Knowledge Extraction from Classification Schemas

Steffen Lamparter, Marc Ehrig, and Christoph Tempich

Institute of Applied Informatics and Formal Description Methods (AIFB),
University of Karlsruhe, Germany
http://www.aifb.uni-karlsruhe.de/WBS/
{sla,meh,cte}@aifb.uni-karlsruhe.de

Abstract. The availability of formal ontologies is crucial for the success of the Semantic Web. Manual construction of ontologies is a difficult and time-consuming task and easily causes a knowledge acquisition bottleneck. Semi-Automatic ontology generation eases that problem. This paper presents a method which allows semi-automatic knowledge extraction from underlying classification schemas such as folder structures or web directories. Explicit as well as implicit semantics contained in the classification schema have to be considered to create a formal ontology. The extraction process is composed of five main steps: Identification of concepts and instances, word sense disambiguation, taxonomy construction, identification of non-taxonomic relations, and ontology population. Finally the process is evaluated by using a prototypical implementation and a set of real world folder structures.

1 Introduction

The amount of digital information saved on hard disks all over the world is estimated from 403 to 1986 Terabyte and increased between 2000 and 2003 by 114%[1]. While search on the web now performs reasonable well, local information becomes increasingly unaccessible. In particular for virtual organizations, in which the stakeholder want to share their local information among each other, this obstructs collaboration. To make the information more accessible a systematical way to organize it is needed, which ontologies can provide. This view is supported by a case study which involved a virtual organization in the tourism domain where we deployed ontologies in a peer-to-peer knowledge sharing environment with promising results (cf. [1]). In the case study a common ontology was available to organize the information the participants wanted to share. Additionally they could extend the common ontology locally with concepts and relations. The participants used mainly the labels of their shared folders to create new ontological entities. Although the participants found it very useful to "customize" the ontology this manual engineering process is very time consuming and costly. In particular when it comes to changes in the folder structures the continuous updating of the "customized" ontology is not practical for the normal user.
To solve this *knowledge acquisition bottleneck* methods are needed that (semi-)automatically generate ontologies. In this context it is especially interesting how existing,

[1] http://www.sims.berkeley.edu/research/projects/how-much-info-2003

legacy information can be used to generate explicit semantical descriptions of a domain. In our case the available information are the local folder structures and existing thesauri/topic hierarchies which provide a vocabulary for the domain. More generally this information can be seen as classification schemas.

Following the ideas presented in [2] in the context of *Emergent Semantics* we have conceived a general process to learn ontologies from classification schemas as an extension of the ontology learning frame work described in [3]. Consequently we consider explicit as well as implicit semantics hidden in the structure of the schema and we combine methods from various different research domains such as natural language processing (NLP), web mining, machine learning, and knowledge representation to learn an ontology. In particular we introduce new methods to deduce concepts, relations and instances from the labels found in folder structures, relations from the arrangement of the schemas, and instantiated relations.
In the remainder of this paper the actual extraction process is presented. The process contains five steps: Identification of concepts and instances, word sense disambiguation, extracting taxonomic and non-taxonomic relations, and finally populating the ontology. Subsequently, we evaluate our process using a prototypical implementation and four real world folder structures. At the end we conclude with a short discussion and outlook.

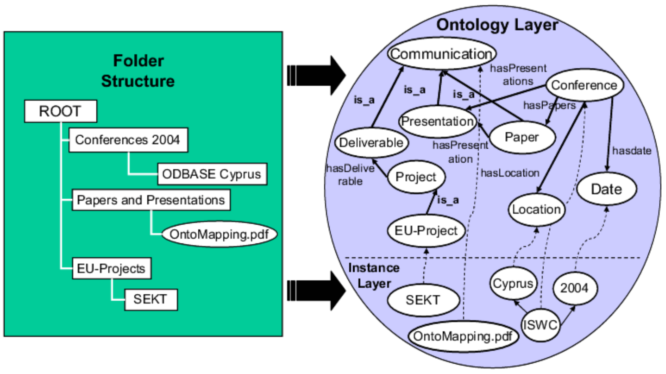


Fig. 1. Example

2 General Knowledge Extraction Process

In this section a general process is introduced that facilitates the creation of a formal and explicit description of semi-structured knowledge obtained from classification schemas (see Figure 2). The result of this method is entirely structured knowledge represented by an ontology.

Subsequently, we describe the input data our extraction process requires, the process steps we carry out, and the results we finally obtain. A more detailed description of this extraction process is presented in [4].

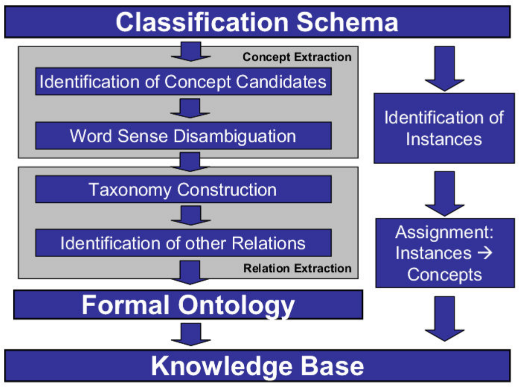

Classification Schema
Concept Extraction
Identification of Concept Candidates
Word Sense Disambiguation
Taxonomy Construction
Identification of other Relations
Relation Extraction
Identification of Instances
Assignment: Instances → Concepts
Formal Ontology
Knowledge Base

Fig. 2. Overview of the extraction process.

2.1 Definition of Input Data Structures

The extraction process presented in this paper is capable to use information from several knowledge sources. First, a classification schema is needed that provides basic information about the domain of interest. Inspired by [5] the term classification schema that is used throughout this paper is defined as follows.

Definition 1 (Classification Schema). *A knowledge structure consisting of a set of labeled nodes arranged in a tree-structure is called a* hierarchy *and can be formally defined as a tuple $\mathcal{H} = (\mathcal{K}, \mathcal{E}, l)$ with $\mathcal{K}$ representing the nodes, $\mathcal{E}$ the set of relations defining the hierarchy, and l the function which assigns a label $l \in \mathcal{L}$ to each node. $(\mathcal{K}, \mathcal{E})$ defines a tree-structure with a unique root.*

Having defined a hierarchy, a classification schema *or* hierarchical classification *can be regarded as a function $\mu : \mathcal{K} \rightarrow 2^{\Delta}$ where Δ represents a set of objects that have to be classified according to the hierarchy $\mathcal{H}$. The set $\mathcal{B} = \{l(k) \mid \forall k \in \mathcal{K}\}$ contains all node labels of the classification schema.*

Figure 1 shows on the left side an example for an classification schema. In this case the white rectangles are the nodes $\mathcal{K}$ and $\mathcal{B} = \{$*ROOT, Conferences 2004, ODBASE Cyprus, Papers and Presentations, EU-Projects, SEKT*$\}$ is the set of node labels. There is one classified object $\Delta = \{ontoMapping.pdf\}$. It is assigned to a node by the function $\mu(ontoMapping.pdf) =$ *'Papers and Presentations'*.

In this context it is important to note that the relations in the set $\mathcal{E}$ do not necessarily have to be taxonomic, i.e. subclass/superclass relations. Hence, our notion of a classification schema covers a wide range of different structures. Classification schemas include for example folder structures on personal computers as well as web directories or product categories.

To extract semantically enriched information from a classification schema further background knowledge is needed. Therefore, a machine readable dictionary (MDR) such as WordNet provides the right means. It can be used to look up and stem words, to retrieve potential meanings of a word and to find taxonomic as well as non-taxonomic associations between these meanings. Additionally, already existing ontologies can be used in order to provide domain-specific knowledge to the extraction process. Ontologies are formally defined in the next section.

2.2 Definition of Output Data Structure

The objective of the process is to represent information found in a classification schema in a formal and explicit way. This is done by defining a knowledge base which includes an ontology together with concrete instances. The formal semantics of ontologies we use throughout this paper is described subsequently (cf. [6]).

Definition 2 (Ontology Layer). *An* ontology *is a tuple* $\mathcal{O} := (\mathcal{C}, \mathcal{P}, \mathcal{H}^{\mathcal{C}}, prop)$ *where the disjoint sets* $\mathcal{C}$ *and* $\mathcal{P}$ *contain concept and relation identifiers.* $\mathcal{H}^{\mathcal{C}}$ *defines taxonomic relations between concepts. I.e.* $\mathcal{H}^{\mathcal{C}} \subseteq \mathcal{C} \times \mathcal{C}$*. The function* $prop : \mathcal{P} \rightarrow \mathcal{C} \times \mathcal{C}$ *defines non-taxonomic relations between concepts.*

A *knowledge base* contains concepts as well as concrete instances of theses concepts. Therefore, an additional instance layer is needed.

Definition 3 (Instance Layer). *The instance layer of an ontology is defined by the tuple* $KB := (\mathcal{O}, \mathcal{I}, \mathcal{L}, inst)$*.* $\mathcal{O}$ *is the ontology the instance layer refers to.* $\mathcal{I}$ *is the set of instance identifiers and set* $\mathcal{L}$ *contains literals. The mapping between the ontology and instance level is done using the functions* $inst : \mathcal{C} \rightarrow 2^{\mathcal{I}}$*.*

On the right side of Figure 1 there is an example for a knowledge base. Here the set of concepts is defined by $\mathcal{C} = \{$*Communication, Conference, Paper,* …$\}$ and taxonomic relations are represented by *isA*-Relations. That means, $\mathcal{H}^{\mathcal{C}} = \{$*(Communication, Presentation), (Paper, Communication),*…$\}$. $\mathcal{P} = \{$*(Conference, Paper), (Paper, Presentation),* …$\}$ specifies non-taxonomic relations. The set of instances $\mathcal{I} = \{$*SEKT, Cyprus, ODBASE, 2004, OntoMapping.pdf*$\}$ is mapped to corresponding concepts using the function $inst$. E.g. $inst(Cyprus) =$ *'Location'*.

2.3 Process Steps

The extraction process includes five major steps. First relevant concepts have to be identified. Therefore, node labels of the classification schema have to be analyzed with respect to a dictionary in order to find potential concept identifiers. This is done in the concept identification step. Then, these concept candidates have to be disambiguated to get the appropriate meanings in the given context. A concept identifier together with a concrete meaning defines a concept for the ontology.

Thereafter, explicit associations between the concepts are defined. First, a taxonomy is constructed. This has to be done from scratch, because hierarchies in classification schemas do not necessarily define a taxonomy in terms of *subClassOf*- or *isA*-relations, respectively. Furthermore, non-taxonomic relations between concepts have to be established.

Having an ontology, instances have to be assigned to get a complete knowledge base. Therefore, instances are identified in the classification schema by means of the dictionary. A further step is needed for the assignment of the instances to the corresponding concepts. In the next section methods that provide the functionalities mentioned above are described in detail.

3 Extraction Methods in Detail

Subsequently, methods for concept and instance identification, word sense disambiguation, taxonomy construction, identification of non-taxonomic relations, and assignment of instances are presented. Mostly these methods are supported by additional background knowledge in terms of dictionaries or domain-specific ontologies.

3.1 Identification of Concepts and Instances

In this step relevant concepts and instances are extracted from the classification schema. A basic problem is to draw the line between concepts and instances. Even for a human ontology engineer this can be a challenging issue.

All labels $\mathcal{B}$ of the classification schema are either classified into the set of concept candidates $\mathcal{B}_C$ or into the set of instances $\mathcal{B}_I$. Therefore, we assume $B_C \cup B_I = B$ and $B_C \cap B_I = \emptyset$. This means all terms which are not concepts are instances and vice versa. In this work we use the assumption that general terms included in a dictionary are concepts and specific terms not contained in a dictionary are instances.

In the following we outline methods that identify potential concepts by analyzing all labels in $\mathcal{B}$. The first method distinguishes the labels in concept candidates B_C^{lex} and instances B_I^{lex}. Thereafter, four methods are applied to revise this segmentation: (1) The sets are scanned for noun phrases, (2) the individual labels are decomposed, (3) entities are recognized by their names, (4) and concepts and instances are identified by domain-specific ontologies.

Due to the special properties of node labels in a classification schema compared to sentences in a normal NLP task, the following methods differ in some points from usual methods applied in NLP.

Lexical analysis of labels. In this step a solely syntactic analysis of the labels $b_j \in \mathcal{B}$ is performed. Therefore, special characters have to be replaced and the individual words have to be stemmed. A word is a set of letters separated form the rest of the label by space characters. In case all atomic words w_i of a label $b_j = w_{j1}, w_{j2}, \ldots, w_{ji}, \ldots, w_{jn}$ are contained in the dictionary as nouns the entire label b_j is a concept candidate. Otherwise b_j is an instance. Thus, if the set $\mathcal{W}_N$ contains only nouns from a dictionary the sets $\mathcal{B}_C^{lex}$ and $\mathcal{B}_I^{lex}$ will be defined as follows:

$$\mathcal{B}_C^{lex} = \{b_j \in \mathcal{B} \mid \forall i : w_{ij} \in \mathcal{W}_N\} \tag{1}$$

$$\mathcal{B}_I^{lex} = \{b_j \in \mathcal{B} \mid \exists i : w_{ij} \notin \mathcal{W}_N\} \tag{2}$$

In Figure 1 for instance, $b_3 =$ *'Papers and Presentations'* is assigned to $\mathcal{B}_I^{lex}$ and $b_4 =$ *'EU Projects'* to the set $\mathcal{B}_C^{lex}$.[2]

[2] Note that a consistent usage of characters and name conventions can improve the results of this step dramatically. If the labels of the nodes are very complex syntactic ambiguousness could arise. This is the case if particular nouns can also be used as adjectives for instance. The problem could be tackled by part-of-speech tagging [7, 8]. For syntactic ambiguousness see also 3.2.

Recognizing noun phrases. Although concepts are mainly represented by one single noun it is also possible that a concept is represented by a whole expression, e.g. compounds (*'credit card'*), prepositional phrases (*'board of directors'*), and adjective-noun relation (*'Semantic Web'*). Such a group of words in a label behaves like a noun and is called *noun phrase*. Due to the fact that noun phrases can be included in both sets, $\mathcal{B}_C^{lex}$ and $\mathcal{B}_I^{lex}$, both sets have to be analyzed for noun phrases. A simple method for doing this is to look up a specific expression in the dictionary. But not all noun phrases should be regarded as concepts (e.g. *last week*). According to the assumption above a noun phrase is a concept candidate if it is contained in the dictionary.

Now, we consider an expression $b_j \in \mathcal{B}_C^{lex}$ containing a noun phrase a_{ji}. a_{ji} has to be marked as a noun phrase to support finding the correct sense in section 3.2. E.g. this would be the case for $b_j = a_{ji} =$ *'Computer Science'*. Here the term has to be marked and no further action is required, because the term is already classified as concept candidate.

Additionally, a_{ji} has to be included in the set $\mathcal{B}_C^{lex}$ as a separate concept candidate, if b_j contains other words beyond the noun phrase ($b_j \neq a_{ji}$). Consider a label $b_j =$ *'Lecture Computer Science'*. In this case the recognized noun phrase is still $a_{ji} =$ *'Computer Science'*. So a_{ji} has to be added as separate concept candidate. This scenario can be described by equation 3 (first line), whereas the set $\mathcal{W}_N$ contains all nouns (and noun phrases) of the dictionary.

In case a expression $b_j \in \mathcal{B}_I^{lex}$ is analyzed and a noun phrase a_{ji} is detected the expression has to be accepted as a concept candidate (see Equation 3, second line). If the label b_j doesn't contain other words beyond the noun phrase a_{ji} the whole label b_j can be removed from the set $\mathcal{B}_I$ (see Equation 4). For example, the phrase $b_j = a_{ji} =$ *'Artificial Intelligence'* can be removed from $\mathcal{B}_I^{lex}$, but $b_j =$ *'Applied Computer Science'* with $a_{ji} =$ *'Computer Science'* cannot be removed.

$$\begin{aligned}\mathcal{B}_C^{np} := \mathcal{B}_C^{lex} &\cup \{a_{ji} \mid \exists i,j : b_j \in \mathcal{B}_C^{lex} \wedge a_{ji} \in \mathcal{W}_N \wedge b_j \neq a_{ji}\} \\ &\cup \{a_{ji} \mid \exists i,j : b_j \in \mathcal{B}_I^{lex} \wedge a_{ji} \in \mathcal{W}_N\}\end{aligned} \tag{3}$$

$$\mathcal{B}_I^{np} := \mathcal{B}_I^{lex} \setminus \{b_j \mid \exists j : b_j \in \mathcal{B}_I^{lex} \wedge a_{ji} \in \mathcal{W}_N \wedge b_j = a_{ji}\} \tag{4}$$

In the unusual case that node labels of the classification schema are very complex and similar to sentences in natural language, it is very hard to recognize proper concepts. The use of a chunk parser can be reasonable to solve this problem [9].

Lexical decomposition of labels. In the last two steps the labels $b_j \in \mathcal{B}$ are analyzed as a whole. Now, based on the lexical analysis done before the label is decomposed into the individual words $w_{j1}, \ldots, w_{ji}, \ldots w_{jn}$. To find out whether a subset of the entire label represents a concept candidate all words w_{ji} are looked up in a dictionary separately. If only one word w_{ji} is found as a noun in the dictionary this word can be accepted as a concept candidate (see Equation 5). For instance the concept $w_{11} =$ *'Conference'* can be extracted from the label $b_1 =$ *'Conferences 2004'*.

If more than one word of a label is found in the dictionary a method will be needed to decide whether these words should form one single multi-word concept c_{j1} or several different concepts c_{jr} with $r = 1, 2, \ldots, m$. Therefore, the non-substantival words between concept candidates can be used as indicator [5]. If two recognized concept candidates are connected by a space character or a preposition, they will be related by a logical 'and'-Relation. In this case objects $\delta \in \Delta$ classified under the label are belonging to both concept candidates. Thus, only one single concept candidate $c_{j1} \in \mathcal{B}_C^{decomp}$ should be composed. E.g. this is the case for $b_j = c_{j1} =$ *'EU Projects'*. On the other hand, if two recognized concepts are connected by the word 'and' or a comma a logical 'or'-Relation is assumed. In this case classified objects belong to either the first or second part of the label and two different concept candidates $c_{j1}, c_{j2} \in \mathcal{B}_C^{decomp}$ are composed, consequently. The label $b_4 =$ *'Papers and Presentations'* produces two separate concepts $c_{41} = w_{41} =$ *'Paper'* and $c_{42} = w_{43} =$ *'Presentation'*. In such a scenario maximal number of $n-1$ concepts are extracted from one label $(m < n-1)$.

$$\mathcal{B}_C^{decomp} := \mathcal{B}_C^{np} \cup \{w_{ji} \mid \forall k \neq i : w_{ji} \in \mathcal{W}_N \wedge w_{jk} \notin \mathcal{W}_N\} \\ \cup \{c_{jr} \mid \forall j : b_j \in \mathcal{B}_C^{np} \wedge \forall r : r \leq m\} \tag{5}$$

Named entity recognition. The task *named entity recognition* is about identifying certain entities as well as temporal and numeric expression by their name or format. That means, instances of generic concepts such as *Person*, *Location*, *Date* or *Time* are identified. Because dictionaries usually include very generic concepts as well as concrete named entities, both sets, B_C^{decomp} and B_I^{decomp}, have to be included in the named entity recognition process.

Named entity recognition can be regarded as a function $\gamma : \mathcal{N} \rightarrow \mathcal{C}_{NER}$ that assigns a concept $c \in \mathcal{C}_{NER}$ to each named entity $e \in \mathcal{N}$. In case a named entity $e_{ji} \in \mathcal{N}$ is found in the label $b_j \in \mathcal{B}_I^{decomp}$ the corresponding concept $c_{ji} = \gamma(e_{ji})$ has to be included in $\mathcal{B}_C^{name}$ (first line of Equation 6). E.g in the label $b_1 =$ *'Conferences 2004'* the word $e_{11} =$ *'2004'* is recognized as date. In this case the concept $c_{11} =$ *'Date'* can be added to the set $\mathcal{B}_C^{name}$.

If a named entity e_{ji} is identified in a label $b_j \in \mathcal{B}_C^{decomp}$ this named entity has to be deleted from concept candidates (second line of Equation 6) and moved to the set of instances $\mathcal{B}_I$ (Equation 7). Additionally, the concept $\gamma(e_{ji})$ has to be accepted as a concept candidate (first line of Equation 6).

$$\mathcal{B}_C^{name} := \mathcal{B}_C^{decomp} \cup \{\gamma(e_{ji}) \mid \exists j, i : e_{ji} = w_{ji} \wedge b_j \in \mathcal{B}_I^{decomp}\} \\ \setminus \{e_{ji} \mid \exists j, i : e_{ji} = w_{ji} \wedge b_j \in \mathcal{B}_C^{decomp}\} \tag{6}$$

$$\mathcal{B}_I^{name} := \mathcal{B}_I^{decomp} \cup \{e_{ji} \mid \exists j, i : e_{ji} = w_{ji} \wedge b_j \in \mathcal{B}_C^{decomp}\} \tag{7}$$

For instance a label $b_j =$ 'Cyprus' would be in the set $\mathcal{B}_C^{decomp}$ although it should be classified as instance. Therefore, *Cyprus* has to be deleted from the set of concept candidates $\mathcal{B}_C^{name}$ and added to the set of instances $\mathcal{B}_I^{name}$. Furthermore, the recognized concept *Location* has to be added to the set $\mathcal{B}_C^{name}$.

For sake of completeness we list some of the most prominent approaches for named entity recognition:

- Pattern-based approach: Context sensitive reduction-rules are defined statically and applied to the labels [10].

– Gazetteers-based approach: Already known entity names are saved in a lists (gazetteers) together with the concept they belong to. With this lists mapping between instances and concepts can be done easily.
– Automatic approaches: Theses are mostly statistical methods like the Hidden-Markow-Model [11] or the Maximum-Entropy-Model [12].

Often all three approaches are combined to achieve a better performance [13].

Mapping to a domain-specific ontology. In this step concept candidates which are not in the dictionary are identified by comparing words retrieved from the classification schema with concepts and instances of domains-specific ontologies. This method is based on the assumption that in a specific domain the same words always have the same meaning. Thus, it is possible to identify concepts simply by comparing the words w_{ij} of labels $b_j \in \mathcal{B}_I^{name}$ with the concepts $c_k \in C_{domain}$ as well as with the instances $inst_{domain}(c_k)$ of a domain specific ontology. A word w_{ij} of a label classified as an instance $b_j \in \mathcal{B}_I^{name}$ that syntactically equals the label of a concept $c_k \in \mathcal{C}_{domian}$ is supposed to be a concept candidate (see Equation 9). E.g. there is a label b_j = *'Associate Professor'* in the set $\mathcal{B}_I^{name}$ as well as a concept c_k = *'Associate Professor'* in the domain-specific ontology $\mathcal{C}_{domain}$. In this case the concept label b_j could be added to the set $\mathcal{B}_C^{name}$.

If the label b_j only consists of the recognized concept candidate w_{ji} the label b_j can be deleted from the set of instances. In case of the label b_j = *'Associate Professor'*, b_j could be deleted from $\mathcal{B}_I^{name}$, because the label contains no other words.

$$\mathcal{B}_I^{onto} := \mathcal{B}_I^{name} \setminus \{b_j \mid b_j \in \mathcal{B}_I^{name} \wedge b_j = w_{ji}\} \tag{8}$$

If there is no match between w_{ji} and the concepts of the domain-specific ontology, w_{ji} is compared to the instances of this ontology $\mathcal{I}_{domain}$. If $w_{ji} \in \mathcal{I}_{domain}$ holds, w_{ji} will still be an instance, but the corresponding concept $c_k = inst^{-1}_{domain}(w_{ji})$ will be accepted as a concept candidate. Assuming the concept c_k = *'Topic'* has an instance which matches the label b_j = *'Information Retrieval'* with $b_j \in \mathcal{B}_I^{name}$. In this case c_k = *'Topic'* can be added to $\mathcal{B}_C^{onto}$.

$$\begin{aligned}\mathcal{B}_C^{onto} := \mathcal{B}_C^{name} &\cup \{w_{ji} \mid \exists j,i,k : w_{ji} = c_k \wedge c_k \in \mathcal{C}_{domain} \wedge b_j \in \mathcal{B}_I^{name}\} \\ &\cup \{c_k \mid \exists i,j,k : w_{ji} = inst_{domain}(c_k)\}\end{aligned} \tag{9}$$

For this method only domain-specific ontologies can be used which have at least the quality level that is claimed for the new ontology.

3.2 Word Sense Disambiguation

Lexical polysemy has been a hot topic since the earliest days of computer-based language processing in the 1950s [14]. Lexical polysemy either arises due to the fact that words can be used in different part-of-speeches (syntactical polysemy) or due to words that have varying meanings in different contexts (semantical polysemy). *Word sense disambiguation* is about solving semantical polysemy.

Having identified the concept candidates $\mathcal{B}_C$ word sense disambiguation algorithms are applied to assign appropriate meanings from the MRD to these concept candidates. Then, concept candidates and their distinct meaning are used as concepts $\mathcal{C}$ for the ontology. Having non-ambiguous concepts is necessary to define a correct taxonomy and to find valid non-taxonomic relations.

In [14] different approaches to word sense disambiguation are described. On the one hand there are global, context-independent approaches, which assign meanings retrieved form an external dictionary by applying special heuristics. E.g. a frequency-based approach where always the most frequently applied sense can be used. On the other hand there are context-sensitive approaches. This kind of methods uses the context of a word to disambiguate it. In our scenario the context is composed by other words in the label and by labels of the subordinated as well as superordinated nodes in the classification schema. For the disambiguation process knowledge-based, corpus-based and hybrid methods can be applied.

3.3 Taxonomy Construction

Having identified the concepts $\mathcal{C}$ of the ontology a taxonomic structure $\mathcal{H}_\mathcal{C} \subseteq \mathcal{C} \times \mathcal{C}$ is needed. Therefore, the concepts have to be associated by irreflexive, acyclic, and transitive relations. The hierarchy already contained in the classification schema cannot be used for this purpose, because the relations in this hierarchy do not have to be taxonomic. There are already various algorithms available tackling the problem of building taxonomies. According to Maedche et. al. [15] they can be categorized in symbolic and statistical algorithms. Symbolic algorithms use pattern recognition to identify taxonomic relation between concepts. Due to the fact that in this scenario only node labels and not sentences are processed, lexico-syntactic patterns from a NLP-scenario can be reused only to a small extend. Alternatively, statistical methods can be applied. Here various kinds of clustering algorithms are available[15].

In the following two algorithms similar to [16] are outlined: One approach based on a semantic net such as WordNet and one symbolic, pattern-based algorithm.

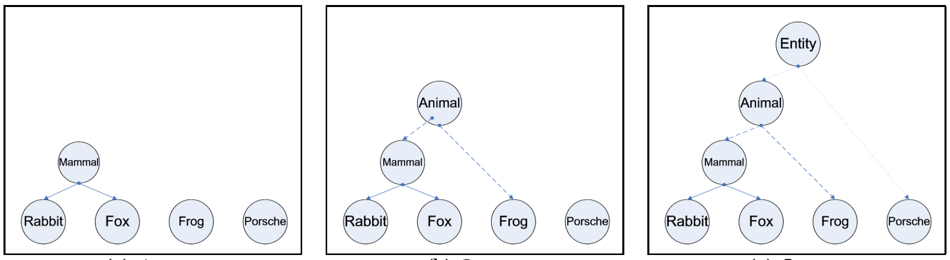
Extracting taxonomic relations by pruning. Starting point for this process step are the individual concepts $\mathcal{C}$. In order to find taxonomic relations between these concepts we use the fact that all concepts are represented by a meaning in the machine readable dictionary. If the used machine readable dictionary defines also hyponym/hyperonym-relations between meanings (like WordNet does) it will easily be possible to find taxonomic relations between the concepts. This is done by comparing all concepts $c_i \in \mathcal{C}$ with the other concepts $c_j \in \mathcal{C}$ $(i \neq j)$ to find out which concepts are directly taxonomic related, which have a common super-concept, and which are not taxonomic related at all. In case two concepts c_i and c_j are not directly connected, but they have common super-concepts, the most specific super-concept c_{ij} as well as two taxonomic relations have to be included in the ontology. In Figure 1, for instance, the concepts $c_1 =$ *'Presentation'* and $c_2 =$ *'Paper'* are not directly connected by a taxonomic relation, but they have the common super-concept $c_{12} =$ *'Communication'*. In the following equations the operator '$\geq$' specifies taxonomic relations between two concepts. E.g. $c_1 \geq c_2$

states that c_2 is a subclass of c_1.

$$\mathcal{H}_{\mathcal{C}}^{new} := \mathcal{H}_{\mathcal{C}}^{old} \cup \{c_i \times c_j \mid \forall i,j : c_i \geq c_j \wedge i \neq j\} \\ \cup \{c_{ij} \times c_i, c_{ij} \times c_j \mid \forall i,j : i \neq j \wedge c_{ij} \geq c_i \wedge c_{ij} \geq c_j\} \tag{10}$$

$$\mathcal{C}^{new} := \mathcal{C}^{old} \cup \{c_{ij} \mid \forall i,j : c_i \ngeq c_j \wedge c_j \ngeq c_i \wedge c_{ij} \geq c_i \wedge c_{ij} \geq c_j\} \tag{11}$$

Iteratively, this step has to be repeated on bases of $\mathcal{C}^{new}$ until no super-concepts are included any more (i.e. $\mathcal{C}^{new} = \mathcal{C}^{old}$). Figure 3 shows an example for this iterative process.



(a) 1. step (b) 2. step (c) 3. step

Fig. 3. Example for extracting taxonomic relations by pruning.

Pattern-based extraction of taxonomic relations. Additionally, a symbolic algorithm can be applied in order to generate taxonomic relations. [17] uses such a method for natural language processing and defines regular expressions that can be used to find relations between words. Designing such regular expressions for complex sentences is a cumbersome task and the results reached in the NLP-domain are not too promising. Nevertheless, for analyzing node labels a symbolic approach could be useful. Here, the label structure is much simpler than the structure of a natural language sentence. Therefore, finding regular expressions that indicate a taxonomic relation in a node label can be done more easily. For example the regular expression *[(NP)*NP]* indicates that the last noun phrase of a label is super-concept of the whole label. This is true, because in most cases the last word of a sequence determines the object type. E.g. *EU Project* is of type *Project*.

3.4 Identification of Non-Taxonomic Relations

This section is about finding non-taxonomic relations $\mathcal{P} = \mathcal{C} \times \mathcal{C}$. Therefore, two tasks have to be performed. Firstly, we have to detect which concepts are related. And secondly, we have to figure out how these concepts are related. Thus, a name for the relation has to be found. For discovering taxonomic relations the second step was not necessary, because the name of the relation was already defined (*subClassOf*, *isA*).

There is a huge number of algorithms dealing with finding non taxonomic relations [18–20]. Mostly these approaches apply co-occurrences analysis in order to find out which concepts are related. Then, the label of the relation is generated using the predicate of the sentence. This is not possible in case of a classification schema, because node labels rarely contain predicates. But classification schemas also contain additional information compared to natural language texts. In the following we outline how this information can be exploited.

Identifying general relations by pruning. Due to the fact that concepts are represented by meanings of a dictionary relations contained in the dictionary can also be reused easily. These relations are mostly very general. E.g. *WordNet* contains relations such as *partOf* or *hasSynonym*. Normally, such general relations are not very relevant for a specific domain or application. In order to avoid inflating the ontology with irrelevant relation only those relation that are useful should be reused. However, domain-specific relations can hardly be found using dictionaries or other general knowledge structures. Therefore, other methods are needed.

Reusing domain-specific ontologies. To identify more specific relations existing ontologies can be used. Here especially ontologies are suitable which model the same domain or are used for the same purpose. Assuming that such an ontology is defined by the tuple $O_{domian} = (\mathcal{C}_d, \mathcal{P}_d, \mathcal{H}^{\mathcal{C}}, prop)$ the starting point $c_a^d \in \mathcal{C}_d$ and the endpoint $c_b^d \in \mathcal{C}_d$ of the relation $r_d \in \mathcal{P}_d$ has to match two concepts $c_a, c_b \in \mathcal{C}$. Then, a relation between c_a and c_b of type r_1^d can be included in the new ontology. Again, we assume that two concepts $c \in \mathcal{C}$ and $c_d \in \mathcal{C}_d$ will match if their labels are identical. E.g. there is a domain-specific ontology with the concepts $c_1^d =$ *'Conference'*, $c_2^d =$ *'Presentation'*, and a relation of type $r_1^d =$ *'hasPresentation'*. In this scenario the relation can be added to the new ontology.

Identifying relations using the classification hierarchy. A concept hierarchy – as mentioned above – is represented by a tuple $\mathcal{H} = (\mathcal{K}, \mathcal{E}, l)$. The set of relations $\mathcal{E}$ contains information about the human domain model underlying the classification schema. Although the relations define no real taxonomy and thus cannot be used for finding taxonomic relations, they are not meaningless. They indicate weather two concepts c_a, c_b are related in some way.

To show this we consider the set $\mathcal{E}' \subseteq \mathcal{E}$ that includes only relations between nodes $k \in \mathcal{K}$ which have corresponding concepts in $\mathcal{C}$. I.e. we will assume that two concepts c_a, c_b are related if the nodes k_a, k_b are also related by an association $r_{dom} \in \mathcal{E}'$. In Figure 1, for instance, *SEKT* is not in $\mathcal{E}'$, but $k_1 =$ 'Conference' and $k_2 =$ *'Location'* are related since *ODBASE Cyprus* is a subfolder of *Conferences 2004*. In case two concepts are related by $r_{dom} \in \mathcal{E}'$ as well as by a general association $r_{general}$ found in the step before we assume r_{dom} is of type $r_{general}$ and include the relation in the ontology. For the remaining relations $\mathcal{E}'^{new} = \mathcal{E}'^{old} \setminus \{r_{dom}\}$ the type of a relation $c_a \rightarrow c_b$ is generated by concatenating 'has' and the label of concept c_b. E.g. the type of the relation between *Conference* and *Location* would be *hasLocation*.

Pattern-based extraction of non-taxonomic relations. Information about relations between concepts is not only contained in the structure of the hierarchy but also in the labels of the nodes itself. If two concepts are extracted from the same node label they are related in some way. E.g. the label *Conferences 2004* includes the two concepts *Conference* and *Date*. Again, we know that there is an association between two concepts, but we do not know which. In the last section we used regular expressions to define patterns that indicate taxonomic relations. Now, we can extend this method to facilitate the discovery of non-taxonomic relations. Therefore, a list of regular expressions is needed together with the relation type they refer to. For instance, the regular expression [NP *within* NP] might indicate an *include*-relation. In order to find relations all node labels containing more than one concept have to be searched for the patterns defined in the list. The use of predicate-based patterns [21] seems to be not very promising due to the fact that predicates are rarely used in node labels. In case there are no additional words in the label that allow the use of pattern-based approaches we can adopt a method similar to that in the paragraph before. Again, we compose the relation type by concatenating *has* with the second concept. I.e. for example above a relation of type *hasDate* is introduced to connect *Conference* and *Date*.

3.5 Ontology Population

In order to populate the ontology $\mathcal{O} = (\mathcal{C}, \mathcal{H}^{\mathcal{C}}, \mathcal{P}, inst)$ the function $inst : \mathcal{C} \rightarrow 2^I$ has to be defined.

Reusing already extracted knowledge. During the generation process of the core ontology knowledge about the mapping between instances $\mathcal{B}_I$ and concept candidates $\mathcal{B}_C$ has already evolved. Now, this knowledge can be incorporated into the ontology population process.

In the concept identification step concepts are extracted from instances. We assume that a concept $c \in \mathcal{C}$ extracted from an instance $i \in \mathcal{B}_I$ represents the concept of this instance ($inst(i) = c$). In this way all instances that produced a concept can be assigned to this concept. Other instances cannot be assigned by this method. E.g. named entity recognition discovers that *Cyprus* is an instance of *Location* and *2004* is an instance of *Date*.

A problem occurs if the mapping is not unique. If two concepts c_1, c_2 are extracted from one instance it will be not clear to which concept the instance has to be assigned. This is case for the file *OntoMapping.pdf*. The problem could be solved by assigning the instance to the most specific common super-class of c_1 and c_2. In this case some information contained in the classification schema is lost. For the file *OntoMapping.pdf* it is not possible to decide whether it is a *Presentation* or a *Paper*. Thus, it has to be assigned to *Communication*.

Populating by means of the classification schema. Now we consider all instances in the set $\mathcal{B}_I$ which have not been assigned in the last step. They are assigned by using the hierarchy of the classification schema. Therefore, we have to analyze the direct super-node of an instance. If a concept is extracted from this super-node the instance will be assigned to that concept. Otherwise the next superior node in the hierarchy has to be considered. If there is no node with a corresponding concept in the entire partial tree, the instance will be assigned to the root concept. In Figure 1 the instance *SEKT* will be assigned to the next superordinated concept. This would be *EU-Project* in this case.

Having described a process for representing knowledge obtained from a classification schema in an explicit and formal way, we will know evaluate this process using real world folder structures.

4 Evaluation

For evaluation purpose we used a prototypical implementation of the knowledge extraction method introduced in this paper. First we outline the architecture of this prototype. Then, the test data and the evaluation measures are introduced. Finally, the results automatically generated by the extraction method are evaluated.

Prototype. The prototype used for evaluation allows to extract an ontology from an underlying directory of a computer system. The extraction process comprises five steps, each including several algorithms. The prototype does not implement all algorithms introduced in the last sections, but it implements at least one for each process step. This guarantees a valid solution, but the performance of the prototype is only a base line for future enhancements.

The prototype includes the following algorithms:

Identification of concepts and instances. Lexical analysis and decomposition of node labels as well as reusing a domain-specific ontology is performed.

Word Sense Disambiguation. There are two alternative methods available. One global, context-independent algorithm, that assigns meanings based on the frequencies of their occurrence. The other method disambiguates words based on the context. The method combines the techniques of Magnini et. al. [5], Rada [22], and the frequency based approach mentioned above.

Taxonomy construction. In this step all algorithms suggested by the extraction method are implemented. Extracting relations by pruning and a pattern-based approach, where only the regular expression [(NP)*NP] is used.

Identification of non-taxonomic relations: A method for identifying non-taxonomic relations by using knowledge from the classification schema is implemented in this step.

Ontology Population: All algorithms for ontology population introduced in the extraction method are implemented.

For the concrete implementation the machine-readable dictionary *WordNet*[3] is used. The current version of this dictionary contains 152059 words and 115424 meanings that are expressed by synonym sets (*synsets*). These synsets are used internally for representing concepts of the ontology. Furthermore the ISWC-Ontology[4] is used which is highly relevant for the domain of the evaluation data set.

[3] http://www.cogsci.princeton.edu/∼wn

[4] http://annotation.semanticweb.org/iswc/iswc.daml

Evaluation data set. We use four real world folder structures to evaluate the prototypical implementation of the extraction method. The directories cover the domains university, project management, and Semantic Web technologies. All structures are working directories of employees of a research institute, which include academic as well as administrative data. We compared the automatically generated ontologies to one which we manually engineered. The 'reference' ontology contained only information which could directly be deduced from the folder structures with common sense.

The folder structures are serialized in RDF(S)-format according to the SWAP-Common ontology[5]. Table 4 contains statistical data about the used directories.

	# folders	# files	max. depth	avg. depth	$\frac{words}{label}$
Directory 1	293	493	15	5.9	1.44
Directory 2	548	1309	12	6.7	1.46
Directory 3	197	552	14	8.0	1.21
Directory 4	189	780	5	3.8	5.53

Fig. 4. Topology statistic of folder structures

Evaluation measures. To evaluate the extraction method we apply the standard measures *Recall* and *Precision* originally used in Information Retrieval. *Recall* shows how much of the existing knowledge is extracted.

$$Recall = \frac{\sharp\ correctly\ extracted\ entities}{\sharp\ entities} \tag{12}$$

To calculate the Recall values we count the number of correct extracted concepts, relations or instances, and divide it by the overall number contained the classification schema. Concepts will count as correct if they are contained in the 'reference' ontology. A relation will be correct if both concepts as well as the relation type is valid. To get a correct instance, the label and the assigned concept have to be correct.

Precision in contrast specifies to which extend the knowledge is extracted correctly. In this case we built the ratio between the correct extracted and the overall extracted concepts, relations, or instances.

$$Precision = \frac{\sharp\ correctly\ extracted\ entities}{\sharp\ exracted\ entities} \tag{13}$$

Since there are no preexisting ontologies for our evaluation data (Gold Standard), the Recall values can only be calculated for the concept and instance identification. In these cases we were able to compare the results to the performance a human ontology engineer reaches based on the information contained in the classification schema. Of course, this measure cannot be completely objective.

Evaluation results. The overall Precision value for concepts, relations, and instances lies between 70% and 81% for the different directories. As mentioned above there is no Recall value for the overall process. Because errors in early process steps could cause cascading errors in the following steps we analyzed the five different steps separately.

[5] http://swap.semanticweb.org/2003/01/swap-common#

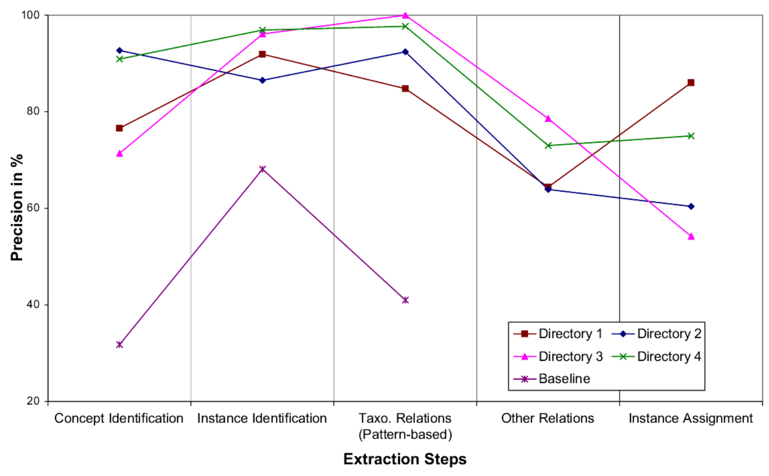


Fig. 5. Precision for each step

Concept and instance identification performs well for all directories (70%–93%). A major part of the errors are due to not recognized named entities. Another issue the prototype cannot handle are complex node labels. If labels similar to sentences are used, concept identification will fail quite often. In such cases NLP-techniques have to be introduced (POS-tagging, chunck-parsing,. . .). We introduced a baseline where we assume that all labels of the classification schema are concepts (concept identification) or instances (instance identification), respectively. Here we achieve average Precision-values of 31% for concept identification and 61% for instance identification. That means our identification algorithms performs much better. Concept and instance identification achieves Recall values well above 80% .

In order to disambiguate the extracted concept candidates we evaluated two different algorithms. One context-sensitive algorithm based on the methods by Magnini et. al. [5] and Rada [22]. The second algorithm we apply is a simple frequency-based method. Except for one directory the frequency-based algorithm performs better than the context-sensitive one. Considering context improves the disambiguation result only in case of a very specific domain (directory 4). However, the difference between both approaches seems to be quite small.

In terms of Precision the extraction of taxonomic relations performs very well. This can be explained by the fact that the first method only reuses relations from WordNet. Thus, Precision of 100% can be reached if errors of earlier steps are neglected. The pattern-based approach achieves Precision values between 84,8% and 100%. Here we generated a baseline by interpreting the hierarchy of the classification schema as valid taxonomy and encountered a Precision value of about 40%.

Finding non-taxonomic relations is probably the most difficult task, because here not only the relation itself but also the label of the relation has to be extracted. The implemented approach based on the classification hierarchy achieves between 63,9% and 78,6% Precision. Errors are caused to the same extend by wrong relations identification and wrong assignment of labels.

The performance of the ontology population method depends highly on the previously generated ontology. I.e. an instance cannot be assigned correctly if the corresponding concept is not extracted. Thus, Precision between 55% and 85% is achieved. If errors that are caused by a wrong ontology are disregarded, we can achieve much better results. Especially the first method (using knowledge of the extraction process) performs very well with Precision values between 80% and 100%.

test directory	# concepts	# instances	# taxonomic relations	other relations	$\frac{concepts}{folder}$	max. depth	avg. depth	Precision
1	175	656	68	90	0.59	6	3.4	80.7%
2	299	1457	115	180	0.55	7	3.5	78.3%
3	262	624	52	14	1.33	6	3.8	80.1%
4	265	776	54	89	1.4	5	3.7	70.4%

Fig. 6. Topology statistic of generated ontologies

Figure 6 contains statistical data about the generated ontologies. It is obvious that the structure of the ontologies depend heavily on the properties of the underlying directories. Thus, the folder structures with shorter labels and a deeper tree structure (directory 1 and 3) achieve the best Precision values. Directory 4 has by far the longest labels and the shallowest tree structure and achieves the worst Precision result. The relative flat and coarse taxonomies of the ontologies are caused by the fact that the extraction of taxonomic relations is only executed once in the prototype. To get more complete taxonomies this algorithm has to be repeated iteratively until all super-concepts are introduced.

In general good results can be achieved although not all algorithms contained in the extraction process have been implemented yet.

5 Related Work

The extraction process depends heavily on the underlying data structures. One can distinguish between different ontology learning approaches: Learning from natural language texts, semi-structured schemas, dictionaries, knowledge bases and from entirely structured information such as relational schemas. Our work focuses on the area of ontology learning from semi-structured schemas. We used explicit as well as implicit semantics hidden in classification schemas in order to generate an ontology. Deitel et. al. [23] also use information in RDF-format to construct an ontology. They extract knowledge from RDF-annotations of Web-resources by applying graph theory techniques. Doan et. al. [24] use a machine learning approach to map between a semi-structured source file and a target ontology. First, mappings have to be defined manually and then the machine learning method tries to find new mapping based on the existing ones. [5] present methods for interpreting schema models on basis of the taxonomic relations as well as the linguistic material they contain. The main difference to our work is the fact that the schema models they build upon include already valid taxonomic relations.

Apart from the work done in the field of ontology learning there has been some effort to build ontologies from taxonomies manually. In [25] the authors describe a case study were they have engineered an ontology based on the Art and Architecture Thesaurus to describe architectonic images. Similarly in [26] the NCI thesaurus was used to model an ontology for medical domain. In contrast to our work they do not consider automated methods to build the ontology.

In [27] a method is presented to generate a global virtual view from database schemas. They use also WordNet as a common vocabulary. However, they do not learn new relations as we do from labels, but integrate different existing schemas.

6 Conclusion

In this paper we presented a method for automatic knowledge extraction from classification schemas. This extracted knowledge is represented by a formal ontology. The integration with methods based on other data structures is made possible by incorporating a generic ontology learning framework.

The extraction method we outlined above combines methods and algorithms from various research domains, which are usually treated separately in literature. Additionally, we introduced several heuristics that exploit the special semantics of classification schemas.

To evaluate this method we built an prototype for knowledge extraction from directories. This prototype implements the five steps of the extraction method and the majority of algorithms they include. Applying the method to real world folder structures we realize Precision values between 70% and 80%. In this scenario the entire method was executed automatically without human intervention. But the evaluation also made clear that there is a lot room for improvements. Especially the implementation of named entity recognition promises further improvement.

Certainly the prototype evaluated here is not suitable for entirely automatic ontology generation, but the results represent a good basis for a human ontology engineer. This enables more economical and efficient ontology engineering and thus saves time and money.

Acknowledgements. Research reported in this paper has been partially financed by the Graduate School for Information Management and Market Engineering (DFG grant no. GRK 895) and the EU in the IST projects SWAP (IST-2001-34103) and SEKT (IST-2003-506826).

References

1. Pinto, S., Staab, S., Sure, Y., Tempich, C.: OntoEdit empowering SWAP: a case study in supporting DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies (DILIGENT). In Bussler, C., Davies, J., Fensel, D., Studer, R., eds.: First European Semantic Web Symposium, ESWS 2004. Volume 3053 of LNCS., Heraklion, Crete, Greece, Springer (2004) 16–30
2. Maedche, A.: Emergent semantics for ontologies - support by an explicit lexical layer and ontology learning. IEEE Intelligent Systems (2002)
3. Maedche, A., Staab, S.: Ontology learning for the semantic web. IEEE Intelligent Systems **16** (2001) 72–79
4. Lamparter, S.: (Semi-)Automatische Wissensextraktion aus Klassifikationsschemata. Master's thesis, Institute AIFB, University of Karlsruhe (TH) (2004)

5. Magnini, B., Serafini, L., Speranza, M.: Making explicit the semantics hidden in schema models. In: Proc. of the 2nd Int. Semantic Web Conference. (2003)
6. Hotho, A., Maedche, A., Staab, S., Zacharias, V.: On knowledgeable unsupervised text mining. In Franke, J., Nakhaeizadeh, G., Renz, I., eds.: Text Mining, Theoretical Aspects and Applications, Physica-Verlag (2003) 131–152
7. Brants, T.: TnT – a statistical part-of-speech tagger. In: Proc. of the 6th Applied Natural Language Processing Conference (ANLP), Seattle, WA (2000)
8. Brill, E.: Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. Computational Lingusitics **21** (1995) 543–565
9. Abney, S.P.: Parsing by chunks. In Berwick, R.C., Abney, S.P., Tenny, C., eds.: Principle-Based Parsing: Computation and Psycholinguistics. Kluwer, Dordrecht (1991) 257–278
10. Grishman, R.: The NYU system for MUC-6 or Where's the Syntax? In: Proc. of the 6th Message Understanding Conference (MUC-6, 1995), Morgan Kaufmann (1995)
11. Bikel, D.M., Miller, S., Schwartz, R., Weischedel, R.: Nymble: a high-performance learning name-finder. In: Proc. of 5th Conf. on Applied Natural Language Processing. (1997) 194–201
12. Borthwick, A.: A Maximum Entropy Approach to Named Entity Recognition. Ph.d. thesis, New York University (1999)
13. Mikheev, A., Moens, M., Grover, C.: Named entity recognition without gazetteers. In: In EACL'99, Bergen, Norway (1999) 1–8
14. Ide, N., Véronis, J.: Introduction to the special issue on word sense disambiguation: The state of the art. Computational Linguistics **24** (1998) 1–40
15. Maedche, A., Pekar, V., Staab, S. In: Ontology Learning Part One - On Discovering Taxonomic Relations from the Web. Springer Verlag (2002) 301–322
16. Missikoff, M., Navigli, R., Velardi, P.: The usable ontology: An environment for building and assessing a domain ontology. In: Proc. of 1st Int. Semantic Web Conference (ISWC 2002). Number 2342 in LNCS, Springer-Verlag, Berlin (2002) 39–53
17. Hearst, M.A.: Automatic acqusition of hyponyms from large text corpora. In: Proceedings of the 14th International Conference on Computational Linguistics, Nantes, France (1992)
18. Resnik, P.: Selection and Information: A Class-based Approach to Lexical Relationships. PhD thesis, University of Pennsylania (1993)
19. Maedche, A., Staab, S.: Discovering conceptual relations from text. In: Proceedings of ECAI 2000, IOS Press (2000)
20. Faure, D., Nedellec, C.: A corpus-based conceptual clustering method for verb frames and ontology acquisition. In: LREC-98 Workshop on Adapting Lexical and Corpus Resources to Sublanguages and Applications, Granada, Spain (1998)
21. Kavalec, M., Maedche, A., Svatek, V.: Discovery of lexical entries for non-taxonomic relations in ontology learning. In: SOFSEM – Theory and Practice of Computer Science, Springer LNCS 2932 (2004)
22. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man and Cybernetics **19** (1989) 17–30
23. Deitel, A., Faron, C., Dieng, R.: Learning ontologies from rdf annotations. In: Proc. of the IJCAI Workshop in Ontology Learning. (2001)
24. Doan, A., Domingos, P., Levy, A.: Learning source descriptions for data integration. In: Proc. of 3rd Int. Workshop on the Web and Databases, Dallas, Texas (2000) 81–86
25. Wielinga, B., Th, A., Wielemaker, S., Sandberg, J.: From thesaurus to ontology. In: Proc. of Int. Conf. on Knowledge Capture, New Yory, NY, USA, ACM Press (2001) 194–201
26. Golbeck, J., Fragoso, G., Hartel, F., Hendler, J., Parsia, B., Oberthaler, J.: The national cancer institute's thesaurus and ontology. Journal of Web Semantics **1** (2003)
27. Beneventano, D., Bergamaschi, S., Guerra, F., Vincini, M.: Synthesizing an integrated ontology. IEEE Internet Computing **XX** (2003) 42–51