Procesamiento del Lenguaje Natural, num. 29 (2002), pp.89-96 recibido 22-05-2002; aceptado 01-07-2002

Algoritmos de análisis para gramáticas de inserción de árboles*

Vicente Carrillo y Víctor J. Díaz
Universidad de Sevilla
Avda. Reina Mercedes s/n
41012 Sevilla
{carrillo,vjdiaz}@lsi.us.es

Miguel A. Alonso
Universidad de la Coruña
Campus de Elviña s/n
15071 La Coruña
alonso@udc.es

Resumen: *Tree Insertion Grammar* (TIG) es un compromiso entre *Context Free Grammar* (CFG) y *Tree Adjoining Grammar* (TAG) que puede ser analizada con un coste temporal de $\mathcal{O}(n^3)$. En la literatura, tan sólo han sido descritos dos algoritmos de análisis para TIGs, basados en los ya conocidos CYK y Earley para CFGs. En este trabajo definimos cuatro nuevos analizadores para TIGs: tres son ascendentes y otro ascendente predictivo.
Palabras clave: Análisis sintáctico, gramáticas de inserción de árboles

Abstract: *Tree Insertion Grammar* (TIG) is a compromise between *Context Free Grammar* (CFG) and *Tree Adjoining Grammar* (TAG) that can be parsed in $O(n^3)-$*time*. In the literature, just two parsers for TIGs have been defined, based on the well-known CYK and Earley algorithms for CFGs. In this paper, we define four new parsers for TIGs: three parsers use a bottom-up strategy and the other one uses a predictive bottom-up strategy.
Keywords: Parsing, tree insertion grammars

1. Introducción

Las gramáticas de adjunción de árboles (*Tree Adjoining Grammar*, TAG) (Joshi y Schabes, 1997) constituyen un formalismo naturalmente lexicalizado muy adecuado para la descripción de la sintaxis de los lenguajes naturales. Como contrapartida, el proceso de análisis para este formalismo suavemente sensible al contexto implica mayores costes computacionales que el mismo proceso para las gramáticas independientes del contexto (*Context Free Grammar*, CFG): la complejidad temporal en el caso peor de los analizadores para TAG es de $\mathcal{O}(n^6)$, donde n es la longitud de la cadena de entrada, frente a la complejidad $\mathcal{O}(n^3)$ que presentan los analizadores para CFG. En los últimos años, se han descrito muchas aproximaciones que intentan mejorar las prestaciones de los analizadores para TAG: unas basadas en la compilación de los árboles elementales en autómatas de estados finitos (Evans y Weir, 1998), otras que aplican ciertos filtros a los algoritmos de análisis (Carrillo et al., 2001; Díaz et al., 2002) y otras, como la empleada en este trabajo, basadas en restricciones sobre el formalismo (Schabes y Waters, 1995).

Las gramáticas de inserción de árboles (*Tree Insertion Grammar*, TIG) (Schabes y Waters, 1995) constituyen un compromiso entre CFG y TAG que combina la eficiencia de análisis de las primeras con la fuerte lexicalización de las segundas, ya que, al igual que ocurre con las CFGs, cualquier TIG se puede analizar con un coste temporal de $O(n^3)$ en el peor caso y, por otra parte, al ser las TIGs una subclase de las TAGs, se encuentran naturalmente lexicalizadas. La importancia del formalismo TIG se fundamenta en el hecho de que la mayoría de las gramáticas de adjunción de árboles de amplia cobertura se corresponden en su mayor parte con dicho formalismo. Esta afirmación se puede comprobar en la gramática del inglés XTAG (Doran et al., 1994), donde el 99 % de los árboles y adjunciones posibles son compatibles con el formalismo TIG.

La mayoría de los analizadores para TAG y TIG son extensiones de analizadores bien conocidos para CFG. En la literatura podemos encontrar multitud de analizadores para TAG, algunos usan una estrategia ascendente (Alonso et al., 1999; Carrillo et al., 2001; Díaz et al., 2000; Noord, 1994) y otros utilizan estrategias ascendentes predictivas de manera similar al algoritmo de Earley para CFG (Alonso et al., 1999; Nederhof, 1999;

* Parcialmente financiado por el Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica (TIC2000-0370-C02-01), Ministerio de Ciencia y Tecnología (HP2001-0044) y Xunta de Galicia (PGIDT01PXI10506PN).

Díaz et al., 2002). Sin embargo, tan sólo se han definido dos analizadores para TIG, uno basado en el algoritmo CYK (Schabes y Waters, 1996) y el otro en el algoritmo de Earley (Schabes y Waters, 1995). En este trabajo pretendemos cubrir esa carencia, definiendo un conjunto de cuatro analizadores como núcleo de una red de analizadores para TIG similar a la definida para TAG en (Alonso et al., 1999). Podría pensarse que los analizadores para TIG pueden ser derivados directamente de los analizadores para TAG, dadas las similitudes entre ambos formalismos. Sin embargo, los aspectos que los diferencian son lo suficientemente significativos como para hacer que tal adaptación no sea sencilla de realizar. Como ilustración, podemos considerar la enorme diferencia existente los analizadores de tipo Earley para TAG y el analizador de tipo Earley para TIG definido en (Schabes y Waters, 1995).

El artículo se encuentra estructurado de la siguiente manera. La sección 2 introduce la notación necesaria para el resto del artículo. En la sección 3 se definen tres analizadores para TIG que emplean estrategias ascendentes para llevar a cabo el análisis. En la sección 4 se define un analizador para TIG basado en el algoritmo de Earley. La sección 5 presenta las conclusiones finales.

2. Notación

2.1. Gramáticas de inserción de árboles (TIG)

Una TIG es una 5-tupla $(V_N, V_T, S, \boldsymbol{I}, \boldsymbol{A})$, donde V_N es un conjunto de símbolos no terminales, V_T es un conjunto de símbolos terminales, $S \in V_N$ es el axioma, $\boldsymbol{I}$ es un conjunto finito de *árboles iniciales* finitos y $\boldsymbol{A}$ es un conjunto finito de *árboles auxiliares* finitos. Al conjunto $\boldsymbol{I} \cup \boldsymbol{A}$ se le denomina *árboles elementales*. Nos referiremos a la raíz de un árbol elemental γ como $\mathbf{R}^\gamma$. En cada árbol elemental, los nodos de la frontera se etiquetan con símbolos terminales, la palabra vacía (ε) o símbolos no terminales marcados para sustitución, excepto un nodo en cada árbol auxiliar, cuya etiqueta es la misma que la de la raíz y que se denomina nodo *pie*. Denotaremos como $\mathbf{F}^\beta$ al nodo pie de un árbol auxiliar β. Denominamos *espina* al camino de la raíz al pie de un árbol auxiliar. Usaremos $label(M^\gamma)$ para denotar la etiqueta asociada al nodo M^γ.

Los árboles auxiliares en los cuales todo nodo frontera está a la izquierda (derecha) del nodo pie se denominan *árboles auxiliares izquierdos (derechos)*. El resto de árboles auxiliares se denominan *árboles wrapping*. Usaremos $\boldsymbol{A_L}$ y $\boldsymbol{A_R}$ para denotar los conjuntos de árboles auxiliares izquierdos y derechos, respectivamente.

Una derivación TIG comienza con un árbol inicial cuya raíz está etiquetada por S. Este árbol se extiende repetidamente usando las operaciones de *adjunción* y *sustitución*. La *adjunción* inserta un árbol auxiliar β en el nodo M^γ de un árbol γ que tenga la misma etiqueta que $\mathbf{R}^\beta$. En concreto, M^γ es reemplazado por β y $\mathbf{F}^\beta$ es reemplazado por el subárbol dominado por M^γ. Usaremos $\beta \in adj(M^\gamma)$ para denotar que un árbol $\beta \in \boldsymbol{A}$ puede ser adjuntado en un nodo M^γ, es decir, M^γ es un nodo de adjunción. Si la adjunción no es obligatoria en M^γ entonces $\mathbf{nil} \in adj(M^\gamma)$, donde **nil** es un símbolo vacío. La adjunción de un árbol auxiliar izquierdo (derecho) se denomina *adjunción izquierda (derecha)*. Usaremos $\beta \in \text{ladj}(M^\gamma)$ ($\beta \in \text{radj}(M^\gamma)$) para denotar que $\beta \in \boldsymbol{A_L}$ ($\beta \in \boldsymbol{A_R}$) se puede adjuntar en el nodo M^γ, es decir, M^γ es un nodo de adjunción izquierda (derecha). Si una adjunción izquierda (derecha) no es obligatoria en el nodo M^γ entonces $\mathbf{nil} \in \text{ladj}(M^\gamma)$ ($\mathbf{nil} \in \text{radj}(M^\gamma)$). La *sustitución* es una operación obligatoria y reemplaza un nodo marcado para sustitución M^γ con una copia de un árbol inicial α cuya raíz esté etiquetada igual que M^γ. Usamos $\alpha \in \text{subst}(M^\gamma)$ para indicar que el nodo M^γ puede ser sustituido por el árbol $\alpha \in \boldsymbol{I}$.

TIG no permite: (1) árboles auxiliares *wrapping*, (2) la adjunción de un árbol auxiliar izquierdo (derecho) en la espina de un árbol auxiliar derecho (izquierdo) y (3) la adjunción en los nodos raíz y pie de los árboles auxiliares. Para incrementar los árboles que se pueden generar, TIG permite un número arbitrario de adjunciones simultáneas sobre un mismo nodo. La adjunción simultánea es una operación esencialmente ambigua y provoca la creación de muchos árboles diferentes. Fácilmente se pueden imaginar variantes de TIG donde la adjunción simultánea esté más limitada. Para no incrementar la ambiguedad de la derivación, hemos elegido la variante de TIG presentada en (Schabes y Waters, 1996) que como máximo permite una adjunción izquierda y

otra derecha sobre un nodo. Además, para mantener los árboles que se pueden generar mediante adjunción simultánea, permitiremos la adjunción en los nodos raíz y pie de los árboles auxiliares.

Con objeto de representar los árboles de análisis parciales, definimos una producción $N^\gamma \rightarrow N_1^\gamma \dots N_g^\gamma$ para cada nodo N^γ y sus secuencia ordenada de g hijos $N_1^\gamma \dots N_g^\gamma$ en un árbol elemental. Denotaremos el conjunto de producciones asociado a un árbol elemental γ como $\mathcal{P}(\gamma)$. Por razones técnicas, y salvo en el analizador basado en el algoritmo CYK, consideramos las producciones adicionales $\top \rightarrow \mathbf{R}^\alpha$, $\top \rightarrow \mathbf{R}^\beta$ y $\mathbf{F}^\beta \rightarrow \bot$ para cada árbol inicial α y cada árbol auxiliar β. Para mantener la capacidad generativa de la gramática, se prohíbe la adjunción y sustitución en los nodos $\top$ y $\bot$.

2.2. Esquemas de análisis

Los algoritmos de análisis sintáctico se pueden definir como sistemas deductivos (Sikkel, 1997; Shieber et al., 1995), donde las fórmulas, llamadas ítems, son conjuntos de constituyentes completos o incompletos. Los *esquemas de análisis* fueron introducidos en (Sikkel, 1997) como un método de alto nivel para la descripción de algoritmos de análisis. Un sistema de análisis abstrae los detalles de implementación, como las estructuras de control y datos.

Formalmente, un *sistema de análisis* para una gramática G y una cadena $a_1 \dots a_n$ es un triple $< \mathcal{I}, \mathcal{H}, \mathcal{D} >$, donde $\mathcal{I}$ es un conjunto de *ítems* que representa resultados de análisis intermedios, $\mathcal{H}$ es el conjunto inicial de ítems llamado *hipótesis* que codifica la frase que se va a analizar, y $\mathcal{D}$ un conjunto de *pasos deductivos* que permite que se generen nuevos ítems a partir de los existentes. Los pasos deductivos son de la forma $\frac{\eta_1,\dots,\eta_k}{\xi}\ cond$, que significa que si están presentes todos los antecedentes η_i de un paso deductivo y se satisface la condición *cond*, entonces el consecuente ξ debería ser generado por el analizador. El conjunto $\mathcal{F} \subseteq \mathcal{I}$ de *ítems finales* representa el reconocimiento de una frase. Un *esquema de análisis* es un sistema de análisis parametrizado para una gramática y una frase.

Los esquemas de análisis nos permiten establecer relaciones entre dos analizadores de manera formal. Un esquema se puede generalizar desde otro por medio de *refinamiento de ítems*, dividiendo un ítem en múltiples ítems, *refinamiento de pasos*, descomponiendo un paso deductivo en una secuencia de pasos, y considerando una clase de gramáticas mayor (*extensión*).

Para disminuir el número de ítems y pasos deductivos en un esquema de análisis, podemos aplicar los siguientes tipos de filtros: *filtro estático*, el cual simplemente elimina las partes redundantes, *filtro dinámico*, que consiste en usar información contextual para determinar la validez de los ítems, y *contracción de pasos*, mediante el cual una secuencia de pasos deductivos se reemplaza por uno solo. En particular, los filtros son relaciones muy interesantes porque se pueden usar para mejorar las prestaciones de los analizadores en casos prácticos.

3. Análisis ascendente

En esta sección presentamos tres nuevos analizadores que aplican estrategias ascendentes. El primero está basado en el conocido algoritmo CYK para CFG y se trata de una adaptación del presentado en forma algorítmica en (Schabes y Waters, 1996). El interés de este algoritmo en casos reales puede no ser excesivo, ya que obliga a que los árboles de las gramáticas se encuentren binarizados, aunque es un buen punto de partida para describir el resto de analizadores.

El siguiente esquema que presentamos es una generalización del CYK, concretamente, se trata de una extensión a una clase de gramáticas mayor con objeto de evitar la restricciones impuestas por el algoritmo CYK.

El tercer esquema es un filtro dinámico del segundo esquema. Este esquema adapta al formalismo TIG la conocida relación de *esquina izquierda* para CFG, con objeto de eliminar ítems innecesarios en el proceso de análisis y, por consiguiente, aumentar las prestaciones en casos prácticos.

3.1. El esquema *CYK*

Definimos a continuación el sistema de análisis sintáctico

$$\mathbb{P}_{\text{CYK}} =< \mathcal{I}_{\text{CYK}}, \mathcal{H}_{\text{CYK}}, \mathcal{D}_{\text{CYK}} >$$

para una gramática TIG arbitraria $G \in TIG$ y una cadena de entrada $a_1 ... a_n$ con $n \geq 0$.

El conjunto de ítems válidos del sistema viene dado por:

$$\mathcal{I}_{\text{CYK}} = \{[M^\gamma, i, j, code]\}$$

donde $M^\gamma \in V_N$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$ y $0 \leq i \leq j$.

El parámetro *code* es el que controla que como máximo se efectúe una adjunción izquierda y otra derecha sobre el nodo M^γ, y puede tomar los siguientes valores:

- $code = \emptyset$ si no se ha completado ninguna adjunción en el nodo M^γ;
- $code = \{L\}$ si se ha completado una adjunción de un árbol auxiliar izquierdo en el nodo M^γ;
- $code = \{R\}$ si se ha completado una adjunción de un árbol auxiliar derecho en el nodo M^γ;
- $code = \{L, R\}$ si se ha completado las adjunciones de un árbol auxiliar izquierdo y otro derecho en el nodo M^γ.

El conjunto de hipótesis es:

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

Este conjunto es el estándar y, por tanto, será el mismo para el resto de esquemas que veremos en este trabajo.

Los pasos deductivos del sistema son:

$$\mathcal{D}_{\text{CYK}} = \mathcal{D}^{\text{Sc}}_{\text{CYK}} \cup \mathcal{D}^{\epsilon}_{\text{CYK}} \cup \mathcal{D}^{\text{Foot}}_{\text{CYK}} \cup \mathcal{D}^{\text{Cmp}_1}_{\text{CYK}} \cup \mathcal{D}^{\text{Cmp}_2}_{\text{CYK}} \cup \mathcal{D}^{\text{LAdj}}_{\text{CYK}} \cup \mathcal{D}^{\text{Radj}}_{\text{CYK}} \cup \mathcal{D}^{\text{Subs}}_{\text{CYK}}$$

$$\mathcal{D}^{\text{Sc}}_{\text{CYK}} = \frac{[a, j, j+1]}{[N^\gamma, j, j+1, \emptyset]} \quad label(N^\gamma) = a$$

$$\mathcal{D}^{\epsilon}_{\text{CYK}} = \frac{}{[N^\gamma, j, j, \emptyset]} \quad label(N^\gamma) = \epsilon$$

$$\mathcal{D}^{\text{Foot}}_{\text{CYK}} = \frac{}{[\mathbf{F}^\beta, j, j, \emptyset]} \quad \beta \in \boldsymbol{A}$$

$$\mathcal{D}^{\text{Cmp}_1}_{\text{CYK}} = \frac{[O^\gamma, i, j, code]}{[M^\gamma, i, j, \emptyset]} \quad M^\gamma \to O^\gamma \in \mathcal{P}(\gamma)$$

$$\mathcal{D}^{\text{Cmp}_2}_{\text{CYK}} = \frac{\begin{array}{c}[O^\gamma_1, i, j, code] \\ [O^\gamma_2, j, k, code']\end{array}}{[M^\gamma, i, k, \emptyset]} \quad M^\gamma \to O^\gamma_1 O^\gamma_2 \in \mathcal{P}(\gamma)$$

$$\mathcal{D}^{\text{LAdj}}_{\text{CYK}} = \frac{\begin{array}{c}[\mathbf{R}^\beta, i, j, code'] \\ [M^\gamma, j, k, code]\end{array}}{[M^\gamma, i, k, \{L\} \cup code]} \quad \begin{array}{l}\beta \in \text{ladj}(M^\gamma) \\ L \notin code\end{array}$$

$$\mathcal{D}^{\text{RAdj}}_{\text{CYK}} = \frac{\begin{array}{c}[\mathbf{R}^\beta, j, k, code'] \\ [M^\gamma, i, j, code]\end{array}}{[M^\gamma, i, k, \{R\} \cup code]} \quad \begin{array}{l}\beta \in \text{radj}(M^\gamma) \\ R \notin code\end{array}$$

$$\mathcal{D}^{\text{Subs}}_{\text{CYK}} = \frac{[\mathbf{R}^\alpha, i, j, \emptyset]}{[M^\gamma, i, j, \emptyset]} \quad \alpha \in \text{subst}(M^\gamma)$$

El conjunto de ítems finales, dado $\alpha \in \boldsymbol{I}$ con $label(\mathbf{R}^\alpha) = S$, se define como:

$$\mathcal{F}_{\text{CYK}} = \{[\mathbf{R}^\alpha, 0, n, code]\}$$

Los pasos deductivos $\mathcal{D}^{\text{Sc}}_{\text{CYK}}$, $\mathcal{D}^{\epsilon}_{\text{CYK}}$ y $\mathcal{D}^{\text{Foot}}_{\text{CYK}}$ son los que inician el reconocimiento ascendente. Una vez reconocido el subárbol dominado por un nodo, los pasos $\mathcal{D}^{\text{Cmp}_1}_{\text{CYK}}$ y $\mathcal{D}^{\text{Cmp}_2}_{\text{CYK}}$ permiten continuar el reconocimiento ascendente. Cuando se ha reconocido un árbol auxiliar izquierdo (derecho), el paso $\mathcal{D}^{\text{LAdj}}_{\text{CYK}}$ ($\mathcal{D}^{\text{RAdj}}_{\text{CYK}}$) efectúa la adjunción en un nodo de adjunción izquierda (derecha), siempre que el reconocimiento lo haya alcanzado y no haya sido ya adjuntado por la izquierda (derecha). La operación $\mathcal{D}^{\text{Subs}}_{\text{CYK}}$ sustituye un árbol inicial que ha sido reconocido en un nodo de sustitución.

3.2. El esquema *bottom-up Earley*

En esta sección definimos el sistema de análisis sintáctico

$$\mathbb{P}_{\text{buE}} = < \mathcal{I}_{\text{buE}}, \mathcal{H}_{\text{CYK}}, \mathcal{D}_{\text{buE}} >$$

para una gramática TIG arbitraria $G \in TIG$ y una cadena de entrada $a_1 ... a_n$ con $n \geq 0$

El conjunto de ítems válidos del sistema viene dado por $\mathcal{I}_{\text{buE}} = \mathcal{I}^{(\text{i})}_{\text{buE}} \cup \mathcal{I}^{(\text{ii})}_{\text{buE}}$:

$$\mathcal{I}^{(\text{i})}_{\text{buE}} = \{[M^\gamma \to \delta \bullet \nu, i, j, code]\}$$

tal que $M^\gamma \to \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $\nu \neq \epsilon$ y $code = \emptyset$ si no se ha completado ninguna adjunción en el nodo M^γ y $code = \{L\}$ si se ha completado una adjunción de un árbol auxiliar izquierdo en el nodo M^γ. Por otra parte,

$$\mathcal{I}^{(\text{ii})}_{\text{buE}} = \{[M^\gamma \to \nu \bullet, i, j, code]\}$$

tal que $M^\gamma \to \nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $code = \emptyset$ si no se completó ninguna adjunción sobre M^γ, $code = \{L\}$ si se completó una adjunción izquierda sobre M^γ, $code = \{R\}$ si se completó una adjunción derecha sobre M^γ y $code = \{L, R\}$ si se completó una adjunción izquierda y otra derecha sobre M^γ.

Los pasos deductivos del esquema son:

$$\mathcal{D}_{\text{buE}} = \mathcal{D}^{\text{Ini}}_{\text{buE}} \cup \mathcal{D}^{\text{Sc}}_{\text{buE}} \cup \mathcal{D}^{\epsilon}_{\text{buE}} \cup \mathcal{D}^{\text{Foot}}_{\text{buE}} \cup \mathcal{D}^{\text{Cmp}}_{\text{buE}} \cup \mathcal{D}^{\text{LAdj}}_{\text{buE}} \cup \mathcal{D}^{\text{Radj}}_{\text{buE}} \cup \mathcal{D}^{\text{Subs}}_{\text{buE}}$$

$$\mathcal{D}^{\text{Ini}}_{\text{buE}} = \frac{}{[N^\gamma \to \bullet\nu, i, i, \emptyset]} \quad \gamma \in \boldsymbol{I} \cup \boldsymbol{A}$$

$$\mathcal{D}^{\text{Sc}}_{\text{buE}} = \frac{[a, j, j+1] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, code]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j+1, code]}$$

tal que $label(M^\gamma) = a$

$$\mathcal{D}^{\epsilon}_{\text{buE}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, code]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j, code]}$$

tal que $label(M^\gamma) = \epsilon$

$$\mathcal{D}^{\text{Foot}}_{\text{buE}} = \frac{}{[\mathbf{F}^\beta \rightarrow \bot \bullet, j, j, \emptyset]} \quad \beta \in \boldsymbol{A}$$

$$\mathcal{D}^{\text{Cmp}}_{\text{buE}} = \frac{[M^\gamma \rightarrow \nu \bullet, j, k, code] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, code']}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, k, code']}$$

donde por una parte (**nil** $\in ladj(M^\gamma)$ y $L \notin code$) o ($\beta \in ladj(M^\gamma)$ y $L \in code$), mientras que por otra parte (**nil** $\in radj(M^\gamma)$ y $R \notin code$) o ($\beta \in radj(M^\gamma)$ y $R \in code$).

$$\mathcal{D}^{\text{LAdj}}_{\text{buE}} = \frac{[\top \rightarrow \mathbf{R}^\beta \bullet, i, j, \emptyset] \quad [M^\gamma \rightarrow \bullet \nu, i, i, \emptyset]}{[M^\gamma \rightarrow \bullet \nu, i, j, \{L\}]} \quad \beta \in \text{ladj}(M^\gamma)$$

$$\mathcal{D}^{\text{RAdj}}_{\text{buE}} = \frac{[\top \rightarrow \mathbf{R}^\beta \bullet, j, k, \emptyset] \quad [M^\gamma \rightarrow \nu \bullet, i, j, code]}{[M^\gamma \rightarrow \nu \bullet, i, k, \{R\} \cup code]}$$

tal que $\beta \in \text{radj}(M^\gamma)$ y $R \notin code$

$$\mathcal{D}^{\text{Subs}}_{\text{buE}} = \frac{[\top \rightarrow \mathbf{R}^\alpha \bullet, j, k, \emptyset] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, code]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, k, code]}$$

tal que $\alpha \in \text{subst}(M^\gamma)$.

El conjunto de ítems finales es:

$$\mathcal{F}_{\text{buE}} = \{\ [\top \rightarrow \mathbf{R}^\alpha \bullet, 0, n, \emptyset]\ \}$$

tal que $\alpha \in \boldsymbol{I}$ y $S = label(\mathbf{R}^\alpha)$.

El paso $\mathcal{D}^{\text{Ini}}_{\text{buE}}$ inicia el reconocimiento desde todos los subárboles de los árboles elementales. El paso $\mathcal{D}^{\text{Sc}}_{\text{buE}}$ reconoce la presencia de un símbolo terminal en la cadena de entrada, mientras que $\mathcal{D}^{\epsilon}_{\text{buE}}$ y $\mathcal{D}^{\text{Foot}}_{\text{buE}}$ reflejan el hecho de que se puede saltar sobre nodos etiquetados con ϵ y nodos pie sin tener que reconocer nada. El paso $\mathcal{D}^{\text{Cmp}}_{\text{buE}}$ continúa el reconocimiento ascendente cuando se ha completado el reconocimiento de un subárbol. El resto de pasos funcionan de forma análoga a los del mismo nombre del esquema anterior.

3.3. El esquema *bottom-up left corner*

Al igual que el esquema **buLC** para TAGs (Carrillo et al., 2001), este esquema elimina del dominio de **buE** aquellos ítems que no aportan nada significativo en el proceso de análisis, y que son aquellos de la forma:

$$[M^\gamma \rightarrow \bullet \nu, i, i, code]$$

Al posibilitar el esquema **buE** este tipo de ítems, se provoca un aumento en el número de ítems deducidos y, por tanto, una merma en el comportamiento práctico del analizador. Por ello proponemos modificar el dominio y los pasos deductivos de dicho esquema para evitar que se generen este tipo de ítems, obteniendo como resultado el esquema **buLC** que describimos a continuación y cuyo comportamiento en casos prácticos mejora a **buE**.

El sistema ascendente con filtro *left corner* se define como

$$\mathbb{P}_{\text{buLC}} =< \mathcal{I}_{\text{buLC}}, \mathcal{H}_{\text{CYK}}, \mathcal{D}_{\text{buLC}} >$$

para una gramática TIG arbitraria $G \in TIG$ y una cadena de entrada $a_1 ... a_n$ con $n \geq 0$.

El conjunto de ítems válidos del sistema es $\mathcal{I}_{\text{buLC}} = \mathcal{I}^{(\text{i})}_{\text{buLC}} \cup \mathcal{I}^{(\text{ii})}_{\text{buLC}} \cup \mathcal{I}^{(\text{iii})}_{\text{buLC}}$:

$$\mathcal{I}^{(\text{i})}_{\text{buLC}} = \{[M^\gamma \rightarrow \delta \bullet \nu, i, j, code]\}$$

tal que $M^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $\delta \neq \epsilon$, $\nu \neq \epsilon$ y $code = \emptyset$ si no se ha completado ninguna adjunción en el nodo M^γ y $code = \{L\}$ si se ha completado una adjunción de un árbol auxiliar izquierdo en el nodo M^γ.

$$\mathcal{I}^{(\text{ii})}_{\text{buLC}} = \{[M^\gamma \rightarrow \bullet \nu, i, j, code]\}$$

tal que $M^\gamma \rightarrow \nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $\nu \neq \epsilon$ y $code$ puede tomar uno de los siguientes valores:

- $code = \emptyset$ si no se ha completado ninguna adjunción en el nodo M^γ pero $\exists \beta \in \text{ladj}(M^\gamma)$:
- $code = \{L\}$ si se ha completado una adjunción de un árbol auxiliar izquierdo en el nodo M^γ.

$$\mathcal{I}^{(\text{iii})}_{\text{buLC}} = \{[M^\gamma \rightarrow \nu \bullet, i, j, code]\}$$

tal que $M^\gamma \to \nu \in \mathcal{P}(\gamma), \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq i \leq j$, $code = \emptyset$ si no se completó ninguna adjunción sobre M^γ, $code = \{L\}$ si se completó una adjunción izquierda sobre M^γ, $code = \{R\}$ si se completó una adjunción derecha sobre M^γ y $code = \{L, R\}$ si se completó una adjunción izquierda y otra derecha sobre M^γ.

Los pasos deductivos del esquema son:

$$\begin{aligned}\mathcal{D}_{\text{buLC}} = \quad & \mathcal{D}^{\text{LC}_\text{t}}_{\text{buLC}} \cup \mathcal{D}^{\text{LC}_\epsilon}_{\text{buLC}} \cup \mathcal{D}^{\text{LC}_\text{n}}_{\text{buLC}} \cup \mathcal{D}^{\text{LC}_{\text{subs}}}_{\text{buLC}} \cup \\ & \mathcal{D}^{\text{LAdj}_{\text{subs}}}_{\text{buLC}} \cup \mathcal{D}^{\text{LAdj}_\text{t}}_{\text{buLC}} \cup \mathcal{D}^{\text{LAdj}_\epsilon}_{\text{buLC}} \cup \\ & \mathcal{D}^{\text{LAdj}_\text{n}}_{\text{buLC}} \cup \mathcal{D}^{\text{Sc}}_{\text{buLC}} \cup \mathcal{D}^{\epsilon}_{\text{buLC}} \cup \mathcal{D}^{\text{Foot}}_{\text{buLC}} \cup \\ & \mathcal{D}^{\text{Cmp}}_{\text{buLC}} \cup \mathcal{D}^{\text{Radj}}_{\text{buLC}} \cup \mathcal{D}^{\text{Subs}}_{\text{buLC}}\end{aligned}$$

$$\mathcal{D}^{\text{LC}_\text{t}}_{\text{buLC}} = \frac{[a, j, j+1]}{[O^\gamma \to M^\gamma \bullet \nu, j, j+1, \emptyset]}$$

tal que $label(M^\gamma) = a$ y $\mathbf{nil} \in adj(O^\gamma)$

$$\mathcal{D}^{\text{LC}_\epsilon}_{\text{buLC}} = \frac{}{[O^\gamma \to M^\gamma \bullet \nu, j, j, \emptyset]}$$

tal que $label(M^\gamma) = \epsilon$ y $\mathbf{nil} \in \text{ladj}(O^\gamma)$

$$\mathcal{D}^{\text{LC}_{\text{subs}}}_{\text{buLC}} = \frac{[\top \to \mathbf{R}^\alpha \bullet, j, k, \emptyset]}{[O^\gamma \to M^\gamma \bullet \nu, j, k, \emptyset]}$$

tal que $\alpha \in \text{subst}(M^\gamma)$ y $\mathbf{nil} \in \text{ladj}(O^\gamma)$

$$\mathcal{D}^{\text{LC}_\text{n}}_{\text{buLC}} = \frac{[O^\gamma \to \delta \bullet, j, k, code]}{[Q^\gamma \to O^\gamma \bullet \nu, j, k, \emptyset]}$$

tal que $\mathbf{nil} \in \text{ladj}(Q^\gamma)$

$$\mathcal{D}^{\text{LAdj}_\text{t}}_{\text{buLC}} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\beta \bullet, i, j, \emptyset] \\ [a, j, j+1]\end{array}}{[Q^\gamma \to M^\gamma \bullet \nu, i, j+1, \{L\}]}$$

tal que $label(M^\gamma) = a$ y $\beta \in \text{ladj}(Q^\gamma)$

$$\mathcal{D}^{\text{LAdj}_\epsilon}_{\text{buLC}} = \frac{[\top \to \mathbf{R}^\beta \bullet, i, j, \emptyset]}{[Q^\gamma \to M^\gamma \bullet \nu, i, j, \{L\}]}$$

tal que $label(M^\gamma) = \epsilon$ y $\beta \in \text{ladj}(Q^\gamma)$

$$\mathcal{D}^{\text{LAdj}_{\text{subs}}}_{\text{buLC}} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\beta \bullet, i, j, \emptyset] \\ [\top \to \mathbf{R}^\alpha \bullet, j, k, \emptyset]\end{array}}{[Q^\gamma \to M^\gamma \bullet \nu, i, k, \emptyset]}$$

tal que $\alpha \in \text{subst}(M^\gamma)$ y $\beta \in \text{ladj}(Q^\gamma)$

$$\mathcal{D}^{\text{LAdj}_\text{n}}_{\text{buLC}} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\beta \bullet, i, j, \emptyset] \\ [O^\gamma \to \delta \bullet, j, k, code]\end{array}}{[Q^\gamma \to O^\gamma \bullet \nu, i, k, \{L\}]}$$

tal que $\beta \in \text{ladj}(Q^\gamma)$ y por una parte ($\mathbf{nil} \in ladj(M^\gamma)$ y $L \notin code$) o ($\beta \in ladj(M^\gamma)$ y $L \in code$), mientras que por otra parte ($\mathbf{nil} \in radj(M^\gamma)$ y $R \notin code$) o ($\beta \in radj(M^\gamma)$ y $R \in code$).

$$\mathcal{D}^{\text{Sc}}_{\text{buLC}} = \frac{\begin{array}{c}[a, j, j+1] \\ [N^\gamma \to P^\gamma \delta \bullet M^\gamma \nu, i, j, code]\end{array}}{[N^\gamma \to P^\gamma \delta M^\gamma \bullet \nu, i, j+1, code]}$$

tal que $label(M^\gamma) = a$

$$\mathcal{D}^{\epsilon}_{\text{buLC}} = \frac{[N^\gamma \to P^\gamma \delta \bullet M^\gamma \nu, i, j, code]}{[N^\gamma \to P^\gamma \delta M^\gamma \bullet \nu, i, j, code]}$$

tal que $label(M^\gamma) = \epsilon$

$$\mathcal{D}^{\text{Foot}}_{\text{buLC}} = \frac{}{[\mathbf{F}^\beta \to \bot \bullet, j, j, \emptyset]} \quad \beta \in \boldsymbol{A}$$

$$\mathcal{D}^{\text{Cmp}}_{\text{buLC}} = \frac{\begin{array}{c}[M^\gamma \to \nu \bullet, j, k, code] \\ [N^\gamma \to P^\gamma \delta \bullet M^\gamma \nu, i, j, code']\end{array}}{[N^\gamma \to P^\gamma \delta M^\gamma \bullet \nu, i, k, code']}$$

donde por una parte ($\mathbf{nil} \in ladj(M^\gamma)$ y $L \notin code$) o ($\beta \in ladj(M^\gamma)$ y $L \in code$), mientras que por otra parte ($\mathbf{nil} \in radj(M^\gamma)$ y $R \notin code$) o ($\beta \in radj(M^\gamma)$ y $R \in code$).

$$\mathcal{D}^{\text{RAdj}}_{\text{buLC}} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\beta \bullet, j, k, \emptyset] \\ [M^\gamma \to \nu \bullet, i, j, code]\end{array}}{[M^\gamma \to \nu \bullet, i, k, \{R\} \cup code]}$$

tal que $\beta \in \text{radj}(M^\gamma)$ y $R \notin code$

$$\mathcal{D}^{\text{Subs}}_{\text{buLC}} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\alpha \bullet, j, k, \emptyset] \\ [N^\gamma \to P^\gamma \delta \bullet M^\gamma \nu, i, j, code]\end{array}}{[N^\gamma \to P^\gamma \delta M^\gamma \bullet \nu, i, k, code]}$$

tal que $\alpha \in \text{subst}(M^\gamma)$.

El conjunto de ítems finales es:

$$\mathcal{F}_{\text{buLC}} = \{\ [\top \to \mathbf{R}^\alpha \bullet, 0, n, \emptyset]\ \}$$

tal que $\alpha \in \boldsymbol{I}$ y $S = label(\mathbf{R}^\alpha)$.

La eliminación del dominio de un determinado tipo de ítems provoca que tengamos que reescribir el paso $\mathcal{D}^{\text{Ini}}_{\text{buE}}$, obteniendo los pasos $\mathcal{D}^{\text{LC}_\text{t}}_{\text{buLC}}$, $\mathcal{D}^{\text{LC}_\epsilon}_{\text{buLC}}$, $\mathcal{D}^{\text{LC}_{\text{subs}}}_{\text{buLC}}$ y $\mathcal{D}^{\text{LC}_\text{n}}_{\text{buLC}}$, que se aplican cuando el símbolo que está a la izquierda de la regla (la esquina izquierda) es un terminal, la cadena vacía, un nodo de sustitución o un no terminal, respectivamente. Actuamos de forma análoga con la regla $\mathcal{D}^{\text{LAdj}}_{\text{buE}}$ para obtener las cuatro reglas de adjunción izquierda de este esquema. El resto de pasos funcionan igual que en el esquema anterior.

4. Análisis ascendente predictivo

Aquí presentamos un algoritmo eficiente de análisis *left-to-right* para TIG que combina predicciones descendentes, como el algoritmo de Earley para CFG, con reconocimiento ascendente. Este algoritmo satisface la propiedad del prefijo válido (Nederhof, 1999).

A continuación definimos el sistema de análisis

$$\mathbb{IP}_{\text{Earley}} =< \mathcal{I}_{\text{Earley}}, \mathcal{H}_{\text{CYK}}, \mathcal{D}_{\text{Earley}} >$$

para una gramática TIG arbitraria $G \in TIG$ y una cadena de entrada $a_1...a_n$ con $n \geq 0$.

El conjunto de ítems válidos es $\mathcal{I}_{\text{Earley}} = \mathcal{I}^{(\text{i})}_{\text{Earley}} \cup \mathcal{I}^{(\text{ii})}_{\text{Earley}}$:

$$\mathcal{I}^{(\text{i})}_{\text{Earley}} = \{[M^\gamma \rightarrow \delta \bullet \nu, i, j, code]\}$$

tal que $M^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $\nu \neq \epsilon$ and $code = \emptyset$ si no se completó ninguna adjunción sobre M^γ y $code = \{L\}$ si se completó una adjunción izquierda sobre M^γ.

$$\mathcal{I}^{(\text{ii})}_{\text{Earley}} = \{[M^\gamma \rightarrow \nu\bullet, i, j, code]\}$$

tal que $M^\gamma \rightarrow \nu \in \mathcal{P}(\gamma), \gamma \in \boldsymbol{I} \cup \boldsymbol{A}, 0 \leq i \leq j$, $code = \emptyset$ si no se completó ninguna adjunción sobre M^γ, $code = \{L\}$ si se completó una adjunción izquierda sobre M^γ, $code = \{R\}$ si se completó una adjunción derecha sobre M^γ y $code = \{L, R\}$ si se completó una adjunción izquierda y otra derecha sobre M^γ.

Para el conjunto de pasos deductivos, definimos subconjuntos *inicio*, *scan*, *predicción* y *compleción* similares al analizador Earley para CFG. El conjunto $\mathcal{D}_{\text{Earley}}$ se define mediante:

$$\begin{aligned}\mathcal{D}_{\text{Earley}} = \quad & \mathcal{D}^{\text{Ini}}_{\text{Earley}} \cup \mathcal{D}^{\text{Sc}}_{\text{Earley}} \cup \mathcal{D}^{\epsilon}_{\text{Earley}} \cup \\ & \mathcal{D}^{\text{Cmp}}_{\text{Earley}} \cup \mathcal{D}^{\text{Foot}}_{\text{Earley}} \cup \mathcal{D}^{\text{LAdjPred}}_{\text{Earley}} \cup \\ & \mathcal{D}^{\text{LAdjCmp}}_{\text{Earley}} \cup \mathcal{D}^{\text{RadjPred}}_{\text{Earley}} \cup \mathcal{D}^{\text{Pred}}_{\text{Earley}} \cup \\ & \mathcal{D}^{\text{RAdjCmp}}_{\text{Earley}} \cup \mathcal{D}^{\text{SubsPred}}_{\text{Earley}} \cup \mathcal{D}^{\text{SubsCmp}}_{\text{Earley}}\end{aligned}$$

El reconocimiento comienza con la predicción de todo árbol inicial $\alpha \in \boldsymbol{I}$ cuya raíz esté etiquetada con el axioma ($label(\mathbf{R}^\alpha) = S$):

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

El conjunto de pasos de *scan* son los tres siguientes:

$$\mathcal{D}^{\text{Sc}}_{\text{Earley}} = \frac{[a, j, j+1] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j, code]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j+1, code]}$$

siempre que $label(M^\gamma) = a$

$$\mathcal{D}^{\epsilon}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j, code]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j, code]}$$

siempre que $label(M^\gamma) = \epsilon$

$$\mathcal{D}^{\text{Foot}}_{\text{Earley}} = \frac{[\mathbf{F}^\beta \rightarrow \bullet\bot, j, j, \emptyset]}{[\mathbf{F}^\beta \rightarrow \bot\bullet, j, j, \emptyset]} \quad \beta \in \boldsymbol{A}$$

En el caso de TIG, tenemos cuatro tipos de predicciones con sus pasos de compleción asociados: subárbol, adjunción izquierda, adjunción derecha y sustitución.

$$\mathcal{D}^{\text{Pred}}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j, code]}{[M^\gamma \rightarrow \bullet\omega, j, j, \emptyset]}$$

tal que $\mathbf{nil} \in \text{ladj}(M^\gamma)$

$$\mathcal{D}^{\text{Cmp}}_{\text{Earley}} = \frac{[M^\gamma \rightarrow \omega\bullet, j, k, code] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j, code']}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, k, code']}$$

donde por una parte ($\mathbf{nil} \in ladj(M^\gamma)$ y $L \notin code$) o ($\beta \in ladj(M^\gamma)$ y $L \in code$), mientras que por otra parte ($\mathbf{nil} \in radj(M^\gamma)$ y $R \notin code$) o ($\beta \in radj(M^\gamma)$ y $R \in code$).

$$\mathcal{D}^{\text{LAdjPred}}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j, code]}{[\top \rightarrow \bullet\mathbf{R}^\beta, j, j, \emptyset]}$$

tal que $\beta \in \text{ladj}(M^\gamma)$

$$\mathcal{D}^{\text{LAdjCmp}}_{\text{Earley}} = \frac{[\top \rightarrow \mathbf{R}^\beta\bullet, j, k, \emptyset] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j, code]}{[M^\gamma \rightarrow \bullet\omega, j, k, \{L\}]}$$

tal que $\beta \in \text{ladj}(M^\gamma)$

$$\mathcal{D}^{\text{RAdjPred}}_{\text{Earley}} = \frac{[M^\gamma \rightarrow \nu\bullet, i, j, code]}{[\top \rightarrow \bullet\mathbf{R}^\beta, j, j, \emptyset]}$$

tal que $\beta \in \text{radj}(M^\gamma)$ y $R \notin code$

$$\mathcal{D}^{\text{RAdjCmp}}_{\text{Earley}} = \frac{[\top \rightarrow \mathbf{R}^\beta\bullet, j, k, \emptyset] \quad [M^\gamma \rightarrow \nu\bullet, i, j, code]}{[M^\gamma \rightarrow \nu\bullet, i, k, \{R\} \cup code]}$$

tal que $\beta \in \text{radj}(M^\gamma)$ y $R \notin code$

$$\mathcal{D}^{\text{SubsPred}}_{\text{Earley}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, code]}{[\top \rightarrow \bullet \mathbf{R}^\alpha, j, j, \emptyset]}$$

tal que $\alpha \in subst(M^\gamma)$

$$\mathcal{D}^{\text{SubsCmp}}_{\text{Earley}} = \frac{[\top \rightarrow \mathbf{R}^\alpha \bullet, j, k, \emptyset] \quad [N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, code]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, k, code]}$$

tal que $\alpha \in subst(M^\gamma)$.

El conjunto de ítems finales se define mediante:

$$\mathcal{F}_{\text{Earley}} = \{\ [\top \rightarrow \mathbf{R}^\alpha \bullet, 0, n, \emptyset]\ \}$$

tal que $\alpha \in \boldsymbol{I}$ y $S = label(\mathbf{R}^\alpha)$.

5. Conclusiones

Hemos descrito cuatro nuevos algoritmos para análisis sintáctico de TIG que constituyen una red de analizadores básica para dicho formalismo. Comenzamos con un algoritmo de tipo CYK y concluimos con un algoritmo de tipo Earley. Como analizadores intermedios definimos dos algoritmos: uno tipo Earley ascendente, que elimina la restricción que presenta el CYK respecto a la forma de la gramática, y otro que aplica un filtro basado en la relación de esquina izquierda (muy conocida en CFG) al algoritmo tipo Earley ascendente, con objeto de mejorar las prestaciones en casos prácticos.

Actualmente, con el fin de extender la red de analizadores descrita en este artículo, trabajamos en la definición de un algoritmo *left corner* predictivo para TIG.

Bibliografía

Alonso, M. A., D. Cabrero, E. de la Clergerie, y M Vilares. 1999. Tabular algorithms for TAG parsing. En *Proc. of EACL'99*, páginas 150–157, Bergen, Noruega, Junio. ACL.

Carrillo, V., V. Díaz, y M. A. Alonso 2001. Análisis sintáctico ascendente de TAGs guiado por la esquina izquierda. *Procesamiento del Lenguaje Natural*, 27:47–54.

Díaz, V., V. Carrillo, y M. A. Alonso 2000. A bidirectional bottom-up parser for TAG. En *Proc. of IWPT 2000*, páginas 299–300, Trento, Italia.

Díaz, V., V. Carrillo, y M. A. Alonso 2002. A left-corner parser for tree adjoining grammars. En *Proc. of TAG+6*, Venecia, Italia.

Doran, C., D. Egedi, B. Hockey, B. Srinivas, y M. Zaidel. 1994. XTAG system — a wide coverage grammar for English. En *Proc. of COLING'94*, páginas 922–928, Kyoto, Japón.

Evans, R. y D. Weir. 1998. A structure-sharing parser for lexicalized grammars. En *Proc. of COLING-ACL'98*, volumen I, páginas 372–378, Montreal, Canadá.

Joshi, A. K. y Y. Schabes. 1997. Tree adjoining grammars. En G. Rozenberg y A. Salomaa, editores, *Handbook of Formal Languages. Vol 3: Beyond Words*. Springer-Verlag, Berlín/Heidelberg/Nueva York, páginas 69–123.

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

van Noord, G. 1994. Head corner parsing for TAG. *Computational Intelligence*, 10(4):525–534.

Schabes, Y. y R. C. Waters. 1995. Tree insertion grammar: A cubic-time parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21(4):479–513.

Schabes, Y. y R. C. Waters. 1996. Sthocastic lexicalized tree insertion grammar. En H. Bunt y M. Tomita, editores, *Recent advances in parsing technologies*. Kluwer Academic Publishers, Dordrecht/Boston/Londres, páginas 281–294.

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

Sikkel, K. 1997. *Parsing Schemata — A Framework for Specification and Analysis of Parsing Algorithms*. Springer-Verlag, Berlín/Heidelberg/Nueva York.