A Predictive Left-Corner Parser for Tree Adjoining Grammars

Vicente Carrillo and Víctor J. Díaz
Department of Computer Languages and Systems, University of Seville
Avda. Reina Mercedes s/n, Seville, 41012 Spain
{carrillo,vjdiaz}@lsi.us.es

Miguel A. Alonso
Department of Computation, University of Coruña
Campus de Elviña, La Coruña, 15071 Spain
alonso@dc.fi.udc.es

Abstract

Tree Adjoining Grammar (TAG) is a formalism that has become very popular for the description of natural languages. However, the parsers for TAG that have been defined on the basis of the Earley's algorithm entail important computational costs. In this article, we propose to extend the *left corner relation* from *Context Free Grammar* (CFG) to TAG in order to define an efficient left corner parser for TAG that improves the performance of the Earley-like parsers guaranteeing the valid prefix property, due to a remarkable reduction in the number of predictions operations with respect to Earley-like parsing algorithms.

Keywords:parsing, left corner, tree adjoining grammar

1 Introduction

Tree Adjoining Grammar (TAG) [6] is a naturally lexicalized formalism adequate to describe the syntax of natural languages. As a counterpart, the parsing process for this mildly context-sensitive formalism entails larger computational costs than the same proccess applied to Context Free Grammar (CFG): the worst case time complexity of TAG parsers is $\mathcal{O}(n^6)$, in contrast with $\mathcal{O}(n^3)$ complexity of CFG parsers. In recent years, there have been described in the literature several approaches that try to improve the perfomance of TAG parsers, most of them based on restrictions in the formalism [10] or compilation of elementary trees into finite-state automata [5].

Earley's [4] and *Left corner* (LC) [9] parsers are probably the most popular parsing algorithms for CFG. Both of them proceeds through the sentence from left to right, but they differs in the way top-down predictions are used to guide the bottom-up recognition. A LC parser reduces the number of predictions applied by an Earley's parser by using a *left corner relation* between grammar symbols. Several parsers for TAG have been defined on the basis of the Earley's algorithm [7, 6, 1, 8] but, to the best of our knowledge, only one parser [3] that does not guarantee the valid prefix property (VPP) has been defined to improve the practical performance of Earley-like parsers for TAG by using a *left corner relation*. In this work, we present a left corner parser for TAGs that satisfies the VPP and remains the time complexity of the algorithm presented in [3].

The article may be outlined as follows. In section 2 we recall the definition of TAG and we introduce the notation used in the rest of the article. In section 3 an Earley-like parsing algorithm for TAG is described.

This algorithm is used as a base to define a predictive left-corner parser in section 4. Section 5 presents final conclusions.

2 Notation

2.1 Tree Adjoining Grammars

A TAG is a five-tuple $(V_N, V_T, S, \boldsymbol{I}, \boldsymbol{A})$, where V_N is a set of nonterminal symbols, V_T is a set of terminal symbols, $S \in V_N$ is the axiom, $\boldsymbol{I}$ is a finite set of finite *initial trees* and $\boldsymbol{A}$ is a finite set of finite *auxiliary trees*. The set $\boldsymbol{I} \cup \boldsymbol{A}$ is referred to as the set of *elementary trees*. Internal nodes in an elementary tree are labeled by nonterminal symbols. We refer to the root of an elementary tree γ as $\mathbf{R}^\gamma$. In each elementary tree the nodes on the frontier are labeled by terminal symbols or the empty string (ε), except that exactly one node in each auxiliary tree which is marked as the foot and whose label is the same as the root. We refer to the foot of an auxiliary tree β as $\mathbf{F}^\beta$. The path from the root to the foot is called the *spine*. We use $label(M^\gamma)$ to denote the label of node M^γ.

The *adjunction* operation inserts an auxiliary tree β into another tree γ on a node M^γ that has the same label as $\mathbf{R}^\beta$. As a result, M^γ is replaced by β and $\mathbf{F}^\beta$ is replaced by the subtree rooted at M^γ. We use $\beta \in adj(M^\gamma)$ to denote that a tree $\beta \in \boldsymbol{A}$ may be adjoined on node M^γ, i.e. M^γ is an adjunction node. If adjunction is not mandatory on M^γ then $\mathbf{nil} \in adj(M^\gamma)$, where **nil** is a dummy symbol.

In order to represent partial parse trees, we define a production $N^\gamma \to N_1^\gamma ... N_g^\gamma$ for every node N^γ and its ordered g children $N_1^\gamma ... N_g^\gamma$ in an elementary tree. We refer to the set of productions related to an elementary tree γ as $\mathcal{P}(\gamma)$.

For technical reasons, we consider additional productions $\top \to \mathbf{R}^\alpha$, $\top \to \mathbf{R}^\beta$ and $\mathbf{F}^\beta \to \bot$ for every initial tree α and auxiliary tree β. To preserve the generative capability of the grammar, the nodes $\top$ and $\bot$ can not be adjunction nodes.

Given two pairs (p, q) and (p', q') of integers, $(p, q) \leq (p', q')$ is satisfied if $p \leq p'$ and $q \leq q'$. In order to simplify the description of parser, we introduce the operation $\cup$ that is defined by: given two integers p and q, we define $p \cup q$ as p if q is undefined and as q if p is undefined, being undefined in other case.

2.2 Parsing schemata

Parsing algorithms can be defined as deduction systems [11],[12] where formulas, called items, are sets of complete or incomplete constituents. Parsing schemata were introduced in [12] as a framework for high-level description of parsing algorithms. A parsing schema abstracts from implementation details of an algorithm, such as data and control structures.

Formally, a *parsing system* for a grammar G and string $a_1 \dots a_n$ is a triple $\langle \mathcal{I}, \mathcal{H}, \mathcal{D} \rangle$, with $\mathcal{I}$ a set of *items* which represent intermediate parse results, $\mathcal{H}$ an initial set of items called *hypothesis* that encodes the sentence to be parsed, and $\mathcal{D}$ a set of *deduction steps* that allow new items to be derived from already known items. Deduction steps are of the form $\frac{\eta_1,\dots,\eta_k}{\xi}\, cond$, meaning that if all antecedents η_i of a deduction step are present and the conditions *cond* are satisfied, then the consequent ξ should be generated by the parser. A set $\mathcal{F} \subseteq \mathcal{I}$ of *final items* represent the recognition of a sentence. A *parsing schema* is a parsing system parameterized by a grammar and a sentence.

The set of items in a parsing system $\mathbb{P}_{\mathrm{Alg}}$ corresponding to the parsing schema **Alg** describing a given parsing algorithm *Alg* is denoted $\mathcal{I}_{\mathrm{Alg}}$, the set of hypotheses $\mathcal{H}_{\mathrm{Alg}}$, the set of final items $\mathcal{F}_{\mathrm{Alg}}$ and the set of deduction steps is denoted $\mathcal{D}_{\mathrm{Alg}}$.

The parsing schemata framework allows us to establish relations between two parsers in a formal way. A parsing schema can be generalized from another one by means of *item refinement*, breaking single items into multiple items, *step refinement*, decomposing a single deduction step in a sequence of steps, and by considering a larger class of grammars (*extension*).

In order to decrease the number of items and deduction steps in a parsing schema, we can apply the following kinds of filtering: *static filtering*, in which redundant parts are simply discarded, *dynamic filtering*, using context information to determine the validity of items, and *step contraction*, in which a sequence of deduction steps is replaced by a single one. In particular, *filters* are very interesting relations because they can be used to improve the performance of parsers in practical cases. An example of filter is the relation between Earley and Left Corner (LC) parsers for CFG [12].

3 An Earley-like parser for TAG

In this section we describe an Earley-like parsing algorithm for TAG preserving the valid prefix property. It is essentialy the same algorithm defined in [8, 1]. Parsers satisfying the VPP guarantee that, as they read the input string from left to right, the substrings read so far are valid prefixes of the language defined by the grammar. More formally, a parser satisfies the VPP if for any substring $a_1 \ldots a_k$ read from the input string $a_1 \ldots a_k a_{k+1} \ldots a_n$ guarantees that there is a string of tokens $b_1 \ldots b_m$, where b_i need not be part of the input string, such that $a_1 \ldots a_k b_1 \ldots b_m$ is a valid string of the language.

To obtain an Earley-like parsing algorithm for TAG preserving the VPP we need to define items of the form

$$\mathcal{I}^1_{\text{Earley}} = \left\{ \; [N^\gamma \rightarrow \delta \bullet \nu, h, i, j \mid p, q] \; \right\}$$

such that $N^\gamma \rightarrow \delta \bullet \nu \in \mathcal{P}(\gamma),\ \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq h \leq i \leq j,\ (p, q) \leq (i, j)$. It must be satisfied that $\mathbf{R}^\gamma$ spans the part $a_{h+1} \ldots a_i \delta\nu\upsilon$ of the input string, with δ spanning $a_i \ldots a_p\ \mathbf{F}^\gamma\ a_{q+1} \ldots a_j \overset{*}{\Rightarrow} a_i \ldots a_j$ if and only if $(p, q) \neq (-, -)$, and spanning $a_i \ldots a_j$ if and only if $(p, q) = (-, -)$. We also need to define a new kind of intermediate pseudo-items

$$\mathcal{I}^2_{\text{Earley}} = \left\{ \; [N^\gamma \rightarrow \delta \bullet, i, j \mid p, q] \; \right\}$$

such that $N^\gamma \rightarrow \delta \bullet \in \mathcal{P}(\gamma),\ \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq i \leq j,\ (p, q) \leq (i, j)$. It must be satisfied that δ spans $a_i \ldots a_p\ \mathbf{F}^\gamma\ a_{q+1} \ldots a_j \overset{*}{\Rightarrow} a_i \ldots a_j$ if and only if $(p, q) \neq (-, -)$, spanning $a_i \ldots a_j$ if and only if $(p, q) = (-, -)$.

The hypotheses defined for this parsing schema are the standard ones and therefore they will be omitted in the next parsing schemata described in this article:

$$\mathcal{H} = \left\{ \; [a, i-1, i] \mid a = a_i, 1 \leq i \leq n \; \right\}$$

The following is a list of the different types of deduction steps used in this parsing strategy:

$$\mathcal{D}^{\text{Init}}_{\text{Earley}} = \frac{}{[\top \rightarrow \bullet \mathbf{R}^\alpha, 0, 0, 0 \mid -, -]}$$

with $\alpha \in \boldsymbol{I}$.

$$\mathcal{D}^{\text{Scan}}_{\text{Earley}} = \frac{\begin{array}{l}[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, j-1 \mid p, q],\\ [a, j-1, j]\end{array}}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, h, i, j \mid p, q]}$$

with $\text{label}(M^\gamma) = a$.

$$\mathcal{D}^{\varepsilon}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, j \mid p, q],}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, h, i, j \mid p, q]}$$

with $\text{label}(M^\gamma) = \varepsilon$.

$$\mathcal{D}^{\text{Pred}}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, j \mid p, q]}{[M^\gamma \rightarrow \bullet \upsilon, h, j, j \mid -, -]}$$

with $\mathbf{nil} \in \text{adj}(M^\gamma)$.

$$\mathcal{D}^{\text{Comp}}_{\text{Earley}} = \frac{\begin{array}{l}[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, k \mid p, q],\\ [M^\gamma \rightarrow \upsilon \bullet, h, k, j \mid p', q']\end{array}}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, h, i, j \mid p \cup p', q \cup q']}$$

with $\mathbf{nil} \in \text{adj}(M^\gamma)$.

$$\mathcal{D}^{\text{AdjPred}}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, j \mid p, q]}{[\top \rightarrow \bullet \mathbf{R}^\beta, j, j, j \mid -, -]}$$

with $\beta \in \text{adj}(M^\gamma)$.

$$\mathcal{D}^{\text{FootPred}}_{\text{Earley}} = \frac{\begin{array}{l}[\mathbf{F}^\beta \rightarrow \bullet \bot, j, k, k \mid -, -],\\ [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, j \mid p, q]\end{array}}{[M^\gamma \rightarrow \bullet \delta, h, k, k \mid -, -]}$$

with $\beta \in \text{adj}(M^\gamma)$.

$$\mathcal{D}^{\text{FootComp}}_{\text{Earley}} = \frac{\begin{array}{l}[M^\gamma \rightarrow \delta \bullet, h, k, l \mid p, q],\\ [\mathbf{F}^\beta \rightarrow \bullet \bot, j, k, k \mid -, -],\\ [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, h, i, j \mid p', q']\end{array}}{[\mathbf{F}^\beta \rightarrow \bot \bullet, j, k, l \mid k, l]}$$

with $\beta \in \text{adj}(M^\gamma)$, $p \cup p'$ is defined, $q \cup q'$ is defined.

$$\mathcal{D}^{\text{AdjComp}^0}_{\text{Earley}} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\beta \bullet, j, j, m \mid k, l],\\ [M^\gamma \to \delta \bullet, h, k, l \mid p, q],\end{array}}{[[M^\gamma \to \delta \bullet, j, m \mid p, q]]}$$

with $\beta \in \text{adj}(M^\gamma)$.

$$\mathcal{D}^{\text{AdjComp}^1}_{\text{Earley}} = \frac{\begin{array}{l}[[M^\gamma \to \delta \bullet, j, m \mid p, q]],\\ [M^\gamma \to \delta \bullet, h, k, l \mid p, q],\\ [N^\gamma \to \delta \bullet M^\gamma \nu, h, i, j \mid p', q']\end{array}}{[N^\gamma \to \delta M^\gamma \bullet \nu, h, i, m \mid p \cup p', q \cup q']}$$

with $\beta \in \text{adj}(M^\gamma)$.

Parsing begins by creating the item corresponding to a production having the root of an initial tree as left-hand side and the dot in the leftmost position of the right-hand side. Then, a set of deductive steps $\mathcal{D}^{\text{Pred}}_{\text{Earley}}$ and $\mathcal{D}^{\text{Comp}}_{\text{Earley}}$ traverse each elementary tree. A step in $\mathcal{D}^{\text{AdjPred}}_{\text{Earley}}$ predicts the adjunction of an auxiliary tree β in a node of an elementary tree γ and starts the traversal of β. Once the foot of β has been reached, the traversal of β is momentary suspended by a step in $\mathcal{D}^{\text{FootPred}}_{\text{Earley}}$, which re-takes the subtree of γ which must be attached to the foot of β. When the traversal of a predicted subtree has finished, a step in $\mathcal{D}^{\text{FootComp}}_{\text{Earley}}$ re-takes the traversal of β continuing at the foot node. When the traversal of β is completely finished, the consecutive application of a step in $\mathcal{D}^{\text{AdjComp}^0}_{\text{Earley}}$ and a step in $\mathcal{D}^{\text{AdjComp}^1}_{\text{Earley}}$ check if the subtree attached to the foot of β corresponds with the adjunction node. The input string has been recognized if a final item $[\top \to \mathbf{R}^\alpha \bullet, 0, 0, n \mid -, -]$, with $\alpha \in \boldsymbol{I}$, is generated. The worst-case time complexity of the algorithm is $\mathcal{O}(n^6)$ [8].

4 A predictive left-corner parser for TAG

In this section we present a parser that uses the left corner relation to filter the predictions on the predictive Earley-like parser for TAGs described in the previous section. The time complexity of the algorithm with respect to the length n of the input string remains $\mathcal{O}(n^6)$, but the practical performance is improved with respect to the Earley-like algorithm, due to the reduction in the size of the set of deduced items.

In a CFG, the left corner of a non-terminal symbol A is the terminal or non-terminal symbol X if and only if there exists a production $A \to X\nu$ in the grammar, where ν is a sequence of symbols. In the case of $A \to \varepsilon$, we consider ε as the left corner of A. The following definition extends the left corner relation to the case of TAG.

Definition 1 Left corner relation on the elementary trees of a TAG
The left corner of a node O^γ is her leftmost daughter P^γ if and only if $adj(P^\gamma) = \{\mathbf{nil}\}$. The relation $>_\ell$ on $(V_N \cup \top) \times (V_N \cup V_T \cup \{\varepsilon, \bot\})$ is defined by $O^\gamma >_\ell P^\gamma$ if there is a production $O^\gamma \to P^\gamma \nu \in \mathcal{P}(\gamma)$ and $adj(P^\gamma) = \{\mathbf{nil}\}$. The transitive and reflexive closure of $>_\ell$ is denoted $>^*_\ell$.

It is worth noting that the left corner relation for TAG always starts on a node labeled with a nonterminal symbol and ends on a adjunction node, a node labeled with a terminal symbol or a node labeled with ε. We use $M^\gamma >_\ell \triangle$ to denote that M^γ is a adjunction node.

In the following, we define a (instantiated item-based) parsing system $\mathbb{P}_{\text{pLC}}$ corresponding to the predictive left corner parsing algorithm for TAG, for an arbitrary tree adjoining grammar G and an input string $a_1 \dots a_n$ with $n \geq 0$:

$$\mathbb{P}_{\text{pLC}} = < \mathcal{I}_{\text{pLC}}, \mathcal{H}_{\text{pLC}}(a_1 \dots a_n), \mathcal{D}_{\text{pLC}} >$$

4.1 Items

The set of items $\mathcal{I}_{\text{pLC}}$ considered in this parsing algorithm is defined as

$$\mathcal{I}_{\text{pLC}} = \mathcal{I}^{\text{p}}_{\text{pLC}} \cup \mathcal{I}^{\text{lc}^1}_{\text{pLC}} \cup \mathcal{I}^{\text{lc}^2}_{\text{pLC}} \cup \mathcal{I}^{\text{lc}^3}_{\text{pLC}}$$

Predictive steps in the Earley-like parser are replaced by goals that the **pLC** parser tries to satisfy in a bottom-up manner. The bottom-up phase of the recognition process is guided towards the correpronding goal by the left corner relation. Therefore, we consider two kinds of items: *predictive* and *left corner items*.

Predictive items in the set $\mathcal{I}^{\mathrm{p}}_{\mathrm{pLC}}$ are used to start the subtree prediction of a node or the adjunction prediction at an adjunction node. These items are of the form $[M^\gamma, h, j]$ and they will be generated if preceding items indicate that a constituent M^γ should be looked for, starting at position j. To preserve the valid prefix property, we include the position h where the tree γ starts at. Thus,

$$\mathcal{I}^{\mathrm{p}}_{\mathrm{pLC}} = \{[M^\gamma, h, j]\}$$

such that $label(M^\gamma) \in V_N$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$ and $0 \leq h \leq j$.

Left corner (lc) items of the form $[C^\gamma; M^\gamma \rightarrow \delta \bullet \nu, h, i, j \mid p, q]$ will be generated if $[C^\gamma, h, j]$ is set as a goal, $C^\gamma >^*_\ell M^\gamma$ and δ spans $a_{i+1} \dots a_p \dots a_q \dots a_j$ if δ dominates the foot of γ and the string spanned by the foot is $a_{p+1} \dots a_q$ or δ spans $a_{i+1} \dots a_j$ in other case. The idea is that each *lc* item incorporates a prefix for a given goal. A graphical representation of this kind of items is shown in figure 1. Thus,

$$\mathcal{I}^{\mathrm{lc}^1}_{\mathrm{pLC}} = \{[C^\gamma; M^\gamma \rightarrow \delta \bullet \nu, h, i, j \mid p, q]\}$$

such that $M^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $C^\gamma >^*_\ell M^\gamma$, $\delta \neq \varepsilon$, $0 \leq h \leq i \leq j$ and $((p, q) \leq (i, j)$ or $(p, q) = (-, -))$. We can notice that the **pLC** parser filters items with the dot preceding the leftmost symbol of the productions. However, this filtering is not possible when the leftmost symbol P^γ of a production is:

1. an adjunction node, because the insertion of an auxiliary tree ends the left corner relation;

2. a node labeled with $\perp$, because the left corner relation is limited to an elementary tree.

To deal with these two special cases, we define the following kind of left corner items:

$$\mathcal{I}^{\mathrm{lc}^2}_{\mathrm{pLC}} = \{[C^\gamma; M^\gamma \rightarrow \bullet P^\gamma \nu, h, j, j \mid -, -]\}$$

such that $M^\gamma \rightarrow P^\gamma \nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $C^\gamma >^*_\ell M^\gamma$, $0 \leq h \leq j$ and $(P^\gamma >_\ell \triangle$ or $label(P^\gamma) = \perp)$.

We also need to define a new kind of intermediate pseudo-items:

$$\mathcal{I}^{\mathrm{lc}^3}_{\mathrm{pLC}} = \{[N^\gamma; N^\gamma \rightarrow \delta \bullet, i, j \mid p, q]\}$$

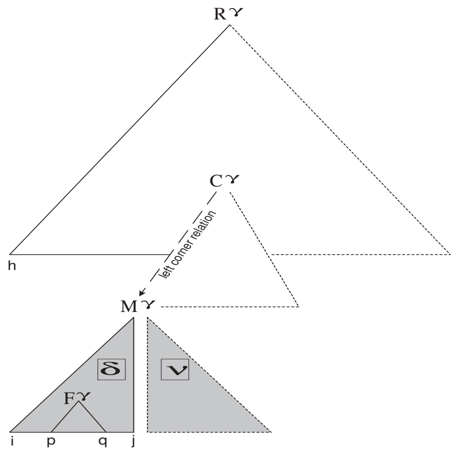

Figure 1: *Left corner items*

such that $N^\gamma \rightarrow \delta \bullet \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq i \leq j$, $(p, q) \leq (i, j)$. It must be satisfied that δ spans $a_i \dots a_p\ \mathbf{F}^\gamma\ a_{q+1} \dots a_j \stackrel{*}{\Rightarrow} a_i \dots a_j$ if and only if $(p, q) \neq (-, -)$, spanning $a_i \dots a_j$ if and only if $(p, q) = (-, -)$.

The set of final items is defined as follows:

$$\mathcal{F}_{\mathrm{pLC}} = \{[\top; \top \rightarrow \mathbf{R}^\alpha \bullet, 0, 0, n \mid -, -] \mid \alpha \in \boldsymbol{I}\}$$

4.2 Deduction steps

With respect to the set of deduction steps $\mathcal{D}_{\mathrm{pLC}}$, we define subsets for *initialize*, *scan* and *complete* similar to the Earley-like parser for TAGs. The left corner relation will be applied for the four cases of prediction: initial, subtree, foot and adjunction. The left corner steps come in three varieties, for terminal, empty and nonterminal left corners. The latter is needed when the left corner is an adjunction or bottom node, due to the auxiliary tree or the subtree excised by an adjunction must be recognized. The set $\mathcal{D}_{\mathrm{pLC}}$ of deduction steps is defined as follows:

$$\mathcal{D}_{\mathrm{pLC}} = \mathcal{D}^{\mathrm{LI_t}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LI_\varepsilon}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LI_{pre}}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{Scan}}_{\mathrm{pLC}} \cup \mathcal{D}^{\varepsilon}_{\mathrm{pLC}} \cup$$

$$\mathcal{D}^{\mathrm{LC_t}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LC_\varepsilon}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LC_{pre}}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LC_n}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{Pre}}_{\mathrm{pLC}} \cup$$

$$\mathcal{D}^{\mathrm{Comp}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LA_t}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LA_\varepsilon}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{LA_{pre}}}_{\mathrm{pLC}} \cup \mathcal{D}^{\mathrm{AdjComp^0}}_{\mathrm{pLC}} \cup$$

$$\mathcal{D}_{\rm pLC}^{\rm AdjComp^1} \cup \mathcal{D}_{\rm pLC}^{\rm LF_t} \cup \mathcal{D}_{\rm pLC}^{\rm LF_\varepsilon} \cup \mathcal{D}_{\rm pLC}^{\rm LF_{pre}} \cup \mathcal{D}_{\rm pLC}^{\rm FootComp}$$

4.2.1 Initialization steps

The recognition starts by predicting every initial tree $\alpha \in \boldsymbol{I}$. As $\top >_\ell^* O^\alpha$ and $O^\alpha \to P^\alpha \nu \in \mathcal{P}(\alpha)$, we can apply a left corner filter and obtain the three following steps:

$$\mathcal{D}_{\rm pLC}^{\rm LI_t} = \frac{[a,0,1]}{[\top; O^\alpha \to P^\alpha \bullet \nu, 0, 0, 1 \mid -, -]}$$

with $label(P^\alpha) = a$.

$$\mathcal{D}_{\rm pLC}^{\rm LI_\varepsilon} = \frac{}{[\top; O^\alpha \to P^\alpha \bullet \nu, 0, 0, 0 \mid -, -]}$$

with $label(P^\alpha) = \varepsilon$.

$$\mathcal{D}_{\rm pLC}^{\rm LI_{pre}} = \frac{}{[\top; O^\alpha \to \bullet P^\alpha \nu, 0, 0, 0 \mid -, -]}$$

with $P^\alpha >_\ell \triangle$.
A step in $\mathcal{D}_{\rm pLC}^{\rm LI_t}$ is used when the left-most daughter P^α is labeled by a terminal symbol. A step in $\mathcal{D}_{\rm pLC}^{\rm LI_\varepsilon}$ is used when P^α is labeled by the empty string . If P^α is an adjunction node then a step in $\mathcal{D}_{\rm pLC}^{\rm LI_{pre}}$ is applied.

4.2.2 Scanning

The scanning steps are analogous to the scanning steps of the Earley-like parser:

$$\mathcal{D}_{\rm pLC}^{\rm Scan} = \frac{\begin{array}{l}[C^\gamma; N^\gamma \to P^\gamma \delta \bullet M^\gamma \nu, h, i, j \mid p, q] \\ [a, j, j+1]\end{array}}{[C^\gamma; N^\gamma \to P^\gamma \delta M^\gamma \bullet \nu, h, i, j+1 \mid p, q]}$$

with $label(M^\gamma) = a$.

$$\mathcal{D}_{\rm pLC}^{\varepsilon} = \frac{[C^\gamma; N^\gamma \to P^\gamma \delta \bullet M^\gamma \nu, h, i, j \mid p, q]}{[C^\gamma; N^\gamma \to P^\gamma \delta M^\gamma \bullet \nu, h, i, j \mid p, q]}$$

with $label(M^\gamma) = \varepsilon$.
A step in $\mathcal{D}_{\rm pLC}^{\rm Scan}$ recognizes the presence of a terminal symbol in the input string. A step in $\mathcal{D}_{\rm pLC}^{\varepsilon}$ encodes the fact that one can skip over a node labeled with ε without having to match anything.

4.2.3 Left corner prediction and completion

When the recognition process reaches a node M^γ that dominates a given node O^γ by means of a left corner relation, i.e. $M^\gamma >_\ell^* O^\gamma$, and adjunction is not mandatory at M^γ, we can apply the left corner filter defined by the following deduction steps:

$$\mathcal{D}_{\rm pLC}^{\rm LC_t} = \frac{\begin{array}{c}[M^\gamma, h, j] \\ [a, j, j+1]\end{array}}{[M^\gamma; O^\gamma \to P^\gamma \bullet \nu, h, j, j+1 \mid -, -]}$$

with $\mathbf{nil} \in adj(M^\gamma),\ label(P^\gamma) = a$.

$$\mathcal{D}_{\rm pLC}^{\rm LC_\varepsilon} = \frac{[M^\gamma, h, j]}{[M^\gamma; O^\gamma \to P^\gamma \bullet \nu, h, j, j \mid -, -]}$$

with $\mathbf{nil} \in adj(M^\gamma),\ label(P^\gamma) = \varepsilon$.

$$\mathcal{D}_{\rm pLC}^{\rm LC_{pre}} = \frac{[M^\gamma, h, j]}{[M^\gamma; O^\gamma \to \bullet P^\gamma \nu, h, j, j \mid -, -]}$$

with $\mathbf{nil} \in adj(M^\gamma),\ label(P^\gamma) = \perp$.

Steps in $\mathcal{D}_{\rm pLC}^{\rm LC_n}$ perform the bottom-up recognition trough the nodes in a left corner relation. It is worth noting that the prefix of left corner items establishes the end of this bottom-up recognition, since the application of a sequence of $\mathcal{D}_{\rm pLC}^{\rm LC_n}$ steps stops when the prefix is reached:

$$\mathcal{D}_{\rm pLC}^{\rm LC_n} = \frac{[M^\gamma; O^\gamma \to \nu \bullet, h, j, k \mid p, q]}{[M^\gamma; Q^\gamma \to O^\gamma \bullet \omega, h, j, k \mid p, q]}$$

with $M^\gamma \neq O^\gamma$.

4.2.4 Prediction

The prediction of nodes is performed by steps in $\mathcal{D}_{\rm pLC}^{\rm Pre}$:

$$\mathcal{D}_{\rm pLC}^{\rm Pre} = \frac{[C^\gamma; N^\gamma \to \delta \bullet M^\gamma \nu, h, i, j \mid p, q]}{[M^\gamma, h, j]}$$

with $label(M^\gamma) \in V_N$.

4.2.5 Normal completion

Steps in the set $\mathcal{D}_{\rm pLC}^{\rm Comp}$ are analogous to the *completor* steps of the Earley-like parser. These steps complete the recognition of a subtree dominated by a node that has not mandatory adjunction:

$$\mathcal{D}^{\text{Comp}}_{\text{pLC}} = \frac{\begin{array}{c}[C^\gamma; N^\gamma \to \delta \bullet M^\gamma\nu, h, i, j \mid p, q] \\ [M^\gamma; M^\gamma \to \omega\bullet, h, j, k \mid p', q']\end{array}}{[C^\gamma; N^\gamma \to \delta M^\gamma \bullet \nu, h, i, k \mid p \cup p', q \cup q']}$$

with $\textbf{nil} \in adj(M^\gamma)$.

4.2.6 Adjunction prediction

When an adjunction node M^γ is reached, we must trigger the recognition of every auxiliary tree β that may be adjoined at M^γ. As $\top >^*_\ell O^\beta$ and $O^\beta \to P^\beta\nu \in \mathcal{P}(\beta)$, we can apply a left corner filter, obtaing the following three sets of deduction steps:

$$\mathcal{D}^{\text{LA}_\text{t}}_{\text{pLC}} = \frac{\begin{array}{c}[M^\gamma, h, j] \\ [a, j, j+1]\end{array}}{[\top; O^\beta \to P^\beta \bullet \nu, j, j, j+1 \mid -, -]}$$

with $\beta \in adj(M^\gamma)$, $label(P^\beta) = a$.

$$\mathcal{D}^{\text{LA}_\varepsilon}_{\text{pLC}} = \frac{[M^\gamma, h, j]}{[\top; O^\beta \to P^\beta \bullet \nu, j, j, j \mid -, -]}$$

with $\beta \in adj(M^\gamma)$, $label(P^\beta) = \varepsilon$.

$$\mathcal{D}^{\text{LA}_\text{pre}}_{\text{pLC}} = \frac{[M^\gamma, h, j]}{[\top; O^\beta \to \bullet P^\beta\nu, j, j, j \mid -, -]}$$

with $\beta \in adj(M^\gamma)$, $(P^\beta >_\ell \triangle$ or $label(P^\beta) = \bot)$.

4.2.7 Adjunction completion

Once the recognition of an auxiliary tree β is exhausted, the parser completes the adjoined node M^γ which it was adjoined on:

$$\mathcal{D}^{\text{AdjComp}^0}_{\text{pLC}} = \frac{\begin{array}{c}[\top; \top \to \mathbf{R}^\beta\bullet, j, j, m \mid k, l] \\ [M^\gamma; M^\gamma \to \omega\bullet, h, k, l \mid p, q]\end{array}}{[M^\gamma; M^\gamma \to \omega\bullet, j, m \mid p, q]}$$

with $\beta \in adj(M^\gamma)$.

$$\mathcal{D}^{\text{AdjComp}^1}_{\text{pLC}} = \frac{\begin{array}{c}[M^\gamma; M^\gamma \to \omega\bullet, j, m \mid p, q] \\ [M^\gamma; M^\gamma \to \omega\bullet, h, k, l \mid p, q] \\ [C^\gamma; N^\gamma \to \delta \bullet M^\gamma\nu, h, i, j \mid p', q']\end{array}}{[C^\gamma; N^\gamma \to \delta M^\gamma \bullet \nu, h, i, m \mid p \cup p', q \cup q']}$$

with $\beta \in adj(M^\gamma)$.

4.2.8 Foot prediction

When the recognition reaches an adjunction node M^γ and an auxiliary tree β, such that $\beta \in adj(M^\gamma)$, is recognized up to the node $\bot$, then the recognition of the excised subtree must be started. As in previous cases, we can apply a left corner filter to this prediction, defining the following set of deduction steps:

$$\mathcal{D}^{\text{LF}_\text{t}}_{\text{pLC}} = \frac{\begin{array}{c}[M^\gamma, h, j] \\ [E^\beta; \mathbf{F}^\beta \to \bullet\bot, j, k, k \mid -, -] \\ [a, k, k+1]\end{array}}{[M^\gamma; O^\gamma \to P^\gamma \bullet \nu, h, k, k+1 \mid -, -]}$$

with $label(P^\gamma) = a$.

$$\mathcal{D}^{\text{LF}_\varepsilon}_{\text{pLC}} = \frac{\begin{array}{c}[M^\gamma, h, j] \\ [E^\beta; \mathbf{F}^\beta \to \bullet\bot, j, k, k \mid -, -]\end{array}}{[M^\gamma; O^\gamma \to P^\gamma \bullet \nu, h, k, k \mid -, -]}$$

with $label(P^\gamma) = \varepsilon$.

$$\mathcal{D}^{\text{LF}_\text{pre}}_{\text{pLC}} = \frac{\begin{array}{c}[M^\gamma, h, j] \\ [E^\beta; \mathbf{F}^\beta \to \bullet\bot, j, k, k \mid -, -]\end{array}}{[M^\gamma; O^\gamma \to \bullet P^\gamma\nu, h, k, k \mid -, -]}$$

with $P^\gamma >_\ell \triangle$ or $label(P^\gamma) = \bot$.

4.2.9 Foot completion

A set in $\mathcal{D}^{\text{FootComp}}_{\text{pLC}}$ starts the recognition of the right context of an auxiliary tree β when the excised subtree rooted at M^γ is exhausted:

$$\mathcal{D}^{\text{FootComp}}_{\text{pLC}} = \frac{\begin{array}{c}[M^\gamma, h, j] \\ [E^\beta; \mathbf{F}^\beta \to \bullet\bot, j, k, k \mid -, -] \\ [M^\gamma; M^\gamma \to \nu\bullet, h, k, l \mid p, q]\end{array}}{[E^\beta; \mathbf{F}^\beta \to \bot\bullet, j, k, l \mid k, l]}$$

with $\beta \in adj(M^\gamma)$.

The time complexity of the algorithm with respect to the length n of the input string is $\mathcal{O}(n^6)$ for this parser, due to the adjunction completion steps that present the maximum number of relevant indices. The correction of the algorithm is proved in [2].

5 Conclusion

We have defined a new parser for TAG that is an extension of the Left Corner parser for Context Free Grammars. The new parser guarantees the VPP and it can be view as a filter on an Earley-like parser for TAGs where the number of predictions is reduced due to the generalized left corner relation that we have established on the nodes of elementary trees. The worst-case complexity with respect to space and time is the standard one for TAG parsing, but preliminary experiments have shown a better performance than classical Earley-like parsers for TAG.

Acknowledgements

The work described in this article has been supported in part by Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica (TIC2000-0370-C02-01), Ministerio de Ciencia y Tecnología (HP2001-0044, FIT-150500-2002-416) and Xunta de Galicia (PGIDT01PXI10506PN).

References

[1] M.A. Alonso, D. Cabrero, E. de la Clergerie, and M. Vilares. Tabular algorithms for TAG parsing. In *Proc. of Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL'99)*, pages 150–157, Bergen, Norway, 1999.

[2] V. Carrillo. Corrección de analizadores basados en left-corner para tags. Technical Report LSI-2002-03, Department of Computer Languages and Systems, University of Seville, 2002.

[3] V. J. Díaz, V. Carrillo, and M. A.Alonso. A left corner parser for tree adjoining grammars. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+6)*, pages 90–95, Venice, Italy, May 2002.

[4] J. Earley. *An efficient context-free parsing algorithm*. PhD thesis, Carnegie-Mellon University, Pittsburg, PA, 1968.

[5] R. Evans and D. Weir. A structure-sharing parser for lexicalized grammars. In *Proc. of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-COLING'98)*, volume I, pages 372–378, Montreal, Canada, 1998.

[6] A.K. Joshi and Y. Schabes. *Handbook of Formal Languages*, volume 3, chapter Tree-adjoining grammars, pages 69–123. G. Rozenberg and A. Salomaa, 1997.

[7] B. Lang. The systematic construction of earley parsers: Application to the production of $o(n^6)$ earley parsers for tree adjoining grammars. In *Proc. of the 1st International Workshop on Tree Adjoining Grammars*, Montreal, Canada, 1990.

[8] M.-J. Nederhof. The computational complexity of the correct-prefix property for TAGs. *Computational Linguistics*, 25(3):345–360, 1999.

[9] D.J. Rosenkrantz and P.M. Lewis. Deterministic left corner parsing. In *Proc. of 11th Annual Symposium on Switching and Automata Theory*, pages 139–152, 1970.

[10] 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):479–513, 1995.

[11] S.M. Shieber, Y. Schabes, and F.C.N. Pereira. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1–2):3–36, 1995.

[12] K. Sikkel. *Parsing schemata — A framework for specification and analysis of parsing algorithms*. Springer-Verlag, Berlin/Heidelberg/New York, 1997.