Concept Adjustment for Description Logics

Yue Ma*
Technische Universität Dresden
Institut für theoretische Informatik
Dresden, Germany
mayue@tcs.inf.tu-dresden.de

Felix Distel
Technische Universität Dresden
Institut für theoretische Informatik
Dresden, Germany
felix@tcs.inf.tu-dresden.de

ABSTRACT

There exist a handful of natural language processing and machine learning approaches for extracting Description Logic concept definitions from natural language texts. Typically, for a single target concept several textual sentences are used, from which candidate concept descriptions are obtained. These candidate descriptions may have confidence values associated with them. In a final step, the candidates need to be combined into a single concept, in the easiest case by selecting a relevant subset and taking its conjunction. However, concept descriptions generated in this manner can contain false information, which is harmful when added to a formal knowledge base. In this paper, we claim that this can be improved by considering formal constraints that the target concept needs to satisfy. We first formalize a reasoning problem for the selection of relevant candidates and examine its computational complexity. Then, we show how it can be reduced to SAT, yielding a practical algorithm for its solution. Furthermore, we describe two ways to construct formal constraints, one is automatic and the other interactive. Applying this approach to the SNOMED CT ontology construction scenario, we show that the proposed framework brings a visible benefit for SNOMED CT development.

Categories and Subject Descriptors

I.2.4 [**Knowledge Representation Formalisms and Methods**]: Representation languages; I.2.6 [**Learning**]: Concept Learning; F.4.3 [**Formal Languages**]: Decision problems

General Terms

Concept Adjustment

Keywords

Description Logics, Biomedical Ontologies, Constraints, Natural Language Processing

*This work has been funded by the DFG Research Unit FOR 1513, project B1.

1. INTRODUCTION

A formal representation of domain information is often the preferred way to express knowledge because formal knowledge allows for automatic manipulation by machines. This is of particular importance when the amount of knowledge is large, such as in the medical domain. Among many promising knowledge representation formalisms are Description Logics, upon which the Ontology Web Languages (OWL2[1]) is based. Besides, Description Logics have been applied to the widely used medical ontology SNOMED CT that contains more than 311,000 formally defined concepts.

Adding new concepts with their formal definitions to an ontology is a tedious, costly and error-prone process, which needs to be performed manually by specially trained knowledge engineers. Different approaches have been proposed for assisting ontology generation. Among them are machine learning and natural language processing based approaches such as [4, 18, 8]. In most settings, several natural language sentences are available for a single target concept, and one concept description is obtained from each of these sentences. Sometimes, a weight representing a confidence degree is returned together with a description. In this work, these automatically generated concept descriptions are called description candidates. Typically, a description candidate obtained from a sentence only captures one aspect of the target concept. The relevant candidates must be selected and combined into a single description.

A drawback of formal knowledge representation formalisms is that they are not very error tolerant. Even small errors can lead to unpredictable consequences. Therefore maintaining correctness of the definitions is crucial [15]. We propose to use formal constraints to ensure the quality of the definitions. Formal constraints include positive and negated logical formulae that a good combination of definition candidates should satisfy. Potential sources for these constraints are design manuals, existing knowledge about the concept from within the ontology, or manually added constraints.

Only those description candidates satisfying the formal constraints should be selected as definitions of concepts. This ensures the quality of the final concept description. In this paper, we formalize the task of automatic selection of good candidates for a given set of candidates and constraints as *concept adjustment for formal definitions in Description Logics*. For illustration, consider the following scenario where an ontology is generated from text.

Definition candidates

In [19] an approach for extracting superclass relationships for a given target concept is presented. This is complemented by our proposed formalism in [8] for extracting other named relationships (so-called DL roles) between concepts. For example, if Baritosis is the target concept, then from the sentences given in Table 1 the

[1]http://www.w3.org/TR/owl2-profiles

Table 1: Sentence Examples

"Baritosis is a benign type of pneumoconiosis, which is caused by long-term exposure to barium dust."
"Baritosis are nonfibrotic forms of pneumoconiosis that result from inhalation of iron oxide."
"Baritosis is one of the benign pneumoconiosis in which inhaled particulate matter lies in the lungs for years."
"Baritosis is due to inorganic dust lies in the lungs."

following relationships are desired results of the text mining.

$$\mathsf{Baritosis} \mid \mathsf{ISA} \mid \mathsf{Pneumoconiosis}, w1$$
$$\mathsf{Baritosis} \mid \mathsf{Finding\_site} \mid \mathsf{Lung}, w2$$
$$\mathsf{Baritosis} \mid \mathsf{Causative\_agent} \mid \mathsf{Barium\_dust}, w3$$
$$\mathsf{Baritosis} \mid \mathsf{Associated\_morphology} \mid \mathsf{Deposition}, w4$$

Here, $A|R|B$ means *concept A has R relationship with concept B* and w_i denotes the confidence degree returned for $A|R|B$.

Each discovered superclass (here denoted by ISA) is a description candidate, i.e. in this example Pneumoconiosis. The other relationships give rise to concept descriptions that are in DL terminology called *existential restrictions*. For the above example, one would obtain the following candidates $\{\exists$Finding_site.Lung, Pneumoconiosis, $\exists$Causative_agent.Barium_dust, $\exists$Associated_morphology.Deposition$\}$. Once the superclasses and existential restrictions of a new concept are known, the concept can be defined as the conjunction of its superclasses and the existential restrictions.

Formal constraints[2]

Definition candidates with their weights returned by natural language processing approaches are usually obtained through analysis of lexical or linguistic features of some given sentences. There is typically no mechanism to guarantee that candidates are logically sound. Logical soundness, is instead characterized by formal constraints. Ideally, formal constraints come from knowledge engineers. For example, the knowledge engineer might state that Baritosis should be a kind of Disease. In particular, manually added constraints can be either too costly or too few to filter out problematic definition candidates. We therefore consider other promising ways to obtain formal constraints.

First, in some fortunate cases partial knowledge about a concept is already available in the ontology. For example, in SNOMED CT there might be a partial definition present in the ontology, that one wants to extend to a full definition. Then the existing partial definition can be used as a constraint within our approach.

Second, large ontologies usually provide a design manual, that codifies design choices that have been made early in the ontology engineering process. Take SNOMED CT as example, its User Guide[3] defines permissible values for each role (range restrictions). For instance, Body_structure is the range for Finding_site. So any candidate that uses the Finding_site role in an existential restriction $\exists$Finding_site.C, where C is not subsumed by Body_structure would violate the User Guide. Such restrictions can be encoded as logical constraints.

Finally, we also consider an interactive way to obtain constraints. It is based on the intuition that new definitions bring new logical consequences. A first combination of candidates is generated (e.g. using only the constraints from the design manual) and the logical consequences for this definition are computed. These are then presented to a knowledge engineer, who can mark some of them as *intended* or *unintended*. This process is similar to formative ontology evaluation described in [11]. The unintended logical consequences will become new negative constraints and the intended ones become new positive constraints for the next round of candidate selection.

[2]Formal constraints are described informally here and will be formalized in Section 3 after the introduction of Description Logics.

[3]http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html

Table 2: Syntax and Semantics of $\mathcal{EL}$

Name	Syntax	Semantics
concept name	A	$A^{\mathcal{I}} \subseteq \Delta_{\mathcal{I}}$
role name	r	$r^{\mathcal{I}} \subseteq \Delta_{\mathcal{I}} \times \Delta_{\mathcal{I}}$
top concept	$\top$	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
conjunction	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$(\exists r.C)^{\mathcal{I}} = \{x \mid \exists y\colon (x,y) \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$
primitive definition	$A \sqsubseteq C$	$A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$
full definition	$A \equiv C$	$A^{\mathcal{I}} = C^{\mathcal{I}}$

Our approach is a hybrid of both information extraction from unstructured resources (such as text) and candidate selection via formal reasoning. Unlike most declarative Information Extraction approaches, that encode the information extraction process in logics [12, 14, 13, 17], we treat them independently, as complements of each other. This allows us to benefit from cutting-edge techniques from both fields.

We formalize our approach in Description Logics terminology in Section 3 after introducing some preliminaries in Section 2. We then analyze the computational complexities of different related problems in Section 4. In Section 5, an encoding from the concept adjustment problem to SAT is given to serve as the basis of an algorithm to compute the desired candidates by benefiting from highly optimized SAT solvers. In Section 6, we apply our approach to the concept definition generation problem for SNOMED CT, showing that the proposed concept adjustment framework can have a significant contribution for generating high quality definitions for a formal ontology. Section 7 discusses the related work and Section 8 concludes the paper with a perspective on the future work.

2. PRELIMINARIES

2.1 The Description Logic $\mathcal{EL}$

Concept descriptions in the Description Logic $\mathcal{EL}$ are built from a set of concept names N_C and a set of role names N_R using the constructors *top concept* $\top$, *conjunction* $\sqcap$, and *existential restrictions* $\exists$. This is shown in the syntax column of Table 2. A concept description C is called an *atom* if it is either a concept name or of the form $C = \exists r.D$ for some concept description D and some role name $r \in N_R$. Every concept description can be written as a conjunction of atoms.

The semantics of $\mathcal{EL}$ is defined using interpretations $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a non-empty *domain* $\Delta_{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$ mapping role names to binary relations on $\Delta_{\mathcal{I}}$ and concept descriptions to subsets of $\Delta_{\mathcal{I}}$ according to Table 2.

Among the *axioms* in $\mathcal{EL}$ are full definitions, primitive definitions and general concept inclusions (GCIs). *Full definitions* are statements of the form $A \equiv C$, *primitive definitions* are statements of the form $A \sqsubseteq C$ and *GCIs* are of the form $C \sqsubseteq D$ where A is a concept name and C and D are concept descriptions. A TBox $\mathcal{T}$ is a set of axioms of these three types. We say that the interpretation

$\mathcal{I}$ is a *model* of $\mathcal{T}$ if $A^\mathcal{I} = C^\mathcal{I}$ (or $A^\mathcal{I} \subseteq C^\mathcal{I}$, $C^\mathcal{I} \subseteq D^\mathcal{I}$) holds for every full definition $A \equiv C$ (primitive definition $A \sqsubseteq C$, GCI $C \sqsubseteq D$, respectively) from $\mathcal{T}$. A concept description C is said to be subsumed by the concept D with respect to the TBox $\mathcal{T}$ (denoted by $\mathcal{T} \models C \sqsubseteq D$) if $C^\mathcal{I} \subseteq D^\mathcal{I}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$. It is well-known that subsumption reasoning in $\mathcal{EL}$ is tractable, i.e. given concept descriptions C and D, and a TBox $\mathcal{T}$ it can be decided in polynomial time if $\mathcal{T} \models C \sqsubseteq D$ [2].

2.2 Satisfiability Problem

For complexity arguments we use the standard boolean satisfiability problem for a propositional language $\mathcal{L}_\mathcal{A}$ with a finite set of propositional variables $\mathcal{A} = \{p_1, \ldots, p_n\}$. A literal is a variable p or its negation $\neg p$. A clause $K = l_1 \vee l_2 \vee \ldots \vee l_k$ is a disjunction of literals. A CNF formula is a conjunction of clauses, which is usually represented as a set of clauses $F = \{K_1, K_2, \ldots, K_m\}$. Deciding if a formula in CNF is satisfiable is called the satisfiability (SAT) problem which is NP-complete. Even though the SAT problem is intractable, state of the art SAT solvers are highly optimized and can deal with large size inputs [1].

3. PROBLEM

In this section we formalize the concept adjustment problem described in the introduction as a DL reasoning problem. In our setting constraints are simply GCIs or negated GCIs where either the left hand side or the right hand side is a concept variable X, i.e. a concept name not occurring in $\mathcal{T}$ or on the other side of a constraint. We distinguish constraints of the following four types:

$$D \sqsubseteq X \quad (1) \qquad D \not\sqsubseteq X \quad (3)$$
$$X \sqsubseteq D \quad (2) \qquad X \not\sqsubseteq D \quad (4)$$

In these constraints D can be a complex concept description. For a complex concept description C that does not use concept variables, we say C satisfies the positive constraint $D \sqsubseteq X$ or $X \sqsubseteq D$ if $\mathcal{T} \models D \sqsubseteq C$ or $\mathcal{T} \models C \sqsubseteq D$, respectively. C satisfies the negative constraint $D \not\sqsubseteq X$ or $X \not\sqsubseteq D$ if $\mathcal{T} \not\models D \sqsubseteq C$ or $\mathcal{T} \not\models C \sqsubseteq D$, respectively.

The task is now straightforward: for a given set of description candidates and a given set of constraints, find a subset of the candidates whose conjunction satisfies the constraints. For complexity considerations we restate this as a decision problem.

Problem 1. (Concept Adjustment (CA)) Input: A set of atomic candidate concept descriptions $\mathcal{S}$, a (possibly empty) ontology $\mathcal{T}$ and a set of constraints $\mathcal{C}$ of the forms (1)–(4).

Question: Is there a subset $\mathcal{S}'$ of $\mathcal{S}$ such that $\bigsqcap \mathcal{S}'$ satisfies all the constraints in $\mathcal{C}$?

We also consider a setting where a weight is associated with each candidate. This is formalized in the following decision problem where the idea is to maximize the least confidence value among the selected candidates.

Problem 2. (Maximal Confidence Concept Adjustment (MCA)) Input: A set of atomic candidate concept descriptions $\mathcal{S}$, a real number $k \in [0, 1]$, a (possibly empty) ontology $\mathcal{T}$ and a set of constraints $\mathcal{C}$ of the forms (1)–(4), together with a confidence function $\mathrm{wt}\colon \mathcal{C} \to [0, 1]$.

Question: Is there a subset $\mathcal{S}'$ of $\mathcal{S}$ such that $\bigsqcap \mathcal{S}'$ satisfies all the constraints in $\mathcal{C}$ and $\min\{\mathrm{wt}(S) \mid S \in \mathcal{S}'\} \geq k$?

4. COMPLEXITY

Both problems CA and MCA are contained in NP, but not all variants are also NP-hard. Containment in NP is clear, since one can simply guess a subset of $\mathcal{S}$ and verify in polynomial time if it is a solution (remember that subsumption reasoning in $\mathcal{EL}$ is tractable). We shall see that the restricted variants of CA and MCA which allow only constraints of types (1)–(3) are tractable (and thus not NP-hard unless P=NP). The variants that allow for the full set of constraints are NP-hard. NP-hardness is thus caused by constraints of type (4).

4.1 Tractable Variants

We show that restricted versions of CA and MCA are tractable. Let an instance $(\mathcal{T}, \mathcal{S}, \mathcal{C})$ of CA be given. We first consider the variant that restricts to constraint types (2) and (3). Notice that if a concept C satisfies a constraint $X \sqsubseteq D$ or $D \not\sqsubseteq X$ and E is a concept description satisfying $E \sqsubseteq C$ then E also satisfies the constraint. In particular, if $\bigsqcap \mathcal{S}'$ satisfies all constraints for some $\mathcal{S}' \subseteq \mathcal{S}$ then $\bigsqcap \mathcal{S}$ also satisfies them. Hence, if only constraint types (2) and (3) occur, then there is a solution to the CA problem iff $\mathcal{S}$ itself is a solution. The latter can be verified in polynomial time since subsumption reasoning in $\mathcal{EL}$ is tractable. The same argument shows that there is a solution to the MCA problem $(\mathcal{T}, \mathcal{S}, k, \mathcal{C}, \mathrm{wt})$ iff $\{S \in \mathcal{S} \mid \mathrm{wt}(S) \geq k\}$ is a solution. Again, this can be verified in polynomial time. Hence, CA and MCA are tractable if we restrict to types (2) and (3).

We now consider the variant that restricts to constraint types (1) to (3). Let $D \sqsubseteq X \in \mathcal{C}$ be a constraint of type (1). A concept $\bigsqcap \mathcal{S}'$ for $\mathcal{S}' \subseteq \mathcal{S}$ satisfies this constraint if and only if $\mathcal{T} \models D \sqsubseteq S$ for all $S \in \mathcal{S}'$. This shows that there is a solution $\mathcal{S}'$ of $(\mathcal{T}, \mathcal{S}, \mathcal{C})$ iff there is a solution $\mathcal{S}'_0$ of $(\mathcal{T}, \mathcal{S}_0, \mathcal{C}_0)$ where

$$\begin{aligned} \mathcal{S}_0 &= \{S \in \mathcal{S} \mid \forall (D \sqsubseteq X) \in \mathcal{C} \colon \mathcal{T} \models D \sqsubseteq S\} \\ \mathcal{C}_0 &= \{c \in \mathcal{C} \mid c \text{ of type (2) or (3)}\}. \end{aligned} \tag{5}$$

Notice that $\mathcal{C}_0$ can be obtained in linear time and $\mathcal{S}_0$ can be computed in polynomial time since subsumption reasoning in $\mathcal{EL}$ is tractable. This shows that restrictions of type (1) can be dealt with in a polynomial time preprocessing step. Tractability of CA and MCA for constraints of types (1)–(3) then follows immediately from tractability of CA and MCA when restricted to (2) and (3).

4.2 NP-hard Variants

Using a reduction from the satisfiability problem for propositional formulae in conjunctive normal form we show that CA is NP-hard, even when restricted to constraints of types (2) and (4).

Let f be a propositional formula in conjunctive normal form over variables $x_1, \ldots, x_n$. Let k be the number of clauses in f. We denote by K_j, $j \in \{1, \ldots k\}$, the set of literals occurring in the j-th clause of f.

We construct an instance of CA as follows. For each literal l we introduce a concept name T_l and for each clause K_j, $j \in \{1, \ldots, k\}$, we introduce a concept name U_j. We use only one role denoted by r. The ontology $\mathcal{T}$ is considered to be empty. The set $\mathcal{S}_f$ consists of the concept descriptions

$$S_l = \exists r.\Big(T_l \sqcap \bigsqcap_{l \in K_j} U_j\Big) \tag{6}$$

for all literals l. A constraint

$$X \sqsubseteq \exists r.U_j \tag{7}$$

is added to the set $\mathcal{C}_f$ for each $j \in \{1, \ldots, k\}$ and a constraint

$$X \not\sqsubseteq \exists r.T_{x_i} \sqcap \exists r.T_{\neg x_i} \tag{8}$$

is added to $\mathcal{C}_f$ for every variable x_i, $i \in \{1, \ldots, n\}$. Intuitively, (7) ensures that a solution must contain S_l for at least one literal l from each clause in f. The constraint (8) ensures that a solution cannot contain both S_{x_i} and $S_{\neg x_i}$ for some $i \in \{1, \ldots, n\}$.

PROPOSITION 1. *The formula f is satisfiable iff $(\mathcal{T}, \mathcal{S}_f, \mathcal{C}_f)$ has a solution.*

PROOF. Assume first that f is satisfiable. Let $\mathcal{V}$ be an assignment of truth values that makes f true. Define $\mathcal{S}_\mathcal{V} = \{S_x \mid \mathcal{V}(x) = 1\} \cup \{S_{\neg x} \mid \mathcal{V}(x) = 0\}$. Let K_k be a clause of f. Since $\mathcal{V}$ makes f true there is one literal $l \in K_k$ such that $\mathcal{V}(l) = 1$. But then

$$\mathcal{T} \models \bigsqcap \mathcal{S}_\mathcal{V} \sqsubseteq S_l = \exists r.\Big(T_l \sqcap \bigsqcap_{l \in K_j} U_j\Big) \sqsubseteq \exists r.U_k.$$

Hence, $\bigsqcap \mathcal{S}_\mathcal{V}$ satisfies all constraints of type (7). To see that $\mathcal{S}_\mathcal{V}$ also satisfies the constraints of type (8) notice that for every literal l the concept S_l is the only candidate that is subsumed by $\exists r.T_l$. Then either $S_{x_i} \notin \mathcal{S}_\mathcal{V}$ and therefore $\mathcal{T} \not\models \bigsqcap \mathcal{S}_\mathcal{V} \sqsubseteq \exists r.T_{x_i}$, or $S_{\neg x_i} \notin \mathcal{S}_\mathcal{V}$ and therefore $\mathcal{T} \not\models \bigsqcap \mathcal{S}_\mathcal{V} \sqsubseteq \exists r.T_{\neg x_i}$. Thus $\mathcal{S}_f$ is a solution.

Next, let $\mathcal{S}'$ be a solution to $(\mathcal{T}, \mathcal{S}_f, \mathcal{C}_f)$. Since $\bigsqcap \mathcal{S}'$ satisfies the constraint of type (8) for all variables x_i, either $\mathcal{T} \not\models \bigsqcap \mathcal{S}' \sqsubseteq \exists r.T_{x_i}$ and thus $S_{x_i} \notin \mathcal{S}'$, or $\mathcal{T} \not\models \bigsqcap \mathcal{S}' \sqsubseteq \exists r.T_{\neg x_i}$ and therefore $S_{\neg x_i} \notin \mathcal{S}_\mathcal{V}$. Hence one can define a valuation $\mathcal{V}'$ that maps each variable x_i to 1 if $S_{x_i} \in \mathcal{S}'$, and x_i to 0 if $S_{\neg x_i} \in \mathcal{S}'$, and is constantly 1 if neither holds.

Let K_j be a clause from f. Since $\mathcal{S}'$ is a solution to the CA problem it satisfies $\bigsqcap \mathcal{S}' \sqsubseteq \exists r.U_j$, and therefore there must be some atom $S_l \in \mathcal{S}'$ such that $S_l \sqsubseteq \exists r.U_j$. By (6 this implies that l occurs in K_j, and the definition of $\mathcal{V}'$ implies that $\mathcal{V}'(l) = 1$. Therefore, for each clause K_j there is a literal l in K_j that $\mathcal{V}'$ evaluates to 1, i.e. the full formula f evaluates to 1. In particular f is satisfiable. □

Hardness of MCA follows immediately from hardness of CA, since CA can be viewed as a special case of MCA when all confidence values are 1. NP-completeness follows from this proposition and containment of CA and MCA in NP.

THEOREM 1. *CA and MCA are NP-complete.*

5. SAT ENCODING

We have argued in Section 4 that CA and MCA can be solved in non-deterministic polynomial time. However, our argument does not yield a practical algorithm for computing a solution. In this section, we propose an encoding of CA in SAT, with the aim of using SAT solvers to come up with a solution, at least when SNOMED CT is used as the underlying ontology.

SNOMED CT is special, since it only contains full definitions and primitive definitions [16]. SNOMED CT further has the property that it is *acyclic*, i.e. the concept name on the left hand side of a definition cannot occur on its right hand side neither explicitly nor implicitly. This means that SNOMED CT can be unfolded by recursively replacing each fully defined concept name (i.e. a concept name occurring on the left hand side of a full definition) by its definition. Unfolding results in a logically equivalent TBox where only the primitive definitions remain.

The encoding itself is relatively straightforward, but it makes use of the following characterization of subsumption in $\mathcal{EL}$.

LEMMA 1. *Let $\mathcal{T}$ be a TBox containing only primitive definitions. Let C and D be concept descriptions that can be written as as $C = C_1 \sqcap \cdots \sqcap C_n$ and $D = D_1 \sqcap \cdots \sqcap D_m$ where C_j, $1 \leq j \leq n$, and D_i, $1 \leq i \leq m$, are atoms. Then $\mathcal{T} \models C \sqsubseteq D$ iff for every atom D_i, $1 \leq i \leq m$, there is an atom C_j, $1 \leq j \leq n$, such that $\mathcal{T} \models C_j \sqsubseteq D_i$.*

PROOF. The "if"-direction is trivial, which is why we only prove the "only if"-direction. Assume that there is an atom D_i for some $1 \leq i \leq m$ such that $\mathcal{T} \not\models C_j \sqsubseteq D_i$ for all $1 \leq j \leq n$. This means there are models $\mathcal{I}_j$, $1 \leq j \leq n$, of $\mathcal{T}$ with elements $x_j \in \Delta_{\mathcal{I}_j}$ such that $x_j \in C_j^{\mathcal{I}_j}$ but $x_j \notin D_i^{\mathcal{I}_j}$. Since $\mathcal{EL}$ is known to have the tree-model property, we can assume that all these models are tree-shaped with x_j as the root. We now create a new interpretation $\mathcal{I}$ by fusing the nodes x_j into one new node x and show that $\mathcal{I}$ is still a model of $\mathcal{T}$ and x is a counterexample to $\mathcal{T} \models C \sqsubseteq D$.

More formally, we can assume wlog that the domains of the tree shaped models $\Delta_{\mathcal{I}_j}$ are mutually disjoint. We define the new model $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ as follows

$$\Delta_\mathcal{I} = \{x\} \cup \bigcup_{1 \leq j \leq n} \left(\Delta_{\mathcal{I}_j} \setminus \{x_j\}\right).$$

For all $y \neq x$ the interpretations of concept names and role names are defined as in the model $\mathcal{I}_j$ that y stems from, i.e. $y \in A^\mathcal{I}$ iff $y \in A^{\mathcal{I}_j}$ where $y \in \Delta_{\mathcal{I}_j}$ and similarly for roles. This property carries over to complex concepts, because the models are tree shaped and $\mathcal{EL}$ can only talk about role successors, not predecessors.

For x we define the interpretations as follows. For every concept name $A \in N_C$ we define $A^\mathcal{I}$ such that $x \in A^\mathcal{I}$ iff $x_j \in A^{\mathcal{I}_j}$ holds for some model $\mathcal{I}_j$, $1 \leq j \leq n$. Likewise we define for every role name $r \in N_R$ the relation $r^\mathcal{I}$ to contain the pair (x, y), $y \neq x$ iff (x_j, y) is contained in $r^{\mathcal{I}_j}$ for some $1 \leq j \leq n$.

We first show that $\mathcal{I}$ is a model of $\mathcal{T}$. Let $A \sqsubseteq E$ be a primitive definition. Consider $y \in A^\mathcal{I}$. If $y \neq x$, $y \in \Delta_{\mathcal{I}_j}$ for some j, then we obtain $y \in E^\mathcal{I}$ from the above-mentioned fact that $\mathcal{I}$ and $\mathcal{I}_j$ coincide on $\Delta_{\mathcal{I}_j}$ except for x_j. If $y = x$ then by definition there must be a model $\mathcal{I}_j$ such that $x_j \in A^{\mathcal{I}_j}$. Since $\mathcal{I}_j$ is a model of $\mathcal{T}$ this implies that $x_j \in E^{\mathcal{I}_j}$. If E is a concept name, then we get $x \in E^\mathcal{I}$ immediately from the definition of $\mathcal{I}$. If $E = \exists r.F$ is an existential then there must be some $z \in F^{\mathcal{I}_j}$, $z \neq x_j$, such that $(x_j, z) \in r^{\mathcal{I}_j}$. But then also $z \in F^\mathcal{I}$ and $(x, z) \in r^\mathcal{I}$ by the definition of $\mathcal{I}$. This yields $y = x \in E^\mathcal{I}$. Finally, if E is a conjunction of atoms, one can first argue for each of its atoms as above, which yields that $y \in E^\mathcal{I}$ also holds for the conjunction. We have thus proven $A^\mathcal{I} \subseteq E^\mathcal{I}$ for all primitive definitions in $\mathcal{T}$. $\mathcal{T}$ contains only primitive definitions, therefore $\mathcal{I}$ is a model of $\mathcal{T}$.

It remains to show that x is a counterexample to $C \sqsubseteq D$. Consider an atom C_j, $1 \leq j \leq n$. We know that $x_j \in C_j^{\mathcal{I}_j}$. We can use similar arguments as above for the concept E to show that $x \in C_j^\mathcal{I}$. Since this holds for all atoms C_j it follows that $x \in C^\mathcal{I}$. Now, assume that $x \in D_i^\mathcal{I}$. In the case where D_i is a concept name, we obtain that $x_j \in D_i^{\mathcal{I}_j}$ for some $1 \leq j \leq n$, a contradiction to the assumption $x_j \notin D_i^{\mathcal{I}_j}$. If $D_i = \exists r.F$ is an existential then there must be some $y \neq x$ satisfying $y \in F^\mathcal{I}$, $(x, y) \in r^\mathcal{I}$. The element y must stem from a model $\mathcal{I}_j$ for some $1 \leq j \leq n$. Via the construction of $\mathcal{I}$ one can argue that $y \in F^{\mathcal{I}_j}$ and $(x_j, y) \in r^{\mathcal{I}_j}$, yielding $x_j \in D_i^{\mathcal{I}_j}$, again a contradiction. This proves $x \notin D_i^\mathcal{I}$ and thus in particular $x \notin D^\mathcal{I}$. Thus $\mathcal{I}$ is a model of $\mathcal{T}$ in which $C^\mathcal{I} \subseteq D^\mathcal{I}$ does not hold. In particular $\mathcal{T} \not\models C \sqsubseteq D$. □

In the following we assume that $\mathcal{T}$ has been completely unfolded with respect to the full definitions such that Lemma 1 is applicable.

We are now ready to describe the actual encoding. We start with an instance $(\mathcal{T}, \mathcal{S}, \mathcal{C})$ of CA. In a first step we do a preprocessing according to (5), i.e. every candidate violating a type (1) constraint

is removed. This leaves us with only constraints of type (2) to (4). For every remaining candidate $S_i \in \mathcal{S}$ we introduce a propositional variable s_i. Furthermore, a clause is introduced for each constraint from $\mathcal{C}$ of types (2) to (4). The final formula f is then the conjunction of these clauses. We construct f in such a way that a solution $\mathcal{S}'$ contains the candidate S_i iff the satisfying assignment to f sets s_i to true. For a type (2) constraint $X \sqsubseteq D$ we can assume that D is atomic, otherwise we can equivalently replace it by a set of constraints, one for each atom in D. We define

$$f_{X \sqsubseteq D} = \bigvee_{\mathcal{T} \models S_i \sqsubseteq D} s_i.$$

For a type (3) constraint $D \not\sqsubseteq X$ we define

$$f_{D \not\sqsubseteq X} = \bigvee_{\mathcal{T} \not\models D \sqsubseteq S_i} s_i,$$

and for a type (4) constraint $X \not\sqsubseteq D$ we define

$$f_{X \not\sqsubseteq D} = \bigvee_{D' \text{ atom of } D} \bigwedge_{\mathcal{T} \models S_i \sqsubseteq D'} \neg s_i.$$

To generate these formulae, we have to perform one subsumption check for each pair of a candidate and an atom from a constraint, totaling to at most $|\mathcal{C}| \cdot |\mathcal{S}|$ subsumption checks. Since subsumption checking in $\mathcal{EL}$ is tractable this can be done in polynomial time. Each subset $\mathcal{S}' \subseteq \mathcal{S}$ gives rise to a truth assignment $\phi_{\mathcal{S}'}$ where $\phi_{\mathcal{S}'}(s_i) = 1$ if $S_i \in \mathcal{S}'$ and $\phi_{\mathcal{S}'}(s_i) = 0$ otherwise.

LEMMA 2. *For any constraint of type (2) to (4) and a set $\mathcal{S}' \subseteq \mathcal{S}$ the concept $\bigsqcap \mathcal{S}'$ satisfies c iff $\phi_{\mathcal{S}}$ makes f_c true.*

PROOF. We prove the for each of the three types of constraints separately. For a constraint $X \sqsubseteq D$ where D is atomic it follows from Lemma 1 that $\mathcal{T} \models \bigsqcap \mathcal{S}' \sqsubseteq D$ iff there is some $S_i \in \mathcal{S}'$ satisfying $\mathcal{T} \models S_i \sqsubseteq D$. By definition of $\phi_{\mathcal{S}'}$ this holds iff $\phi_{\mathcal{S}'}(s_i) = 1$ for some $S_i \in \mathcal{S}$ with $\mathcal{T} \models S_i \sqsubseteq D$, i.e. iff $\phi_{\mathcal{S}'}$ makes $f_{X \sqsubseteq D}$ true.

Consider now a constraint $D \not\sqsubseteq X$. In this case, we do not need Lemma 1. It follows directly from the definition of conjunction that $\mathcal{T} \models D \sqsubseteq \bigsqcap \mathcal{S}'$ iff $\mathcal{T} \models D \sqsubseteq S_i$ for all $S_i \in \mathcal{S}'$. Conversely, $\mathcal{T} \not\models D \sqsubseteq \bigsqcap \mathcal{S}'$ iff $\mathcal{T} \not\models D \sqsubseteq S_i$ for some $S_i \in \mathcal{S}'$. By definition of $\phi_{\mathcal{S}'}$ this holds iff $\phi_{\mathcal{S}'} = 1$ for some $S_i \in \mathcal{S}$ satisfying $\mathcal{T} \not\models D \sqsubseteq S_i$, i.e. iff $\phi_{\mathcal{S}'}$ satisfies $f_{D \not\sqsubseteq X}$.

For a constraint $X \not\sqsubseteq D$ the claim is a direct translation of Lemma 1, which states that $\mathcal{T} \models \bigsqcap \mathcal{S}' \sqsubseteq D$ iff for all atoms D' of D there is some $S_i \in \mathcal{S}'$ such that $\mathcal{T} \models S_i \sqsubseteq D'$, i.e. iff $\phi_{\mathcal{S}'}$ makes

$$\bigwedge_{D' \text{ atom of } D} \bigvee_{\mathcal{T} \models S_i \sqsubseteq D'} s_i$$

true. The claim is simply the negation of the above statement. □

The following theorem is a simple consequence of Lemma 2.

THEOREM 2. *$\mathcal{S}'$ is a solution to the CA problem $(\mathcal{T}, \mathcal{S}, \mathcal{C})$ iff $\phi_{\mathcal{S}'}$ makes the following formula true:*

$$f = \bigwedge_{c \in \mathcal{C}} f_c.$$

This provides us with a practical algorithm for solving CA problems. To solve MCA problems, we can use an iterative approach: First solve the CA problem obtained by ignoring the confidence values. In a next step set all variables corresponding to a candidate with confidence lower or equal to the minimal confidence in the first solution to false. This process is repeated until the resulting CA problem can no longer be solved. The last solution must be optimal. At the latest, this process terminates after $|\mathcal{S}|$ iterations.

6. EVALUATION

In this section, we apply the above theory to the setting of learning logical definitions from texts. For this, we need two steps: one to learn definition candidates from texts, and the other to refine concepts by formal constraints, as detailed in the following.

6.1 Experiment Set Up

We carry out a one-concept-leave-out evaluation. That is, in each round of experiments one concept is removed from the complete SNOMED CT ontology. This concept is then used as the target concept in the learning process, during which a new description is learned from text. The learned concept description is compared to the original concept description which has been removed, essentially using the original description as the gold standard. The text corpus and the ontology examined in the experiment are given below.

6.1.1 Ontology and Text Corpus

We take SNOMED CT as the ontology for our experiments because it is widely used and written in $\mathcal{EL}$ (if role hierarchies are neglected). In our experiments we restrict to the concepts that are descendants of Disease(disorder)[4]. Among the 65,073 descendants of Disease, 853 concepts are mentioned in our available text corpus. In the experiment, only these 853 concepts are considered.

For the experiment we chose a combination of two text corpora: WIKI and D4D. WIKI is obtained by querying Wikipedia with one-word SNOMED CT concept names, resulting in a document consisting of around 53,943 distinct sentences with 972,038 words. D4D contains textual definitions extracted by querying DOG4DAG[5] [19] over concepts that have relationships via the most frequent attributes[6] used for Disease, obtaining 7,092 distinct sentences with 112,886 words. So in all, our textual data contains 61,035 sentences with 1,084,924 words.

6.1.2 Learning Definition Candidates from Texts

Following [8], we use the *distance supervision* [10] approach for the definition candidate extraction. This approach is independent of manual annotation of textual data, which would be costly. The process is sketched below. For more details, refer to [8].

The text corpus is first annotated by *Metamap*[7], a tool developed to identify SNOMED CT concepts occurring in texts. This is illustrated in the 1st row of Table 3, where "Baritosis" and "barium dust" in the sentence are annotated with concepts Baritosis and Barium_Dust, respectively, by Metamap.

Then the annotated sentences are aligned with the SNOMED CT relationship base. That is, if a sentence contains two concepts that are in a relationship in SNOMED CT, this sentence is aligned with the corresponding role. Using DL reasoning we first generate the set of all triples $A|R|B$ that are entailed by SNOMED CT in the following sense: $\mathcal{RB} = \{A|R|B \colon \text{SNOMED CT} \models A \sqsubseteq \exists R.B\}$. Reasoning provides a way to use implicit information encoded in SNOMED CT[8]. Because the inferred role base $\mathcal{RB}$ contains the relationship Baritosis | Causative_agent | Barium_dust, the sentence is aligned to Causative_agent. Once the sentence is aligned,

[4]The type information attached to each concept (i.e. disorder) is the same for all. For simplicity, we ignore it below if not necessary.

[5]DOG4DAG is a system capable of retrieving and ranking textual definitions from the web. However, it has query number restrictions so that we cannot query as many as SNOMED CT concepts.

[6]Three are used: Associated_morphology, Causative_agent, and Finding_site

[7]http://metamap.nlm.nih.gov/

[8]For example, for Finding_site 630,547 relation pairs are obtained through reasoning compared to only 43,079 explicitly given [8].

Table 3: Text Alignment and Features

Annotated Sentence	"*Baritosis*/Baritosis is pneumoconiosis caused by *barium dust*/Barium_Dust."
SNOMED CT relationship	Baritosis | Causative_agent | Barium_Dust

features are extracted from it. As a typical feature for relation extraction, the between-words of annotated phrases are used.

During the candidate extraction for a target concept, all the aligned sentences are divided into training and test sets as follows: If an annotated sentence does not contain mentions of the target concept, then it goes into the training set; otherwise, it becomes a test sentence. This way, no information about the target concept will be allowed in the training data. Since several sentences can be aligned with the same role, weights for different features extracted from different sentences are learned by the Stanford multi-class classifier [9] based on the training set.

Description candidates for the target concept are obtained by running the classifier on the test data. For each test sentence, it predicts a relationship between the annotated concepts, one of which is the target concept. This yields a triple $A|R|B$ which is interpreted as a description candidate $\exists R.B$ for the target concept A.

6.1.3 Construction of Formal Constraints

For evaluation, we construct formal constraints based on the following observations: (1) The existing SNOMED CT concept hierarchy is assumed to be correct and we introduce constraints that ensure that the hierarchy is preserved. (2) Explicitly defined domain and range information is available for the limited number of roles used in SNOMED CT.

For the sake of the experiment, we use the original SNOMED CT, i.e. without the target concept removed, to simulate an expert. Assuming that in a real world scenario they could be provided by a human expert we take into account the direct superclasses and subclasses of the target concept. This is shown in Lines 4–11 of Algorithm 1, the resulting constraints are of types (1) and (2). The role domain and range restrictions can be gathered from the SNOMED CT User Guide, as shown in Lines 13–15 and Lines 16–18 of Algorithm 1, respectively, resulting in constraints of type (4).

Once the constraints are ready, we can use the concept adjustment approach proposed in the previous sections to select good candidates for the definition of the target concept. Note that the concept adjustment should only use those parts of SNOMED CT that do not contain information about the target concept. For this, we use a module extraction approach [6, 5] to extract a module about the target concept (Line 20). Then we delete the obtained module from SNOMED CT ensuring that the rest is of no relevance to the target concept (Line 21). Note that one cannot simply extract a module about all the concept names of SNOMED CT except for the target concept because the resulting module would include some information about the target concept via links between the target concept and other concepts. Finally, the optimized group of candidates is generated by solving the concept adjustment task (Line 22 in Algorithm 1), taking the given candidates, the generated constraints, and the background knowledge from SNOMED CT without information about the target concept.

EXAMPLE 1. *Take* Baritosis *as the target concept. From the textual data, we get 10 sentences that have annotation* Baritosis *by Metamap, from which the following relationships were extracted*

Algorithm 1 Definition Selection (with automatic constraints)

```
1: procedure DefSeleAutoCons(TargetCon)
2:     Candidates ← candidateExtract(TextualData)
3:     AutoConstraints ← ∅
4:     for all C ∈ SNOMED do
5:         if SNOMED ⊨ TargetCon ⊑ C then
6:             AutoConstraints.add(TargetCon ⊑ C)
7:         end if
8:         if SNOMED ⊨ C ⊑ TargetCon then
9:             AutoConstraints.add(C ⊑ TargetCon)
10:        end if
11:    end for
12:    for all R ∈SNOMED do
13:        for all B ∉ Range(R) do
14:            AutoConstraints.add(TargetCon ⋢ ∃R.B)
15:        end for
16:        if TargetCon ∉ Domain(R) then
17:            AutoConstraints.add(TargetCon ⋢ ∃R.⊤)
18:        end if
19:    end for
20:    Module ← ModuleExtract(TargetCon, SNOMED)
21:    PartSnd ← SNOMED \ Module
22:    return CandidateSelection(PartSnd, AutoConstraints, Candidates)
23: end procedure
```

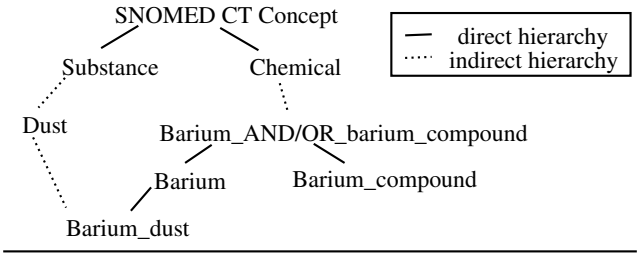

Figure 1: Example on Concept Adjustment

with high weights by the learning approach given in Section 6.1.2:

Baritosis | Causative_agent | Barium_compound,	0.92140
Baritosis | Causative_agent | Dust,	0.97038
Baritosis | Finding_site | Lung_structure,	0.99999
Baritosis | Causative_agent | Barium_dust,	0.99997

Therefore, we have the following definition candidates: {∃Causative_agent.Barium_compound, ∃Causative_agent.Dust, ∃Finding_site.Lung_structure, ∃Causative_agent.Barium_dust}. *However, by* SNOMED CT, ∃Causative_agent.Barium_compound *is an undesired candidate. Due to the high weights returned by the learning approach, it is hardly possible to exclude the undesired candidate by looking at the weights alone.*[9] *Indeed, it can be*

[9]Note that even the concept type information is used as

Algorithm 2 Definition Selection (with interactive constraints)

```
 1: procedure DefSele(TargetCon,AutoConstraints)
 2:     Candidates ← DefSeleAutoCons(TargetCon)
 3:     Module ← ModuleExtract(TargetCon, SNOMED)
 4:     PartSnd ← SNOMED \ Module
 5:     InteractiveConstraints ← ∅
 6:     MoreConstraints ← true
 7:     Definition ← TargetCon = ⊓_{C∈Candidates} C
 8:     while MoreConstraints do
 9:         NewImplication ← Classification(PartSnd∪Definition)\
    Classification(PartSnd)
10:         for all p in NewImplication do
11:             question ← Is p is desired?
12:             if yes then  ▷ a new positive (Type 1, 2) constraint
13:                 InteConstraints.add(p)
14:             else         ▷ a new negated (Type 3, 4) constraint
15:                 InteConstraints.add(¬ p)
16:             end if
17:         end for
18:         Constraints ← AutoConstraints ∪ InteConstraints
19:         Candidates ← CandidateSelection(Candidates, Con-
    straints, PartSnd)
20:         MoreConstraints ← Stop interaction?
21:     end while
22: end procedure
```

discarded by the concept adjustment framework. This is because, Algorithm 1 yields a constraint on Causative_agent*, which ensures that the value cannot be a sort of* Chemical*. But as shown in Figure 1,* Barium_compound *is indeed a descendant of* Chemical *in the background knowledge base, thus being discarded.*

Obviously, in real life applications it is not possible to query an existing ontology for constraints as in Algorithm 1. In this case, we propose to use interactive communication with knowledge engineers, as detailed in Algorithm 2. Line 1 in Algorithm 2 simply runs Algorithm 1 to obtain a first set of definition candidates. The conjunction over the selected candidates yields a definition for the target concept (Line 7). If the knowledge engineer wants to continue to find more constraints, new implications are computed (Line 9). These are obtained by comparing the classifications of PartSnd, the module with no information about the target concept, and PartSnd plus the newly computed definition of the target concept. Then for each new implication, the expert decides if it is desired (Line 11). If yes, it will be added as a positive constraint (Line 12–13); otherwise, it will be a new negated constraint (Line 14–15). Once the constraint set is updated using interactive constraints (Line 18), new candidates will be computed again by Algorithm 1 (Line 19). This interactive process proceeds until the expert does not want more constraints.

In our preliminary evaluation, we only consider the scenario that constraints are built automatically as described in Algorithm 1. The evaluation is done automatically by comparing the learned and the adjusted candidates to those originally given in SNOMED CT.

6.2 Evaluation and Discussion

In this section we discuss the evaluation metric and results. As the evaluation metric, we consider an extended precision to measure what percentage of a set of candidates is correct with respect to SNOMED CT semantics. It must take implicit knowledge from SNOMED CT into account, since there is no unique way to define concepts. Different definitions may lead to the same meaning under Description Logics semantics. For example, our learning approach might return a candidate ∃Causative_agent.Dust for the target concept Baritosis. When looking up the definition given in SNOMED CT, this candidate is not explicitly mentioned. However, it is a no harm candidate because in the definition of Baritosis as given in SNOMED CT we have ∃Causative_agent.Barium_dust and SNOMED CT $\models$ Barium_dust $\sqsubseteq$ Dust holds (Figure 1). From this Baritosis $\sqsubseteq$ ∃Causative_agent.Dust follows.

Formally, given a set of candidates $\text{Cands} = \{A|R|B\colon A, B$ are concept names and R is a role name$\}$, the reasoning based precision can be defined as follows:

$$\text{Precision} = \frac{|\{A \mid R \mid B \in \text{Cands}\colon \text{SNOMED} \models A \sqsubseteq \exists R.B\}|}{|\text{Cands}|}.$$

Among the 853 examined concepts are 276 concepts whose candidates generated in the first phase come with Precision < 1. Figure 2 compares the precision values of the sets of description candidates for these 276 target concepts, before and after concept adjustment. Interestingly, we can see that the candidates for all the concepts become 100% correct after the adjustment under the constraints obtained by Algorithm 1, but none of them is so beforehand (most of them have a precision value less than 0.8 without adjustment).

The great increase in precision is partly due to the set of relatively strong constraints that Algorithm 1 constructed. In applications where the design manual of an ontology is not as exhaustive as SNOMED CT's, we believe that an interactive approach as given in Algorithm 2 should be used to generate a rich set of constraints.

In this preliminary evaluation, we do not consider a recall metric because it is unclear how the semantic closure of definitions should be defined. By contrast, we have designed the candidate extraction step (Line 2 in Algorithm 1) to keep as many candidates as possible, such that the information given in the available textual data about the target concept can be exploited to a large extent. The candidates are then verified by the proposed concept adjustment approach to discard potentially erroneous ones.

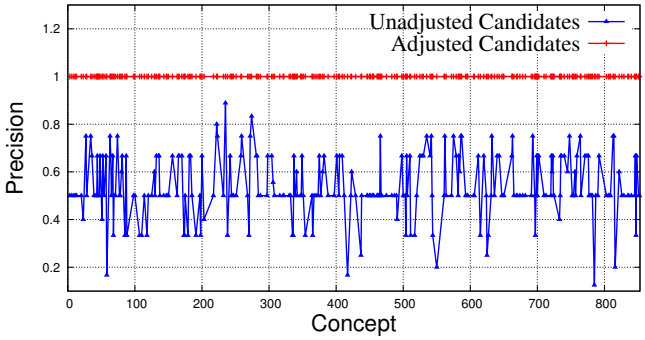

Figure 2: Precision with/without Concept Adjustment

7. RELATED WORK

Formal ontology generation is an important but non-trivial task [4]. Often ontology languages allow to express knowledge both at the instance and the terminology levels, and existing automatic systems for ontology construction may differ in the type of knowledge that is learned. Like [18, 8], our work is at the terminological level. Völker [18] describes some first approaches for generating OWL

a feature during the description learning, the candidate ∃Causative_agent.Barium_compound still cannot be excluded because its type is the same as that of the correct candidate Barium_dust by SNOMED CT. And note that the type information is different from the concept hierarchy in SNOMED CT.

DL concept definitions for generic domains by applying syntactic transformation rules. For a fixed set of relations, [8] presents a method to learn concept definitions via a relation extraction based approach. By contrast, [7, 3] are at the instance level, that is, they find logical descriptions for given instances.

General approaches are often difficult to apply to specific domains, such as SNOMED CT. For example, the approach in [18] is inappropriate for our system because it encounters unresolved reference roles such as ∃Of, which does not have a representation in SNOMED CT. Moreover, different formal expressions (e.g. ∃Caused_by, ∃Due_to, ∃Result_from) will be generated from variant expressions (e.g. "caused by", "due to", "result from"), even if they all express the same relation ∃Causative_agent according to SNOMED CT. Following [8], the candidate extraction procedure used in this paper is based on relation extraction techniques [10]. This can be done for our specific scenario because SNOMED CT has a relatively stable and limited set of roles compared to its large increasing number of concept names. Compared to [8], we assigned less strict algorithm parameters in the experiments for the aim that more potential candidates can be extracted. And then an extra concept adjustment layer, as proposed in this paper, is used to increase the precision of the definition candidates.

In addition, ontology construction systems can differ in the DL languages they consider. For instance, [18] does not specifically consider $\mathcal{EL}$++ constructors, Similar to [3, 8], we take $\mathcal{EL}$ as the target language in this paper with the aim to construct a system for assisting SNOMED CT development. Normally, more expressive languages are more difficult to deal with. But specific techniques for restricted languages can be of particular benefits [8]. In particular, we show that the proposed concept adjustment problems can be tractable with respect to $\mathcal{EL}$ if the constraint types are restricted, and it is NP-Complete when all types are allowed, which is still below the complexity of many other DL reasoning problems.

8. CONCLUSION AND FUTURE WORK

In this paper, the problem of automatic generation of formal definitions of an ontology is considered. We have proposed a novel framework to get formal definition candidates of a good precision. The approach combines both techniques from natural language processing and ontology reasoning. First, based on the distance supervision approach, definition candidates are extracted from textual data about the target concept. Then the concept adjustment approach is proposed to remove those which violate some formal constraints.

Precisely formalized in Description Logic terminology, the proposed concept adjustment problems (*CA* and *MCA*) are analyzed theoretically. We have showed that different constraint types may lead to problems of different complexities: it remains tractable if the constraints are restricted to the first three types, but becomes NP-complete once constraints of the fourth type appear. For implementation, we have studied an encoding based approach which can reduce the concept adjustment problems to a SAT instance. This way, we can benefit from the highly optimized modern SAT solvers for our problem. As evaluation, we have set up experiments where different concepts from the SNOMED CT ontology are considered as the target concept with the formal constraints constructed automatically. Under the defined extended precision, we can see that the definition candidates after the concept adjustment have significantly improved the precision. Thus, the output of the whole framework can provide more reliable formal definitions.

In the future, we will study a proper definition of *recall* which can take into account the reasoning over background knowledge. Based on this, we will improve our approach to achieve both better precision and recall. Meanwhile, we are interested in evaluations about the interactive way for formal constraint construction by examining other ontologies. The extension of the current system to more expressive Description Logics is also under consideration.

9. REFERENCES

[1] C. Ansótegui, M. L. Bonet, and J. Levy. SAT-based maxsat algorithms. *Artificial Intelligence*, 196:77–105, 2013.

[2] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proceedings of IJCAI'05*, 2005.

[3] M. Chitsaz, K. Wang, M. Blumenstein, and G. Qi. Concept learning for $\mathcal{EL}$++ by refinement and reinforcement. In *Proceedings of PRICAI'12*, pages 15–26, 2012.

[4] P. Cimiano. *Ontology learning and population from text - algorithms, evaluation and applications*. Springer, 2006.

[5] B. Konev, M. Ludwig, D. Walther, and F. Wolter. The logical difference for the lightweight description logic $\mathcal{EL}$. *J. Artif. Intell. Res. (JAIR)*, 44:633–708, 2012.

[6] B. Konev, C. Lutz, D. Walther, and F. Wolter. Semantic modularity and module extraction in description logics. In *Proceedings of ECAI*, pages 55–59, 2008.

[7] J. Lehmann and P. Hitzler. Concept learning in description logics using refinement operators. *Machine Learning*, 78(1-2):203–250, 2010.

[8] Y. Ma and F. Distel. Learning formal definitions for Snomed CT from text. In *Proceedings of AIME'13*, 2013.

[9] C. Manning and D. Klein. *Optimization, Maxent Models, and Conditional Estimation without Magic*. Tutorial at HLT-NAACL 2003 and ACL 2003, 2003.

[10] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL/AFNLP'09*, pages 1003–1011, 2009.

[11] V. Pammer, C. Ghidini, M. Rospocher, L. Serafini, and S. Lindstaedt. Automatic support for formative ontology evaluation. In *Proceedings of EKAW'10 Poster and Demo Track*, 2010.

[12] H. Poon and P. Domingos. Joint inference in information extraction. In *Proceedings of AAAI'07*, pages 913–918, 2007.

[13] F. Reiss, S. Raghavan, R. Krishnamurthy, H. Zhu, and S. Vaithyanathan. An algebraic approach to rule-based information extraction. In *Proceedings of ICDE'08*, pages 933–942, 2008.

[14] W. Shen, A. Doan, J. F. Naughton, and R. Ramakrishnan. Declarative information extraction using Datalog with embedded extraction predicates. In *Proceedings of VLDB'07*, pages 1033–1044, 2007.

[15] E. P. B. Simperl, C. Tempich, and Y. Sure. A cost estimation model for ontology engineering. In *Proceedings of ISWC'06*, pages 625–639, 2006.

[16] K. Spackman, K. Campbell, and R. Cote. Snomed RT: A reference terminology for health care. In *Proceedings of the 1997 AMIA Annual Fall Symposium*, page 640–644, 1997.

[17] F. M. Suchanek, M. Sozio, and G. Weikum. Sofie: A self-organizing framework for information extraction. Technical Report 5-004, Max Planck Institute, Saarbrücken, 2008.

[18] J. Völker. *Learning expressive ontologies*. PhD thesis, Universität Karlsruhe, 2009.

[19] T. Wächter, G. Fabian, and M. Schroeder. DOG4DAG: semi-automated ontology generation in OBO-Edit and Protégé. In *Proceedings of SWAT4LS'11*, pages 119–120, 2011.