PARSING TECHNIQUES FOR LEXICALIZED CONTEXT-FREE GRAMMARS

Giorgio Satta
Dip. di Elettronica e Informatica
Università di Padova
via Gradenigo 6/A
35131 Padova, Italy
satta@dei.unipd.it

Abstract

State-of-the art parsers use lexicalized grammars to achieve high accuracy on real-world texts. Most of these systems are based on traditional parsing algorithms that were originally developed for the unlexicalized versions of the adopted grammar formalisms. We show that these parsing algorithms do not perform efficiently when processing lexicalized grammars. We then develop novel parsing algorithms that overcome the computational inefficiencies of the standard algorithms .

1 Introduction

In recent years, much of the parsing literature has focused on so-called lexicalized grammars, that is grammars in which each individual rule is specialized for one or more lexical items. Formalisms of this sort include dependency grammar [14], lexicalized tree-adjoining grammar [18], link grammar [21], head-driven phrase-structure grammar [15], tree insertion grammar [19], combinatorial categorial grammar [22] and bilexical grammar [8]. Probabilistic lexicalized grammars have also been exploited in state-of-the-art, real-world parsers, as reported in [12], [1], [7], [3], [5], and [10]. Other parsers or language models for speech recognition that do not directly exploit a generative grammar, still are heavily based on lexicalization, as for instance the systems presented in [11], [13], [16] and [4].

The wide diffusion of lexicalized grammars is mainly due to the capability of these formalisms to control syntactic acceptability, when this is sensitive to individual words in the language, and word selection, accounting for genuinely lexical factors as well as semantic and world knowledge conditions. More precisely, lexicalized grammars can select the complements and modifiers that a constituent can take, on the basis of special words playing a particularly informative role within the given constituent and the given complements and modifiers. These special words are typically identified with the lexical head and co-heads of a constituent, where with the term co-head of a constituent A we denote a lexical head of any of the subconstituents of A. To give a simple example (from [9]), the word *convene* requires an NP object to form a VP, but some NPs are more lexically or semantically appropriate than others, and the appropriateness depends largely on the NP's head, e.g., the word *meeting* vs. the word *party*. In this way, the grammar is able to make stipulations on the acceptability of input sentences like *Nora convened the meeting* and *Nora convened the party*. This was not satisfactorily captured by earlier formalisms that did not make use of lexicalization mechanisms. See [6] for further discussion.

Within the parsing community, and in the speech community as well, a considerable research effort has been devoted to the important problem of defining statistical parameters associated with lexicalized grammars, and to the problem of the specification of algorithms for the statistical estimation of these parameters. In contrast, not much has been done with respect to the sentence processing problem. Most of the above mentioned systems parse input strings using naïve adaptations of existing algorithms developed for the unlexicalized version of the adopted formalism, possibly in combination with heuristics specially tailored to cut down the parsing search space. However, the internal structure and the large size of formalisms derived by means of some lexicalization mechanism render these systems unsuitable to be processed with traditional parsing techniques. In the talk we address these problems and present novel parsing algorithms for lexicalized grammars that overcome the computational inefficiencies that arise when standard algorithms are used. We will focus on lexicalized context-free grammars (LCFGs), a class of grammars originally introduced in [9]. LCFGs are a convenient abstraction that allows us to study important computational and generative properties that are common to several of the above mentioned lexicalized grammars. In what follows, we report a formal definition of LCFGs and give a brief outline of the results that will be more carefully presented in the talk. Other abstractions of lexicalized grammars have been presented in [2] and in [10]. In those works, however, the major emphasis is on the study of linguistic expressiveness and of parameter estimation problems, respectively, for different lexicalized grammar formalisms.

2 Lexicalized context-free grammars

We can think of a lexicalized context-free grammar as a particular kind of CFG obtained by applying a lexicalization procedure to some underlying CFG.[1] Before presenting the formal definition of LCFG, we briefly discuss an example. Consider the sample phrase *dumped sacks into a bin*. In order to be able to capture the lexical and syntactic relations holding among the words in this phrase, we pair each of the standard nonterminals NP, VP, etc., with a tuple of words from the phrase itself. The nonterminals of the resulting grammar are therefore of the following kind: V[dump], NP[sack], VP[dump][sack], etc. Using these new lexicalized nonterminals we can write new context-free productions of the form VP[dump][sack] $\rightarrow$ V[dump] NP[sack]. The main idea here is that lexical items appearing in the right-hand side nonterminals can be inherited by the left-hand side nonterminal. A possible derivation of the above phrase is depicted in Figure 1.

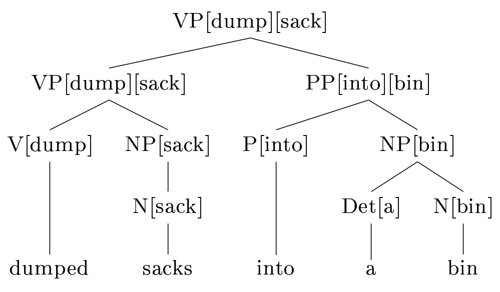

Figure 1: A sample derivation in an LCFG.

[1]The particular way we lexicalize CFGs differs from the proposal in [19], where a CFG is transformed into a tree substitution grammar. Also, we note here in passing that an LCFG could be alternatively defined as a very restricted kind of attribute grammar with only synthesized attributes [17].

We now give a formal definition of LCFG. A **lexicalized context-free grammar** is a CFG $G = (V_{\rm N}, V_{\rm T}, P, S[\$])$, with $V_{\rm N}$ and $V_{\rm T}$ the set of nonterminal and terminal symbols, respectively, and with $S[\$] \in V_{\rm N}$ a special start symbol. The following conditions are all satisfied by G:

(i) there exists a set $V_{\rm D}$, called the set of **delexicalized nonterminals**, such that

$$V_{\rm N} \subseteq \{A[a_1][a_2]\cdots[a_r] \mid A \in V_{\rm D},\ r \geq 1,\ a_j \in V_{\rm T},\ 1 \leq j \leq r\};$$

(ii) every production in P has one of the following two forms:

(a) $A_0[a_{0,1}]\cdots[a_{0,r_0}] \rightarrow A_1[a_{1,1}]\cdots[a_{1,r_1}]\ A_2[a_{2,1}]\cdots[a_{2,r_2}]\ \cdots\ A_q[a_{q,1}]\cdots[a_{q,r_q}]$,
where $q \geq 1$ and multiset $\{a_{0,1},\ldots,a_{0,r_0}\}$ is included in multiset $\{a_{1,1},\ldots,a_{1,r_1},a_{2,1},\ldots,a_{q,r_q}\}$;

(b) $A[a] \rightarrow a$.

The multiset condition in (ii)a states that the lexical items in the left-hand side nonterminal are always inherited from the nonterminals in the right-hand side. Note also that the start symbol $S[\$]$ is a lexicalized nonterminal. As a convention, we assume that \$ is a dummy lexical item that does not appear anywhere else in G, and disregard it in the definition of $L(G)$.

In current practice, the set of delexicalized nonterminals $V_{\rm D}$ is kept separate from set $V_{\rm T}$. Set $V_{\rm D}$ usually encodes word sense information and other features as number, tense, etc., structural information as bar-level and subcategorization requirements, and any additional information that does not explicitly refer to individual lexical items, as for instance contextual constraints for parent node category [3], or constraints on the constituent's yield, expressed through finite information about distribution of some lexical category [5].

Let G be some LCFG and let p be some production in P. If p has the form in (ii)a above, let $k_p = \sum_{j=1}^{q} r_j$; otherwise, let $k_p = 1$. The **degree of lexicalization** of an LCFG G is defined as $k_G = \max_{p \in P} k_p$. We also say that G is a k_G-lexical CFG. Note that from condition (ii)a it directly follows that G has productions with right-hand side length bounded by k_G. When the set of delexicalized nonterminals is fixed, the degree of lexicalization induces a proper hierarchy on the class LCFG. More precisely, it can be shown that, for any non-empty set $V_{\rm D}$ and any $k \geq 3$, there exists a k-lexical CFG G defined on $V_{\rm D}$ such that $L(G)$ cannot be generated by any k'-lexical CFG defined on $V_{\rm D}$ and with $k' < k$.

The class of 2-lexical CFG, also called bilexical CFG, turns out to be of major importance in current parsing practice. In fact, the probabilistic formalisms adopted in [1], [7], [3] and [5] are strongly equivalent to bilexical grammars, in the following sense. For each of the above formalisms, we can effectively construct a corresponding probabilistic bilexical grammar with the following properties. There is a one-to-one mapping between derivations in the source formalism and derivations in the target bilexical grammar. This map preserves the associated probabilities and can be computed through a homomorphism (homomorphisms can be implemented as real-time, memoryless processes). Most important here, when computational problems related to parsing must be investigated, bilexical grammars are a useful abstraction of the above mentioned formalisms, and parsing algorithms developed for bilexical grammars can directly be adapted to these formalisms. In Section 3, we will mainly focus on bilexical grammars.[2]

[2]The adoption of bilexical formalisms in state-of-the-art, real-world parsers is related to the fact that currently available natural language annotated corpora are still limited in their size, so that the estimation of probabilities for lexical relations of arity greater than two is still quite problematic.

3 LCFG Parsing

The cost of the expressiveness of an LCFG is a very large production set. In the simplest case of bilexical CFGs, the size of set P usually grows with the square of the size of V_{T}, and thus can be very large. Standard context-free parsing algorithms, which run in time linear with the size of the input grammar, are inefficient in such cases. Therefore, a first goal in the design of a parsing algorithm for LCFGs is the one of achieving running time sublinear with respect to the grammar size. As a first result, we show that algorithms satisfying the so-called correct-prefix property [20] are unlikely to achieve this goal, even in case the grammar is precompiled in an amount of time polynomial in its size.

In order to achieve the sublinear time goal, a common practice is to use standard CFG parsing algorithms and to select only those rules in the grammar that are lexically grounded in the input sentence. However, this adds a factor of n^2 for an input string of length n, resulting in $O(n^5)$ running time for bilexical CFGs and related formalisms. We show how dynamic programming can be exploited to achieve an $O(n^4)$ result. Also, we specify an $O(n^3)$ time algorithm for a restricted kind of bilexical CFGs. We argue that the proposed restriction leaves enough power to the formalism to capture most common natural language constructions. We discuss how these results generalize to LCFG with higher degree of lexicalization and to lexicalized tree-adjoining grammars [18] as well.

The above algorithms exploit bottom-up strategies, which are the most common in parsing lexicalized grammars. We show how top-down strategies can be applied in parsing bilexical CFGs, still retaining the desired condition on sublinear running time with respect to the input grammar size. This is done by exploiting generalized look-ahead techniques. We argue that the resulting strategies are even more selective than standard top-down strategies satisfying the correct-prefix property.

Acknowledgements

This abstract presents ongoing research in collaboration with Jason Eisner, University of Rochester, and Mark-Jan Nederhof, DFKI. The author is supported by MURST under project *PRIN: BioInformatica e Ricerca Genomica* and by University of Padua, under project *Sviluppo di Sistemi ad Addestramento Automatico per l'Analisi del Linguaggio Naturale*.

References

[1] H. Alshawi. Head automata and bilingual tiling: Translation with minimal representations. In *Proc. of the 34th ACL*, pages 167–176, Santa Cruz, CA, 1996.

[2] J. Carroll and D. Weir. Encoding frequency information in lexicalized grammars. In *Proceedings of the 5th Int. Workshop on Parsing Technologies*, MIT, Cambridge, MA, September 1997.

[3] E. Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proc. of AAAI-97*, Menlo Park, CA, 1997.

[4] C. Chelba and F. Jelinek. Exploiting syntactic structure for language modeling. In *Proc. of the 36th ACL*, Montreal, Canada, 1998.

[5] M. Collins. Three generative, lexicalised models for statistical parsing. In *Proc. of the 35th ACL*, Madrid, Spain, 1997.

[6] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing.* PhD thesis, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 1999.

[7] J. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proc. of the 16th COLING*, pages 340–345, Copenhagen, Denmark, 1990.

[8] J. Eisner. Bilexical grammars and a cubic-time probabilistic parser. In *Proceedings of the 5th Int. Workshop on Parsing Technologies*, MIT, Cambridge, MA, September 1997.

[9] J. Eisner and G. Satta. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proc. of the 37th ACL*, pages 457–464, College Park, Maryland, 1999.

[10] J. Goodman. Probabilistic feature grammars. In *Proceedings of the 5th Int. Workshop on Parsing Technologies*, MIT, Cambridge, MA, September 1997.

[11] F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, A. Ratnaparkhi, and S. Roukos. Decision tree parsing using a hidden derivation model. In *Proceedings of the 1994 Human Language Technology Workshop*, pages 272–277, 1994.

[12] J. Lafferty, D. Sleator, and D. Temperley. Grammatical trigrams: A probabilistic model of link grammar. In *Proc. of the AAAI Conf. on Probabilistic Approaches to Nat. Lang.*, October 1992.

[13] D. Magerman. Statistical decision-tree models for parsing. In *Proc. of the 33rd ACL*, pages 276–283, Cambridge, MA, 1995.

[14] I. Mel'čuk. *Dependency Syntax: Theory and Practice.* State University of New York Press., 1988.

[15] C. Pollard and I. Sag. *Head-Driven Phrase Structure Grammar.* University of Chicago Press., 1994.

[16] A. Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *Second Conference on Empirical Methods in Natural Language Processing*, Brown University, Providence, Rhode Island, 1997.

[17] G. Rozenberg and A. Salomaa. *Handbook of Formal Languages*, volume 1. Springer-Verlag, Berlin, Germany, 1997.

[18] Y. Schabes, A. Abeillé, and A. K. Joshi. Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars. In *Proc. of the 12th COLING*, pages 578–583, Budapest, Hungary, 1988.

[19] Y. Schabes and R. C. Waters. Tree insertion grammar: A cubic-time parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21(4):479–515, 1995.

[20] S. Sippu and E. Soisalon-Soininen. *Parsing Theory: $LR(k)$ and $LL(k)$ Parsing*, volume 2. Springer-Verlag, Berlin, Germany, 1990.

[21] D. Sleator and D. Temperley. Parsing english with a link grammar. In *Proceedings of the 3rd Int. Workshop on Parsing Technologies*, Tilburg, Durbuy, Germany, August 1993.

[22] M. Steedman. *Surface Structure and Interpretation.* The MIT Press, Cambridge, MA, 1996.