Procesamiento del Lenguaje Natural, num. 29 (2002), pp.65-72 recibido 08-05-2002; aceptado 01-07-2002

Análisis sintáctico combinado de gramáticas de adjunción de árboles y de gramáticas de inserción de árboles*

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

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

Resumen: La operación de adjunción es el mecanismo que hace de las Gramáticas de Adjunción de Árboles (TAG) un formalismo adecuado para la descripción de la estructura sintáctica de los lenguajes naturales. Sin embargo, en la práctica, una porción importante de las gramáticas de amplia cobertura creadas siguiendo este formalismo está formada por árboles que pueden ser combinados mediante un tipo de adjunción más simple, el definido para las Gramáticas de Inserción de Árboles (TIG). En este artículo se describe un algoritmo de análisis sintáctico que hace uso de esta característica para reducir la complejidad práctica del análisis sintáctico de TAG, de tal modo que la costosa operación de adjunción estándar se utiliza sólo en aquellos casos en los que la versión TIG de adjunción no puede ser aplicada.
Palabras clave: Análisis sintáctico, gramáticas de adjunción de árboles, gramáticas de inserción de árboles

Abstract: Adjunction is a powerful operation that makes Tree Adjoining Grammar (TAG) useful for describing the syntactic structure of natural languages. In practice, a large part of wide coverage grammars written following the TAG formalism is formed by trees that can be combined by means of the simpler kind of adjunction defined for Tree Insertion Grammar. In this article, we describe a parsing algorithm that makes use of this characteristic to reduce the practical complexity of TAG parsing: the expensive standard adjunction operation is only considered in those cases in which the simpler cubic-time adjunction cannot be applied.
Keywords: Parsing, tree adjoining grammars, tree insertion grammars

1. Introducción

Las Gramáticas de Adjunción de Árboles (TAG) (Joshi y Schabes, 1997) y las Gramáticas de Inserción de Árboles (TIG) (Schabes y Waters, 1995) son formalismos gramaticales basados en árboles que hacen uso de dos operaciones denominadas adjunción y sustitución. Sin embargo, la operación de adjunción es más simple en el caso de TIG que en el de TAG, hecho que conlleva importantes consecuencias:

- TAG genera lenguajes de adjunción de árboles, un superconjunto estricto de los lenguajes independientes del contexto. La complejidad en el peor caso de su análisis es de orden $\mathcal{O}(n^6)$ en tiempo y $\mathcal{O}(n^4)$ en espacio, donde n representa la longitud de la cadena de entrada.
- TIG genera lenguajes independientes del contexto que se analizan, en el peor caso, en tiempo $\mathcal{O}(n^3)$ y en espacio $\mathcal{O}(n^2)$.

Aunque en gran medida la expresividad de las TAG se basa en la potencia de su operación de adjunción, se ha observado que gran parte de cada una de las gramáticas de amplia cobertura escritas siguiendo dicho formalismo se corresponde realmente con una TIG. Por ejemplo, Schabes y Waters (1995) informan de que más del 99 % de los árboles y de las adjunciones permitidas en la gramática del inglés XTAG (Doran et al., 1994) son compatibles con el formalismo TIG. En consecuencia, no parece lógico aplicar un analizador sintáctico para TAG, cuyo coste temporal es de orden $\mathcal{O}(n^6)$, en todas las operaciones de adjunción cuando la mayoría se pueden resolver con un analizador que presenta un coste temporal de orden cúbico.

En este artículo proponemos un analizador sintáctico combinado que toma lo mejor de ambos formalismos, de tal modo que aque-

* 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).

llas partes de la gramática que se corresponden con una TIG se analizan con una complejidad temporal cúbica y una complejidad espacial cuadrática, mientras que sólo aquellas partes de la gramática en las que debe aplicarse obligatoriamente la operación de adjunción definida para TAG son analizadas en tiempo $\mathcal{O}(n^6)$ y en espacio $\mathcal{O}(n^4)$.

1.1. El formalismo TAG

Formalmente, una TAG es una quíntupla $(V_N, V_T, S, \boldsymbol{I}, \boldsymbol{A})$, donde V_N es un conjunto finito de no-terminales, V_T es un conjunto finito de terminales, S es el axioma de la gramática, $\boldsymbol{I}$ es un conjunto finito de *árboles iniciales* y $\boldsymbol{A}$ es un conjunto finito de *árboles auxiliares*. $I \cup A$ es el conjunto de *árboles elementales*. Los nodos internos de dichos árboles están etiquetados por no-terminales mientras que los nodos hoja lo están por terminales o bien por la cadena vacía ϵ, excepto un único nodo hoja, denominado *pie*, de cada árbol auxiliar que está etiquetado por el mismo no-terminal que etiqueta la raíz del árbol. El camino que recorre los nodos desde la raíz hasta el pie se denomina *espina*. Denotaremos por etiq(N^γ) la etiqueta correspondiente al nodo N del árbol γ.

La derivación de nuevos árboles se realiza mediante la operación de *adjunción*: dado un árbol elemental α con un nodo N^γ etiquetado por el mismo no-terminal que la raíz y el pie de un árbol auxiliar β, la adjunción de β en N^γ se realiza escindiendo el subárbol de γ que tiene como raíz el nodo N^γ (que recibe el nombre de *nodo de adjunción*), pegando β a dicho nodo y pegando el subárbol escindido al pie de β. Notaremos mediante $\beta \in \text{adj}(N^\gamma)$ la posibilidad de que el nodo N de γ sea nodo de adjunción de β. Si la adjunción no es obligatoria en un nodo N^γ entonces $\textbf{nil} \in \text{adj}(N^\gamma)$, donde $\textbf{nil} \notin \boldsymbol{I} \cup \boldsymbol{A}$ es un símbolo vacío. Si la adjunción no está permitida en N^γ entonces $\{\textbf{nil}\} = \text{adj}(N^\gamma)$.

La operación de *sustitución* no incrementa la capacidad generativa del formalismo pero se considera habitualmente cuando se trabaja con TAGs lexicalizadas. En este caso, algunos nodos hoja (denominados *nodos de sustitución*) de los árboles elementales pueden estar etiquetados por símbolos no-terminales. Un árbol inicial α puede ser sustituido en un nodo N^γ, hecho denotado por $\alpha \in \text{subs}(N^\gamma)$, si su raíz está etiquetada por el mismo no-terminal que etiqueta N^γ. Un nodo de sustitución no puede serlo de adjunción.

1.2. El formalismo TIG

Consideraremos el conjunto $\boldsymbol{A}$ formado por la unión de los conjuntos $\boldsymbol{A}_L$, que contendrá *árboles auxiliares izquierdos* en los cuales todos los nodo de la frontera están situados a la izquierda del nodo pie, $\boldsymbol{A}_R$, que contendrá *árboles auxiliares derechos* en los que los nodos de la frontera están situados a la derecha del nodo pie, y $\boldsymbol{A}_W$, que contendrá *árboles wrapping* en los que los nodos de la frontera están ubicados tanto a la derecha como a la izquierda del nodo pie.

Dado un árbol auxiliar, denominaremos *nodos espina* a aquellos nodos ubicados en la espina y *nodos izquierdos* (resp. *nodos derechos*) a aquellos nodos situados a la izquierda (resp. a la derecha) de la espina. El conjunto $\boldsymbol{A}_{SL} \subseteq \boldsymbol{A}_L$ (resp. $\boldsymbol{A}_{SR} \subseteq \boldsymbol{A}_R$) de árboles auxiliares *fuertemente izquierdos* (resp. *fuertemente derechos*) está formado por aquellos árboles en los cuales no se permite realizar adjunciones en los nodos derechos (resp. nodos izquierdos) y en los cuales los nodos espina sólo admiten la adjunción de árboles fuertemente izquierdos (resp. fuertemente derechos). En esencia, una TIG es una TAG restringida en la cual el conjunto de árboles auxiliares es igual a $\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$ y en la que no se permite realizar adjunciones en los nodos raíz y pie de los árboles auxiliares. Adicionalmente, el formalismo TIG permite la realización de varias adjunciones en un mismo nodo (Schabes y Shieber, 1994).

1.3. Notación de los algoritmos

Describiremos los algoritmos de análisis mediante esquemas de análisis sintáctico (Sikkel, 1997). Un *sistema de análisis sintáctico* para una gramática $\mathcal{G}$ y una cadena de entrada $a_1 \ldots a_n$ es un triple $\text{P} = \langle \mathcal{I}, \mathcal{H}, \mathcal{D} \rangle$, donde $\mathcal{I}$ es un conjunto de *ítems* que representan resultados intermedios del proceso de análisis, $\mathcal{H}$ es un conjunto inicial de ítems denominado *hipótesis* que representan la cadena que va a ser analizada y $\mathcal{D}$ es un conjunto de reglas de inferencia denominadas *pasos deductivos*, mediante las cuales se derivan nuevos ítems ξ a partir de los ítems η_i existentes. Estos pasos deductivos tienen la forma $\frac{\eta_1, \ldots, \eta_k}{\xi}$ *cond* y su significado es el siguiente: si todos los antecedentes $\eta_i \in \mathcal{H} \cup \mathcal{I}$ de un paso deductivo ya existen y se satisfacen las condiciones *cond*, entonces el con-

secuente ξ deberá ser generado por el analizador sintáctico. La presencia de un conjunto $\mathcal{F} \subseteq \mathcal{I}$ de *ítems finales* indica el reconocimiento de la cadena de entrada. Un *esquema de análisis sintáctico* es un sistema de análisis sintáctico parametrizado para una gramática y una cadena de entrada.

En la descripción de los algoritmos de análisis para TAG y TIG , tenemos que representar el reconocimiento parcial de los árboles elementales. Para ello consideraremos cada árbol elemental γ constituido por un conjunto de producciones independientes del contexto $\mathcal{P}(\gamma)$: un nodo N^γ y sus hijos $N_1^\gamma \dots N_g^\gamma$ se representarán mediante una producción $N^\gamma \rightarrow N_1^\gamma \dots N_g^\gamma$. Indicaremos la posición del punto en el árbol marcando su posición en la producción correspondiente.

Con el fin de simplificar la descripción de los algoritmos consideraremos una producción adicional $\top \rightarrow \mathbf{R}^\alpha$ para cada árbol inicial α y dos producciones adicionales para cada árbol auxiliar β: $\top \rightarrow \mathbf{R}^\beta$ y $\mathbf{F}^\beta \rightarrow \bot$, donde $\mathbf{R}^\beta$ y $\mathbf{F}^\beta$ se refieren a los nodos raíz y pie de β, respectivamente. Los nuevos nodos $\top$ y $\bot$ no pueden ser nodos de adjunción.

La relación $\Rightarrow$ de derivación en $\mathcal{P}(\gamma)$ se define como $\delta \Rightarrow \nu$ si existen $\delta', \delta'', M^\gamma, \upsilon$ tales que $\delta = \delta' M^\gamma \delta''$, $\nu = \delta' \upsilon \delta''$ y existe $M^\gamma \rightarrow \upsilon \in \mathcal{P}(\gamma)$. El cierre reflexivo y transitivo de $\Rightarrow$ se denota como $\overset{*}{\Rightarrow}$. En caso de realizar una adjunción, $\delta \overset{*}{\Rightarrow} \nu$ si existen $\delta', \delta'', M^\gamma, \upsilon$ tal que $\delta = \delta' M^\gamma \delta''$, $\mathbf{R}^\beta \overset{*}{\Rightarrow} \upsilon_1 \mathbf{F}^\beta \upsilon_3$, $\beta \in \text{adj}(M^\gamma)$, $M^\gamma \rightarrow \upsilon_2$ y $\nu = \delta' \upsilon_1 \upsilon_2 \upsilon_3 \delta''$.

Dados dos pares de enteros (p, q) y (i, j), se cumple que $(p, q) \leq (i, j)$ si $i \leq p$ y $q \leq j$. Dados dos enteros p y q definiremos la operación $p \cup q$ como p si q está indefinido y como q si p está indefinido, estando indefinida en otro caso.

2. Análisis sintáctico de TIG

El algoritmo de análisis sintáctico definido en esta sección se basa en el analizador de complejidad temporal cúbica presentado por Schabes y Waters (1995). Puesto que la motivación de nuestro algoritmo es la de ser integrado con un analizador sintáctico para TAG del estilo de los descritos en (Alonso Pardo et al., 1999), hemos determinado seguir las especificaciones propias de TAG en caso de conflicto entre las definiciones para TAG y para TIG. Así, no se permitirá la realización de varias adjunciones en un mismo nodo, pero sí se permitirá adjuntar en el nodo raíz de un árbol auxiliar.

El algoritmo de análisis sintáctico para TIG trabaja con los tres conjuntos siguientes de ítems:

- Un conjunto $\mathcal{I}_{\text{TIG}}^{(a)}$ con ítems de la forma $[N^\gamma \rightarrow \delta \bullet \nu, i, j, adj]$, donde $N^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$, $0 \leq i \leq j$ y adj $\in \{\text{true}, \text{false}\}$. Se cumple que $\delta \neq \epsilon$, $\delta \overset{*}{\Rightarrow} a_{i+1} \dots a_j$ y adj = true si y sólo si se ha realizado una adjunción en el nodo N^γ, en otro caso adj = false. Los dos índices i y j indican la porción de la cadena de entrada que se deriva de δ.
- Un conjunto $\mathcal{I}_{\text{TIG}}^{(b)}$ con ítems de la forma $[N^\gamma \rightarrow \bullet \upsilon, j, j, \text{false}]$, donde $N^\gamma \rightarrow \upsilon \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$ y $0 \leq j$, que indican que no se ha realizado ninguna adjunción en N^γ.
- Un conjunto $\mathcal{I}_{\text{TIG}}^{(c)}$ con ítems de la forma $[N^\gamma \rightarrow \bullet \upsilon, i, j, \text{true}]$, donde $N^\gamma \rightarrow \upsilon \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$ y $0 \leq i \leq j$. Se cumple que existe un $\beta \in \boldsymbol{A}_{SL}$ tal que $\beta \in \text{adj}(N^\gamma)$ y $\mathbf{R}^\beta \overset{*}{\Rightarrow} a_{i+1} \dots a_j$. En este caso, i y j indican la parte de la cadena de entrada expandida por el árbol auxiliar izquierdo adjuntado en N^γ.

El componente booleano final de los ítems es necesario para bloquear la realización de más de una adjunción en un nodo: un valor de *true* indica que se ha realizado una adjunción en el nodo N^γ y por consiguiente cualquier otra adjunción en el mismo nodo queda descartada; por el contrario, un valor de *false* indica que no se ha realizado ninguna adjunción en dicho nodo, en cuyo caso el ítem puede jugar en un futuro el papel del ítem que reconoce la parte escindida de un árbol elemental que debe ser pegada al nodo pie de un árbol auxiliar derecho. Otras ventajas del algoritmo son el manejo explícito de las restricciones de adjunción, al contrario de lo que ocurría en el analizador de Schabes y Waters (1995), y que ya no es necesario que todo árbol auxiliar tenga al menos un símbolo terminal en su frontera.

Esquema 1 *El sistema de análisis sintáctico* $\text{P}_{\text{TIG}} = \langle \mathcal{I}_{\text{TIG}}, \mathcal{H}_{\text{TIG}}, \mathcal{D}_{\text{TIG}} \rangle$ *correspondiente a un algoritmo de tipo Earley se define tal y como se muestra a continuación:*

$$\mathcal{I}_{\text{TIG}} = \mathcal{I}_{\text{TIG}}^{(a)} \cup \mathcal{I}_{\text{TIG}}^{(b)} \cup \mathcal{I}_{\text{TIG}}^{(c)}$$

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

$$\mathcal{D}^{\text{Init}}_{\text{TIG}} = \frac{}{[\top \to \bullet \mathbf{R}^{\alpha}, 0, 0, \text{false}]} \quad \begin{array}{l} \alpha \in \boldsymbol{I} \\ S = \text{etiq}(\mathbf{R}^{\alpha}) \end{array}$$

$$\mathcal{D}^{\text{Scan}}_{\text{TIG}} = \frac{\begin{array}{l}[N^{\gamma} \to \delta \bullet M^{\gamma}\nu, i, j, adj], \\ [a, j, j+1]\end{array}}{[N^{\gamma} \to \delta M^{\gamma} \bullet \nu, i, j+1, adj]} \ a = \text{etiq}(M^{\gamma})$$

$$\mathcal{D}^{\epsilon}_{\text{TIG}} = \frac{[N^{\gamma} \to \delta \bullet M^{\gamma}\nu, i, j, adj]}{[N^{\gamma} \to \delta M^{\gamma} \bullet \nu, i, j, adj]} \ \epsilon = \text{etiq}(M^{\gamma})$$

$$\mathcal{D}^{\text{Pred}}_{\text{TIG}} = \frac{[N^{\gamma} \to \delta \bullet M^{\gamma}\nu, i, j, adj]}{[M^{\gamma} \to \bullet \upsilon, j, j, \text{false}]}$$

tal que $\mathbf{nil} \in \text{adj}(M^{\gamma})$ *ó* $(\exists \beta \in \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$ *con* $\beta \in \text{adj}(M^{\gamma}))$

$$\mathcal{D}^{\text{Comp}}_{\text{TIG}} = \frac{\begin{array}{l}[N^{\gamma} \to \delta \bullet M^{\gamma}\nu, i, j, adj], \\ [M^{\gamma} \to \upsilon \bullet, j, k, adj']\end{array}}{[N^{\gamma} \to \delta M^{\gamma} \bullet \nu, i, k, adj]}$$

tal que $(\mathbf{nil} \in \text{adj}(M^{\gamma})$ *y* $adj' = \text{false})$ *ó* $(\exists \beta \in \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$ *con* $\beta \in \text{adj}(M^{\gamma})$ *y* $adj' = \text{true})$

$$\mathcal{D}^{\text{SubsPred}}_{\text{TIG}} = \frac{[N^{\gamma} \to \delta \bullet M^{\gamma}\nu, i, j, adj]}{[\top \to \bullet \mathbf{R}^{\alpha}, j, j, \text{false}]} \ \alpha \in \text{subs}(M^{\gamma})$$

$$\mathcal{D}^{\text{SubsComp}}_{\text{TIG}} = \frac{\begin{array}{l}[N^{\gamma} \to \delta \bullet M^{\gamma}\nu, i, j, adj], \\ [\top \to \mathbf{R}^{\alpha} \bullet, j, k, \text{false}]\end{array}}{[N^{\gamma} \to \delta M^{\gamma} \bullet \nu, i, k, adj]} \ \alpha \in \text{subs}(M^{\gamma})$$

$$\mathcal{D}^{\text{LAdjPred}}_{\text{TIG}} = \frac{[M^{\gamma} \to \bullet \upsilon, i, i, \text{false}]}{[\top \to \bullet \mathbf{R}^{\beta}, i, i, \text{false}]} \quad \begin{array}{l} \beta \in \text{adj}(M^{\gamma}) \\ \beta \in \boldsymbol{A}_{SL} \end{array}$$

$$\mathcal{D}^{\text{LAdjComp}}_{\text{TIG}} = \frac{\begin{array}{l}[M^{\gamma} \to \bullet \upsilon, i, i, \text{false}], \\ [\top \to \mathbf{R}^{\beta} \bullet, i, j, \text{false}]\end{array}}{[M^{\gamma} \to \bullet \upsilon, i, j, \text{true}]} \quad \begin{array}{l} \beta \in \text{adj}(M^{\gamma}) \\ \beta \in \boldsymbol{A}_{SL} \end{array}$$

$$\mathcal{D}^{\text{RAdjPred}}_{\text{TIG}} = \frac{[M^{\gamma} \to \upsilon \bullet, i, j, \text{false}]}{[\top \to \bullet \mathbf{R}^{\beta}, j, j, \text{false}]} \quad \begin{array}{l} \beta \in \text{adj}(M^{\gamma}) \\ \beta \in \boldsymbol{A}_{SR} \end{array}$$

$$\mathcal{D}^{\text{RAdjComp}}_{\text{TIG}} = \frac{\begin{array}{l}[M^{\gamma} \to \upsilon \bullet, i, j, \text{false}], \\ [\top \to \mathbf{R}^{\beta} \bullet, j, k, \text{false}]\end{array}}{[M^{\gamma} \to \upsilon \bullet, i, k, \text{true}]} \quad \begin{array}{l} \beta \in \text{adj}(M^{\gamma}) \\ \beta \in \boldsymbol{A}_{SR} \end{array}$$

$$\mathcal{D}^{\text{LRFoot}}_{\text{TIG}} = \frac{[\mathbf{F}^{\beta} \to \bullet \bot, j, j, \text{false}]}{[\mathbf{F}^{\beta} \to \bot \bullet, j, j, \text{false}]} \ \beta \in \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$$

$$\begin{aligned} \mathcal{D}_{\text{TIG}} = \ & \mathcal{D}^{\text{Init}}_{\text{TIG}} \cup \mathcal{D}^{\text{Scan}}_{\text{TIG}} \cup \mathcal{D}^{\epsilon}_{\text{TIG}} \cup \mathcal{D}^{\text{Pred}}_{\text{TIG}} \cup \\ & \mathcal{D}^{\text{Comp}}_{\text{TIG}} \cup \mathcal{D}^{\text{SubsPred}}_{\text{TIG}} \cup \mathcal{D}^{\text{SubsComp}}_{\text{TIG}} \cup \\ & \mathcal{D}^{\text{LAdjPred}}_{\text{TIG}} \cup \mathcal{D}^{\text{LAdjComp}}_{\text{TIG}} \cup \\ & \mathcal{D}^{\text{RAdjPred}}_{\text{TIG}} \cup \mathcal{D}^{\text{RAdjComp}}_{\text{TIG}} \cup \mathcal{D}^{\text{LRFoot}}_{\text{TIG}} \end{aligned}$$

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

tal que $\alpha \in \boldsymbol{I}$ *y* $S = \text{etiq}(\mathbf{R}^{\alpha})$

La definición de las hipótesis realizada en este sistema de análisis sintáctico se corresponde con la estándar y es la misma que se utilizará en los restantes, por lo que no nos volveremos a referir explícitamente a ellas.

Los pasos en $\mathcal{D}^{\text{Scan}}_{\text{TIG}}$ y $\mathcal{D}^{\epsilon}_{\text{TIG}}$ son los encargados de leer los símbolos terminales de la entrada y la cadena vacía, respectivamente. Por su parte, los pasos en $\mathcal{D}^{\text{Pred}}_{\text{TIG}}$ y $\mathcal{D}^{\text{Comp}}_{\text{TIG}}$ se encargan de realizar el recorrido de los árboles elementales al estilo Earley, mientras que los pasos en $\mathcal{D}^{\text{SubsPred}}_{\text{TIG}}$ y $\mathcal{D}^{\text{SubsComp}}_{\text{TIG}}$ se encargan de iniciar y de finalizar el recorrido de los árboles elementales que participan en las operaciones de sustitución. Los restantes pasos gestionan las operaciones de adjunción:

- Si un árbol auxiliar izquierdo $\beta \in \boldsymbol{A}_{SL}$ puede ser adjuntado en un nodo M^{γ}, un paso en $\mathcal{D}^{\text{LAdjPred}}_{\text{TIG}}$ se encarga de iniciar el recorrido de β. Cuando dicho recorrido finaliza, un paso en $\mathcal{D}^{\text{LAdjComp}}_{\text{TIG}}$ inicia el recorrido del subárbol correspondiente a M^{γ} y establece el último componente del ítem a *true* con el fin de prohibir posteriores adjunciones en dicho nodo.

- En el caso de que un árbol auxiliar derecho $\beta \in \boldsymbol{A}_{SR}$ pueda ser adjuntado en un nodo M^{γ}, un paso de $\mathcal{D}^{\text{RAdjPred}}_{\text{TIG}}$ iniciará el recorrido de β en cuanto termine de recorrerse el subárbol correspondiente a tal nodo. Cuando se termine el recorrido de β, un paso en $\mathcal{D}^{\text{RAdjComp}}_{\text{TIG}}$ se encargará de actualizar las posiciones correspondientes a la parte de la cadena de entrada reconocida por M^{γ} incluyendo aquella parte reconocida por β, al tiempo que establece el último componente del ítem a *true* para evitar otras adjunciones posteriores en el mismo nodo.

El nodo pie de los árboles auxiliares no recibe ningún tratamiento especial, tal y como se percibe en los pasos de $\mathcal{D}^{\text{LRFoot}}_{\text{TIG}}$.

La complejidad cúbica del algoritmo queda establecida por los pasos deductivos $\mathcal{D}^{\text{Comp}}_{\text{TIG}}$, $\mathcal{D}^{\text{SubsComp}}_{\text{TIG}}$ y $\mathcal{D}^{\text{RAdjComp}}_{\text{TIG}}$, ya que todos ellos combinan tres posiciones distintas de la cadena de entrada. La complejidad espacial cuadrática viene dada por la definición de los ítems, ya que cada uno de ellos almacena dos posiciones de la cadena de entrada.

3. Análisis sintáctico combinado de TIG y TAG

En esta sección integraremos el algoritmo descrito en la sección 2 con el analizador para TAG descrito por Alonso Pardo et al. (1999) que incorpora una estrategia de tipo Earley, con el fin de definir un analizador sintáctico combinado para TIG y TAG en el cual la adjunción de los árboles auxiliares fuertemente izquierdos o derechos[1] se realice mediante pasos deductivos equivalentes a los definidos en $\mathrm{P}_{\mathrm{TIG}}$, mientras que el resto de las adjunciones se realizará de la forma descrita en (Alonso Pardo et al., 1999).

El algoritmo resultante considera los siguientes conjuntos de ítems:

- Un conjunto $\mathcal{I}^{(a)}_{\mathrm{Mix}}$ de ítems de la forma $[N^\gamma \rightarrow \delta \bullet \nu, i, j \mid p, q \mid adj]$, donde $N^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$, $0 \leq i \leq j$, $(p,q) = (-,-)$ ó $(p,q) \leq (i,j)$, y $adj \in \{\mathrm{true}, \mathrm{false}\}$. Debe satisfacerse una de las siguientes condiciones:
  1. $\gamma \in \boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$, $\delta \neq \epsilon$, $(p,q) \neq (-,-)$ y $\delta \overset{*}{\Rightarrow} a_{i+1} \ldots a_p$ $\mathbf{F}^\gamma$ $a_{q+1} \ldots a_j \overset{*}{\Rightarrow} a_{i+1} \ldots a_j$
  2. $\delta \neq \epsilon$, $(p,q) = (-,-)$ y $\delta \overset{*}{\Rightarrow} a_{i+1} \ldots a_j$.

  Los dos índices i and j indican la parte de la cadena de entrada reconocida por δ. Si $\gamma \in \boldsymbol{A}$, p y q indicarán la parte de la cadena reconocida por el nodo pie de γ si éste es un descendiente de δ, ya que en otro caso estarán indefinidos, hecho que se representará mediante $p = q = -$. Con respecto al elemento booleano final:
  - Si $adj = \mathrm{true}$ y $\nu \neq \epsilon$, indicará que un árbol fuertemente izquierdo $\beta \in \boldsymbol{A}_L$ ha sido adjuntado en N^γ.
  - Si $adj = \mathrm{true}$ y $\nu = \epsilon$, indicará que un árbol auxiliar ha sido adjuntado en N^γ.
  - Si $adj = \mathrm{false}$, indicará que no se ha realizado ninguna adjunción en N^γ.
- Un conjunto $\mathcal{I}^{(b)}_{\mathrm{Mix}}$ de ítems de la forma $[N^\gamma \rightarrow \bullet\upsilon, j, j \mid -,- \mid \mathrm{false}]$, donde $M^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$ y $0 \leq i \leq j$, que indican que no se ha realizado ninguna adjunción en N^γ.
- Un conjunto $\mathcal{I}^{(c)}_{\mathrm{Mix}}$ de ítems de la forma $[N^\gamma \rightarrow \bullet\upsilon, i, j \mid -,- \mid \mathrm{true}]$, donde $M^\gamma \rightarrow \delta\nu \in \mathcal{P}(\gamma)$, $\gamma \in \boldsymbol{I} \cup \boldsymbol{A}$ y $0 \leq i \leq j$. Para estos ítems se cumple que existe un $\beta \in \boldsymbol{A}_{SL}$ tal que $\beta \in \mathrm{adj}(N^\gamma)$ y $\mathbf{R}^\beta \overset{*}{\Rightarrow} a_{i+1} \ldots a_j$, esto es, β ha sido adjuntado en N^γ.

Esquema 2 *El sistema de análisis sintáctico* $\mathrm{P}_{\mathrm{Mix}} = \langle \mathcal{I}_{\mathrm{Mix}}, \mathcal{H}_{\mathrm{TIG}}, \mathcal{D}_{\mathrm{Mix}} \rangle$ *correspondiente a un algoritmo de análisis sintáctico combinado para TIG y TAG se define tal y como se muestra a continuación:*

$$\mathcal{I}_{\mathrm{Mix}} = \mathcal{I}^{(a)}_{\mathrm{Mix}} \cup \mathcal{I}^{(b)}_{\mathrm{Mix}} \cup \mathcal{I}^{(c)}_{\mathrm{Mix}}$$

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

tal que $\alpha \in \boldsymbol{I}$ *y* $S = \mathrm{etiq}(\mathbf{R}^\alpha)$

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

tal que $a = \mathrm{etiq}(M^\gamma)$

$$\mathcal{D}^{\epsilon}_{\mathrm{Mix}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j \mid p, q \mid adj]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, j \mid p, q \mid adj]}$$

tal que $\epsilon = \mathrm{etiq}(M^\gamma)$

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

tal que $\mathbf{nil} \in \mathrm{adj}(M^\gamma)$ *ó* $(\exists \beta \in \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$ *con* $\beta \in \mathrm{adj}(M^\gamma))$

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

tal que $(\mathbf{nil} \in \mathrm{adj}(M^\gamma)$ *y* $adj' = \mathrm{false})$ *ó* $(\exists \beta \in \boldsymbol{A}$ *con* $\beta \in \mathrm{adj}(M^\gamma)$ *y* $adj' = \mathrm{true})$

$$\mathcal{D}^{\mathrm{SubsPred}}_{\mathrm{Mix}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j \mid p, q \mid adj]}{[\top \rightarrow \bullet\mathbf{R}^\alpha, j, j \mid -,- \mid \mathrm{false}]}$$

tal que $\alpha \in \mathrm{subs}(M^\gamma)$

$$\mathcal{D}^{\mathrm{SubsComp}}_{\mathrm{Mix}} = \frac{[N^\gamma \rightarrow \delta \bullet M^\gamma\nu, i, j \mid p, q \mid adj], [\top \rightarrow \mathbf{R}^\alpha\bullet, j, k \mid -,- \mid \mathrm{false}]}{[N^\gamma \rightarrow \delta M^\gamma \bullet \nu, i, k \mid p, q \mid adj]}$$

[1] Dado el conjunto $\boldsymbol{A}$ de una TAG, determinaremos el conjunto $\boldsymbol{A}_{SL}$ de la siguiente manera: primero, estableceremos el conjunto $\boldsymbol{A}_L$ mediante el examen de la frontera de los árboles de $\boldsymbol{A}$ y haremos $\boldsymbol{A}_{SL} := \boldsymbol{A}_L$; en segundo lugar, eliminaremos de $\boldsymbol{A}_{SL}$ aquellos árboles que permitan realizar adjunciones en nodos situados a la derecha de su espina; y en tercer lugar, eliminaremos iterativamente de $\boldsymbol{A}_{SL}$ aquellos árboles que permitan adjuntar en los nodos de su espina árboles del conjunto $\boldsymbol{A} - \boldsymbol{A}_{SL}$. El conjunto $\boldsymbol{A}_{SR}$ se determina de una manera análoga.

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

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

tal que $\beta \in \boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{FootPred}} = \frac{[\mathbf{F}^\beta \to \bullet\bot, k, k \mid -, - \mid \text{false}]}{[M^\gamma \to \bullet\upsilon, k, k \mid -, - \mid \text{false}]}$$

tal que $\beta \in \boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{FootComp}} = \frac{\begin{array}{c}[\mathbf{F}^\beta \to \bullet\bot, k, k \mid -, - \mid \text{false}], \\ [M^\gamma \to \upsilon\bullet, k, l \mid p', q' \mid \text{false}]\end{array}}{[\mathbf{F}^\beta \to \bot\bullet, k, l \mid k, l \mid \text{false}]}$$

tal que $\beta \in \boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{AdjComp}_0} = \frac{\begin{array}{c}[\top \to \mathbf{R}^\beta\bullet, j, m \mid k, l \mid \text{false}], \\ [M^\gamma \to \upsilon\bullet, k, l \mid p', q' \mid \text{false}]\end{array}}{[M^\gamma \to \upsilon\bullet, j, m \mid p', q' \mid \text{true}]}$$

tal que $\beta \in \boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{LAdjPred}} = \frac{[M^\gamma \to \bullet\upsilon, i, i \mid -, - \mid \text{false}]}{[\top \to \bullet\mathbf{R}^\beta, i, i \mid -, - \mid \text{false}]}$$

tal que $\beta \in \text{adj}(M^\gamma)$ *con* $\beta \in \boldsymbol{A}_{SL}$

$$\mathcal{D}_{\text{Mix}}^{\text{LAdjComp}} = \frac{\begin{array}{c}[M^\gamma \to \bullet\upsilon, i, i \mid -, - \mid \text{false}], \\ [\top \to \mathbf{R}^\beta\bullet, i, j \mid -, - \mid \text{false}]\end{array}}{[M^\gamma \to \bullet\upsilon, i, j \mid -, - \mid \text{true}]}$$

tal que $\beta \in \boldsymbol{A}_{SL}$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{RAdjPred}} = \frac{[M^\gamma \to \upsilon\bullet, i, j \mid p, q \mid \text{false}]}{[\top \to \bullet\mathbf{R}^\beta, j, j \mid -, - \mid \text{false}]}$$

tal que $\beta \in \boldsymbol{A}_{SR}$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{RAdjComp}} = \frac{\begin{array}{c}[M^\gamma \to \upsilon\bullet, i, j \mid p, q \mid \text{false}], \\ [\top \to \mathbf{R}^\beta\bullet, j, k \mid -, - \mid \text{false}]\end{array}}{[M^\gamma \to \upsilon\bullet, i, k \mid p, q \mid \text{true}]}$$

tal que $\beta \in \boldsymbol{A}_{SR}$ *con* $\beta \in \text{adj}(M^\gamma)$

$$\mathcal{D}_{\text{Mix}}^{\text{LRFoot}} = \frac{[\mathbf{F}^\beta \to \bullet\bot, j, j \mid -, - \mid \text{false}]}{[\mathbf{F}^\beta \to \bot\bullet, j, j \mid -, - \mid \text{false}]}$$

tal que $\beta \in \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$

$$\begin{aligned}\mathcal{D}_{\text{Mix}} = \quad & \mathcal{D}_{\text{Mix}}^{\text{Init}} \cup \mathcal{D}_{\text{Mix}}^{\text{Scan}} \cup \mathcal{D}_{\text{Mix}}^{\epsilon} \cup \mathcal{D}_{\text{Mix}}^{\text{Pred}} \cup \\ & \mathcal{D}_{\text{Mix}}^{\text{Comp}} \cup \mathcal{D}_{\text{Mix}}^{\text{SubsPred}} \cup \mathcal{D}_{\text{Mix}}^{\text{SubsComp}} \cup \\ & \mathcal{D}_{\text{Mix}}^{\text{AdjPred}} \cup \mathcal{D}_{\text{Mix}}^{\text{FootPred}} \cup \mathcal{D}_{\text{Mix}}^{\text{FootComp}} \cup \\ & \mathcal{D}_{\text{Mix}}^{\text{AdjComp}_0} \cup \mathcal{D}_{\text{Mix}}^{\text{LAdjPred}} \cup \mathcal{D}_{\text{Mix}}^{\text{LAdjComp}} \cup \\ & \mathcal{D}_{\text{Mix}}^{\text{RAdjPred}} \cup \mathcal{D}_{\text{Mix}}^{\text{RAdjComp}} \cup \mathcal{D}_{\text{Mix}}^{\text{LRFoot}}\end{aligned}$$

$$\mathcal{F}_{\text{Mix}} = \{\ [\top \to \mathbf{R}^\alpha\bullet, 0, n \mid -, - \mid \text{false}]\ \}$$

tal que $\alpha \in \boldsymbol{I}$ *y* $S = \text{etiq}(\mathbf{R}^\alpha)$

El proceso de análisis comienza por la creación del ítem correspondiente a la producción que tiene la raíz de un árbol inicial en su lado izquierdo y el punto en el extremo izquierdo de su lado derecho. A continuación, un conjunto de pasos deductivos en $\mathcal{D}_{\text{Mix}}^{\text{Pred}}$ y $\mathcal{D}_{\text{Mix}}^{\text{Comp}}$ se encargan de recorrer los árboles elementales, mientras que los pasos en $\mathcal{D}_{\text{Mix}}^{\text{Scan}}$ y $\mathcal{D}_{\text{Mix}}^{\epsilon}$ se encargan de leer los símbolos terminales de la cadena de entrada y la cadena vacía, respectivamente. Las operaciones de sustitución se realizan mediante los pasos deductivos de $\mathcal{D}_{\text{Mix}}^{\text{SubsPred}}$ y $\mathcal{D}_{\text{Mix}}^{\text{SubsComp}}$, de forma similar a como ocurría en el caso de las TIG.

Las adjunciones de árboles auxiliares fuertemente izquierdos y derechos son realizadas por los mismos pasos que en el caso del algoritmo para TIG. En lo que respecta a las adjunciones de tipo TAG, un paso en $\mathcal{D}_{\text{Mix}}^{\text{AdjPred}}$ se encarga de predecir la adjunción de un árbol auxiliar $\beta \in \boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$ en un nodo M^γ, al tiempo que inicia el recorrido de β. En cuanto se alcanza el nodo pie de β, un paso de $\mathcal{D}_{\text{Mix}}^{\text{FootPred}}$ suspende el recorrido de dicho árbol para retomar el recorrido del árbol γ en el nodo M^γ. Puesto que en ese momento no hay información disponible acerca del nodo concreto en el cual se realizó la adjunción de β, en realidad se predicen todos los posibles nodos en los que se puede adjuntar dicho árbol auxiliar. Cuando se termina de recorrer el subárbol enraizado en M^γ, un paso $\mathcal{D}_{\text{Mix}}^{\text{FootComp}}$ se encarga de retomar el recorrido de β a partir de su nodo pie. Una vez que se termina de recorrer β, un paso deductivo $\mathcal{D}_{\text{Mix}}^{\text{AdjComp}_0}$ comprueba que el subárbol pegado al nodo pie se corresponde con el del nodo de adjunción, en cuyo caso la operación de adjunción se da por finalizada con la aplicación de un paso deductivo $\mathcal{D}_{\text{Mix}}^{\text{Comp}}$, teniendo en cuenta que p' y q' tienen valor si y sólo si el nodo de adjunción está ubicado en la espina de γ. Como se puede observar, para la finalización de la operación de adjunción se ha seguido el enfoque de dos pasos propuesto por Nederhof (1999).

4. Complejidad computacional

La complejidad espacial en el peor caso es $\mathcal{O}(n^4)$, pues cada ítem almacena a lo sumo cuatro posiciones de la cadena de entrada. Este caso se da en aquellos árboles pertenecientes al conjunto $\boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$, ya que tanto los árboles iniciales como

los fuertemente izquierdos y derechos contribuyen con una complejidad $\mathcal{O}(n^2)$ al resultado final.

En lo que respecta a la complejidad temporal en el peor caso:

- Las adjunciones TIG, esto es, la adjunción de un árbol fuertemente izquierdo o derecho en un nodo de un árbol en $\boldsymbol{I} \cup \boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR}$, presentan una complejidad $\mathcal{O}(n^3)$, debido a los pasos en $\mathcal{D}_{\text{Mix}}^{\text{RAdjComp}}$ y $\mathcal{D}_{\text{Mix}}^{\text{Comp}}$.
- Las adjunciones de tipo TAG son realizadas en tiempo $\mathcal{O}(n^6)$ por los pasos de $\mathcal{D}_{\text{Mix}}^{\text{AdjComp}_0}$, que son los encargados de tratar con los árboles auxiliares pertenecientes a $\boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$. De hecho, $\mathcal{O}(n^6)$ se alcanza sólo cuando un árbol *wrapping* es adjuntado en un nodo espina de un árbol auxiliar *wrapping*. Por su parte, la adjunción de un árbol *wrapping* en un nodo derecho de un árbol *wrapping* es realizada en tiempo $\mathcal{O}(n^5)$ por los pasos de $\mathcal{D}_{\text{Mix}}^{\text{Comp}}$. La misma complejidad se obtiene para la adjunción de un árbol auxiliar fuertemente derecho en un nodo derecho o espina de un árbol *wrapping*, debido a los pasos de $\mathcal{D}_{\text{Mix}}^{\text{RAdjComp}}$.
- Los demás casos de adjunción, tales como la adjunción de un árbol fuertemente izquierdo o derecho en un nodo espina de un árbol en $(\boldsymbol{A}_L - \boldsymbol{A}_{SL}) \cup (\boldsymbol{A}_R - \boldsymbol{A}_{SR})$, se realizan en tiempo $\mathcal{O}(n^4)$.
- Las operaciones de sustitución se realizan en tiempo $\mathcal{O}(n^3)$, excepto en el caso de que el nodo de sustitución se encuentre a la derecha del nodo pie en un árbol perteneciente a $\boldsymbol{A} - (\boldsymbol{A}_{SL} \cup \boldsymbol{A}_{SR})$, en cuyo caso se realiza en tiempo $\mathcal{O}(n^5)$

5. Resultados experimentales

Los algoritmos de análisis sintáctico descritos en este artículo han sido comparados en una implementación en Prolog de la máquina deductiva presentada por (Shieber, Schabes, y Pereira, 1995). Con ellos hemos realizado varios experimentos. En primer lugar, hemos comparado el rendimiento del algoritmo de análisis sintáctico para TIG con el diseñado para TAG cuando ambos trabajan sobre gramáticas TIG. Para ello se han diseñado dos gramáticas artificiales G_l (con $\boldsymbol{A}_{SR} = \emptyset$) y G_r (con $\boldsymbol{A}_{SL} = \emptyset$). Recordamos que para una TIG, la complejidad temporal del paso de compleción de adjunción de un analizador TAG (equivalente a $\mathcal{D}_{\text{Mix}}^{\text{AdjComp}_0}$) es $\mathcal{O}(n^4)$, en contraste con la complejidad $\mathcal{O}(n^2)$ de $\mathcal{D}_{\text{TIG}}^{\text{LAdjComp}}$ y la complejidad $\mathcal{O}(n^3)$ de $\mathcal{D}_{\text{TIG}}^{\text{RAdjComp}}$. En consecuencia, esperábamos que el analizador sintáctico para TIG fuese considerablemente más rápido que el otro. En efecto, para G_l hemos observado que dicho analizador es hasta 18 veces más rápido que el analizador para TAG. Sin embargo, en el caso de G_r las diferencias entre ambos analizadores son mínimas.

Estos resultados fueron corroborados por un segundo experimento realizado sobre varias TAG artificiales utilizando el analizador sintáctico definido por P_{Mix} y el analizador específico para TAG: el rendimiento del analizador combinado mejora ostensiblemente cuando el análisis de la cadena de entrada involucra la realización de adjunciones de árboles fuertemente izquierdos.

En un tercer experimento hemos tomado un subconjunto de la gramática XTAG (Doran et al., 1994) formado por 27 árboles elementales que cubren varias construcciones presentes en la lengua inglesa, como son las cláusulas relativas, los verbos auxiliares, los dependencias no acotadas, la extracción, etc. Para evitar perturbaciones debidas al mecanismo de unificación, hemos reemplazado las estructuras de rasgos presentes en los árboles elementales por un conjunto equivalente de restricciones de adjunción locales. No se ha realizado ningún tipo de filtrado de los árboles antes del análisis de cada frase. En la tabla 1 se muestran los resultados obtenidos para algunas frases representativas. Observamos como la aplicación del analizador definido por P_{Mix} conlleva una reducción del tiempo de análisis que llega al 31 % en algunos casos, ya que dicha mejora depende del tipo de árboles auxiliares que se utilice en cada frase.

6. Conclusiones

Hemos definido un algoritmo de análisis sintáctico que reduce la complejidad práctica del análisis de TAG al tener en cuenta que gran parte de una gramática de adjunción de árboles real puede ser tratada como una TIG.

El algoritmo de análisis sintáctico definido por P_{Mix} no preserva la propiedad del prefijo válido (Nederhof, 1999). Es posible ob-

Frase	P_{TAG}	P_{Mix}	Mejora
Srini bought a book	0.61	0.49	19.67 %
Srini bought Beth a book	0.77	0.71	7.79 %
Srini bought a book at the bookstore	0.94	0.93	1.06 %
he put the book on the table	0.83	0.71	14.46 %
the sun melted the ice	0.71	0.66	7.04 %
Elmo borrowed a book	0.55	0.49	10.91 %
he hopes Muriel wins	0.93	0.77	17.20 %
he hopes that Muriel wins	1.26	1.16	7.94 %
the man who Muriel likes bought a book	2.14	1.48	30.84 %
the man that Muriel likes bought a book	1.21	1.04	14.05 %
Clove caught a frisbee	0.55	0.49	10.91 %
who caught a frisbee	0.55	0.44	20.00 %
the emu thinks that the aardvark smells terrible	1.48	1.32	10.81 %
who does the emu think smells terrible	0.99	0.77	22.22 %
who did the elephant think the panda heard the emu said smells terrible	3.13	2.36	24.60 %
Herbert is angry and furious	0.55	0.55	0.00 %
Herbert is more livid than angry	0.50	0.44	12.00 %

Cuadro 1: Resultados con XTAG, en segundos, del analizador para TAG y del definido por P_{Mix}

tener una variante de dicho algoritmo que preserve tal propiedad mediante la introducción en los ítems de un elemento adicional h, que será utilizado para indicar la posición de la cadena de entrada en la cual comenzó el recorrido del árbol elemental indicado en cada ítem. La complejidad espacial se incrementará hasta $\mathcal{O}(n^5)$, aunque la complejidad temporal en el peor caso permanecerá en $\mathcal{O}(n^6)$ si se modifican los pasos AdjComp_0 y Comp tal y como se indica en (Nederhof, 1999).

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.

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, Agosto.

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.

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

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.

Shieber, S. M., Y. Schabes, y F. C.Ñ. 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.