1} \cup A^i$ and $u_2 \in (A_{joint}^{ij})_{p-1} \cup A^j$ or $u_1 \in (A_{joint}^{ij})_{p-1} \cup A^j$ and $u_2 \in (A_{joint}^{ij})_{p-1} \cup A^i$*
- *there is $p_0 \geq 0$ such that $(A_{joint}^{ij})_{p_0} = \emptyset$ and $(A_{joint}^{ij})_{p_0} = (A_{joint}^{ij})_{p_0+1} = \ldots$*

Remark 1. From the Proposition 3 we obtain that A_{joint}^{ij} is a finite reunion of finite sets $(A_{joint}^{ij})_p$, $p = \overline{0, p_0 - 1}$ and thus, the set A_{joint}^{ij} is also a finite set of elements.

Definition 4. *We define the joining operation of two θ-schemas as the mapping $\otimes : dom(\otimes) \rightarrow \mathcal{S}_\otimes$ such that:*

$dom(\otimes) = \{(\mathcal{S}_i, \mathcal{S}_j) \mid X^i \cap X^j \neq \emptyset \text{ and } \exists A_{joint}^{ij} \neq \emptyset\}$
$\mathcal{S}_\otimes$ *is the set of the θ-schemas over $A_0^i \cup A_0^j$ for $\forall (\mathcal{S}_i, \mathcal{S}_j) \in dom(\otimes)$*

where $\mathcal{S}_i = (X^i, A_0^i, A^i, R^i)$ and $\mathcal{S}_j = (X^j, A_0^j, A^j, R^j)$ with $A_0^i \cap A_0^j = \emptyset$. We note:

$$\mathcal{S}_i \otimes \mathcal{S}_j = \mathcal{S}_{ij}$$

where $\mathcal{S}_{ij}$ is the semantic θ-schema: $\mathcal{S}_{ij} = (X^{ij}, A_0^{ij}, A^{ij}, R^{ij})$ such that:
$X^{ij} = X^i \cup X^j$
$A_0^{ij} = A_0^i \cup A_0^j$
$R_0^{ij} = R_0^i \cup R_0^j$
$A^{ij} = A^i \cup A^j \cup A_{joint}^{ij}$
$R^{ij} \subseteq X^{ij} \times A_{joint}^{ij} \times X^{ij}$

Proposition 4. *The semantic schema obtained by joining two schemas is not necessarily the supremum of them.*

Proof. Let us consider two disjunctive θ-schemas $\mathcal{S}_i = (X^i, A_0^i, A^i, R^i)$ and $\mathcal{S}_j = (X^j, A_0^j, A^j, R^j)$ with $A_0^i \cap A_0^j = \emptyset$.

If we note by $\mathcal{S} = (X, A_0, A, Z_{n_0})$ the supremum of $\mathcal{S}_i$ and $\mathcal{S}_j$ then from the way the supremum of two schemas is defined in [2] we obtain:
$X = X^i \cup X^j$, $A_0 = A_0^i \cup A_0^j$, $A = A^i \cup A^j$ and $Z_{n_0} = \bigcup_{m \geq 0} Z_m$ where:

$$\begin{cases} Z_0 = R_0^i \cup R_0^j \\ Z_{m+1} = \{(x, \theta(u,v), y) \in X \times A \times X \mid \exists z : \\ \qquad\qquad (x,u,z) \in Z_m, (z,v,y) \in Z_m\}, m \geq 0 \end{cases} \tag{1}$$

The sequence $\{Z_m\}_{m \geq 0}$ is a finite one because it satisfies the following property: there is a natural number m_0 such that $Z_{m_0} = Z_{m_0+1} = \ldots = \emptyset$

Let us consider that $\mathcal{S}_{ij} = \mathcal{S}_i \otimes \mathcal{S}_j$, $\mathcal{S}_{ij} = (X^{ij}, A_0^{ij}, A^{ij}, R^{ij})$.

From the way the $\mathcal{S}_{ij}$ is constructed by the joining operation we have that $X = X^{ij} = X^i \cup X^j$, $A_0 = A_0^{ij} = A_0^i \cup A_0^j$ and $Z_0 = R_0^{ij} = R_0^i \cup R_0^j$ but because one of the conditions for existing $\mathcal{S}_{ij}$ is that $A_{joint}^{ij} \neq \emptyset$ we obtain that:

$$A = A^i \cup A^j \subset A^{ij} = A^i \cup A^j \cup A_{joint}^{ij} \tag{2}$$

We know that $Z_{n_0} \subseteq X \times A \times X$ and $R^{ij} \subseteq X^{ij} \times A^{ij} \times X^{ij}$. Because $A_0^i \cap A_0^j = \emptyset$ (from the premise) we obtain that $Z_{n_0} = R^i \cup R^j$ which implies that:

$$Z_{n_0} \subseteq R^{ij}$$

Moreover, based on the way the relation "$\sqsubseteq$" is defined in Definition 3, from $X = X^{ij}$ and $Z_{n_0} \subseteq R^{ij}$ we obtain

$$\mathcal{S} \sqsubseteq \mathcal{S}_{ij} \tag{3}$$

which means:

$$sup\{\mathcal{S}_i, \mathcal{S}_j\} \sqsubseteq \mathcal{S}_i \otimes \mathcal{S}_j \tag{4}$$

4 The JOINER algorithm

In this section we present the algorithm that receives as input two schemas $\mathcal{S}_i$ and $\mathcal{S}_j$ and obtains at output the schema $\mathcal{S}_{ij} = \mathcal{S}_i \otimes \mathcal{S}_j$.

But before that we must introduce the notations used in the algorithm. Thus:

- $(M)_n$ where M is a set and n is a natural number, $n \geq 0$, represents the set M obtained at the n-th step of the algorithm.
- $(R_{new})_n$ represents the set of the relations labeled by elements of $(A_{joint}^{ij})_n$ that are obtained by composing:

  - some relations of R^i with some relations of R^j, that share common nodes of $(X_{com})_n$ if $n = 0$
  - some relations of $R^i \cup (R_{new})_{n-1}$ with some relations of $R^j \cup (R_{new})_{n-1}$ that share common nodes of $(X_{com})_n$, if $n \geq 1$.

The JOINER algorithm:
Input: $\mathcal{S}_i = (X^i, A_0^i, A^i, R^i)$, $\mathcal{S}_j = (X^j, A_0^j, A^j, R^j)$
if $(A_0^i \cap A_0^j \neq \emptyset)$
  return $error\_code$
if $((X^i \cap X^j = \emptyset) \vee (A_{joint}^{ij} = \emptyset))$
  return $error\_code$
step 0 :
$(X_{com})_0 \leftarrow X^i \cap X^j$
$X^{ij} \leftarrow X^i \cup X^j$
if $((X_{com})_0 \neq \emptyset)$
 $(R_{new})_0 \leftarrow \{(x, \theta(u, v), y) \in X^i \times (A_{joint}^{ij})_0 \times X^j \mid \exists z \in (X_{com})_0 : (x, u, z) \in R^i \wedge (z, v, y) \in R^j\}$
 $\cup \{(x, \theta(u, v), y) \in X^j \times (A_{joint}^{ij})_0 \times X^i \mid \exists z \in (X_{com})_0 : (x, u, z) \in R^j \wedge (z, v, y) \in R^i\}$
 $X_{new} \leftarrow pr_1((R_{new})_0) \cup pr_3((R_{new})_0)$
 $n \leftarrow 1$
 step n : **while** $((R_{new})_{n-1} \neq \emptyset)$
  $(X_{com})_n \leftarrow (X_{new} \cap X^i) \cup (X_{new} \cap X^j)$
  $(R_{new})_n \leftarrow \emptyset$
  if $((X_{com})_n \neq \emptyset)$
   $(R_{new})_n \leftarrow \{(x, \theta(u_1, u_2), y) \in X^i \times (A_{joint}^{ij})_n \times X^j \mid \exists z \in (X_{com})_n : (x, u_1, z) \in R^i \cup (R_{new})_{n-1} \wedge (z, u_2, y) \in R^j \cup (R_{new})_{n-1}\}$
   $\cup \{(x, \theta(u_1, u_2), y) \in X^j \times (A_{joint}^{ij})_n \times X^i \mid \exists z \in (X_{com})_n : (x, u_1, z) \in R^j \cup (R_{new})_{n-1} \wedge (z, u_2, y) \in R^i \cup (R_{new})_{n-1}\}$
   $X_{new} \leftarrow pr_1((R_{new})_n) \cup pr_3((R_{new})_n)$
  endif
  $n \leftarrow n + 1$
 endwhile
 $R_0^{ij} \leftarrow R_0^i \cup R_0^j$
 $A_0^{ij} \leftarrow pr_2(R_0^{ij})$
 $R^{ij} \leftarrow \bigcup_{n \geq 0} (R_{new})_n \cup R^i \cup R^j$
 $A^{ij} \leftarrow pr_2(R^{ij})$
endif
Output: $\mathcal{S}_{ij} = (X^{ij}, A_0^{ij}, A^{ij}, R^{ij})$

Remark 2. An algorithm is consider *well-defined* if it fulfills the following conditions:
- $(A1)$ its execution ends in a finite time
- $(A2)$ the resulted output is correctly defined

Proposition 5. *The* **JOINER algorithm** *is a well-defined algorithm.*

Proof. From Remark 2 we obtain that for each input $\mathcal{S}_i = (X^i, A_0^i, A^i, R^i)$ and $\mathcal{S}_j = (X^j, A_0^j, A^j, R^j)$, with $A_0^i \cap A_0^j = \emptyset$ the JOINER algorithm must:
- $(A1)$ end in a finite time
- $(A2)$ obtain at output the semantic schema S_{ij}

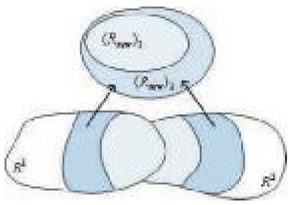

Fig. 1. A graphical illustration of the R_{new} defining method

From the way the set R^{ij} is constructed by the algorithm, we have:
$R_{new} \leftarrow \bigcup_{n\geq 0}(R_{new})_n$
$R^{ij} \leftarrow R^i \cup R^j \cup R_{new}$

In order to prove $(A1)$ we have to prove that there is $n_0 \geq 0$ such that $(R_{new})_{n_0} = \emptyset$ and thus we obtain:

$$R_{new} \leftarrow \bigcup_{n\geq 0}^{n_0}(R_{new})_n$$

But $(R_{new})_n \subseteq X^{ij} \times (A^{ij}_{joint})_n \times X^{ij}$, $n \geq 0$. As we have underlined in Remark 1 the set A^{ij}_{joint}:

$$A^{ij}_{joint} \leftarrow \bigcup_{p\geq 0}(A^{ij}_{joint})_p$$

is a finite set because $\exists p_0 \geq 0$ such that $(A^{ij}_{joint})_{p_0} = \emptyset$. This implies that $\exists n_0$, $n_0 \leq p_0$ such that $(R_{new})_{n_0} = \emptyset$. The condition $(A1)$ is it proved.

The condition $(A2)$ implies to prove that the system $\mathcal{S}_{ij} = (X^{ij}, A^{ij}_0, A^{ij}, R^{ij})$ obtained using the JOINER algorithm is a semantic θ-schema. This means we have to prove that $\mathcal{S}_{ij}$ respects the conditions (C1) $\div$ (C3).

For $\mathcal{S}_{ij}$, the condition (C1) is:
If $(x, \theta(u, v), y) \in R^{ij}$ then there is $z \in X^{ij}$ such that $(x, u, z) \in R^{ij}$ and $(z, v, y) \in R^{ij}$.

Because $R^{ij} = R^i \cup R^j \cup_{n\geq 0} (R_{new})_n$ then one of the following inclusions is fulfilled for $(x, \theta(u, v), y)$:
1) $(x, \theta(u, v), y) \in R^i \cup R^j$
2) $\exists n \geq 0 :\ (x, \theta(u, v), y) \in (R_{new})_n$

For the case 1): lets suppose that $(x, \theta(u, v), y) \in R^i$.

Because $\mathcal{S}_i$ is a θ-schema implies that R^i fulfills the condition (C1), thus: $\exists z \in X^i : (x, u, z) \in R^i$ and $(z, v, y) \in R^i$

Obviously, $X^i \subseteq X^{ij}$ and $R^i \subseteq R^{ij}$ which implies: $\exists z \in X^{ij} : (x, u, z) \in R^{ij}$ and $(z, v, y) \in R^{ij}$

Similar proof for $(x, \theta(u, v), y) \in R^j$. Thus, we obtain that the case 1) fulfills the condition (C1).

For the case 2): if $(x, \theta(u, v), y) \in (R_{new})_n$ then $\exists z \in (X_{com})_n$, $n \geq 0$ such that: $(x, u, z) \in (R^i \cup R^j) \cup (R_{new})_{n-1}$ and $(z, v, y) \in (R^i \cup R^j) \cup (R_{new})_{n-1}$.

Because $R^i \cup R^j \subseteq R^{ij}$ and $(R_{new})_{n-1} \subseteq R^{ij}$ (if $n = 0$ we can consider that $(R_{new})_{-1} = \emptyset$), results $(R^i \cup R^j) \cup (R_{new})_{n-1} \subseteq R^{ij}$.

This implies $(x, u, z) \in R^{ij}$ and $(z, v, y) \in R^{ij}$ and because $(X_{com})_n \subseteq X^{ij}$ we obtain that the condition (C1) is true also for the case 2).

The condition (C2) for the schema $\mathcal{S}_{ij}$ is:
If $\theta(u, v) \in A^{ij}$, $(x, u, z) \in R^{ij}$, $(z, v, y) \in R^{ij}$ then $(x, \theta(u, v), y) \in R^{ij}$.

Because $A^{ij} = pr_2(R^{ij}) = pr_2(R^i \cup R^j \cup_{n\geq 0} (R_{new})_n) = pr_2(R^i \cup R^j) \cup_{n\geq 0} pr_2((R_{new})_n)$ then $\forall \theta(u, v) \in A^{ij}$ only one of the following cases arises:
1) $\theta(u, v) \in pr_2(R^i \cup R^j)$
2) $\exists n \geq 0$: $\theta(u, v) \in pr_2((R_{new})_n)$

For the first case we take $\theta(u, v) \in pr_2(R^i)$. The way the set R^i is defined asks for $u \in pr_2(R^i)$ and $v \in pr_2(R^i)$.

Thus $\{(x, u, z), (z, v, y)\} \subseteq R^i$ which implies $(x, \theta(u, v), y) \in R^i \subseteq R^{ij}$.

For $\theta(u, v) \in pr_2(R^j)$ we obtain a similar result. Thus, the case 1) fulfills the condition (C2).

The second case $\theta(u, v) \in pr_2((R_{new})_n)$ implies:
- $u \in pr_2((R^i \cup R^j) \cup (R_{new})_{n-1}) \Rightarrow (x, u, z) \in (R^i \cup R^j) \cup (R_{new})_{n-1}$
- $v \in pr_2((R^i \cup R^j) \cup (R_{new})_{n-1}) \Rightarrow (z, v, y) \in (R^i \cup R^j) \cup (R_{new})_{n-1}$

where $\exists z \in (X_{com})_n \subseteq X^{ij}$. Thus, we obtain $(x, \theta(u, v), y) \in (R_{new})_n \subseteq R^{ij}$ and the condition (C2) is fulfilled again.

The condition (C3): $A^{ij} = pr_2(R^{ij})$ comes from the construction of $\mathcal{S}_{ij}$.

Remark 3. The order in which the two schemas S_i and S_j are given as input for the **JOINER algorithm** is not important because the joining operation is commutative:

$$\mathcal{S}_i \otimes \mathcal{S}_j = \mathcal{S}_j \otimes \mathcal{S}_i$$

5 A case study

As an exemplification of how this method works we will consider the following two θ-schemas:

$\mathcal{S}_1 = (X^1, A_0^1, A^1, R^1)$ where:
- $X^1 = \{P, Y, Z, Q\}$
- $A^1 = A_0^1 = \{a, b\}$
- $R^1 = R_0^1 = \{(P, a, Y), (Z, b, Q)\}$

The second schema is $\mathcal{S}_2 = (X^2, A_0^2, A^2, R^2)$ where:
- $X^2 = \{Y, Z, W\}$
- $A_0^2 = \{u, m\}$, $A^2 = A_0^2 \cup \{\theta(u, m)\}$
- $R_0^2 = \{(Y, u, Z), (Z, m, W)\}$, $R^2 = R_0^2 \cup \{Y, \theta(u, m), W\}$

We consider for both schemas interpretations in the domain of the relations that occur between the members of a family. Thus, the interpretation for $\mathcal{S}_1$ is $\mathcal{I}_1 = (Ob_1, ob_1, \{Alg_u^1\}_{u \in A^1})$ for:
- $Ob_1 = \{$"$Peter$", "$John$", "$Mary$", "Ana"$\}$
- $ob_1 : X^1 \rightarrow Ob_1$, $ob_1(P) =$ "$Peter$", $ob_1(Y) =$ "$John$", $ob_1(Z) =$ "$Mary$", $ob_1(Q) =$ "Ana"

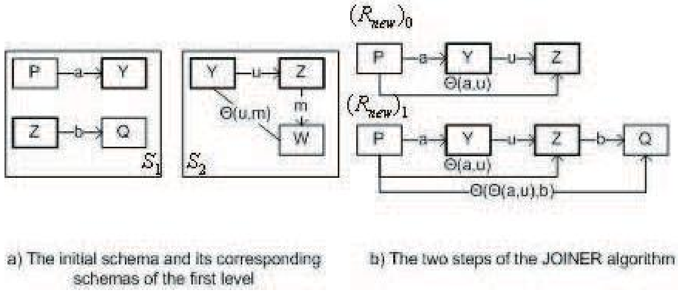

Fig. 2. A case study

- **Algorithm** $Alg_a^1(ob_1(P), ob_1(Y))$
  $return\ ob_1(P) + \text{“}is\ father\ of'' + ob_1(Y)$
  End of algorithm
  Algorithm $Alg_b^1(ob_1(Z), ob_1(Q))$
  $return\ ob_1(Z) + \text{“}is\ mother\ of'' + ob_1(Q)$
  End of algorithm

The interpretation of the θ-schema $\mathcal{S}_2$ is noted $\mathcal{I}_2 = (Ob_2, ob_2, \{Alg_u^2\}_{u\in A^2})$ where:
-$Ob_2 = \{\text{“}John'', \text{“}Mary'', \text{“}Nicolas''\} \subseteq Ob$
-$ob_2 : X^2 \to Ob_2$, $ob_2(Y) = \text{“}John''$, $ob_2(Z) = \text{“}Mary''$, $ob_2(W) = \text{“}Nicolas''$
-**Algorithm** $Alg_u^2(ob_2(Y), ob_2(Z))$
  $return\ ob_2(Z) + \text{“}is\ husband\ of'' + ob_2(Q)$
  End of algorithm
  Algorithm $Alg_m^2(ob_2(Z), ob_2(W))$
  $return\ ob_2(Z) + \text{“}is\ sister\ of'' + ob_2(W)$
  End of algorithm
  Algorithm $Alg_{\theta(u,m)}^2(o_1, o_2)$
  $return\ o_1 + \text{“}is\ brother-in-law\ of'' + o_2$
  End of algorithm

For $A_{joint}^{12} = \{\theta(a, u), \theta(\theta(a, u), b)\}$ the θ-schema $\mathcal{S}_{12} = (X^{12}, A_0^{12}, A^{12}, R^{12})$ is constructed in two steps:
step 1:
$(X_{com})_0 = \{Y, Z\}$ and $(R_{new})_0 = \{(P, \theta(a, u), Z)\}$
step 2:
$(X_{com})_1 = \{P, Z\}$ and $(R_{new})_1 = \{(P, \theta(\theta(a, u), b), Q)\}$

Thus, we obtain:
$X^{12} = X^1 \cup X^2$

$R^{12} = R^1 \cup R^2 \cup (R_{new})_0 \cup (R_{new})_1$, $R_0^{12} = R_0^1 \cup R_0^2$
$A^{12} = A^1 \cup A^2 \cup \{\theta(a,u), \theta(\theta(a,u),b)\}$, $A_0^{12} = A_0^1 \cup A_0^2$.

The interpretation of $\mathcal{S}_{12}$ is
$\mathcal{I}_{12} = (Ob_{12}, ob_{12}, \{Alg_u^{12}\}_{u \in A^{12}})$ where:
- $Ob_{12} = Ob_1 \cup Ob_2$
- $ob_{12} : X^{12} \to Ob_{12}$, $ob_{12}(x) = ob_i(x)$, $\forall x \in X^i$, $i = \overline{1,2}$
-

$$Alg_c^{12}(o_1, o_2) = \begin{cases} Alg_c^1(o_1, o_2), c \in A^{12} \cap A^1 \\ Alg_c^2(o_1, o_2), c \in A^{12} \cap A^2 \end{cases} \quad (5)$$

Algorithm $Alg_{\theta(a,u)}^{12}(o_1, o_2)$
$return\ o_1 + \text{“}is\ father - in - law\ of'' + o_2$
End of algorithm
Algorithm $Alg_{\theta(\theta(a,u),b)}^{12}(o_1, o_2)$
$return\ o_1 + \text{“}is\ grandfather\ of'' + o_2$
End of algorithm

6 Conclusions

As we have underlined from the very beginning of this paper this work will be continued in order to be implemented in the reasoning mechanism of a distributed system which uses the semantic schema concept as the representation and processing model.

References

[1] **N. Ţăndăreanu**:- Semantic Schemas and Applications in Logical representation of Knowledge, Proceedings of the 10th International Conference on Cybernetics and Information Technologies, System and Applications (CITSA2004), July 21-25, 2004, Orlando, Florida

[2] **N. Ţăndăreanu, M.Ghindeanu**:- A three-level distributed knowledge system based on semantic schemas, Proceedings of DEXA2005, IEEE Computer Society, Copenhagen, 2005, 423-427