Extending XML Conditional Schema Representations with WordNet Data

Nicolae Țăndăreanu, Mihaela Colhon, and Cristina Zamfir

***Abstract*—**Conditional Knowledge Representation and Reasoning represents a new brand of KR&R, for which several formalisms have been developed. In this paper we define XML Language Specifications for a graph-based representation formalism of such knowledge enriched with WordNet linguistic knowledge. Our task is to detect when pairs of words (in our formalism they are named objects) could be linked by means of *is_a* and *part_of* relationships.

***Keywords*—**Binary Relation, Conditional Knowledge, Conditional Schema, XML Schema Language, WordNet, hypernyms, meronyms.

I. Introduction

THE Semantic Web is the abstract representation of data on the World Wide Web, based on the Resource Description Framework (RDF) standards and other standards to be defined. It is being developed by the World Wide Web Consortium (W3C) in collaboration with a large number of researchers and industrial partners.

XML - derived from *eXtensible Markup Language* - is a general purpose markup language which supports the sharing of structured data across different information systems. As the popularity of XML increases substantially, the importance of XML languages to describe the structure and semantics of human knowledge also increases ([9]). Although there have proposed about dozen XML languages, no comprehensive mathematical analysis of them have been made.

The database and web community use terms such as schema and XML document (or file). *XML Schema* is considered to be an abstract definition of the set of conforming XML documents.

XML Schema is the XML schema language recommended by the W3C in order to define constraints which are further used to describe a class of documents ([10]).

The spirit and the development approach behind the Semantic Web require as much as possible formal data/knowledge to be provided in formats that others can read and interpret for unforeseen purposes. In other words:

- Automatically processable **meta-data**;
- Presented in a **standard form**;
- Allow flexible and **dynamic interpretation for unforeseen purposes**.

We believe that providing a framework in abstract terms is important to understand the various aspects of a XML language description in order to facilitate its efficient implementation. Towards this goal, in the present paper we propose a XML Schema formalism for representing and processing conditional knowledge texts. We also believe that is important to have a mathematical framework to study when an efficient operation is possible and when it is not.

Many natural language processing applications depend on ontologies such as WordNet in order to obtain prior knowledge about the semantic relationships between words. Enriching the XML Schema structure with WordNet linguistic knowledge in order to integrate new information that can be useful in the reasoning process is another goal of our proposal.

A. Our purpose

Let us consider a collection of knowledge pieces, more precisely of conditional knowledge pieces (texts in natural languages that contains if-then sentences), for which a conditional schema formalism has been developed ([2], [13] and [14]). One may want to find the union of two (or more) such schemas. Then, it could be interested in finding out if a specific knowledge piece (KP) that is valid with respect to both schemas remains valid in the resulted union schema, and conversely, if a KP that is invalid with respect to both schemas remains invalid in the union schema.

Another problem could appear if a schema representation would evolve in a new version during an update process. In this case, one can be interested in constructing the intersection of the old and new schema representation to determine, for example, which knowledge remains unchanged (the closure properties of conditional schemas under boolean operations: union, intersection and difference).

In XML query processing, computing answers from multiple documents (in our case, knowledge pieces) may require to compute union of the internal representation systems – that is, union of the conditional schemas. Such issues are directly related with the efficient implementation of an XML language and following this idea, we consider that the formalism we propose here meet the needs of this formalism.

B. Conditional schema structure

The conditional knowledge representation and processing mechanism was developed under the name of *conditional schema*. The formal concept of conditional schema was introduced in [13], being defined by means of a tuple of eight

components:

$$S = (Ob, C_s, E_r, A, V, B_{cr}, h, f)$$

such that:

- *Ob* is a set of the object names. This set is divided into two subsets corresponding to the individual and abstract objects as follows:
  $Ob = Ob_{ind} \cup Ob_{abstr}$ and $Ob_{ind} \cap Ob_{abstr} = \varnothing$
- C_s is a finite set of symbols named conditional symbols; every symbol $t \in C_s$ denotes a boolean mapping $t : Ob \to \{true, false\}$ such that for an arbitrary object *n* we consider *t*(*n*)=*true* if the condition attached to *t* is satisfied for *n* and *t*(*n*)=*false*, otherwise
- E_r is a finite set of symbols used to designate conditional binary relations over *Ob*×*Ob*;
- *A* is a set of attribute name for the elements of *Ob*;
- *V* is a set of values for the elements of *A*;
- $B_{cr} \subseteq 2^{((Ob \times I) \times (Ob \times I) \times (C_s \cup \{T\}))}$ is the set of the conditional binary relations and $I = \{i, a\}$, where *i* is used to designate individual objects and *a* is used to specify abstract objects.
- $h : E_r \to B_{cr}$ is a mapping that assigns a conditional binary relation for every symbol of E_r;
- $f : Ob \to 2^{A \times V}$ is a mapping that assigns initial knowledge to the objects of *Ob*.

In this formalisms, all the individual objects are characterized by pairs of the form *(attr,value)*, where *attr* represents an attribute name and *value* gives the value of the corresponding attribute.

In what concerns the conditional symbols or mappings of the set C_s we use the following notations:

$$((n, \omega_1), (m, \omega_2)) \in_c h(r)$$

$$p = Cond_r((n, \omega_1), (m, \omega_2))$$

to denote that $((n, \omega_1), (m, \omega_2), p(n))$ belongs to the conditional relation *h*(*r*).

C. Conditional graph structure

The inference mechanism corresponding to this structure is based on a graph model of the schema representations, named *conditional graph* ([13]), model which can be easily identified in the XML specifications we define in the following section in order to encapsulate the conditional schema entities.

The conditional graph ([13]) generated by a conditional schema $S = (Ob, C_s, E_r, A, V, B_{cr}, h, f)$ is the system:

$$G_S = (X \cup Z, \Gamma_X \cup \Gamma_Z)$$

where:

- $X \subseteq Ob \times I$ is the set of nodes, such that $x \in X$ if and only if there are $r \in E_r$, $y \in X$ such that $(x, y) \in_c h(r)$ or $(y, x) \in_c h(r)$
- $\Gamma_X \subseteq X \times E_r \times X$ and $((n, \omega_1), r, (m, \omega_2)) \in \Gamma_X$ if and only if $((n, \omega_1), (m, \omega_2)) \in_c h(r)$
- $Z = \{f(x) \mid x \in Ob\}$ and $\Gamma_Z = \{(f(x), x) \mid x \in Ob\}$
- There are two kinds of arcs specified by Γ_X and Γ_Z respectively. An arc from Γ_X is named **arc of first category** and an arc from Γ_Z is an **arc of second category**.

The graphical representation of a conditional graph $G_S = (X \cup Z, \Gamma_X \cup \Gamma_Z)$ illustrates each element of *X* inside a rectangle and for each $(x, r, y) \in \Gamma_X$ an arc from the node $x \in X$ to $y \in X$ is drawn and we put the label $r \in E_r$ on this arc.

Let $G_S = (X \cup Z, \Gamma_X \cup \Gamma_Z)$ be the conditional graph generated by the conditional schema $S = (Ob, C_s, E_r, A, V, B_{cr}, h, f)$. Let n_1 and n_{k+1} be two arbitrary objects of the set *Ob*. A *path* from n_1 to n_{k+1} in G_S is considered as the pair of the form:

$$([(n_1, \omega_1), ..., (n_{k+1}, \omega_{k+1})], [r_1, ..., r_k])$$

such that the following conditions are fullfiled:

- $(n_1, \omega_1), ..., (n_{k+1}, \omega_{k+1}) \in X$
- $r_1, ..., r_k \in E_r$
- $((n_j, \omega_j), r_j, (n_{j+1}, \omega_{j+1})) \in \Gamma_X, j = \overline{1,k}$

We denote by $Path(n_1, n_{k+1})$ the set of all paths from n_1 to n_{k+1} in G_S. For each

$$d = ([(n_1, \omega_1), ..., (n_{k+1}, \omega_{k+1})], [a_1, ..., a_k]) \in Path(n_1, n_{k+1})$$

there are $t_1, ..., t_k \in C_s$, such that:

$$t_j = Cond_{a_j}((n_j, \omega_j), (n_{j+1}, \omega_{j+1})), j = \overline{1,m}$$

We note by *CS*(*d*) the list of all conditional mappings of the path *d*, that is, *CS*(*d*) = [t_1,..., t_k] and with $Nodes(d)$ the set of all nodes listed on the path *d*, that is:

$$Nodes(d) = \{(n_1, \omega_1), ..., (n_{k+1}, \omega_{k+1})\}$$

As it will be detalied in what follows, the reasoning process developed on this graph-based structure is formalised based on the graph paths with respect to a certain node of each path. For this reason we will note this special node with *Target*(*d*), for each path *d* in G_S.

Let us reconsider the conditional graph $G_S = (X \cup Z, \Gamma_X \cup \Gamma_Z)$ and

$$d = ([(n_1, \omega_1), ..., (n_{k+1}, \omega_{k+1})], [a_1, ..., a_k])$$

a path in G_S. The target node *Target*(*d*) is determined as follows:

1) if there are more than one individual node in *Nodes*(*d*) with $(n_1, i) \in Nodes(d)$ then *Target*(*d*) is the latest individual node that follows n_1 in the set *Nodes*(*d*)

2) otherwise, *Target*(*d*)= n_1

For the conditional mappings related to the paths of a conditional graph we use the following notations:

1) For each conditional mapping t_j of *CS*(*d*), $1 \le j \le k$ we say

$$t_j[d] = on$$

if and only if $t_j = T$ or $\exists x \in Near_{t_j}(Target(d)) : t_j(x) = on$

where:

- $Near_{t_j}(Target(d)) = \{Target(d)\}$ if the object *Target*(*d*) includes the features asked by the conditional symbol t_j; otherwise
- $Near_{t_j}(Target(d)) = \{n\}$, $n \neq Target(d)$, if *Target*(*d*) is connected with *n* by means of *is_a* or *is_part_of* relations and *n* includes the features asked by t_j;

2) otherwise, we note

$$t_j[d] = off$$

D. Conditional schema inference mechanism

We consider a superset E_r^* of the set E_r, such that $E_r \subseteq E_r^*$ and a partial binary operation:

$$\varphi : E_r^* \times E_r^* \to E_r^*$$

We denote by $List(E_r^*)$ the list of nonempty lists containing elements from E_r^*. In [13] we defined the partial mapping

$$\Phi : List(E_r^*) \to E_r^*$$

as follows:

(1) $\Phi([a_1]) = b_1$

(2) $\Phi([a_1,...,a_k]) = b_k, k \geq 2$ such that:

$$b_1 = a_1, b_2 = \varphi(b_1, a_2), ..., b_k = \varphi(b_{k-1}, a_k)$$

In [13] we give all the proofs regarding the correctness of the definition for the mapping Φ.

The answer of a conditional schema is a sentence in a natural language. Because we choose to define this system for knowledge pieces written in English, the answer is also given in English. If we note by *Sen* the set of such sentences, then the mapping that generates these sentences was defined as:

$$g : Ob \times E_r^* \times Ob \times \{on, off\} \to Sen$$

such that $g(x, a, y, on)$ specifies the property given by the semantics corresponding to the label *a*, while $g(x, a, y, off)$ specifies the contrary property.

If we consider the path

$$d = ([(n_1, \omega_1), ..., (n_{k+1}, \omega_{k+1})], [a_1, ..., a_k]) \in Path(n_1, n_{k+1})$$

in the conditional graph G_S of a conditional schema *S* such that:

- $[a_1, ..., a_k] \in dom(\Phi)$
- $CS(d) = [t_1, ..., t_k]$

we define *ans*(*d*) as follows:

- if $t_1[d] = ... = t_k[d] = on$, we take
  $ans(d) = g(n_1, \Phi([a_1, ..., a_k]), n_{k+1}, on)$
  for $(n_1, \Phi([a_1, ..., a_k]), n_{k+1}, on) \in dom(g)$
- if there is $t_j[d] = off$, for $j \in \{1, ..., k\}$ then
  $ans(d) = g(n_1, \Phi([a_1, ..., a_k]), n_{k+1}, off)$
  for $(n_1, \Phi([a_1, ..., a_k]), n_{k+1}, off) \in dom(g)$
- *ans*(*d*) = *undefined*, otherwise.

Finally, the answer mapping corresponding to the path of a conditional graph G_S generated from the conditional schema *S* is the mapping:

$$Ans : Ob \times Ob \to 2^{Sen}$$

defined as follows:

1) If there is $d \in Path(n_1, n_{k+1})$ such that *ans*(*d*) ≠ *undefined* then

$$Ans(n_1, n_{k+1}) = \bigcup_{d \in Path(n_1, n_{k+1})} \{ans(d)\}$$

2) Otherwise, $Ans(d) = unknowm$

II. WORDNET AS LEXICAL RESOURCE

WordNet ([17],[18]) is a thesaurus with defined semantic vocabulary set. Many natural language processing applications depend on ontologies such as WordNet in order to obtain prior knowledge about the semantic relationships between words.

WordNet is a lexical network of English words organized based on their parts of speech (nouns, verbs, adjectives and adverbs) into networks of synomym sets named *synsets*. A synset is a group of words, connected by meaning. It represents one underlying lexical concept which is interlinked with a variety of relations (for this reason, a polysemous word will appear in more than one synset, one for each sense). Synsets of WordNet are connected with semantic relationships. For example, nouns are linked in terms of *hypernym*, *hyponym*, *coordinate term*, *holonym*, and *meronym* relationship; verbs have the relationships of *hypernym*, *troponym*, *entailment*, and *coordinate terms*.

A synset ID is assigned to every word. Words in the same synset have the same synset ID. In this manner, each (English) word gets several entries in the WordNet database, and for each entry a different synset ID is assigned.

It is considered that WordNet represents a valuable resource for human language technology and more generally, for knowledge processing communities.

In the latest years, the development of WordNets for languages other than English was encouraged by the desire and the necessity to built a uniform ontological infrastructure across languages that will simplify machine translation. By storing the wordnets in a central lexical database system, a large multilingual database, named Euro-WordNet (EWN) was created, where the English synsets from WordNet 1.5 function as an Inter-Lingual Index (ILI).

In this vision, translation from a language to another will exploit the EWN principle of multilingual conceptual alignment of monolingual semantic networks via ILI. More precisely, ILI is a set of pivot nodes that allows the linkage between concepts belonging to different wordnets. The advantages of an interlingua mechanism as ILI are well known in Machine Translation (MT). In this database it is possible to go from one synset in a wordnet to a synset in another wordnet that is linked by the same WordNet 1.5 concept. In principle, multilinguality is achieved by adding an equivalence relation for each synset in a language to the closest synset in WordNet 1.5.

The backbone of the noun WordNet subnetwork is the subsumption hierarchy made from the *hyponymy* and *hypernymy* relations. The directions of links (relations) inside this subnetwork may vary ([6], [15]):

- among upwords:
  - ***hypernymy*** (IS-A): *Y is a hypernym of X if every X is a (kind of) Y*
  - ***meronymy*** (PART-OF): *Y is a meronym of X if Y is a part of X*
- downwords:
  - ***hyponymy*** (REVERSE IS-A): *Y is a hyponym of X if every Y is a (kind of) X*
  - ***holonymy***: *Y is a holonym of X if X is a part of Y*
- horizontal: ***antonymy*** and ***synonymy***

a) Hypernyms in WordNet. ([6], [8]) A hypernym is a word that is more generic than a given word. Only verbs and nouns can have hypernyms. Hypernymy is a relation between synsets. Thus, it is a semantical relation.

b) Meronyms in WordNet ([8], [11]). A meronym corresponds to the semantic meronym relation, also called the *part-whole* relation. A word X is a meronym of a word Y, if you can apply the sentence *An Y has an X* or *An X is a part of a Y*. This relation only holds for nouns. The reflexive relation is called *holonym* relation. Noun pairs are labeled as meronyms if there exists a path traversing only meronym and hypernym links between the nouns.

III. XML CONDITIONAL SCHEMA SPECIFICATIONS

Data structural information defined by our XML Conditional Schema formalism describes precisely the concepts and the relations contained in the represented Conditional Knowledge Piece as will be detailed in what follows.

	XML Conditional Schema
Data Type	XML Conditional Schema data are of a large number of built-in data types including string, boolean, numbers, etc.
Structure	XML Conditional Schema representations are organized in a nested data structure, each element having several attributes. The top-most elements are the objects and the relations that exist between them. The rules attached to the conditional relations are identified by the conditional symbols.
Relation	XML Conditional Schema supports inheritance through the resulted objects hierarchy. It does not support multiple-inheritance.

Starting from the conditional schema representation proposed in [9], we defined the XML Conditional Schema formalism as the abstract data model with the following components (or building blocks):

- blocks to encapsulate individual and abstract objects

<**node**>
<**type**> individual/abstract </**type**>
<**object**> object name </**object**>
<**parameters**>
<**parameter**>
<**type**> value type </**type**>
<**name**> attribute name </**name**>
<**value**> attribute value </**value**>
</**parameter**>
...
</**parameters**>
</**node**>

The types for attributes value can be: boolean (that corresponds to the pairs *yes*/*no* or *true*/*false*), string and numeric.

- blocks for binary predicates relating two objects/entities; the conditional symbol tag is only for conditional relations

<**relation**>
<**from**> start node </**from**>
<**to**> end node </**to**>
<**label**> relation name </**label**>
<**cond_symb**> conditional symbol id </**cond_symb**>
</**relation**>

- block for describing the conditions related to the conditional symbols; these conditional symbols are seen as boolean functions applied on individual objects and described by means of a set of condition that must be all satisfied in order to get the **on** value:

<**cond_symb**>
<**id**> conditional symbol id </**id**>
<**condition**>
...
</**condition**>
...
</**cond_symb**>

In the proposed formalism, the conditional symbols can be of two types:

  - one type points out to the existence of a relation between the object to which the conditional symbols is applied and another one (noted here with object2)

<**condition**>
<**type**> relational </**type**>
<**object2**> object2 name </**object2**>
<**relation**> relation name </**relation**>
</**condition**>

This kind of condition ask for a certain relation to exist between the object for which the condition is verified and the object specified by the **object2** tag.

Example: "*Peter is student of Nicolae if Nicolae is his teacher*."

The condition "*if Nicolae is his teacher*" is a relational condition and can be represented by:

<**condition**>
<**type**> relational </**type**>
<**object2**> Nicolae </**object2**>
<**relation**> is_teacher_of </**relation**>
</**condition**>

- the second kind request a particular value attribute for the object the conditional symbol is applied or another object of the conditional schema

<**condition**>
<**type**> attr/value </**type**>
<**attribute**> attribute name </**attribute**>
<**value**> attribute value </**value**>
</**condition**>

This type of condition addresses for a certain pair of attribute and value to be fulfilled by the object the condition is verified.

Example: "*John can join the team if his medical tests are good*"

The condition "*if his medical tests are good*" can be repesented as follows:

<**condition**>
<**type**> attr/value </**type**>
<**attribute**> medical_tests </**attribute**>
<**value**> good </**value**>
</**condition**>

A. XML-conditional knowledge base

As an exemplification for the formalisms already introduced, in this section we give an example of a XML-conditional knowledge base using the XML Conditional Schema specifications.

A XML-conditional knowledge base is an XML file which contains the instances for the XML Conditional Schema entities and their corresponding values. As every XML file, such knowledge base begins with a prolog which contains a declaration that identifies the document as a XML document:

```
<?xml version="1.0" encoding="UTF-8"
          standalone="no"?>
```

Example of XML Conditional Knowledge Base. Let us take the following conditional knowledge piece:

Generally, birds fly but penguins do not fly. Every animal is a flying animal if it has wings and flies. Every flying animal is a mobile animal. Also every animal which has legs is considered mobile animal. Bob is a penguin and Tweety is a small bird. Bob is friend with Tweety.

A1. The Conditional Schema Representation

The knowledge described in text corresponds to the following conditional schema formalism:

- $\begin{cases} Ob = \{Bob, Tweety, penguin, bird, animal, mob\_animal, \\ flying\_animal\} \end{cases}$

such that $Ob_{ind} = \{Bob, Tweety\}$ and Ob_{abstr} = {*penguin, bird, animal, mobile_animal,flying_animal*}

- $C_s = \{q1, q2\}$ where $q1$ stands for the condition "*if it has wings and flies*" and for $q2$ the condition is "*if it has legs*".
- $E_r = \{is\_a, is\_friend\_with\}$ where:

$h(is\_a) = \{(((Bob,i),(penguin,a)),T),$
$(((Tweety,i),(bird,a)),T),$
$(((animal,a),(flying\_animal,a)),q1),$
$(((animal,a),(mob\_animal,a)),q2)\}$
$h(is\_friend\_with) = \{(((Bob,i),(Tweety,i)),T)\}$

where T stands for *true* and denotes a classical binary relation (that is unconditionally fulfilled)

- $A = \{fly, wings, legs, size\}$
- $\begin{cases} R_1(x): IF\ V_x(fly) = yes \wedge V_x(wings) = yes\ THEN \\ q1(x) = on\ ELSE\ q1(x) = off \\ R_2(x): IF\ V_x(legs) = yes\ THEN\ q2(x) = on\ ELSE \\ q2(x) = off \end{cases}$
- $f(Tweety) = \{(size, small)\}$

Automatic extraction of semantic relations between nouns or objects from text corpora is important to many Natural Language Processing tasks.

In the considered conditional knowledge piece, no relation is specified between *penguin*, *bird* and *animal*. Also, no feature for these objects is specified. In this case, it will be useful if we can enrich the information described in text with other knowledge related to these ones. Following this idea, the lexical data existing in WordNet can be a very useful resource for this updating process.

Indeed, we can make use of declarative hypernym/hyponym (*is-a*) and meronym/holonym (*part-of*) hierarchies provided by WordNet. The *is-a* relations will be represented in the conditional graph by arcs of first category, linking pairs of objects from the conditional schema, while *part-of* relationships will be identified by arcs of second category, connecting a node with attribute-value pairs.

A2. Update the Conditional Schema with hypernym (is_a) relationships

The hypernyms of **bird** for the first sense of this word, that is the sense "*warm-blooded egg-laying vertebrates characterized by feathers and forelimbs modified as wings*":

Sense 1
bird --
=> vertebrate, craniate
=> chordate
=> **animal**, animate being, beast, brute, creature, fauna
=> organism, being
=> living thing, animate thing
=> object, physical object
=> physical entity
=> entity

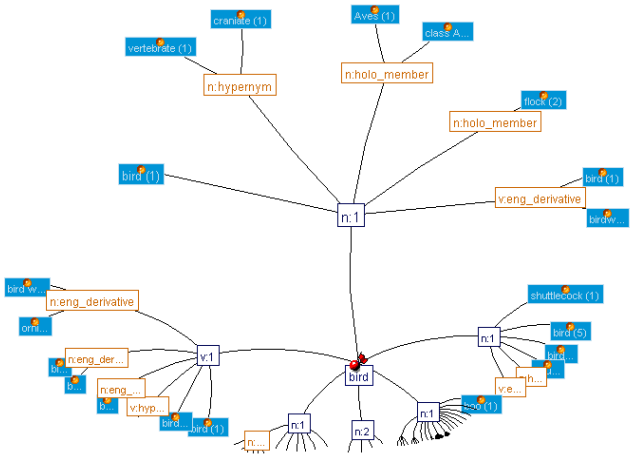

Figure 1 First level of hypernyms for **bird** (© www.racai.ro)

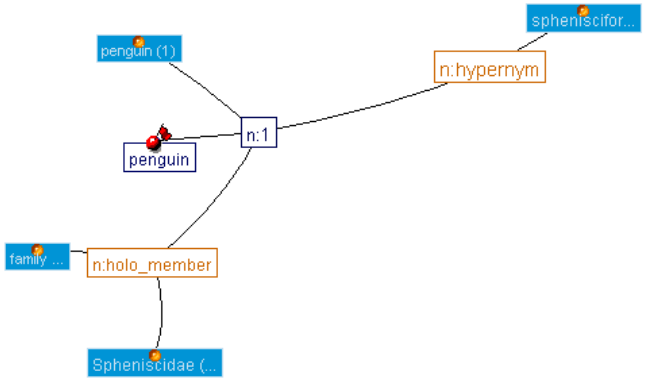

Figure 2 First level of hypernyms for **penguin** (© www.racai.ro)

The hypernyms of **penguin** for the first sense of it: "*short-legged flightless birds of cold southern especially Antarctic regions having webbed feet and wings modified as flippers*"

Sense 1
penguin
=> sphenisciform seabird
=> seabird, sea bird, seafowl
=> aquatic bird
=> **bird**
=> vertebrate, craniate
=> chordate
=> animal, animate being, beast, brute, creature, fauna
=> organism, being
=> living thing, animate thing
=> object, physical object
=> physical entity
=> entity

A3. Update the Conditional Schema with meronym and holonym relationships

We list the meronyms of **bird** for the first sense of this word, that is the sense "*warm-blooded egg-laying vertebrates characterized by feathers and forelimbs modified as wings*":

Sense 1
bird
HAS PART: beak, bill, neb, nib, pecker
HAS PART: furcula
HAS PART: feather, plume, plumage
HAS PART: **wing**
HAS PART: pennon, pinion
HAS PART: bird's **foot**
HAS PART: uropygium
HAS PART: air sac
HAS PART: uropygial gland, preen gland
HAS PART: syrinx
HAS PART: bird, fowl

One can see that no **leg** feature for **bird**s is declared. But it can be found using the holonym relationships of **foot** as it is illustrated below:

Sense 2
foot, human foot, pes -- (the part of the leg of a human being below the ankle joint; "his bare feet projected from his trousers"; "armored from head to foot")
PART OF: **leg**
PART OF: homo, man, human being, human

The meronymy relation is surprisingly ambiguous as the authors of WordNet discovered when trying to manually label sentences. Unlike other relations, meronymy relations often apply only to a specific instantiation of an entity rather than the general case. In the present article, we propose an algorithm that uses only meronyms in order to identify the objects' attributes.

The algorithm for updating conditional schema knowledge with WordNet relationships is described in the subsequel:

```
1. Collect all abstract objects pairs (X,Y)
from a conditional schema and identify
hypernym/hyponym/holonym relations using
WordNet:
   1.1 if X is a hypernym of Y then:
      h(is_a)← h(is_a)∪{(((Y,a),(X,a)),T)}
   1.2 if X is a hyponym of Y then:
      h(is_a)← h(is_a)∪{(((X,a),(Y,a)),T)}
   1.3 if X is a holonym of Y then:
      h(is_part_of)←h(is_part_of)∪
                    {(((Y,a),(X,a)),T)}
2. for each abstract object X∈Ob_abstr
   for each meronym Y of X:
   if ∃ q: an attr/value type condition and
       q.attribute=Y
   then f(X)←f(X)∪{(Y,yes)}
```

A4. The XML Schema Representation of the Conditional Schema

We provide the XML file also with the natural language text of the Conditional Knowledge Piece. An example of such a XML-conditional knowledge base is shown below:

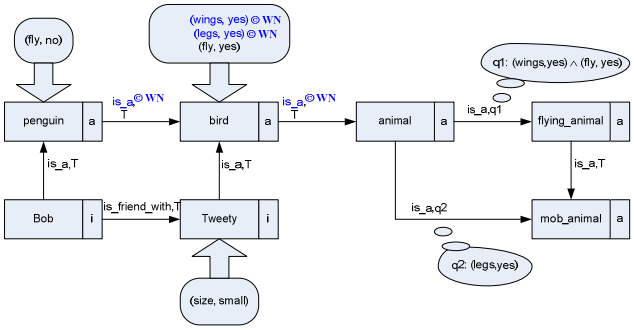

Figure 3 **The conditional graph**

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<knowledge_base>
  <natural_language_text>
  Generally, birds fly but penguins do not
  fly. Every animal is a flying animal if it
  has wings and flies. Every flying animal
  is a mobile animal. Also every animal
  which has legs is considered mobile
  animal. Bob is a penguin and Tweety is a
  small bird. Bob is friend with Tweety.
  </natural_language_text>
<node>
  <type> individual </type>
  <object> Bob </object>
</node>
<node>
  <type> individual </type>
  <object> Tweety </object>
  <parameters>
    <parameter>
      <type> attr/value </type>
      <name> size </name>
      <value> small </value>
    </parameter>
  </parameters>
</node>
<node>
  <type> abstract </type>
  <object> penguin </object>
  <parameters>
      <parameter>
        <type> attr/value </type>
        <name> fly </name>
        <value> no </value>
      </parameter>
  </parameters>
</node>
<node>
  <type> abstract </type>
  <object> bird </object>
  <parameters>
    <parameter>
      <type> attr/value </type>
      <name> fly </name>
      <value> yes </value>
    </parameter>
    <parameter>
      <type> attr/value </type>
      <name> wings </name>
      <value> yes </value>
    </parameter>
    <parameter>
      <type> attr/value </type>
      <name> legs </name>
      <value> yes </value>
    </parameter>
  </parameters>
</node>
<node>
  <type> abstract </type>
  <value> animal </value>
</node>
<node>
  <type> abstract </type>
  <value> flying_animal </value>
</node>
<node>
  <type> abstract </type>
  <value> mobile_animal </value>
</node>
<relation>
  <from> Bob </from>
  <to> Tweety </to>
  <label> is_friend_with </label>
</relation>
<relation>
  <from> Bob </from>
  <to> penguin </to>
  <label> is_a </label>
</relation>
<relation>
  <from> Tweety </from>
  <to> bird </to>
  <label> is_a </label>
</relation>
<relation>
  <from> penguin </from>
  <to> bird </to>
  <label> is_a </label>
</relation>
<relation>
  <from> bird </from>
  <to> animal </to>
  <label> is_a </label>
</relation>
<relation>
  <from> animal </from>
  <to> flying_animal </to>
  <label> is_a </label>
  <cond_symb> q1 </cond_symb>
</relation>
<relation>
  <from> animal </from>
  <to> mob_animal </to>
  <label> is_a </label>
  <cond_symb> q2 </cond_symb>
</relation>
<relation>
  <from> flying_animal </from>
  <to> mob_animal </to>
  <label> is_a </label>
</relation>
<cond_symb>
  <id> q1 </id>
```

```
    <condition>
      <type> attr/value </type>
      <attribute> wings </attribute>
      <value> yes </value>
    </condition>
    <condition>
      <type> attr/value </type>
      <attribute> fly </attribute>
      <value> yes </value>
    </condition>
  </cond_symb>
  <cond_symb>
    <id> q2 </id>
    <condition>
      <type> attr/value </type>
      <attribute> legs </attribute>
      <value> yes </value>
    </condition>
  </cond_symb>
</knowledge_base>
```

B. XML query language

The main role of an XML query language is to allow the formulation of queries and determine the result set of the XML elements that should be returned ([12]). We assume that a query input is a set of known nodes within multiple XML Conditional Knowledge Bases. Following this indea, in order to execute an XML query, the query engine should be supplied with

(1) the query string that describes the objects for which the relations existing between them have to be identified

```
<object1> name object </object1>
<object2> name object </object2>
```

For this version, we consider that `object1` is always an individual object and for this reason all the inputs address querries expressed for particular cases.

(2) optionally, the URL of the Conditional Knowledge Base on which the query must perform; if it not included in the query, than all the Conditional Knowledge Bases will be considered.

C. Processing XML Conditional Schema data

The XML file corresponding to a certain Conditional Knowledge Base is processed with respect to the user interrogation in order to determine the corresponding answer from the knowledge it represents. For this task, a path -driven reasoning is performed using an inference mechanism developed in [2] for performing reasoning based on conditional schema representations. Because conditional relations are used, determining the relationships between pairs of objects usually implies to verify the conditions identified by the conditional symbols. Thus, if we have an `attr/value` type condition, then the value(s) indicated must be fulfilled by the corresponding pairs (*attr*, *value*) of the individual object of the interrogation. The other case corresponds to the `relation` type condition in which a specific relation must exist between the individual object of the interrogation and another object (individual or abstract) indicated by the `<object2>` tag.

We have that for each input an XML file is generated, this file containing the conditional schema representation of the Conditional Knowledge Piece text together with the new relations deduced by the inference engine in order to construct the answer to the user interrogation.

The conditions attached to the conditional schema relations can be used also for providing explanations regarding the generated answers. Indeed, if we consider the following interrogations for the system:

Query_1:

```
<object1> Bob </object1>
<object2> mobile animal </object2>
```

Query_2:

```
<object1> Tweety </object1>
<object2> mobile animal </object2>
```

the system will attach to the deduced relations, the conditional symbols what were fullfiled during the inference:

```
  <relation>
    <from> Bob </from>
    <to> mobile_animal </to>
    <label> is_a </label>
    <cond_symb> q2 </cond_symb>
  </relation>
and
    <relation>
    <from> Bob </from>
    <to> mobile_animal </to>
    <label> is_friend_with </label>
    <cond_symb> q2 </cond_symb>
</relation>
```

for the Query_1 and, respectively,

```
    <relation>
    <from> Tweety </from>
    <to> mobile_animal </to>
    <label> is_a </label>
    <cond_symb> q1 </cond_symb>
</relation>
```

for Query_2.

We will exemplify the computations that result in the system in order to generate answers for the received queries only for the first considered one that is:

Query_1:

```
<object1> Bob </object1>
<object2> mobile_animal </object2>
```

In order to compute $Ans(Bob, mobile\_animal)$ we have to find all paths from *Bob* to *mobile_animal*, that is, to calculate:

$$Path(Bob, mobile\_animal)$$

There are four paths

$$d_1, d_2, d_3, d_4 \in Path(Bob, mobile\_animal)$$

such as:

$$d_1 = ([(Bob,i),(penguin,a),(bird,a),(animal,a),(mobile\_animal,a)],[is\_a,is\_a,is\_a,is\_a])$$

$$d_2 = ([(Bob,i),(penguin,a),(bird,a),(animal,a),(flying\_animal,a),(mobile\_animal,a)],[is\_a,is\_a,is\_a,is\_a,is\_a])$$

$d_3 = ([(Bob,i),(Tweety,i),(bird,a),(animal,a),$
$(mobile\_animal,a)],[is\_friend\_with,is\_a,is\_a,is\_a])$
$d_4 = ([(Bob,i),(Tweety,i),(bird,a),(animal,a),$
$(flying\_animal,a),(mobile\_animal,a)],$
$[is\_friend\_with,is\_a,is\_a,is\_a,is\_a])$

For d_1 we have:

- $\Phi([is\_a,is\_a,is\_a,is\_a]) = is\_a$
- $T = Cond_{is\_a}((Bob,i),(penguin,a))$
- $T = Cond_{is\_a}((penguin,a),(bird,a))$
- $T = Cond_{is\_a}((bird,a),(animal,a))$
- $q_2 = Cond_{is\_a}((animal,a),(mobile\_animal,a))$
- $CS(d_1) = [T,T,T,q_2]$

In order to generate:

$$ans(d_1) = g(Bob,is\_a,mobile\_animal,on)$$

we have to verify that $q_2[d_1] = on$. For this, we have to verify the rule of q_2 for the target object $Target(d_1)$= *Bob*, that is:

$R_2(x): IF\ V_x(legs) = yes\ THEN\ q_2(x) = on\ ELSE\ q_2(x) = off$

where $x \in Near_{q_2}(Bob) = \{bird\}$, that is *x*= *bird*. Results that $q_2(Bob) = on$ and thus $q_2[d_1] = on$ which implies:

$$Ans(d_1) = "Bob\ is\ a\ mobile\ animal"$$

For the path $d_2 \in Path(Bob,mobile\_animal)$ we have:

$$CS(d_2) = [T,T,T,q_1,T]$$

The conditional symbols are:

- $T = Cond_{is\_a}((Bob,i),(penguin,a))$
- $T = Cond_{is\_a}((penguin,a),(bird,a))$
- $T = Cond_{is\_a}((bird,a),(animal,a))$
- $q_1 = Cond_{is\_a}((animal,a),(flying\_animal,a))$
- $T = Cond_{is\_a}((flying\_animal,a),(mobile\_animal,a))$

In this case, we have to verify the rule of q_1 for the target object $Target(d_2)$= *Bob*, that is:

$R_1(x): IF\ V_x(fly) = yes \wedge V_x(wings) = yes\ THEN$
$q1(x) = on\ ELSE\ q1(x) = off$

for $x \in Near_{q_1}(Bob) = \{penguin\}$, that is, *x*= *penguin*. But penguins do not fly, and thus $q_1(Bob) = off$, that is, $q_1[d_2] = off$ which implies:

$$Ans(d_2) = "Bob\ is\ not\ a\ mobile\ animal"$$

For d_3 we have:

- $\Phi([is\_friend\_with,is\_a,is\_a,is\_a]) =$ is_friend_with
- $T = Cond_{is\_friend\_with}((Bob,i),(Tweety,i))$
- $T = Cond_{is\_a}((Tweety,i),(bird,a))$
- $T = Cond_{is\_a}((bird,a),(animal,a))$
- $q_2 = Cond_{is\_a}((animal,a),(mobile\_animal,a))$
- $CS(d_3) = [T,T,T,q_2]$

We have to verify the rule of q_2 for the target object $Target(d_3)$= *Tweety*, aplying the rule $R_1(x)$ for $x \in Near_{q_2}(Tweety) = \{bird\}$, that is *x*= *bird*. Results that $q_2(Tweety) = on$ and thus $q_2[d_3] = on$ which implies:

$$Ans(d_1) = "Bob\ is\ friend\ with\ a\ mobile\ animal"$$

The same answer result also for the path d_4.

The natural language explanations for the conditional symbols `q1` and `q2` corresponding to the considered interrogations, that is, to the individual objects, can be formulated as follows:

- for Query_1: *Because Bob has legs*.
- for Query_2*: Because Tweety flies and has wings*.

The algorithm for computing the answer is a Breadth-First routine, which searches for a path between the two objects specified in the interrogation. It uses a FIFO queue (noted here with `q`) in order to store the temporal paths in the considered conditional schema.

```
Procedure Query(Schema, obj1, obj2)
1. q.push([obj1])
2. while (!q.IsEmpty())
3.   TmpPath ← q.pop()
4.   obj ← TmpPath[len(TmpPath)-1]
5.   If (obj=obj2) then
6.     composeRelation(TmpPath)
7.     return;
8.   EndIf
9.   For each(link_obj in Schema\TmpPath)
10.    If(rel←relation(link_obj,obj)and
         (rel.cond_symb = null or
          evalCond(obj1,rel) = true)) then
11.        NewPath ← TmpPath + [link_obj]
12.        q.push(NewPath);
13.    EndIf
14.  EndFor
15.  EndWhile
EndProcedure
```

The routine would be called like this:

```
NameRel ← Query(Schema, obj1, obj2);
```

where `Schema` is the set of nodes (individual and abstract objects) corresponding to a particular conditional schema.

IV. Conclusion

This paper investigates the challenges of automatically extracting holonym-meronymy relationships from WordNet lexical database.

Also, an XML Schema based-formalism for conditional knowledge representation and processing is described. The present article follows our work dedicated to this type of knowledge by mapping XML representations to *conditional schema* components and implementing the inference process developed for this schema with respect to the XML specifications. We consider that our proposal covers all kinds

of the conditional schema matching problems.

The overall proposals treated in the present paper are emblematic of many of the challenges faced in NLP research today, and the proposed solutions will enable an order of magnitude improvement in conditional knowledge reasoning and discovery technologies.

ACKNOWLEDGMENT

The author Mihaela Colhon has been founded for this research by the strategic grant POSDRU/89/1.5/S/61968, Project ID 61986 (2009), co-financed by the European Social Fund within the Sectorial Operational Program Human Resources Development 2007-2013.

The author Cristina Zamfir has produced this article under the project "Supporting young Ph.D students with frequency by providing doctoral fellowships", co-financed from the EUROPEAN SOCIAL FUND through the Sectorial Operational Program Development of Human Resources.

REFERENCES

[1] S. Bringsjord, M. Clark, and J. Taylor, "Sophisticated Knowledge Representation and Reasoning Requires Philosophy", 2009

[2] M. Colhon, N. Ţăndăreanu, The Inference Mechanism in Conditional Schemas, *Annals of the University of Craiova, Mathematics and Computer Science Series,* 37(1), pp. 55-70, 2010, ISSN: 1223-6934

[3] M. Colhon, XML Semantic Schema Annotation for Dependency Relationships, *Annals of the University of Craiova, Mathematics and Computer Science Series*, 37(3), pp. 64-70, 2010, ISSN: 1223-6934

[4] M. Colhon, N. Tăndăreanu: An Approach for Contextual Translation based on Semantic Schemas, Latest Trends on Computers, vol. I - Proceedings of *The 14th WSEAS International Conference on Computers (part of the 14th WSEAS CSCC Multiconference)*, ISSN: 1792-4251, ISBN: 978-960-474-201-1, pp. 294-299, 2010

[5] M. Colhon, N. Ţăndăreanu: A Semantic Schema - based Approach for Natural Language Translation, *WSEAS Journal Transactions on Computers*, Issue 11, vol. 9, ISSN: 1109-2750, pp. 1307 - 1317, (2010)

[6] Christiane Fellbaum, editor. WordNet: An Electronic Lexical Database. *MIT Press*, Cambridge, MA, 1998.

[7] E. Kotsakis, K. Bohm, "XML Schema Directory: A Data Structure for XML Data Processing", in *Proc. of First International Conference on Web Information Systems Engineering (WISE'00)*, Hong Kong, 2000, pp. 62-69.

[8] J. Kwak and H.-S. Yong: Ontology matching based on hypernym, hyponym, holonym, and meronym sets in WordNet, International journal of Web & Semantic Technology (IJWesT), Vol.1, No.2, April 2010

[9] D. Lee, M. Mani, and M. Murata, "Reasoning about XML Schema Languages using Formal Language Theory," IBM Almaden Research Center, Tech. Rep. RJ# 10197, Log# 95071, Nov. 2000.

[10] F. Michel, "Representation of XML Schema Components," Master Thesis, Dept. Elect. Eng. and Inf. Tech, California Univ., Berkeley, 2007.

[11] A. Shoemaker and V. Ganapathi: Learning to Automatically Discover Meronyms, 2005

[12] N. Ţăndăreanu, M. Colhon, and Cristina Zamfir, Embedding Conditional Knowledge Bases into Question Answering Systems and Java Implementation, *INTERNATIONAL JOURNAL OF COMPUTERS* [Online], 4(4), pp. 169-176, 2010, ISSN: 1998-4308

[13] N. Ţăndăreanu, M. Colhon: Conditional graphs generated by conditional schemas , *Annals of University of Craiova, Mathematics and Computer Science Series*, 36(1), pp. 1-11, 2009, ISSN: 1223-6934

[14] N. Ţăndăreanu, M. Ghindeanu, Towards a Mathematical Modelling of Conditional Knowledge, Proceedings of *The 3rd Romanian Conference on Artificial Intelligence and Digital Communications*, Craiova, Romania, ISBN 973-8419-71-9, vol. 103, pp. 82-96, 2003

[15] S. Witzig: Accessing WordNet from Prolog, *Technical report*, The University of Georgia, May (2003)

[16] K. Yang, R. Steele, and A. Lo "An Ontology for XML Schema to Ontology Mapping Representation," in *Proc. iiWAS2007*, New York, 2007, pp. 8–16.

[17] http://wordnet.princeton.edu/

[18] http://wordnet.princeton.edu/wordnet/documentation/