Coupling grammar and knowledge base: Range Concatenation Grammars and Description Logics

Benoît Sagot[1,2] and Adil El Ghali[1]

[1] Lattice/PPS - Université Paris 7
2, place Jussieu, 75251 Paris Cedex 05, France
[2] Projet ATOLL - INRIA
Domaine de Voluceau, Rocquencourt, B.P. 105, 78153 Le Chesnay, France
{elghali,sagot}@linguist.jussieu.fr

Abstract. In this paper we introduce a novel framework to compute jointly syntactic parses and semantic representations of a written sentence. To achieve this goal, we couple a syntactico-semantic grammar and a knowledge base. The knowledge base is implemented in Description Logics, in a polynomial variant. The grammar is a Range Concatenation Grammar, which combines expressive power and polynomial parsing time, and allows external predicate calls. These external calls are sent to the knowledge base, which is able either to answer these calls or to learn new information, this process taking place during parsing. Thus, only semantically acceptable parses are built, avoiding the costly *a posteriori* semantic check of all syntactically correct parses.

1 Introduction and motivations

Natural Language Understanding [1] involves a number of tasks, among which we focus here on syntactic analysis and semantic interpretation and on the interface between two components realizing these tasks. On the one hand, an RCG parser [2] is in charge of the syntactic analysis. On the other hand, a Description Logic [3] component deals with some aspects of lexical interpretation. We address the problem of Logical Form computation in an incremental way during the parsing process, which lets us use Description Logics reasoning capabilities to guide and verify the results of parsing. The semantic representation we build is a Description Logics A-Boxs, where individuals correspond to semantically relevant ranges and predicates of an RCG analysis. A-Boxs are well known to be adapted to express logical forms, they have the same expressivity as conceptual graphs [4] and can express the content of DRSs [5]. The particularity of our work is to build the semantic representation in parallel with the syntactic analysis. In particular, this allows a very early check of the semantic and conceptual correctness of the input text in the understanding process. Semantically or conceptually incoherent texts are rejected very soon, avoiding their complete syntactic parsing.

Moreover, our approach make it possible to learn some lexical semantic information, when using the system on previously validated corpora, which could help the costly development of a wide-coverage knowledge base. When we can ensure the syntactic and semantic validity of the parsed texts, we can enrich our knowledge base with new concepts. This is done by adding new concepts that semantically validate an RCG analysis to the knowledge base and by adding axioms corresponding to the input sentence.

2 Parsing with Range Concatenation Grammars

Range Concatenation Grammars (RCGs) have been introduced by [6], and more thoroughly described for example in [2]. They define exactly the class of languages which are recognizable in deterministic polynomial time, PTIME. Therefore, it is strictly more powerful than, for example, Mildly Context-Sensitive grammars, while staying computationally tractable, since polynomial. For a formal definition of RCGs, see the above-mentioned papers. Informally, an RCG can be seen as a set of rewriting rules called *clauses* which apply to non-terminal symbols, called *predicates*, that have arguments representing ranges of the input string. Like CFGs, any rewriting sequence can be seen as a tree whose root is the object to be derived from. If such a tree is rooted at a pre-defined starting object (a given predicate named *axiom* with one argument) and is complete, it is a *parse tree*. Like CFGs, the set of all parse trees for a given sentence can be expressed in a polynomial sized structure called *parse forest*.

The formalism of RCGs has several interesting properties, including closure by intersection, union, complementation, concatenation, and Kleene iteration. These operations are achieved without the need to change the component grammars, thus giving to RCGs modularity with respect to these operations. In addition, RCGs are not linear, in the sense that a given range can be (a part of) several arguments of several predicates in the same clause. This allows the use of the same range of the input sentence more than once, e.g. in order to express different points of view on the input sentence that interact with each other.

Because of these properties, and with linguistic and operational arguments, [7] shows that RCGs are a good candidate for the design of efficient linguistic grammars which can both cover all known syntactic phenomena in a linguistically satisfying way and deal simultaneously with predicates representing facts about morphology, syntax and lexical semantics, thanks to non-linearity. The modularity of RCGs makes this possible without defining a huge and/or redundant grammar that would be used anyway only very partially to parse a given sentence. Indeed, it is possible to associate to each lexical entry a *pseudo-RCG* (an RCG without axiom) that describes all its properties. Furthermore, these pseudo-RCGs can be computed dynamically in an inheritance-driven ontology. Parsing a sentence is then a three step process : we first compute the pseudo-RCGs associated with all lexical entries that are used in the sentence, then we put them together with a general non-lexicalized RCG (or appropriate parts thereof), and finally we use the resulting RCG to parse the sentence (see [7]).

Using Boullier's RCG parser, one author implemented this process in a fully polynomial way with a limited French grammar and a toy lexicon, thus building an efficient purely RCG-driven semantico-syntactic parser.

Furthermore, the predicative structure of RCGs makes it possible, in the right hand side of a clause, to call external predicates that can be answered by an independent system. Indeed, the aim of this work is to show how this mechanism can be used to invoke a knowledge base described in an appropriate Description Logic (DL) to answer a certain class of predicates.

A simple RCG is given as an example later in this paper.

3 Description Logics

Description Logics (DL) [3] are a knowledge representation formalism following *semantic networks* and *frames*, and based on first order logic with tarskian semantics. In this framework, the theory is divided in two components, *(i)* the **T-Box** containing *intentional (terminological) knowledge*, namely domain concepts and relations, and *(ii)* the **A-Box** containing *extensional (assertional) knowledge* describing the domain individuals and the relationships among them.

Concepts may be seen as domain classes of individuals and *roles* as binary relations between concepts/individuals. The formalism offers a number of operators (generally called constructors) allowing the definition of complex concepts and roles using simpler ones. The description language we use is $\mathcal{ALCH}$ with role hierarchy and concrete domain. This DL is expressive enough for our needs in the first experiments. The syntax and semantic of this language are given bellow.

$$\mathsf{C,D} \rightarrow \mathsf{A} \mid \top \mid \bot \mid \neg\,\mathsf{C} \mid \mathsf{C} \vee \mathsf{D} \mid \mathsf{C} \wedge \mathsf{D} \mid \forall \mathsf{R}.\mathsf{C} \mid \exists \mathsf{R}.\mathsf{C}$$

$\top^{\mathcal{I}} = \Delta$	$(\mathsf{C} \wedge \mathsf{D})^{\mathcal{I}} = \mathrm{C}^{\mathcal{I}} \cap \mathrm{D}^{\mathcal{I}}$
$\bot^{\mathcal{I}} = \emptyset$	$(\mathsf{C} \vee \mathsf{D})^{\mathcal{I}} = \mathrm{C}^{\mathcal{I}} \cup \mathrm{D}^{\mathcal{I}}$
$(\mathsf{C})^{\mathcal{I}} = \mathsf{C}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	$(\forall \mathsf{R}.\mathsf{C})^{\mathcal{I}} = \{\ \mathrm{x} \mid \forall \mathrm{y}\ (\mathrm{x},\mathrm{y}) \in \mathrm{R}^{\mathcal{I}} \Rightarrow \mathrm{y} \in \mathrm{C}^{\mathcal{I}}\}$
$(\neg\,\mathsf{C})^{\mathcal{I}} = \Delta^{\mathcal{I}} - \mathrm{C}^{\mathcal{I}}$	$(\exists \mathsf{R}.\mathsf{C})^{\mathcal{I}} = \{\ \mathrm{x} \mid \exists \mathrm{y}\ (\mathrm{x},\mathrm{y}) \in \mathrm{R}^{\mathcal{I}} \Rightarrow \mathrm{y} \in \mathrm{C}^{\mathcal{I}}\}$

DL offer several reasoning services [8, 9]; those we use are defined as follows. Let Σ be a knowledge base, we refer to complex concepts by C, D, ..., to roles by $R, Q, ...$ and to individuals by $a, b, ...$

Subsumption ($\Sigma \vdash \mathrm{C} \sqsubseteq \mathrm{D}$) This service tests whether C is subsumed by D in the knowledge base Σ, *i.e.* whether $\mathrm{C}^{\mathcal{I}} \subseteq \mathrm{D}^{\mathcal{I}}$ in any model $\mathcal{I}$ of Σ. For content determination needs, subsumption allows to decide whether a concept is more specific than another. An *off-line* service associated with subsumption is the classification of the knowledge base concepts. Moreover, C and D are said to be equivalent ($\mathrm{C} \equiv \mathrm{D}$) if $\mathrm{C} \sqsubseteq \mathrm{D}$ and $\mathrm{D} \sqsubseteq \mathrm{C}$.

Consistency ($\Sigma \not\models$) This checks whether Σ is satisfiable, *i.e.* has a model.

Instance test ($\Sigma \vdash \mathrm{C}(a)$ resp. $\Sigma \vdash R(a, b)$) This service allows to check if the assertion $\mathrm{C}(a)$ (resp. $R(a, b)$) is satisfied in all models of Σ, *i.e.* if a belongs to the concept C (resp. $\langle$a,b$\rangle$ belongs to the relation R).

Unification This service introduced by [10] aims to find, given two concepts C and D, a substitution σ called *unifier* such that $\sigma(\mathrm{C}) \equiv \sigma(\mathrm{D})$. We use

this service during knowledge base enrichment to detect redundancy and to compute new concepts.

4 Lexical semantics in an RCG and in a DL knowledge base: an example

To exemplify the two previous parts, we now study the following (very simple) sentence both in a pure RCG point of view and with a DL formalization of its lexical semantics:

Mary loves the thriller that Peter wrote. (1)

4.1 The RCG and the parse forest

As explained in Part 2, parsing this sentence begins with the dynamic construction of its related RCG, by the concatenation of a general non-lexicalized RCG with pseudo-RCGs associated with each lexeme of the sentence. The result, very much simplified for readability and space reasons, is the following (`S` is the axiom, clauses are grouped according to their origin, the % symbol begins comments):

```
S(Subj Vrb Obj)          → SUBJ(Subj,Vrb) V(Vrb) VP(Vrb Obj,Vrb) .
SUBJ(PreS SHead PostS,V) → AGT(SHead,V) AGRT(SHead,V)
                           NP(PreS SHead PostS,SHead) .
VP(V Obj,V)              → OBJ(Obj,V) .
OBJ(PreO OHead PostO,V)  → PAT(OHead,V) NP(PreO OHead PostO,OHead) .
NP(PrNoun, PrNoun)       → PN(PrNoun) .
NP(Det Noun Rel,Noun)    → DET(Det,Noun) N(Noun) REL(Rel,Noun) .
DET(Det,Noun)            → ARTDEF(Det) COUNTABLE(Noun) AGRT(Det,Noun) .
REL("that" Subj V,Ante)  → SUBJ(Subj,V) V(V) PAT(Ante,V) .
```

```
PN("Mary")               → . %"Mary" is a proper noun
HUMAN("Mary")            → . %"Mary" is a human being
PERS3S("Mary")           → . %"Mary" is 3rd p. sing.
V("loves")               → . %"loves" is a verbal form
AGRT(Subj,"loves")       → PERS3S(Subj) . %"wrote" is 3rd p. sing.
AGT(Agt,"loves")         → HUMAN(Agt) . %"loves" has a human agent
PAT(Pat,"loves")         → . %"loves" can have any patient
ARTDEF("the")            → . % "the" is a definite article
AGRT("the",Noun)         → . % "the" can be singular or plural
N("thriller")            → . % "thriller" is a common noun
WRITING("thriller")      → . % a thriller is a writing (dynamically computed from
% the inheritance fact thriller < book and the fact that a book is a writing, see 2)
V("wrote")               → . %"wrote" is a verbal form
AGRT(Subj,"wrote")       → . %"wrote" is in agreement with any subject
AGT(Agt,"wrote")         → HUMAN(Agt) . %"wrote" has a human agent
PAT(Pat,"wrote")         → WRITING(Pat) . %"wrote"'s patient is a writing
PN("Peter")              → . %"Peter" is a proper noun
HUMAN("Peter")           → . %"Peter" is a human being
```

```
COUNTABLE(X)             → WRITING(X) . % writings are countable
```

Such a grammar can be usually read as follows: unary predicates represent linguistic properties and binary predicates represent (binary) linguistic relations (usually oriented from the first argument towards the second)[3].

Parsing (1) with this RCG gives a unique analysis, representable by a CFG whose non-terminals are named by instantiated RCG predicates. For example (for better readability, ranges have been identified by their corresponding substrings):

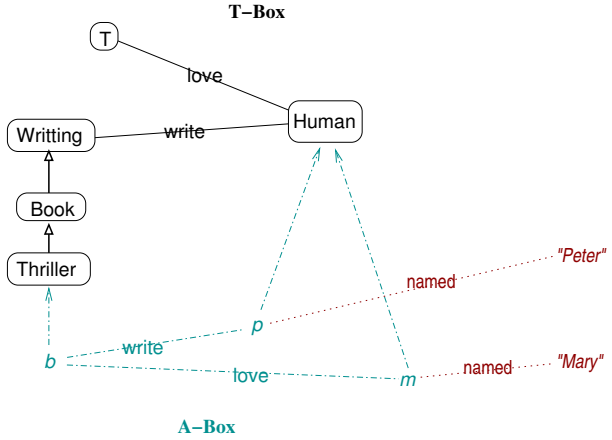
$$\texttt{S}(0..7_{\text{Mary ... wrote}}) \rightarrow \texttt{SUBJ}(0..1_{\text{Mary}}, 1..2_{\text{loves}})\ \texttt{V}(1..2_{\text{loves}})\ \texttt{VP}(1..7_{\text{loves the thriller that Peter wrote}}, 1..2_{\text{loves}})\ .$$

4.2 The DL representation

The task of the DL component is to produce a semantic representation of the input text and to complete or validate the analysis process. The representation is an A-Box in our knowledge base, with an individual for some parsed range. The missing information/verifications ensure that the constructed clauses are consistent w.r.t. the knowledge base.

For instance, giving the following T-Box: $\mathcal{T} = \{\ \mathsf{Writing} \equiv \exists \mathsf{writer}.\mathsf{Human}, \mathsf{Book} \subseteq \mathsf{Writing}, \mathsf{Thriller} \subseteq \mathsf{Book}, \mathsf{Humain} \subseteq \exists \mathsf{love}.\top\ \}$ we can verify that $\texttt{WRITING}(3..4_{\text{thriller}})$ is consistent. To do so we compute the subsumers of Thriller in the T-Box $\mathcal{T}$: Book and Writing. We also have in $\mathcal{T}$ the fact that the role writer links the concepts Writing and Human. A possible A-Box representing the sentence (1) can be: $\mathsf{Human}(p) \wedge \mathsf{named}(p, \text{“}Peter\text{”}) \wedge \mathsf{Thriller}(b) \wedge \mathsf{write}(p, b)$, $\mathsf{Human}(m) \wedge \mathsf{named}(m, \text{“}Mary\text{”}) \wedge \mathsf{love}(m, b)$

The figure bellow summarize the necessary knowledge needed to deal with our example.



[3] In fact, in our RCGs, predicates have usually more than two arguments. The linguistic interpretation of such predicates is a binary relation between two of the arguments, in a context defined by the other arguments.

5 Coupling an RCG with a DL knowledge base

As shown in the previous Part, some RCG predicates used in the pure-RCG approach correspond to lexical semantic facts that can be expressed in a DL knowledge base. These predicates (typically `HUMAN`, `WRITING`, and possibly `AGT` and `PAT` in the RCG given above) have to be unary or binary predicates, representing respectively properties or relations. For theoretical as well as practical reasons, it is more consistent to try and express these facts inside a DL knowledge base rather than in the RCG itself, and to couple the resulting grammar and knowledge base. This leads to a bidirectional interaction: the RCG uses the DL knowledge base through external predicates, and the knowledge base expands itself thanks to queries sent by the RCG about unknown lexemes.

5.1 Invoking the knowledge base during RCG parsing

The first way to couple a DL knowledge base representing lexical semantics facts and a linguistic RCG is to introduce external predicate calls in the right-hand-side of RCG clauses which are boolean queries to the knowledge base. Since inference services of the DL chosen are PSPACE in the worst case, the polynomial parse time achieved by the RCG is not deteriorated. This allows to replace predicates calls such as `HUMAN(Agt)` (in the above example) by calls to the knowledge base, to move from the grammar to the knowledge base clauses such as `HUMAN("Peter")` $\rightarrow$ `.`, and to define properties expressed by predicates such as `WRITING` in an appropriate way inside the knowledge base. As in the pure-RCG approach, a parse tree will be dropped as soon as a right-hand-side predicate returns $False$. It is a more efficient and more satisfying approach than a sequential approach which would first find all possible syntactic analyses for a sentence before filtering them with lexical semantics.

Furthermore, during the parsing and in parallel with the verifications we incrementally build an A-Box representing the input text. For each range in a query sent by the RCG, an A-Box assertion ($\mathsf{C}(a)$ or $\mathsf{r}(a,b)$) is created, each RCG unary predicate corresponding to a concept and each binary predicate to a role in the knowledge base. An instance check for the assertion and a consistency check of the knowledge base containing the T-Box and the so created A-Box is done. The textscdl-component returns to the RCG parser the result of these checks.

For instance, given Σ the knowledge base corresponding to the accomplished steps of parsing, let us consider the query `HUMAN(Agt)`. An individual a is created for the range `Agt`, we first check if a is an instance of Human, if it is `False` we give that answer to the RCG parser, otherwise we add the assertion $\mathsf{Human}(a)$ to the A-Box of the current sentence and check the consistency of the knowledge base $\Sigma \cup \{\mathsf{Human}(a)\}$ and give the answer to the RCG parser.

5.2 Enriching the knowledge base thanks to RCG parsing

In the second way to couple a DL knowledge base with an RCG, the information goes in the opposite direction: if the RCG sends to the DL-component a

query about an unknown lexeme, the knowledge base can create a new concept, partially defined by a positive answer to the query.

The learning configuration of the system is used with previously validated texts, so as to ensure their syntactic and semantic correctness. When parsing a sentence containing an unknown lexeme, the RCG parser works as usual and sends a query to the DL-component, which assigns a concept C to the individual a corresponding to the range of the unknown lexeme, and tries to construct T-Box axioms validating the instance test $\mathsf{C}(a)$.

For instance, in our example, let us suppose that the axiom

$$\mathsf{Thriller} \subseteq \mathsf{Book}$$

is not in the T-Box. In order to validate the semantic representation of the sentence we would need an axiom saying that a Thriller is a kind of Writing. So we can enrich our T-Box with the concept Thriller and an axiom

$$\mathsf{Thriller} \subseteq \mathsf{Writing}$$

However, the so computed concepts and axioms should have a special *"no totaly trustable"* status preventing using them to validate an analysis or to add new concepts/axioms to the knowledge base. They should be validated by a human operator or by statistical methods. After this validation, we try to find existing unifiers for the new concept in our knowledge base to avoid redundancy.

subsectionUsing the knowledge base to build the RCG associated with a sentence

As explained in the first part, parsing with dynamically generated RCGs is an efficient way to proceed. RCG clauses are then constructed by an inheritance process. The DL knowledge base can be used to store all necessary information and host this inheritance process. It is a way to represent inheritance mechanisms, for example inheritance of relations. Such an knowledge base represents in the same nodes both relational properties (lexical semantics) and ontological properties (in the case of nodes representing grammatical concepts), thus unifying a semantic lexicon, a basic semantic representation formalism, and something close to a metagrammar.

6 Conclusion and perspectives

Using description logics as a framework for natural language is not new, it used to be one of the main applications of DL since KL-ONE. Recently, a number of projects using DL for semantic interpretation were realized [3, Ch. 15]. In [11], Franconi makes a correspondence between a previously constructed syntactic analysis and a *logical form* à la Chomsky expressed in DL, this approach is widely used in similar projects.

However, the framework presented in this paper try to go further in at least two ways. First, it does not use a previously built structure such as a full syntactic analysis to build a semantic representation of the input sentence. Secondly,

and most importantly, it combines the syntactico-semantic parsing and the construction of the semantic representation, thus avoiding the costly building of several syntactically acceptable parses that would have to be checked for semantic consistency afterwards, and allowing deep interaction at the syntax-semantics interface which probably leads to better result and parsing efficiency.

An issue that we do not discuss in this paper is the expressivity of our semantic representation. We are exploring two direction to increase its expressive power. Firstly, by considering a more expressive DL, $\mathcal{SHIQ}(D_n)$, in which the inference services are in ExpTime in the worst case. But with the optimization in DL reasoners like FaCT and Racer, experiments suggest that it is feasable to use such a DL for our purpose. Secondly, we also want to investigate how a richer A-Box like a *Boolean A-Boxs* [12] can be used to improve our semantic representation to represent ambiguities for example.

References

1. Allen, J.: Natural language understanding. Benjamin-Cummings Publishing Co., Inc. (1988)
2. Boullier, P.: Counting with range concatenation grammars. Theoretical Computer Science **293** (2003) 391–416
3. Baader, F., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: Description Logics Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
4. Coupey, P., Faron, C.: Towards correspondence between conceptual graphs and description logics. In: Proceedings of the 6th International Conference on Conceptual Structures, Springer-Verlag (1998) 165–178
5. Ludwig, B., Gorz, G., Niemann, H.: Combining expression and content in domains for dialog managers (1998)
6. Boullier, P.: Range concatenation grammars. In: Proceedings of IWPT '00, Trento, Italy (2000) 53–64
7. Sagot, B., Boullier, P.: Les RCG comme formalisme grammatical pour la linguistique. In: Proceedings of TALN '04, Fez, Morocco (2004) 403–412
8. Donini, F., Lenzerini, M., Nardi, D., Schaerf, A.: Reasoning in description logics. In Brewka, G., ed.: Principles of Knowledge Representation and Reasoning. Studies in Logic, Language and Information. CLSI Publications (1996) 193–238
9. Horrocks, I.: Reasoning with expressive description logics: Theory and practice (2002)
10. Baader, F., Narendran, P.: Unification of concept terms in description logics. Journal of Symbolic Computation **31** (2001) 277–305
11. Franconi, E.: Logical form and knowledge representation: towards a reconciliation. In: Working Notes of the AAAI Fall Symposium on Knowledge Representation Systems based on Natural Language, Cambridge, US (1996) 20–24
12. Areces, C., Blackburn, P., Hernandez, B.M., Marx, M.: Handling boolean ABoxes. In: Proceedings of DL'03, Roma, Italy. (2003)