Lexicalized Tree Automata-based Grammars for Translating Conversational Texts

Kiyoshi YAMABANA  Shinichi ANDO  Kiyomi MIMURA
Computer & Communication Media Research, NEC Corporation
4-1-1, Miyazaki, Miyamae-ku, Kawasaki 216-8555, JAPAN
k-yamabana@ct.jp.nec.com  s-ando@cw.jp.nec.com  k-mimura@da.jp.nec.com

Abstract

We propose a new lexicalized grammar formalism called Lexicalized Tree Automata-based Grammar, which lexicalizes tree acceptors instead of trees themselves. We discuss the properties of the grammar and present a chart parsing algorithm. We have implemented a translation module for conversational texts using this formalism, and applied it to an experimental automatic interpretation system (speech translation system).

1 Introduction

Achieving both broad coverage for general texts and better quality for texts from a restricted domain has been an important issue in practical natural language processing. Conversational language is a typical domain this problem has been notable, since they often include idioms, colloquial expressions and/or extra-grammatical expressions while a majority of utterances still obey a standard grammar.

Furuse and Iida (1994) proposed an approach to spoken-language translation based on pattern matching on the surface form, combined with an example-based disambiguation method. Since the grammar rules are simple patterns containing surface expressions or constituent boundaries, they are easy to write, and domain-specific knowledge can be easily accumulated in the grammar. On the other hand, relationships between two trees are not easy to describe, especially when they are separated apart on a larger tree. This might become an obstacle in expanding a domain-specific grammar into a general grammar with a wide coverage.

Brown (1996) approached to this problem employing a multi-engine architecture, where outputs from Transfer Machine Translation (MT), Knowledge-based MT and Example-based MT are combined on the chart during parsing. Ruland et al. (1998) employs a multi-parser multi-strategy architecture for robust parsing of the spoken language, where the results from different engines are combined on the chart using probability-based scores. A difficult part with these hybrid architectures is that it is not easy to properly compare and combine the results from different engines designed on different principles. In addition, these methods will require much computational power, since multiple parsers have to be run simultaneously.

A third approach, such as Takeda (1996), is grammar-based. In this approach, a method is provided to associate a grammar rule to a word or a set of words in order to encode their idiosyncratic syntactic behaviour. An associated grammar rule can be seen as a kind of example if it is described mostly by the surface level information. As is apparent from this description, this approach is an application of strong lexicalization of a grammar (Schabes, Abeillé and Joshi, 1988).

This approach allows coexistence of general rules and surface-level patterns in a uniform framework. Combination of both types of rules is naturally defined. These advantages are a good reason to employ strongly lexicalized grammars as the basic grammar formalism. However, we feel there are some points to be improved in the current strongly lexicalized grammar formalisms.

The first point is the existence of globally defined special tree operation, which requires a special parsing algorithm. In a strongly lexicalized grammar formalism, each word is associated with a finite set of trees anchored by that word. The tree operations usually include substitution of a leaf node by another tree, corresponding to expansion of a nonterminal symbol by a rewriting rule in CFG. However, if the tree operation is limited to substitution, the resulting grammar, namely Lexicalized Tree Substitution Grammar (LTSG), cannot even reproduce the trees obtained from non-lexicalized context free grammars. This will be obvious from the fact that for any LTSG, there is a
constant such that, in any trees built by the grammar, the distance of the root node and the nearest lexical item is less than that constant, while this property does not always hold for CFG. Tree Insertion Grammar (TIG), introduced by Schabes et al. (1995), had to be equipped with the insertion operation in addition to substitution, so that it can be strongly equivalent to an arbitrary CFG. The insertion operation is a restricted form of the adjoining operation in the Lexicalized Tree Adjoining Grammar (LTAG) (Joshi and Schabes, 1992).

Thus, a special tree operation other than substitution is inevitably to strongly lexicalized grammars. It is needed to grow an infinite number of trees from a finitely ambiguous set of initial trees representing the extended domain of locality (EDOL) of the word.

However, such special tree operation requires a specially devised parsing algorithm. In addition, the algorithm will be operation-specific and we have to devise a new algorithm if we want to add or modify the operation at all. Our first motivation was to eliminate the need for globally defined special tree operations other than substitution whenever possible, without losing the existence of EDOL.

Another point is the fact that lexicalization is applied only to trees, not to the tree operations. For example, in LTAG, initial tree sets anchored to a word is not enough to describe the whole set of trees anchored by that word, since initial trees are grown by adjunction of auxiliary trees. Since an auxiliary tree is in the EDOL of another word, the former word has limited direct control over which auxiliary tree can be adjoined to certain node. For detailed control, the grammar writer has to give additional adjoining restrictions to the node, and/or detailed attribute-values to the nodes that can control adjunction through node operations such as unification.

In short, we would like to define a lexicalized grammar such that 1) tree operation is substitution only, 2) it has extended domain of locality, and 3) tree operations as well as trees are lexicalized whenever possible. In the next section, we propose a grammar formalism that has these properties.

2 Lexicalized Tree Automata-based Grammars

In this section we introduce Lexicalized Tree Automata-based Grammar (LTA-based Grammar) and present its parsing algorithm.

First, we define some basic terminologies. A grammar is strongly lexicalized if it consists of 1) a finite set of structures each associated with a lexical item; each lexical item will be called the anchor of the corresponding structure, and 2) an operation or operations for composing the structures (Schabes, Abeillé and Joshi, 1988).

In the following, the word “tree automaton” (TA) will be used as a generic term for an automaton that accepts trees as input. It can be a finite tree automaton, a pushdown tree automaton, or any tree-accepting automaton having a state set, state transitions, initial and final states, and optional memories associated with states. Although our argument below does not necessarily require understanding of these general TAs, definitions and properties of finite and pushdown TAs can be found in Gécseg and Steinby (1997) for example.

2.1 Definition of LTA-based Grammars

The basic idea of an LTA-based grammar is to associate a tree automaton to each word that defines the set of local trees anchored to the word, instead of associating the trees themselves. The lexicalized tree automaton (LTA) provides a finite representation of a possibly non-finite set of local trees. This differs from other lexicalized grammars as LTAG, where non-finiteness of local trees is introduced through a global tree operation such as adjunction of auxiliary trees.

We define a lexicalized tree automata-based grammar as follows. Let $\Sigma$ be a set of terminal symbols (words), and $NT$ be the set of nonterminal symbols disjoint from $\Sigma$. Let $T_w$ be a set of trees (elementary trees) associated with a word $w$ in $\Sigma$. A tree in $T_w$ has nodes either from $\Sigma$ or from $NT$, and its root and one of its leaves are marked by a distinguished symbol self in $NT$. Let $A_w$ be the tree automaton lexicalized to the word $w$, which accepts a subset of trees obtained by repeatedly joining two trees in $T_w$ at the special nodes labelled self, one at the root of a tree and the other at a foot of another tree. From this definition, $A_w$ can be identified with a string automaton; its alphabets are the trees in $T_w$, and a string of the elementary trees are identified with the tree obtained by joining the elementary trees in a bottom-up manner. $S_w$ is a set of nonterminal symbols associated with the word $w$. They are assigned to the root of a tree when the tree is accepted by $A_w$.  
