Tabular Algorithms for TAG Parsing

Miguel A. Alonso
Departamento de Computación
Univesidad de La Coruña
Campus de Elviña s/n
15071 La Coruña
SPAIN
alonso@dc.fi.udc.es

David Cabrero
Departamento de Computación
Univesidad de La Coruña
Campus de Elviña s/n
15071 La Coruña
SPAIN
cabrero@dc.fi.udc.es

Eric de la Clergerie
INRIA
Domaine de Voluceau
Rocquencourt, B.P. 105
78153 Le Chesnay Cedex
FRANCE
Eric.De_La_Clergerie@inria.fr

Manuel Vilares
Departamento de Computación
Univesidad de La Coruña
Campus de Elviña s/n
15071 La Coruña
SPAIN
vilares@dc.fi.udc.es

Abstract

We describe several tabular algorithms for Tree Adjoining Grammar parsing, creating a continuum from simple pure bottom-up algorithms to complex predictive algorithms and showing what transformations must be applied to each one in order to obtain the next one in the continuum.

1 Introduction

Tree Adjoining Grammars are a extension of CFG introduced by Joshi in (Joshi, 1987) that use trees instead of productions as the primary representing structure. Several parsing algorithms have been proposed for this formalism, most of them based on tabular techniques, ranging from simple bottom-up algorithms (Vijay-Shanker and Joshi, 1985) to sophisticated extensions of the Earley's algorithm (Schabes and Joshi, 1988; Schabes, 1994; Nederhof, 1997). However, it is difficult to inter-relate different parsing algorithms. In this paper we study several tabular algorithms for TAG parsing, showing their common characteristics and how one algorithm can be derived from another in turn, creating a continuum from simple pure bottom-up to complex predictive algorithms.

Formally, a TAG is a 5-tuple $\mathcal{G} = (V_N, V_T, S, \boldsymbol{I}, \boldsymbol{A})$, where V_N is a finite set of non-terminal symbols, V_T a finite set of terminal symbols, S the axiom of the grammar, $\boldsymbol{I}$ a finite set of *initial trees* and $\boldsymbol{A}$ a finite set of *auxiliary trees*. $\boldsymbol{I} \cup \boldsymbol{A}$ is the set of *elementary trees*. Internal nodes are labeled by non-terminals and leaf nodes by terminals or ε, except for just one leaf per auxiliary tree (the *foot*) which is labeled by the same non-terminal used as the label of its root node. The path in an elementary tree from the root node to the foot node is called the *spine* of the tree.

New trees are derived by *adjoining*: let α be a tree containing a node N^α labeled by A and let β be an auxiliary tree whose root and foot nodes are also labeled by A. Then, the adjoining of β at the *adjunction node* N^α is obtained by excising the subtree of α with root N^α, attaching β to N^α and attaching the excised subtree to the foot of β. We use $\beta \in adj(N^\alpha)$ to denote that a tree β may be adjoined at node N^α of the elementary tree α.

In order to describe the parsing algorithms for TAG, we must be able to represent the partial recognition of elementary trees. Parsing algorithms for context-free grammars usually denote partial recognition of productions by dotted productions. We can extend this approach to the case of TAG by considering each elementary tree γ as formed by a set of context-free productions $\mathcal{P}(\gamma)$: a node N^γ and its children $N_1^\gamma \ldots N_g^\gamma$ are represented by a production $N^\gamma \rightarrow N_1^\gamma \ldots N_g^\gamma$. Thus, the position of the dot in the tree is indicated by the position of the dot in a production in $\mathcal{P}(\gamma)$. The elements of the productions are the nodes of

the tree, except for the case of elements belonging to $V_T \cup \{\varepsilon\}$ in the right-hand side of production. Those elements may not have children and are not candidates to be adjunction nodes, so we identify such nodes labeled by a terminal with that terminal.

To simplify the description of parsing algorithms we consider an additional production $\top \rightarrow \mathbf{R}^{\alpha}$ for each initial tree and the two additional productions $\top \rightarrow \mathbf{R}^{\beta}$ and $\mathbf{F}^{\beta} \rightarrow \bot$ for each auxiliary tree β, where $\mathbf{R}^{\beta}$ and $\mathbf{F}^{\beta}$ correspond to the root node and the foot node of β, respectively. After disabling $\top$ and $\bot$ as adjunction nodes the generative capability of the grammars remains intact.

The relation $\Rightarrow$ of derivation on $\mathcal{P}(\gamma)$ is defined by $\delta \Rightarrow \nu$ if there are $\delta', \delta'', M^{\gamma}, \upsilon$ such that $\delta = \delta' M^{\gamma} \delta''$, $\nu = \delta' \upsilon \delta''$ and $M^{\gamma} \rightarrow \upsilon \in \mathcal{P}(\gamma)$ exists. The reflexive and transitive closure of $\Rightarrow$ is denoted $\overset{*}{\Rightarrow}$.

In a abuse of notation, we also use $\overset{*}{\Rightarrow}$ to represent derivations involving an adjunction. So, $\delta \overset{*}{\Rightarrow} \nu$ if there are $\delta', \delta'', M^{\gamma}, \upsilon$ such that $\delta = \delta' M^{\gamma} \delta''$, $\mathbf{R}^{\beta} \overset{*}{\Rightarrow} \upsilon_1 \mathbf{F}^{\beta} \upsilon_3$, $\beta \in adj(M^{\gamma})$, $M^{\gamma} \rightarrow \upsilon_2$ and $\nu = \delta' \upsilon_1 \upsilon_2 \upsilon_3 \delta''$.

Given two pairs (p,q) and (i,j) of integers, $(p,q) \leq (i,j)$ is satisfied if $i \leq p$ and $q \leq j$. 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.

1.1 Parsing Schemata

We will describe parsing algorithms using *Parsing Schemata*, a framework for high-level description of parsing algorithms (Sikkel, 1997). An interesting application of this framework is the analysis of the relations between different parsing algorithms by studying the formal relations between their underlying parsing schemata. Originally, this framework was created for context-free grammars but we have extended it to deal with tree adjoining grammars.

A *parsing system* for a grammar G and string $a_1 \ldots a_n$ 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 $\frac{\eta_1, \ldots, \eta_k}{\xi}$ *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.

Parsing schemata are closely related to *grammatical deduction systems* (Shieber et al., 1995), where items are called *formula schemata*, deduction steps are *inference rules*, hypothesis are *axioms* and final items are *goal formulas*.

A parsing schema can be generalized from another one using the following transformations (Sikkel, 1997):

- *Item refinement*, breaking single items into multiple items.
- *Step refinement*, decomposing a single deduction step in a sequence of steps.
- *Extension* of a schema by considering a larger class of grammars.

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.
- *Step contraction*, in which a sequence of deduction steps is replaced by a single one.

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

2 A CYK-like Algorithm

We have chosen the CYK-like algorithm for TAG described in (Vijay-Shanker and Joshi, 1985) as our starting point. Due to the intrinsic limitations of this pure bottom-up algorithm, the grammars it can deal with are restricted to those with nodes having at most two children.

The tabular interpretation of this algorithm works with items of the form

$$[N^{\gamma}, i, j \mid p, q \mid adj]$$

such that $N^{\gamma} \overset{*}{\Rightarrow} a_{i+1} \ldots a_p \ \mathbf{F}^{\gamma} \ a_{q+1} \ldots a_j \overset{*}{\Rightarrow} a_{i+1} \ldots a_j$ if and only if $(p,q) \neq (-,-)$ and $N^{\gamma} \overset{*}{\Rightarrow} a_{i+1} \ldots a_j$ if and only if $(p,q) = (-,-)$, where N^{γ} is a node of an elementary tree with a label belonging to V_N.

The two indices with respect to the input string i and j indicate the portion of the input string that has been derived from N^{γ}. If $\gamma \in \boldsymbol{A}$, p and q are two indices with respect to the input string that indicate that part of the input string recognized

by the foot node of γ. In other case $p = q = -$ representing they are undefined. The element *adj* indicates whether adjunction has taken place on node N^γ.

The introduction of the element *adj* taking its value from the set {true, false} corrects the items previously proposed for this kind of algorithms in (Vijay-Shanker and Joshi, 1985) in order to avoid several adjunctions on a node. A value of *true* indicates that an adjunction has taken place in the node N^γ and therefore further adjunctions on the same node are forbidden. A value of *false* indicates that no adjunction was performed on that node. In this case, during future processing this item can play the role of the item recognizing the excised part of an elementary tree to be attached to the foot node of an auxiliary tree. As a consequence, only one adjunction can take place on an elementary node, as is prescribed by the tree adjoining grammar formalism (Schabes and Shieber, 1994). As an additional advantage, the algorithm does not need to require the restriction that every auxiliary tree must have at least one terminal symbol in its frontier (Vijay-Shanker and Joshi, 1985).

Schema 1 *The parsing systems* $\mathbb{P}_{\text{CYK}}$ *corresponding to the CYK-line algorithm for a tree adjoining grammar* $\mathcal{G}$ *and an input string* $a_1 \dots a_n$ *is defined as follows:*

$$\mathcal{I}_{\text{CYK}} = \{\ [N^\gamma, i, j \mid p, q \mid adj]\ \}$$
such that $N^\gamma \in \mathcal{P}(\gamma)$, $label(N^\gamma) \in V_N$, $\gamma \in I \cup A$, $0 \le i \le j$, $(p,q) \le (i,j)$, $adj \in \{true, false\}$

$$\mathcal{H}_{\text{CYK}} = \{\ [a, i-1, i] \mid a = a_i, 1 \le i \le n\ \}$$

$$\mathcal{D}^{\text{Scan}}_{\text{CYK}} = \frac{[a, i-1, i]}{[N^\gamma, i-1, i \mid -, - \mid false]} \quad N^\gamma \to a$$

$$\mathcal{D}^{\varepsilon}_{\text{CYK}} = \frac{}{[N^\gamma, i, i \mid -, - \mid false]} \quad N^\gamma \to \varepsilon$$

$$\mathcal{D}^{\text{Foot}}_{\text{CYK}} = \frac{}{[\mathbf{F}^\gamma, i, j \mid i, j \mid false]}$$

$$\mathcal{D}^{\text{LeftDom}}_{\text{CYK}} = \frac{[M^\gamma, i, k \mid p, q \mid adj],\ [P^\gamma, k, j \mid -, - \mid adj]}{[N^\gamma, i, j \mid p, q \mid false]}$$
such that $N^\gamma \to M^\gamma P^\gamma \in \mathcal{P}(\gamma)$, $M^\gamma \in spine(\gamma)$

$$\mathcal{D}^{\text{RightDom}}_{\text{CYK}} = \frac{[M^\gamma, i, k \mid -, - \mid adj],\ [P^\gamma, k, j \mid p, q \mid adj]}{[N^\gamma, i, j \mid p, q \mid false]}$$
such that $N^\gamma \to M^\gamma P^\gamma \in \mathcal{P}(\gamma)$, $P^\gamma \in spine(\gamma)$

$$\mathcal{D}^{\text{NoDom}}_{\text{CYK}} = \frac{[M^\gamma, i, k \mid -, - \mid adj],\ [P^\gamma, k, j \mid -, - \mid adj]}{[N^\gamma, i, j \mid -, - \mid false]}$$
such that $N^\gamma \to M^\gamma P^\gamma \in \mathcal{P}(\gamma)$, $M^\gamma, P^\gamma \notin spine(\gamma)$

$$\mathcal{D}^{\text{Unary}}_{\text{CYK}} = \frac{[M^\gamma, i, j \mid p, q \mid adj]}{[N^\gamma, i, j \mid p, q \mid false]} \quad N^\gamma \to M^\gamma \in \mathcal{P}(\gamma)$$

$$\mathcal{D}^{\text{Adj}}_{\text{CYK}} = \frac{[\mathbf{R}^\beta, i', j' \mid i, j \mid adj],\ [N^\gamma, i, j \mid p, q \mid false]}{[N^\gamma, i', j' \mid p, q \mid true]}$$
such that $\beta \in A$, $\beta \in adj(N^\gamma)$

$$\mathcal{D}_{\text{CYK}} = \mathcal{D}^{\text{Scan}}_{\text{CYK}} \cup \mathcal{D}^{\varepsilon}_{\text{CYK}} \cup \mathcal{D}^{\text{Foot}}_{\text{CYK}} \cup \mathcal{D}^{\text{LeftDom}}_{\text{CYK}} \cup \mathcal{D}^{\text{RightDom}}_{\text{CYK}} \cup \mathcal{D}^{\text{NoDom}}_{\text{CYK}} \cup \mathcal{D}^{\text{Unary}}_{\text{CYK}} \cup \mathcal{D}^{\text{Adj}}_{\text{CYK}}$$

$$\mathcal{F}_{\text{CYK}} = \{\ [\mathbf{R}^\alpha, 0, n \mid -, - \mid adj] \mid \alpha \in I\ \}$$

The hypotheses defined for this parsing system are the standard ones and therefore they will be omitted in the next parsing systems described in this paper.

The key steps in the parsing system $\mathbb{P}_{\text{CYK}}$ are $\mathcal{D}^{\text{Foot}}_{\text{CYK}}$ and $\mathcal{D}^{\text{Adj}}_{\text{CYK}}$, which are in charge of the recognition of adjunctions. The other steps are in charge of the bottom-up traversal of elementary trees and, in the case of auxiliary trees, the propagation of the information corresponding to the part of the input string recognized by the foot node.

The set of deductive steps $\mathcal{D}^{\text{Foot}}_{\text{CYK}}$ make it possible to start the bottom-up traversal of each auxiliary tree, as it predict all possible parts of the input string that can be recognized by the foot nodes. Several parses can exist for an auxiliary tree which only differs in the part of the input string which was predicted for the foot node. Not all of them need take part on a derivation, only those with a predicted foot compatible with an adjunction. The compatibility between the adjunction node and the foot node of the adjoined tree is checked by a derivation step $\mathcal{D}^{\text{Adj}}_{\text{CYK}}$: when the root of an auxiliary tree β has been reached, it checks for the existence of a subtree of an elementary tree rooted by a node N^γ which satisfies the following conditions:

1. β can be adjoined on N^γ.
2. N^γ derives the same part of the input string derived from the foot node of β.

If the conditions are satisfied, further adjunctions on N are forbidden and the parsing process continues a bottom-up traverse of the rest of the elementary tree γ containing N^γ.

3 A Bottom-up Earley-like Algorithm

To overcome the limitation of binary branching in trees imposed by CYK-like algorithms, we define a bottom-up Earley-like parsing algorithm for TAG. As a first step we need to introduce the dotted rules into items, which are of the form

$$[N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]$$

such that $\delta \stackrel{*}{\Rightarrow} a_{i+1} \ldots a_p \ \mathbf{F}^\gamma \ a_{q+1} \ldots a_j \stackrel{*}{\Rightarrow} a_{i+1} \ldots a_j$ if and only if $(p,q) \neq (-,-)$ and $\delta \stackrel{*}{\Rightarrow} a_{i+1} \ldots a_j$ if and only if $(p,q) = (-,-)$.

The items of the new parsing schema, denoted $\mathbf{buE_1}$, are obtained by refining the items of **CYK**. The dotted rules eliminate the need for the element *adj* indicating whether the node in the left-hand side of the production has been used as adjunction node.

Schema 2 *The parsing system $\mathbb{P}_{\text{buE}}$ corresponding to the bottom-up Earley-like parsing algorithm, given a tree adjoining grammar $\mathcal{G}$ and a input string $a_1 \ldots a_n$ is defined as follows:*

$$\mathcal{I}_{\text{buE}} = [N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]$$
such that $N^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $(p,q) \leq (i,j)$

$$\mathcal{D}^{\text{Init}}_{\text{buE}} = \frac{}{[N^\gamma \rightarrow \bullet\delta, i, i \mid -, -]}$$

$$\mathcal{D}^{\text{Foot}}_{\text{buE}} = \frac{}{[\mathbf{F}^\beta \rightarrow \bot\bullet, i, j \mid i, j]}$$

$$\mathcal{D}^{\text{Scan}}_{\text{buE}} = \frac{[N^\gamma \rightarrow \delta \bullet a\nu, i, j-1 \mid p, q], \ [a, j-1, j]}{[N^\gamma \rightarrow \delta a \bullet \nu, i, j \mid p, q]}$$

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

$$\mathcal{D}^{\text{AdjComp}}_{\text{buE}} = \frac{[\top \rightarrow \mathbf{R}^\beta\bullet, k, j \mid l, m], \ [M^\gamma \rightarrow \upsilon\bullet, l, m \mid p', q'], \ [N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, k \mid p, q],}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j \mid p \cup p', q \cup q']}$$
such that $\beta \in \boldsymbol{A}$, $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{buE}} = \mathcal{D}^{\text{Init}}_{\text{buE}} \cup \mathcal{D}^{\text{Foot}}_{\text{buE}} \cup \mathcal{D}^{\text{Scan}}_{\text{buE}} \cup \mathcal{D}^{\text{Comp}}_{\text{buE}} \cup \mathcal{D}^{\text{AdjComp}}_{\text{buE}}$$

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

The deduction steps of $\mathbb{P}_{\text{buE}}$ are obtained from the steps in $\mathbb{P}_{\text{CYK}}$ applying the following refinement:

- LeftDom, RightDom and NoDom deductive steps have been split into steps Init and Comp.
- Unary and ε steps are no longer necessary, due to the uniform treatment of all productions independently of the length of the production.

The algorithm performs a bottom-up recognition of the auxiliary trees applying the steps $\mathcal{D}^{\text{Comp}}_{\text{buE}_1}$. During the traversal of auxiliary trees, information about the part of the input string recognized by the foot is propagated bottom-up. A set of deductive steps $\mathcal{D}^{\text{Init}}_{\text{buE}}$ are in charge of starting the recognition process, predicting all possible start positions for each rule.

A filter has been applied to the parsing system $\mathbb{P}_{\text{CYK}}$, contracting the deductive steps Adj and Comp in a single AdjComp, as the item generated by a deductive step Adj can only be used to advance the dot in the rule which has been used to predict the left-hand side of its production.

4 An Earley-like Algorithm

An Earley-like parsing algorithm for TAG can be obtained by incorporating top-down prediction. To do so, two dynamic filters must be applied to $\mathbb{P}_{\text{buE}}$:

- The deductive steps in $\mathcal{D}^{\text{Init}}_{\text{E}}$ will only consider productions having the root of an initial tree as left-hand side.
- A new set $\mathcal{D}^{\text{Pred}}_{\text{E}}$ of predictive steps will be in charge of controlling the generation of new items, considering only those new items which are potentially useful for the parsing process.

Schema 3 *The parsing system $\mathbb{P}_{\text{E}}$ corresponding to an Earley-like parsing algorithm for TAG without the valid prefix property, given a tree adjoining grammar $\mathcal{G}$ and a input string $a_1 \ldots a_n$ is defined as follows:*

$$\mathcal{I}_{\text{E}} = \mathcal{I}_{\text{buE}}$$

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

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

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

such that $\beta \in adj(M^\gamma)$

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

such that $\beta \in adj(M^\gamma)$

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

such that $\beta \in adj(M^\gamma)$, $p \cup p'$ *and* $q \cup q'$ *are defined*

$$\mathcal{D}_{\mathrm{E}}^{\mathrm{AdjComp}} = \frac{\begin{array}{l}[\top \rightarrow \mathbf{R}^\beta\bullet, j, m \mid k, l], \\ [M^\gamma \rightarrow \upsilon\bullet, k, l \mid p, q], \\ [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid p', q']\end{array}}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, m \mid p \cup p', q \cup q']}$$

such that $\beta \in adj(M^\gamma)$

$$\begin{array}{rl}\mathcal{D}_{\mathrm{E}} = & \mathcal{D}_{\mathrm{E}}^{\mathrm{Init}} \cup \mathcal{D}_{\mathrm{buE}}^{\mathrm{Scan}} \cup \mathcal{D}_{\mathrm{E}}^{\mathrm{Pred}} \cup \mathcal{D}_{\mathrm{buE}}^{\mathrm{Comp}} \cup \\ & \mathcal{D}_{\mathrm{E}}^{\mathrm{AdjPred}} \cup \mathcal{D}_{\mathrm{E}}^{\mathrm{FootPred}} \cup \mathcal{D}_{\mathrm{E}}^{\mathrm{FootComp}} \cup \\ & \mathcal{D}_{\mathrm{E}}^{\mathrm{AdjComp}}\end{array}$$

$$\mathcal{F}_{\mathrm{E}} = \mathcal{F}_{\mathrm{buE}}$$

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 $\mathcal{D}_{\mathrm{E}}^{\mathrm{Pred}}$ and $\mathcal{D}_{\mathrm{E}}^{\mathrm{Comp}}$ traverse each elementary tree. A step in $\mathcal{D}_{\mathrm{E}}^{\mathrm{AdjPred}}$ 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 $\mathcal{D}_{\mathrm{E}}^{\mathrm{FootPred}}$, which re-takes the subtree of γ which must be attached to the foot of β. At this moment, there is no information available about the node in which the adjunction of β has been performed, so all possible nodes are predicted. When the traversal of a predicted subtree has finished, a step in $\mathcal{D}_{\mathrm{E}}^{\mathrm{FootComp}}$ re-takes the traversal of β continuing at the foot node. When the traversal of β is completely finished, a deduction step in $\mathcal{D}_{\mathrm{E}}^{\mathrm{AdjComp}}$ checks if the subtree attached to the foot of β corresponds with the adjunction node. With respect to steps in $\mathcal{D}_{\mathrm{E}}^{\mathrm{AdjComp}}$, p and q are instantiated if and only if the adjunction node is in the spine of γ.

5 The Valid Prefix Property

Parsers satisfying the *valid prefix property* 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 valid prefix property 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 maintain the valid prefix property, the parser must recognize all possible derived trees in prefix form. In order to do that, two different phases must work coordinately: a top-down phase that expands the children of each node visited and a bottom-up phase grouping the children nodes to indicate the recognition of the parent node (Schabes, 1991).

During the recognition of a derived tree in prefix form, node expansion can depend on adjunction operations performed in the previously visited part of the tree. Due to this kind of dependencies the set path is a context-free language (Vijay-Shanker et al., 1987). A bottom-up algorithm (e.g. CYK-like or bottom-up Earley-like) can stack the dependencies shown by the context-free language defining the path-set. This is sufficient to get a correct parsing algorithm, but without the valid prefix property. To preserve this property the algorithm must have a top-down phase which also stacks the dependencies shown by the language defining the path-set. To transform an algorithm without the valid prefix property into another which preserves it is a difficult task because stacking operations performed during top-down and bottom-up phases must be correlated some way and it is not clear how to do so without augmenting the time complexity (Nederhof, 1997).

CYK-like, bottom-up Earley-like and Earley-like parsing algorithms described above do not preserve the valid prefix property because foot-prediction (a top-down operation) is not restrictive enough to guarantee that the subtree attached to the foot node really corresponds with a instance of the tree involved in the adjunction.

To obtain a Earley-like parsing algorithm for tree adjoining grammars preserving the valid prefix property we need to refine the items by including a new element to indicate the position of

the input string corresponding to the left-most extreme of the frontier of the tree to which the dotted rule in the item belongs:

$$[h, N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]$$

such that $\mathbf{R}^\gamma \stackrel{*}{\Rightarrow} a_{h+1} \ldots a_i \delta \nu v$ and $\delta \stackrel{*}{\Rightarrow} a_i \ldots a_p$ $\mathbf{F}^\gamma$ $a_{q+1} \ldots a_j \stackrel{*}{\Rightarrow} a_i \ldots a_j$ if and only if $(p, q) \neq (-, -)$ and $\delta \stackrel{*}{\Rightarrow} a_i \ldots a_j$ if and only if $(p, q) = (-, -)$.

Thus, an item $[N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]$ of $\mathbb{P}_{\mathrm{E}}$ corresponds now with a subset of $\{[h, N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]\}$ for all $h \in [0, n]$.

Schema 4 *The parsing system $\mathbb{P}_{\mathrm{Earley}}$ corresponding to a Earley-like parsing algorithm with the valid prefix property, for a tree adjoining grammar $\mathcal{G}$ and a input string $a_1 \ldots a_n$ is defined as follows:*

$$\mathcal{I}_{\mathrm{Earley}} = [h, N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]$$
$$N^\gamma \rightarrow \delta \bullet \nu \in \mathcal{P}(\gamma),\ \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq h \leq i \leq j,\ (p, q) \leq (i, j)$$

$$\mathcal{D}^{\mathrm{Init}}_{\mathrm{Earley}} = \frac{}{\vdash [0, \top \rightarrow \bullet \mathbf{R}^\alpha, 0, 0 \mid -, -]} \quad \alpha \in \boldsymbol{I}$$

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

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

$$\mathcal{D}^{\mathrm{Comp}}_{\mathrm{Earley}} = \frac{[h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, k \mid p, q], [h, M^\gamma \rightarrow v \bullet, k, j \mid p', q']}{[h, N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j \mid p \cup p', q \cup q']}$$

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

$$\mathcal{D}^{\mathrm{FootPred}}_{\mathrm{Earley}} = \frac{[j, \mathbf{F}^\beta \rightarrow \bullet \bot, k, k \mid -, -], [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid p, q]}{[h, M^\gamma \rightarrow \bullet \delta, k, k \mid -, -]}$$
such that $\beta \in adj(M^\gamma)$

$$\mathcal{D}^{\mathrm{FootComp}}_{\mathrm{Earley}} = \frac{[h, M^\gamma \rightarrow v \bullet, k, l \mid p, q], [j, \mathbf{F}^\beta \rightarrow \bullet \bot, k, k \mid -, -], [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid p', q']}{[j, \mathbf{F}^\beta \rightarrow \bot \bullet, k, l \mid k, l]}$$
$\beta \in adj(M^\gamma)$, $p \cup p'$ *and* $q \cup q'$ *are defined*

$$\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}} = \frac{[j, \top \rightarrow \mathbf{R}^\beta \bullet, j, m \mid k, l], [h, M^\gamma \rightarrow v \bullet, k, l \mid p, q], [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid p', q']}{[h, N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, m \mid p \cup p', q \cup q']}$$
such that $\beta \in adj(M^\gamma)$

$$\mathcal{D}_{\mathrm{Earley}} = \mathcal{D}^{\mathrm{Init}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{Scan}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{Pred}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{Comp}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{AdjPred}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{FootPred}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{FootComp}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$$

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

Time complexity of the Earley-like algorithm with respect to the length n of input string is $\mathcal{O}(n^7)$, and it is given by steps $\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$. Although 8 indices are involved in a step $\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$, partial application allows us to reduce the time complexity to $\mathcal{O}(n^7)$.

Algorithms without the valid prefix property have a time complexity $\mathcal{O}(n^6)$ with respect to the length of the input string. The change in complexity is due to the additional index in items of $\mathbb{P}_{\mathrm{Earley}}$. That index is needed to check the trees involved in steps $\mathcal{D}^{\mathrm{FootPred}}_{\mathrm{Earley}}$ and $\mathcal{D}^{\mathrm{FootComp}}_{\mathrm{Earley}}$. In the other steps, that index is only propagated to the generated item. This feature allows us to refine the steps in $\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$, splitting them into several steps generating intermediate items without that index. To get a correct splitting, we must first differentiate steps in $\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$ in which p and q are instantiated from steps in $\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$ in which p' and q' are instantiated. So, we must define two new sets $\mathcal{D}^{\mathrm{AdjComp}^1}_{\mathrm{Earley}}$ and $\mathcal{D}^{\mathrm{AdjComp}^2}_{\mathrm{Earley}}$ of steps instead of the single set $\mathcal{D}^{\mathrm{AdjComp}}_{\mathrm{Earley}}$. Additionally, in steps in $\mathcal{D}^{\mathrm{AdjComp}^1}_{\mathrm{Earley}}$ we need to introduce a new item (dynamic filtering) to guarantee the correctness of the steps.

$$\mathcal{D}^{\mathrm{AdjComp}^1}_{\mathrm{Earley}} = \frac{[j, \top \rightarrow \mathbf{R}^\beta \bullet, j, m \mid k, l], [h, M^\gamma \rightarrow v \bullet, k, l \mid p, q], [h, \mathbf{F}^\gamma \rightarrow \bot \bullet, p, q \mid p, q], [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid -, -]}{[h, N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, m \mid p, q]}$$
such that $\beta \in adj(M^\gamma)$

$$\mathcal{D}^{\mathrm{AdjComp}^2}_{\mathrm{Earley}} = \frac{[j, \top \rightarrow \mathbf{R}^\beta \bullet, j, m \mid k, l], [h, M^\gamma \rightarrow v \bullet, k, l \mid -, -], [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid p', q']}{[h, N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, m \mid p', q']}$$
such that $\beta \in adj(M^\gamma)$

$$\mathcal{D}_{\mathrm{Earley}} = \mathcal{D}^{\mathrm{Init}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{Scan}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{Pred}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{Comp}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{AdjPred}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{FootPred}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{FootComp}}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{AdjComp}^1}_{\mathrm{Earley}} \cup \mathcal{D}^{\mathrm{AdjComp}^2}_{\mathrm{Earley}}$$

Now, we must refine steps in $\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^1}$ into steps in $\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^0}$ and $\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^{1'}}$, and refine steps in $\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^2}$ into steps in $\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^0}$ and $\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^{2'}}$. Correctness of these splittings is guaranteed by the *context-free property of TAG* (Vijay-Shanker and Weir, 1993) establishing the independence of each adjunction with respect to any other adjunction.

After step refinement, we get the Earley-like parsing algorithm for TAG described in (Nederhof, 1997), which preserves the valid prefix property having a time complexity $\mathcal{O}(n^6)$ with respect to the input string. In this schema we also need to define a new kind of intermediate pseudo-items

$$[[N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q]]$$

such that $\delta \stackrel{*}{\Rightarrow} a_i \ldots a_p\ \mathbf{F}^\gamma\ a_{q+1} \ldots a_j \stackrel{*}{\Rightarrow} a_i \ldots a_j$ if and only if $(p,q) \neq (-,-)$ and $\delta \stackrel{*}{\Rightarrow} a_i \ldots a_j$ if and only if $(p,q) = (-,-)$.

Schema 5 *The parsing system* $\mathbb{P}_{\text{Earley}}$ *corresponding to a the final Earley-like parsing algorithm with the valid prefix property having time complexity* $\mathcal{O}(n^6)$*, for a tree adjoining grammar* $\mathcal{G}$ *and a input string* $a_1 \ldots a_n$ *is defined as follows:*

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

such that $N^\gamma \rightarrow \delta \bullet \nu \in \mathcal{P}(\gamma),\ \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq h \leq i \leq j,\ (p,q) \leq (i,j)$

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

such that $N^\gamma \rightarrow \delta \bullet \nu \in \mathcal{P}(\gamma),\ \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq i \leq j,\ (p,q) \leq (i,j)$

$$\mathcal{I}_{\text{Earley}} = \mathcal{I}_{\text{Earley}}^1 \cup \mathcal{I}_{\text{Earley}}^2$$

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

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

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

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

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

such that $\beta \in adj(M^\gamma)$

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

such that $\beta \in adj(M^\gamma)$

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

such that $\beta \in adj(M^\gamma)$, $p \cup p'$ *and* $q \cup q'$ *are defined*

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

such that $\beta \in adj(M^\gamma)$

$$\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^{1'}} = \frac{[[M^\gamma \rightarrow \delta \bullet, j, m \mid p, q]],\ [h, \mathbf{F}^\gamma \rightarrow \bot \bullet, p, q \mid p, q],\ [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid -, -]}{[h, N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, m \mid p, q]}$$

such that $\beta \in adj(M^\gamma)$

$$\mathcal{D}_{\text{Earley}}^{\text{AdjComp}^{2'}} = \frac{[[M^\gamma \rightarrow \delta \bullet, j, m \mid p, q]],\ [h, N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j \mid p, q]}{[h, N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, m \mid p, q]}$$

such that $\beta \in adj(M^\gamma)$

$$\begin{aligned}\mathcal{D}_{\text{Earley}} = \ & \mathcal{D}_{\text{Earley}}^{\text{Init}} \cup \mathcal{D}_{\text{Earley}}^{\text{Scan}} \cup \mathcal{D}_{\text{Earley}}^{\text{Pred}} \cup \\ & \mathcal{D}_{\text{Earley}}^{\text{Comp}} \cup \mathcal{D}_{\text{Earley}}^{\text{AdjPred}} \cup \mathcal{D}_{\text{Earley}}^{\text{FootPred}} \cup \\ & \mathcal{D}_{\text{Earley}}^{\text{FootComp}} \cup \mathcal{D}_{\text{Earley}}^{\text{AdjComp}^0} \cup \\ & \mathcal{D}_{\text{Earley}}^{\text{AdjComp}^{1'}} \cup \mathcal{D}_{\text{Earley}}^{\text{AdjComp}^{2'}}\end{aligned}$$

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

6 Conclusion

We have described a set of parsing algorithms for TAG creating a continuum which has the CYK-like parsing algorithm by (Vijay-Shanker and Joshi, 1985) as its starting point and the Earley-like parsing algorithm by (Nederhof, 1997) preserving the valid prefix property with time

complexity $O(n^6)$ as its goal. As intermediate algorithms, we have defined a bottom-up Earley-like parsing algorithm and an Earley-like parsing algorithm without the valid prefix property, which to our knowledge has not been previously described in literature[1]. We have also shown how to transform one algorithm into the next using simple transformations.Other algorithms could also has been included in the continuum, but for reasons of space we have chosen to show only the algorithms we consider milestones in the development of parsing algorithms for TAG.

An interesting project for the future will be to translate the algorithms presented here to several proposed automata models for TAG which have an associated tabulation technique: Strongly Driven 2–Stack Automata (de la Clergerie and Alonso, 1998), Bottom-up 2–Stack Automata (de la Clergerie et al., 1998) and Linear Indexed Automata (Nederhof, 1998).

7 Acknowledgments

This work has been partially supported by FEDER of European Union (1FD97-0047-C04-02) and Xunta de Galicia (and XUGA20402B97).

References

Eric de la Clergerie and Miguel A. Alonso. 1998. A tabular interpretation of a class of 2-Stack Automata. In *COLING-ACL'98, 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Proceedings of the Conference*, volume II, pages 1333–1339, Montreal, Quebec, Canada, August. ACL.

Eric de la Clergerie, Miguel A. Alonso, and David Cabrero. 1998. A tabular interpretation of bottom-up automata for TAG. In *Proc. of Fourth International Workshop on Tree-Adjoining Grammars and Related Frameworks (TAG+4)*, pages 42–45, Philadelphia, PA, USA, August.

Aravind K. Joshi. 1987. An introduction to tree adjoining grammars. In Alexis Manaster-Ramer, editor, *Mathematics of Language*, pages 87–115. John Benjamins Publishing Co., Amsterdam/Philadelphia.

Mark-Jan Nederhof. 1997. Solving the correct-prefix property for TAGs. In T. Becker and H.-V. Krieger, editors, *Proc. of the Fifth Meeting on Mathematics of Language*, pages 124–130, Schloss Dagstuhl, Saarbruecken, Germany, August.

Mark-Jan Nederhof. 1998. Linear indexed automata and tabulation of TAG parsing. In *Proc. of First Workshop on Tabulation in Parsing and Deduction (TAPD'98)*, pages 1–9, Paris, France, April.

Yves Schabes and Aravind K. Joshi. 1988. An Earley-type parsing algorithm for tree adjoining grammars. In *Proc. of 26th Annual Meeting of the Association for Computational Linguistics*, pages 258–269, Buffalo, NY, USA, June. ACL.

Yves Schabes and Stuart M. Shieber. 1994. An alternative conception of tree-adjoining derivation. *Computational Linguistics*, 20(1):91–124.

Yves Schabes. 1991. The valid prefix property and left to right parsing of tree-adjoining grammar. In *Proc. of II International Workshop on Parsing Technologies, IWPT'91*, pages 21–30, Cancún, Mexico.

Yves Schabes. 1994. Left to right parsing of lexicalized tree-adjoining grammars. *Computational Intelligence*, 10(4):506–515.

Stuart M. Shieber, Yves Schabes, and Fernando C. N. Pereira. 1995. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1&2):3–36, July-August.

Klaas Sikkel. 1997. *Parsing Schemata — A Framework for Specification and Analysis of Parsing Algorithms.* Texts in Theoretical Computer Science — An EATCS Series. Springer-Verlag, Berlin/Heidelberg/New York.

Krishnamurti Vijay-Shanker and Aravind K. Joshi. 1985. Some computational properties of tree adjoining grammars. In *23rd Annual Meeting of the Association for Computational Linguistics*, pages 82–93, Chicago, IL, USA, July. ACL.

Krishnamurti Vijay-Shanker and David J. Weir. 1993. Parsing some constrained grammar formalisms. *Computational Linguistics*, 19(4):591–636.

Krishnamurti Vijay-Shanker, David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *Proc. of the 25th Annual Meeting of the Association for Computational Linguistics*, pages 104–111, Buffalo, NY, USA, June. ACL.

[1] Other different formulations of Earley-like parsing algorithms for TAG has been previously proposed, e.g. (Schabes, 1991).