Análisis sintáctico bidireccional de TAGs

Víctor J. Díaz
Depto. Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Avda. Reina Mercedes s/n
41012 Sevilla
vjdiaz@lsi.us.es

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

Resumen La mayoría de los analizadores para gramáticas de adjunción de árboles descritos en la literatura son adaptaciones de conocidos métodos para gramáticas independientes del contexto que utilizan una estrategia de lectura en un sólo sentido. Sin embargo, la forma en que actúa la operación de adjunción da pie a pensar que las estrategias bidireccionales pueden ajustarse de forma natural a esta clase de gramáticas. En este trabajo, presentamos una extensión para gramáticas de adjunción de árboles del analizador ascendente bidireccional definido por de Vreught y Honig para gramáticas independientes del contexto y mostramos su relación con el método bidireccional propuesto por Van Noord. Si bien es cierto que este nuevo algoritmo no mejora la complejidad asociada a los analizadores clásicos para TAGs, mostraremos que resuelve algunas de las limitaciones que presentan los algoritmos bidireccionales propuestos en la literatura.

1 *Introducción*

Las gramáticas de adjunción de árboles (TAG, *Tree Adjoining Grammars*) [4] son una extensión de las gramáticas independientes del contexto donde las estructuras básicas de representación son árboles en vez de producciones. Desde el punto de vista del procesamiento del lenguaje, una de las características más atractivas de este formalismo es su carácter lexicalizado, es decir, cada árbol debe presentar, al menos, un ítem léxico (ancla).

En la literatura han sido descritos varios algoritmos de análisis para este formalismo. La mayor parte de ellos son adaptaciones de otros ya conocidos para gramáticas independientes del contexto donde la lectura se efectúa en un único sentido de izquierda a derecha (por ejemplo, el de tipo CYK y el de tipo Earley presentados por Alonso et al. en [1]). Sin embargo, no ha sido frecuente la adaptación de analizadores bidireccionales a pesar de que esta clase de estrategia parece muy apropiada a la hora de simular la operación de adjunción. Podemos citar el algoritmo de Lavelli y Satta [5], que presenta la limitación de que sólo puede ser aplicado cuando los árboles elementales incluyen tan sólo un ancla, y el algoritmo de Van Noord [9], que al introducir el concepto de núcleo permite sortear esta dificultad. Sin embargo, la noción de núcleo no es completamente satisfactoria en algunas situaciones donde se presentan más de un ancla en los árboles elementales.

En este trabajo presentamos un nuevo analizador bidireccional ascendente basándonos es el analizador descrito por Vreught and Honig [2, 8] para gramáticas independientes del contexto. Este nuevo analizador presenta las siguientes características:

1. La estrategia bidireccional permite una fácil implementación de la operación de adjunción.
2. La estrategia ascendente es aprovechada para sacar partido de la lexicalización, ya que reduce el espacio de búsqueda al considerar durante el proceso de reconocimiento tan sólo aquellos árboles cuyas anclas concuerdan con algún símbolo de la entrada.
3. En el caso de entradas incorrectas, el analizador es capaz de extraer análisis incompletos que se ajusten a algún segmento de la cadena de entrada.
4. Es aplicable a cualquier gramática de adjunción de árboles lexicalizada, incluyendo el caso donde los árboles elementales

presentan más de un ancla.

Con respecto a la organización general del artículo, en la parte restante de esta sección introducimos las gramáticas de adjunción de árboles y los esquemas de análisis. En la sección 2 describimos el algoritmo propuesto, mientras que en la sección 3 mostramos el algoritmo de Van Noord. Ambos son comparados en la sección 4 conjuntamente con algunos de los algoritmos unidireccionales clásicos.

1.1 Gramáticas de adjunción de árboles

Una gramática de adjunción de árboles es una tupla $\mathcal{G} = (V_N, V_T, S, \boldsymbol{I}, \boldsymbol{A})$, donde V_N es un conjunto finito de símbolos no terminales, V_T es un conjunto finito de símbolos terminales, S es el axioma, $\boldsymbol{I}$ es un conjunto finito de árboles denominados iniciales y $\boldsymbol{A}$ es un conjunto finito de árboles denominados auxiliares. Los árboles contenidos en $\boldsymbol{I} \cup \boldsymbol{A}$ se denominan árboles elementales. En los árboles, los nodos interiores estarán etiquetados con no terminales mientras que las hojas estarán etiquetadas con terminales o la palabra vacía, ε, salvo un nodo en los auxiliares, denominado nodo pie, cuya etiqueta coincidirá con la de su raíz. La raíz de los árboles iniciales estará etiquetada con el axioma. El camino que conduce desde la raíz de los árboles auxiliares hasta su nodo pie se denomina la espina del árbol.

Para la composición de árboles elementales se hace uso de la operación de adjunción: sea γ un árbol que incluye un nodo N^γ etiquetado mediante $X \in V_N$ y sea β un árbol auxiliar cuya raíz y nodo pie está etiquetada también con el símbolo X. Entonces, tal como podemos ver en la figura 1, la adjunción de β en el nodo N^γ se obtiene podando el subárbol de γ cuya raíz es N^γ, colgando β en N^γ y posteriormente colgando el subárbol podado en el nodo pie de β. Mediante $\beta \in \mathrm{adj}(N^\gamma)$ denotaremos que el auxiliar β puede ser adjuntado en el nodo N^γ. Mediante $\mathbf{nil} \in \mathrm{adj}(N^\gamma)$ indicaremos que no es obligatoria la adjunción de ningún árbol auxiliar en N^γ.

Como podemos observar en la figura, la adjunción de β en un nodo de γ puede ser considerada una doble sustitución combinada de las dos subcadenas ω_L y ω_R a la izquierda y derecha de ω, respectivamente. Es justo esta característica la que consideramos puede ser aprovechada por los analizadores bidireccionales de la siguiente forma: un vez reconocido ω se pasa a reconocer a izquierda y derecha las cadenas ω_L y ω_R. Finalmente, se procederá a reconocer a izquierda y derecha las cadenas restantes ω_0 y ω_1.

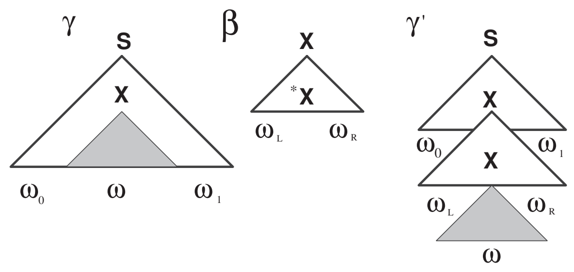

Figura 1: Operación de adjunción

Con objeto de describir los analizadores sintácticos para las gramáticas de adjunción de árboles, representaremos cada árbol elemental γ mediante un conjunto de producciones $\mathcal{P}(\gamma)$ de la siguiente forma: un nodo interior de γ, N^γ, junto con sus g nodos hijos $N_1^\gamma \ldots N_g^\gamma$ serán representados mediante la producción $N^\gamma \to N_1^\gamma \ldots N_g^\gamma$. Los elementos en las producciones serán los nodos del árbol, salvo en los símbolos situados en la parte derecha de una producción cuya etiqueta sea un terminal o la palabra vacía. Puesto que no es posible adjuntar ningún árbol auxiliar en dichos nodos y teniendo en cuenta que no pueden dominar a ningún otro nodo en un árbol elemental, podemos identificarlos mediante la propia etiqueta.

Para simplificar la definición de los analizadores consideraremos incluida la producción $\top \to \mathbf{R}^\alpha$ para cada árbol inicial, siendo $\mathbf{R}^\alpha$ su raíz, y las producciones $\top \to \mathbf{R}^\beta$ y $\mathbf{F}^\beta \to \bot$ para cada árbol auxiliar β, tal que $\mathbf{R}^\beta$ y $\mathbf{F}^\beta$ se corresponden con la raíz y el nodo pie del árbol auxiliar β, respectivamente. Tras exigir que no puede ser adjuntado ningún árbol auxiliar en $\top$ y $\bot$, la capacidad generativa de la gramática permanecerá intacta.

1.2 Esquemas de análisis

Para la definición de los analizadores adoptaremos el modelo basado en sistemas deductivos propuesto por Sikkel (*Parsing Schemata*, [8]) ya que aporta una notación con suficiente grado de abstracción para la descripción de esta clase de algoritmos. También cabe destacar que dentro de este marco teórico pueden establecerse con gran rigor relaciones entre diferentes estrategias de análisis, lo que nos servirá para mostrar la relación entre los esquemas propuestos.

Un sistema para el análisis sintáctico de una gramática G y una cadena de entrada $a_1 \dots a_n$ es una tupla $\langle \mathcal{I}, \mathcal{H}, \mathcal{D} \rangle$, siendo $\mathcal{I}$ un conjunto de *ítems* donde se representan los resultados intermedios en el análisis, $\mathcal{H}$ un conjunto inicial de *hipótesis* donde se representa la cadena de entrada a analizar y $\mathcal{D}$ un conjunto de *pasos deductivos* que determinan la forma en que son combinados y derivados nuevos ítems.

El conjunto de hipótesis para los analizadores que vamos a definir será siempre el mismo y se corresponde con:

$$\begin{aligned}\mathcal{H} = \ & \{[a, i-1, i] \mid a = a_i,\ 1 \leq i \leq n\} \cup \\ & \{[\sharp, -1, 0]\} \cup \{[\$, n, n+1]\}\end{aligned}$$

donde incluimos dos nuevos símbolos terminales, $\sharp$ y \$, con objeto de delimitar la cadena de entrada.

Los pasos deductivos serán de la forma $\frac{\eta_1, \dots, \eta_k}{\xi}\ cond$, indicando que si todos sus antecedentes η_i han sido previamente deducidos y las condiciones *cond* son satisfechas, entonces el consecuente ξ será deducido por el analizador. El conjunto de ítems finales $\mathcal{F} \subseteq \mathcal{I}$ contendrá aquellos ítems que determinarán si una cadena de entrada es o no correcta.

Un esquema para el análisis sintáctico, o simplemente esquema, es un sistema para el análisis sintáctico parametrizado por una gramática y cadena de entrada. Un esquema puede ser generalizado a partir de otro mediante un refinamiento de sus ítems (dividiendo un ítem en varios), mediante un refinamiento de pasos deductivos (dividiendo un paso deductivo en una secuencia de deducciones) o mediante una extensión (aumentando la clase de gramáticas a la que es aplicable).

Si deseamos reducir el número de ítems o pasos deductivos, podemos aplicar un filtro estático (descartando elementos redundantes), un filtro dinámico (utilizando información contextual para determinar la validez de un ítem) o una contracción de pasos (reemplazando una secuencia de deducciones por un sólo paso deductivo).

2 *El esquema* dVH

El esquema **dVH**, que presentaremos ahora, es la extensión del analizador para gramáticas independientes del contexto descrito por los autores De Vreught y Honig. El esquema se define mediante ítems de la forma:

$$[N^\gamma \to \nu \bullet \delta \bullet \omega, i, j, p, q]$$

donde $N^\gamma \to \nu\delta\omega \in P(\gamma)$ es una producción del árbol elemental γ y los dos puntos delimitando a δ indican la parte del subárbol con raíz en N^γ que ha sido reconocida. Si el subárbol ha sido reconocido completamente entonces se verificará $\nu = \omega = \varepsilon$. Los dos índices $0 \leq i \leq j$ determinan el segmento de la cadena de entrada reconocido por δ. Si $\gamma \in \boldsymbol{A}$ y p y q están definidos, es decir, $p \neq -$ y $q \neq -$, se cumple $i \leq p \leq q \leq j$ y los dos índices p y q determinan el segmento de la cadena de entrada reconocido por el nodo pie de γ. Con objeto de transmitir de forma ascendente los valores de los índices p y q desde el nodo pie hasta la raíz del árbol auxiliar, utilizaremos la función $r \cup r'$ que valdrá r si r' no está definido y valdrá r' si r no está definido. En cualquier otro caso, su valor será indefinido.

Los pasos deductivos del esquema son los siguientes:

$$\begin{aligned}\mathcal{D}_{\text{dVH}} = \ & \mathcal{D}^{\text{Ini}}_{\text{dVH}} \cup \mathcal{D}^{\varepsilon}_{\text{dVH}} \cup \mathcal{D}^{\text{Inc}}_{\text{dVH}} \cup \\ & \mathcal{D}^{\text{Con}}_{\text{dVH}} \cup \mathcal{D}^{\text{Pie}}_{\text{dVH}} \cup \mathcal{D}^{\text{Adj}}_{\text{dVH}}\end{aligned}$$

El análisis comienza deduciendo aquellos ítems asociados con producciones cuyo lado derecho incluye, o tan sólo la palabra vacía ($\mathcal{D}^{\varepsilon}_{\text{dVH}}$), o algún terminal participante en la cadena de entrada ($\mathcal{D}^{\text{Ini}}_{\text{dVH}}$). La posición del símbolo terminal en la cadena de entrada determina los valores de los índices en el consecuente. Las producciones nulas son incluidas considerando que ε puede estar situado en cualquier posición de la cadena de entrada. Los índices asociados al nodo pie en ambos pasos deductivos no están definidos ya que todavía no ha intervenido ningún nodo pie en el reconocimiento.

$$\mathcal{D}^{\text{Ini}}_{\text{dVH}} = \frac{[a, j-1, j]}{[N^\gamma \to \nu \bullet a \bullet \omega, j-1, j, -, -]}$$

$$\mathcal{D}^{\varepsilon}_{\text{dVH}} = \frac{}{[N^\gamma \to \bullet\bullet, j, j, -, -]}$$

El reconocimiento de los árboles elementales cuando no se efectúan operaciones de adjunción se realiza mediante los pasos deductivos de inclusión $\mathcal{D}^{\text{Inc}}_{\text{dvH}}$ y concatenación $\mathcal{D}^{\text{Con}}_{\text{dvH}}$. El primer paso deductivo continúa con el reconocimiento ascendente del superárbol respecto a un nodo M^γ siempre que éste no presente una restricción de adjunción obligatoria. Es decir, su aplicación exige que se verifique **nil** $\in$ adj(M^γ). El segundo paso deductivo combina dos reconocimientos parciales y adyacentes δ_1 y δ_2 (ver figura 2). Los índices p y q, son transmitidos al consecuente de forma ascendente a través de los nodos de la espina (junto con sus nodos hermanos por la derecha) en el caso de que estén definidos.

$$\mathcal{D}^{\text{Inc}}_{\text{dVH}} = \frac{[M^\gamma \rightarrow \bullet\delta\bullet, i, j, p, q]}{[N^\gamma \rightarrow \nu \bullet M^\gamma \bullet \omega, i, j, p, q]}$$

$$\mathcal{D}^{\text{Con}}_{\text{dVH}} = \frac{\begin{array}{c}[N^\gamma \rightarrow \nu \bullet \delta_1 \bullet \delta_2\omega, i, j', p, q],\\ [N^\gamma \rightarrow \nu\delta_1 \bullet \delta_2 \bullet \omega, j', j, p', q']\end{array}}{[N^\gamma \rightarrow \nu \bullet \delta_1\delta_2 \bullet \omega, i, j, p \cup p', q \cup q']}$$

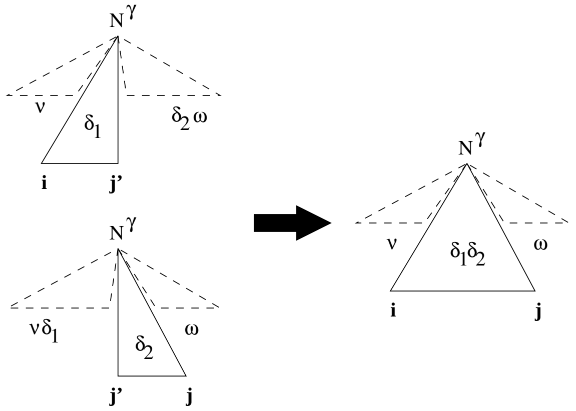

Figura 2: Concatenación

La operación de adjunción es efectuada mediante los pasos deductivos de inclusión de nodo pie $\mathcal{D}^{\text{Pie}}_{\text{dVH}}$ y de adjunción $\mathcal{D}^{\text{Adj}}_{\text{dVH}}$. Dado $\beta \in$ adj(M^γ), el reconocimiento ascendente del árbol auxiliar β comienza mediante $\mathcal{D}^{\text{Pie}}_{\text{dVH}}$ desde su nodo pie, de forma que el segmento de cadena reconocido por el nodo M^γ entre las posiciones k y l determina los valores de los índices en el consecuente (ver figura 3). Los índices p y q del antecedente serán transitoriamente ignorados hasta que la adjunción haya sido completada. Cuando el reconocimiento del árbol auxiliar β alcanza su nodo raíz, mediante $\mathcal{D}^{\text{Adj}}_{\text{dVH}}$ se concluye la adjunción sobre el nodo M^γ y se continuará con

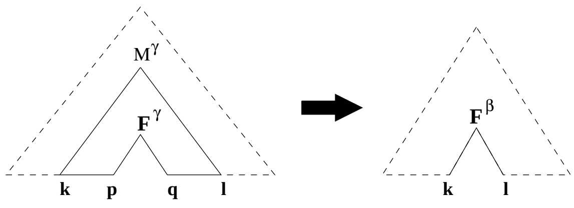

Figura 3: Inclusión de nodo pie

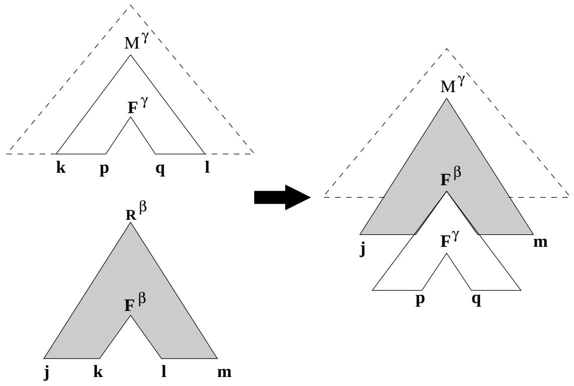

Figura 4: Compleción de adjunción

el reconocimiento ascendente del superárbol de γ respecto al nodo M^γ. Este paso deductivo es aplicado tan sólo cuando el segmento de cadena reconocido por el nodo pie de β coincide con el segmento reconocido por el nodo M^γ (ver figura 4). Los índices p y q del consecuente son nuevamente tomados del antecedente asociado al nodo donde se ha adjuntado. Finalmente, el segmento de cadena reconocido por el nodo M^γ se corresponde, tras la aplicación del paso deductivo, con el segmento reconocido por la raíz del árbol auxiliar β:

$$\mathcal{D}^{\text{Pie}}_{\text{dVH}} = \frac{[M^\gamma \rightarrow \bullet\delta\bullet, k, l, p, q]}{[\mathbf{F}^\beta \rightarrow \bullet\bot\bullet, k, l, k, l]}$$

$$\mathcal{D}^{\text{Adj}}_{\text{dVH}} = \frac{\begin{array}{c}[\top \rightarrow \bullet\mathbf{R}^\beta\bullet, j, m, k, l],\\ [M^\gamma \rightarrow \bullet\delta\bullet, k, l, p, q]\end{array}}{[N^\gamma \rightarrow \nu \bullet M^\gamma \bullet \omega, j, m, p, q]}$$

tal que $\beta \in$ adj(M^γ) en ambos pasos.

Si una cadena de entrada pertenece al lenguaje definido por la gramática deberá ser deducido un ítem asociado a la raíz de un árbol inicial completamente reconocido y que cubra toda la cadena:

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

Una vez introducido el algoritmo, podemos indicar que la extensión efectuada con respecto al algoritmo original consiste, esencialmente, en la incorporación de los índices p y q en los items y la ampliación del conjunto de pasos deductivos, mediante $\mathcal{D}^{\mathrm{Pie}}_{\mathrm{dVH}}$ y $\mathcal{D}^{\mathrm{Adj}}_{\mathrm{dVH}}$, para el reconocimiento de las operaciones de adjunción.

El algoritmo descrito tal cual es un reconocedor. Sin embargo, no es difícil obtener un analizador a partir de la anterior especificación. Partiendo del conjunto de ítems deducidos (*chart*) puede obtenerse el bosque sintáctico aplicando de forma inversa los pasos deductivos que han dado origen a la solución. Otro modo de obtener el bosque sintáctico consiste en ir anotando en cada ítem cierta información que indique cómo fue deducido. Cuando todos los ítems hayan sido deducidos, éstos estarán relacionados formando un grafo a partir del cual puede obtenerse el bosque sintáctico.

La complejidad temporal en el peor de los casos del algoritmo respecto al tamaño n de la cadena de entrada es $O(n^6)$. Esta complejidad procede del paso deductivo $\mathcal{D}^{\mathrm{Adj}}_{\mathrm{dVH}}$ ya que presenta el máximo número de variables relevantes en un paso deductivo: (j, m, k, l, p, q). La complejidad espacial del algoritmo en el peor de los casos es $O(n^4)$ debido a que cada ítem está compuesto de cuatro índices cuyos valores están comprendidos entre 0 y n.

3 El algoritmo de Van Noord

Van Noord [9] define un analizador bidireccional para TAGs que introduce el concepto de núcleo en los árboles elementales: de entre los nodos dominados por un nodo interior en un árbol elemental habrá uno al que se deberá considerar como su núcleo. La relación de núcleo entre nodos es reflexiva y transitiva, y presenta las dos siguientes restricciones:

1. El ancla de un árbol inicial debe ser núcleo de la raíz.
2. El nodo pie de un árbol auxiliar debe ser núcleo de la raíz.

A la hora de realizar el análisis, el método de Van Noord comienza con la predicción de cada árbol inicial cuyo núcleo-ancla participa en la cadena de entrada. A partir de aquí, el reconocimiento procede de forma ascendente a través del recorrido que impone la relación de núcleo. El recorrido se expande a izquierda y derecha (en este orden) a través de la predicción de los núcleos de cada nodo no terminal del árbol. Cuando se alcanza un nodo donde puede ser efectuada una adjunción se empezará el reconocimiento del árbol auxiliar a partir de su nodo pie. El reconocimiento del árbol auxiliar procede de forma análoga al de los árboles iniciales salvo que ahora el recorrido utilizado es el que conduce del nodo pie hasta la raíz mediante la relación de núcleo. Cuando el árbol auxiliar alcanza la raíz se proseguirá con el reconocimiento del árbol al que pertenece el nodo donde se adjuntó.

Pasaremos a la descripción del esquema **VN** definido a partir del método original de Van Noord. El esquema utiliza dos clases de ítems:

1. $[N^\gamma, i, j]$ donde N^γ es un nodo no terminal del árbol elemental γ y donde $0 \leq i \leq j$ son índices que delimitan las posiciones de la cadena de entrada entre las que debe efectuarse la predicción de su núcleo.
2. $[N^\gamma \rightarrow \nu \bullet \delta \bullet \omega, i, j, p, q]$ con una interpretación similar a los análogos en el esquema **dVH** salvo que ahora se exige que la parte central debe incluir el núcleo de N^γ, es decir: $\delta = \delta_1 \upsilon \delta_2$ y $\upsilon = \text{núcleo}(N^\gamma)$

Con respecto a los pasos deductivos, tenemos que

$$\begin{aligned}\mathcal{D}_{\mathrm{VN}} = \; & \mathcal{D}^{\mathrm{Ini}}_{\mathrm{VN}} \cup \\ & \mathcal{D}^{\mathrm{PreHC}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{PreHC^L}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{PreHC^R}}_{\mathrm{VN}} \cup \\ & \mathcal{D}^{\mathrm{HC(a)}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{HC(\varepsilon)}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{HC(M)}}_{\mathrm{VN}} \cup \\ & \mathcal{D}^{\mathrm{Sc^L}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{Sc^R}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{Con^L}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{Con^R}}_{\mathrm{VN}} \cup \\ & \mathcal{D}^{\mathrm{Pie}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{AdjHC}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{Adj^L}}_{\mathrm{VN}} \cup \mathcal{D}^{\mathrm{Adj^R}}_{\mathrm{VN}}\end{aligned}$$

Los cuatro pasos deductivos siguientes realizan la predicción de los núcleos:

$$\mathcal{D}^{\mathrm{Ini}}_{\mathrm{VN}} = \frac{}{[\mathbf{R}^\alpha, 0, n]} \quad \alpha \in \boldsymbol{I}$$

$$\mathcal{D}^{\mathrm{PreHC}}_{\mathrm{VN}} = \frac{[N^\gamma, i, j]}{[M^\gamma, i, j]} \quad M^\gamma = \text{núcleo}(N^\gamma)$$

$$\mathcal{D}^{\mathrm{PreHC^L}}_{\mathrm{VN}} = \frac{[N^\gamma \rightarrow \nu M^\gamma \bullet \mu \bullet \omega, i, j, p, q]}{[M^\gamma, 0, i]}$$

$$\mathcal{D}^{\mathrm{PreHC^R}}_{\mathrm{VN}} = \frac{[N^\gamma \rightarrow \bullet \nu \bullet M^\gamma \omega, i, j, p, q]}{[M^\gamma, j, n]}$$

Los pasos deductivos $\mathcal{D}_{\rm VN}^{\rm HC(a)}$, $\mathcal{D}_{\rm VN}^{\rm HC(\varepsilon)}$ y $\mathcal{D}_{\rm VN}^{\rm HC(M)}$ proceden a efectuar el reconocimiento ascendente a través del camino establecido por los núcleos, dependiendo respectivamente de que el núcleo sea un terminal a, la palabra vacía ε o un nodo interior M^γ:

$$\mathcal{D}_{\rm VN}^{\rm HC(a)} = \frac{[N^\gamma, i, j'],\ [a, j-1, j]}{[N^\gamma \rightarrow \nu \bullet a \bullet \omega, j-1, j, -, -]}$$

$$\mathcal{D}_{\rm VN}^{\rm HC(\varepsilon)} = \frac{[N^\gamma, i, j']}{[N^\gamma \rightarrow \bullet\bullet, j, j, -, -]}$$

$$\mathcal{D}_{\rm VN}^{\rm HC(M)} = \frac{[M^\gamma \rightarrow \bullet\delta\bullet, i, j, p, q]}{[N^\gamma \rightarrow \nu \bullet M^\gamma \bullet \omega, i, j, p, q]}$$

tal que $\mathbf{nil} \in \mathrm{adj}(M^\gamma)$ y $i < j \leq j'$. Si ignoramos las restricciones impuestas por la noción de núcleo estos tres pasos deductivos se corresponden con los pasos deductivos $\mathcal{D}_{\rm dVH}^{\rm Init}$, $\mathcal{D}_{\rm dVH}^{\varepsilon}$ y $\mathcal{D}_{\rm dVH}^{\rm Inc}$ respectivamente.

Los pasos deductivos $\mathcal{D}_{\rm VN}^{\rm Sc^L}$, $\mathcal{D}_{\rm VN}^{\rm Sc^R}$, $\mathcal{D}_{\rm VN}^{\rm Con^L}$ y $\mathcal{D}_{\rm VN}^{\rm Con^R}$ expanden a izquierda o derecha el reconocimiento de los nodos hermanos de un núcleo en los árboles elementales dependiendo de que sean símbolos terminales o no terminales.

$$\mathcal{D}_{\rm VN}^{\rm Sc^L} = \frac{[a, i-1, i],\ [N^\gamma \rightarrow \nu a \bullet \mu \bullet \omega, i, j, p, q]}{[N^\gamma \rightarrow \nu \bullet a\mu \bullet \omega, i-1, j, p, q]}$$

$$\mathcal{D}_{\rm VN}^{\rm Sc^R} = \frac{[a, j, j+1],\ [N^\gamma \rightarrow \bullet\nu \bullet a\omega, i, j, p, q]}{[N^\gamma \rightarrow \bullet\nu a \bullet \omega, i, j+1, p, q]}$$

$$\mathcal{D}_{\rm VN}^{\rm Con^L} = \frac{\begin{array}{c}[M^\gamma \rightarrow \bullet\delta\bullet, i, j', p, q],\\ [N^\gamma \rightarrow \nu M^\gamma \bullet \mu \bullet \omega, j', j, p', q']\end{array}}{[N^\gamma \rightarrow \nu \bullet M^\gamma\mu \bullet \omega, i, j, p \cup p', q \cup q']}$$

$$\mathcal{D}_{\rm VN}^{\rm Con^R} = \frac{\begin{array}{c}[N^\gamma \rightarrow \bullet\nu \bullet M^\gamma\omega, i, j', p, q],\\ [M^\gamma \rightarrow \bullet\delta\bullet, j', j, p', q']\end{array}}{[N^\gamma \rightarrow \bullet\nu M^\gamma \bullet \omega, i, j, p \cup p', q \cup q']}$$

Dado $\beta \in \mathrm{adj}(M^\gamma)$, la adjunción es reconocida de forma ascendente por los pasos deductivos $\mathcal{D}_{\rm VN}^{\rm Pie}$, $\mathcal{D}_{\rm VN}^{\rm AdjHC}$, $\mathcal{D}_{\rm VN}^{\rm Adj^L}$ y $\mathcal{D}_{\rm VN}^{\rm Adj^R}$. El primer paso introduce un nuevo ejemplar del auxiliar a partir de su nodo pie y es idéntico al paso deductivo $\mathcal{D}_{\rm dVH}^{\rm Pie}$. El resto de los pasos deductivos finalizan la adjunción dependiendo de que ésta se haya efectuado en un nodo que es núcleo o que se haya efectuado en un nodo hermano situado a la izquierda o derecha de un núcleo. Podemos observar que el paso deductivo $\mathcal{D}_{\rm VN}^{\rm AdjHC}$ es similar a $\mathcal{D}_{\rm dVH}^{\rm Adj}$, salvo que ahora se exige además que se verifique $M^\gamma = \text{núcleo}(N^\gamma)$.

$$\mathcal{D}_{\rm VN}^{\rm Pie} = \frac{[M^\gamma \rightarrow \bullet\delta\bullet, k, l, p, q]}{[\mathbf{F}^\beta \rightarrow \bullet\bot\bullet, k, l, k, l]} \quad \beta \in \mathrm{adj}(M^\gamma)$$

$$\mathcal{D}_{\rm VN}^{\rm AdjHC} = \frac{\begin{array}{c}[\top \rightarrow \bullet\mathbf{R}^\beta\bullet, j, m, k, l],\\ [M^\gamma \rightarrow \bullet\delta\bullet, k, l, p, q]\end{array}}{[N^\gamma \rightarrow \nu \bullet M^\gamma \bullet \omega, j, m, p, q]}$$

$$\mathcal{D}_{\rm VN}^{\rm Adj^L} = \frac{\begin{array}{c}[\top \rightarrow \bullet\mathbf{R}^\beta\bullet, i, j, k, l],\\ [M^\gamma \rightarrow \bullet\delta\bullet, k, l, p, q],\\ [N^\gamma \rightarrow \nu M^\gamma \bullet \mu \bullet \omega, j, m, p', q']\end{array}}{[N^\gamma \rightarrow \nu \bullet M^\gamma\mu \bullet \omega, i, m, p \cup p', q \cup q']}$$

$$\mathcal{D}_{\rm VN}^{\rm Adj^R} = \frac{\begin{array}{c}[\top \rightarrow \bullet\mathbf{R}^\beta\bullet, j, m, l, k, l],\\ [M^\gamma \rightarrow \bullet\delta\bullet, k, l, p, q],\\ [N^\gamma \rightarrow \bullet\nu \bullet M^\gamma\omega, i, j, p'q']\end{array}}{[N^\gamma \rightarrow \bullet\nu M^\gamma \bullet \omega, i, m, p \cup p', q \cup q']}$$

A partir de los pasos deductivos anteriores podemos concluir que el esquema **VN** es una contracción de los pasos deductivos respecto a **dVH** donde se reduce el número de ítems deducidos relacionados con los nodos ε y los símbolos no terminales que no son anclas. El paso deductivo $\mathcal{D}_{\rm dVH}^{\rm Con}$ queda diluido entre los pasos deductivos restantes del esquema **VN** donde se efectúa la expansión a izquierda o derecha impuesta por la relación de núcleo.

4 Resultados experimentales

La propuesta de Van Noord presenta limitaciones de índole lingüística en el caso de árboles elementales con más de un ancla. Efectivamente, si el árbol es inicial debemos escoger cuál de ellas será el núcleo de su raíz. También sucederá lo mismo si un nodo domina directamente a más de un ancla (anclas hermanas). Por otro lado, el esquema **dVH** introduce la noción de núcleo a expensas de no distinguir entre terminales anclas y no anclas. En el contexto de gramáticas lexicalizadas para lenguajes naturales, si bien es frecuente la aparición de árboles con más de un ancla, sin embargo es raro la aparición de terminales que no sean anclas. Esta observación es patente si miramos la gramática inglesa de amplia cobertura descrita en [10].

El comportamiento del esquema **dVH** puede ser mejorado si tenemos en cuenta las características de los árboles elementales descritos en dicha gramática. Las mejoras se basan en la técnicas propuestas en [3] donde se dividen los pasos deductivos con objeto de restringir su aplicación de forma que sean deducidos tan sólo aquellos ítems que realmente son necesarios para el reconocimiento de la cadena de entrada. Denominaremos **dVH′** al esquema obtenido tras aplicar dichas técnicas, que no son aplicables al algoritmo de Van Noord debido a las restricciones inpuestas por los núcleos.

Los resultados que mostraremos han sido obtenidos a partir de una implementación en Prolog de la máquina deductiva de análisis descrita en [7] ejecutándose en un Pentium II. Hemos implementado los siguientes analizadores: **E** es un analizador basado en el método de Earley que no garantiza la propiedad del prefijo válido [1], **Ned** es un analizador basado en el método de Earley que garantiza la propiedad del prefijo válido [6], **VN**, **dVH** y **dVH′** son los analizadores discutidos en este trabajo.

El experimento ha sido realizado sobre un subconjunto de la gramática inglesa constituido por 27 árboles elementales donde se cubren distintas clases de construcciones: cláusulas de relativo, verbos auxiliares, dependencias no locales, fenómenos de extracción, etc. Puesto que el objetivo del experimento es el comportamiento sintáctico, han sido ignorados los rasgos en los árboles elementales. Dichas restricciones han sido sustituidas por restricciones locales en los árboles elementales. Del mismo documento hemos seleccionado 25 oraciones correctas e incorrectas agrupadas respecto a los distintos aspectos tratados. La tabla 1 muestra el tiempo en segundos que ha consumido cada analizador para cada oración.

Input	**dVH**	**dVH′**	**VN**	**E**	**Ned**
Transitivos y Ditransitivos					
1	0.16	0.05	0.10	0.33	0.33
2	0.27	0.05	0.16	0.38	0.44
Argumentos y Adjuntos					
3	0.38	0.11	0.22	0.49	0.55
4	0.33	0.05	0.16	0.44	0.49
5	0.16	0.01	0.05	0.27	0.33
Ergativos e Intransitivos					
6	0.33	0.11	0.22	0.38	0.44
7	0.16	0.05	0.11	0.27	0.27
8	0.16	0.05	0.11	0.33	0.33
9	0.16	0.05	0.11	0.27	0.27
Complementos sentenciales					
10	0.16	0.05	0.11	0.55	0.44
11	0.22	0.05	0.17	0.66	0.49
Cláusulas de relativo					
12	0.60	0.16	0.38	0.77	0.88
13	0.55	0.16	0.38	0.66	0.77
Verbos auxiliares					
14	0.60	0.22	0.44	0.66	0.77
Extracción					
15	0.16	0.05	0.16	0.33	0.33
16	0.22	0.05	0.11	0.33	0.33
17	0.16	0.05	0.11	0.38	0.33
Dependencias no locales					
18	0.22	0.05	0.11	0.22	0.27
19	0.39	0.11	0.22	0.71	0.61
20	0.28	0.11	0.16	0.55	0.49
21	0.82	0.16	0.49	1.54	1.26
Adjetivos					
22	0.11	0.05	0.05	0.22	0.27
23	0.16	0.05	0.11	0.27	0.27
24	0.22	0.05	0.11	0.27	0.27
25	0.33	0.05	0.16	0.33	0.33

Tabla 1: Tiempo de análisis en segundos

De la tabla 1, podemos observar que **VN**, **dVH** y **dVH′** obtienen mejores resultados que los analizadores predictivos **E** y **Ned**. Sin embargo, debemos indicar que en términos de la operación más costosa, la adjunción, los analizadores predictivos efectúan igual o menos operaciones de adjunción que los analizadores bidirecionales. Por tanto, deducimos que la reducción de árboles elementales en los analizadores bidireccionales, debido a que sólo se consideran aquellos cuyas anclas pueden participar en el análisis, juega un papel importante en la mejora del comportamiento.

Por otra parte, podemos ver que aunque el esquema **dVH** presenta peores resultados que el esquema **VN**, sin embargo **dVH′** los mejora respecto a **VN**. Puesto que el número de operaciones de adjunción efectuadas por los tres analizadores es prácticamente el mismo, podemos concluir que las modificaciones efectuadas en el esquema **dVH**, que han conducido al esquema **dVH′**, han acelerado significativamente los tiempos de respuesta.

5 Conclusiones

En el presente trabajo se presenta un analizador bidireccional ascendente basado en el algoritmo para gramáticas independientes del contexto descrito por De Vreught y Honig y se muestra la relación existente entre el mismo y el algoritmo de Van Noord basado en núcleos. Las características del nuevo analizador hacen que presente un buen comportamiento cuando es aplicado a gramáticas lexicalizadas, al tiempo que permite una gran flexibilidad, pues no requiere de la noción de núcleo y puede ser aplicado independientemente del número y composición de las anclas en los árboles elementales.

6 Agradecimientos

Este trabajo ha sido financiado en parte por los fondos FEDER de la Unión Europea a través del proyecto 1FD97-0047-C04-02 y por la Xunta de Galicia mediante el proyecto PGIDT99XI10502B.

References

[1] Miguel A. Alonso, David Cabrero, Eric de la Clergerie, y Manuel Vilares. Tabular algorithms for TAG parsing. In *Proc. of EACL'99, Ninth Conference of the European Chapter of the Association for Computational Linguistics*, páginas 150–157, Bergen, Noruega, junio de 1999. ACL.

[2] J. P. M. de Vreught y H. J. Honig. A tabular bottom-up recognizer. Technical Report 89-78, Department of Applied Mathematics and Informatics, Delft University of Technology, Delft, Holanda, 1989.

[3] Víctor J. Díaz, Miguel A. Alonso, y Vicente Carrillo. Bidirectional parsing of TAG without heads. In *Proc. of 5th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5)*, páginas 67–72, París, Francia, mayo de 2000.

[4] Aravind K. Joshi y Yves Schabes. Tree-adjoining grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages. Volumen 3: Beyond Words*, capítulo 2, páginas 69–123. Springer-Verlag, Berlín/Heidelberg/Nueva York, 1997.

[5] Alberto Lavelli y Giorgio Satta. Bidirectional parsing of lexicalized tree adjoining grammars. In *Proceedings of the 5th Conference of the European Chapter of the Association for Computational Linguistics (EACL'91)*, Berlín, Alemania, Abril de 1991. ACL.

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

[7] Stuart M. Shieber, Yves Schabes, y Fernando C. N. Pereira. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1–2):3–36, julio-agosto de 1995.

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

[9] Gertjan van Noord. Head-corner parsing for TAG. *Computational Intelligence*, 10(4):525–534, 1994.

[10] The XTAG Research Group. A lexicalized tree adjoining grammar for English. http://www.cis.upenn.edu/~xtag. Technical Report IRCS 95-03, IRCS, Institute for Research in Cognitive Science, University of Pennsylvania, Filadelfia PA, EE.UU. 1999.