Revisando el Reconocedor con Prefijo Válido para TAGs de Schabes

Díaz Madrigal, Víctor J. Carrillo Montero, Vicente Toro Bonilla, Miguel
Facultad de Informática y Estadística. Universidad de Sevilla
Avda. Reina Mercedes s/n 41012- SEVILLA
{vjdiaz,carrillo}@lsi.us.es

Resumen

Garantizar la propiedad del prefijo válido en el reconocimiento de lenguajes para Gramáticas de Adjunción de Árboles (TAG, *Tree Adjoining Grammars*) supone una dificultad añadida en la definición de reconocedores. El reconocedor propuesto por Nederhof es actualmente la mejor solución, $O(n^6)$, siendo n el tamaño de la cadena de entrada. Existe una propuesta anterior debida a Schabes con coste $O(n^9)$. Este trabajo presenta un estudio de este último reconocedor, que lo adaptará dentro del paradigma de analizadores deductivos utilizando la misma notación del reconocedor de Nederhof. Después se presentará una reducción de su complejidad de $O(n^9)$ a $O(n^7)$ y, finalmente, se concluirá que los reconocedores de Nederhof y de Schabes están intimamente relacionados.

1. Introducción

Informalmente, un reconocedor con prefijo válido es aquel que es capaz de reconocer los errores sintácticos tan pronto como suceden. Garantizar esta propiedad en las **Gramáticas de Adjunción de Árboles** (TAG, *Tree Adjoining Grammars*) supone una dificultad añadida en la definición de reconocedores, frente a lo que sucede en las **Gramáticas Independientes del Contexto** (CFG, *Context Free Grammars*). Este trabajo es un estudio del reconocedor con prefijo válido para TAGs propuesto por Schabes **[Sch90]**, que reducirá su coste computacional de $O(n^9)$ a $O(n^7)$, siendo n el tamaño de la cadena de entrada. La ganancia obtenida no supera la de otro reconocedor con similares características propuesto por Nederhof **[Ned97]** con coste $O(n^6)$. En este trabajo mostraremos la relación entre ambos reconocedores.

Una **TAG** **[JLT75]** es una tupla (Σ, N, S, I, A) donde Σ y N son los alfabetos de símbolos terminales y no terminales, $S \in N$ el axioma, y $I \cup A$ es el conjunto de **árboles elementales**, denominándose **árboles iniciales** los primeros y **árboles auxiliares** los segundos. La letra α denotará un árbol inicial, β o β' árboles auxiliares y, en general, γ o γ' árboles elementales. Los nodos interiores de un árbol γ estarán etiquetados con no terminales (en los iniciales, la raíz deberá ser S) y los nodos frontera serán terminales o la palabra vacía ε, salvo uno en los auxiliares (el **nodo pie**) que estará etiquetado con el mismo no terminal que su raíz. Usaremos la notación A^γ, B^γ, etc., para representar los nodos de γ, y mediante $\mathbf{Etiq}(A^\gamma) \in (\Sigma \cup N \cup \{\varepsilon\})$ representaremos su etiqueta. Denotaremos la raíz de un árbol γ mediante R^γ, y el nodo pie de un auxiliar β mediante F^β.

La **adjunción** es la operación de composición en las TAGs y consiste en introducir una instancia de un auxiliar β en un no terminal A^γ, etiquetado igual que la raíz de β, siempre que cumpla una cierta propiedad denotada mediante $\mathbf{Adj}(\beta, A^\gamma)$. Al introducir el auxiliar, el árbol original γ es partido en dos subárboles respecto a A^γ. El lenguaje de una TAG se define como el conjunto de cadenas correspondientes a la frontera de todo árbol inicial o derivado a partir de un inicial mediante posibles adjunciones.

Los reconocedores serán definidos mediante un **sistema deductivo** **[SSP95]** basado en ítems. Los **ítems** representan resultados intermedios en el proceso de reconocimiento. Las **reglas deductivas** serán de la forma siguiente:

$$\frac{\begin{array}{c} A_1 \\ \dots \\ A_k \end{array}}{N \quad B} \qquad P(A_1, \dots, A_k B)$$

donde N es el nombre de la regla, y A_i junto con B son ítems cuyas metavariables son instanciadas en términos al aplicar la regla. $P(A_1,...A_k,B)$ son predicados que se deben evaluar a *cierto* para poder aplicar la regla. Los ítems objetivo establecerán cuando una cadena es gramatical respecto a una gramática de entrada. Por tanto, analizar una cadena de entrada es equivalente a encontrar una derivación que alcance un ítem objetivo. En **[Sik97]** se describen con detalle los mecanismos de transformación entre distintos analizadores deductivos.

cadena de entrada $a_1...a_n$, los ítems serán de la forma (figura 2):

$$[A^\gamma, pos, l, k, fl, fr, star, tl, bl\,]$$

donde A^γ es un nodo del árbol γ, $pos \in \{la,lb,ra,rb\}$ identifica la situación del punto en A^γ, *star* el nodo más profundo que domina a A^γ y sobre el que se ha reconocido parcialmente una adjunción - si no ha existido ninguna adjunción en los nodos que van desde la raíz de γ hasta A^γ, entonces estará no instanciado - y los otros componentes (índices) $l \le k$, $tl \le bl$, $fl \le fr$ son números enteros (instanciados o no) en el rango $[0..n]$ que representan posiciones sobre la

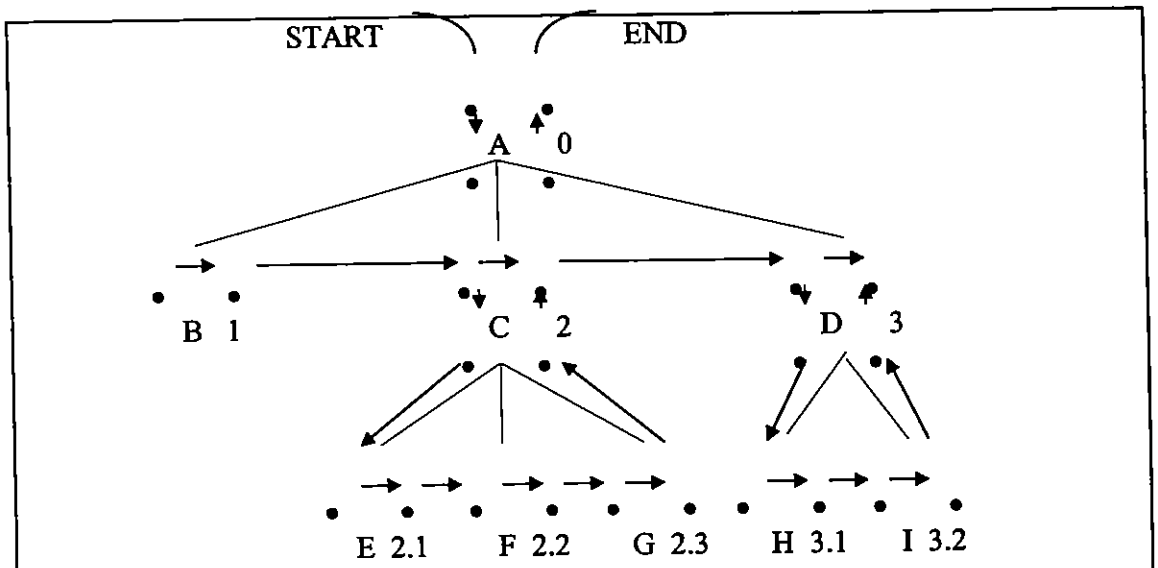

Figura 1. Recorrido del punto sobre un árbol elemental

cadena de entrada. El índice l indica la posición de la entrada donde empezó el reconocimiento de γ. El índice k indica la posición actual de reconocimiento. El par (fl, fr), si está instanciado, indica la subcadena reconocida por un nodo pie, dominado por A^γ, y perteneciente a un auxiliar todavía no reconocido completamente. Y el par (tl, bl), que estará instanciado tan sólo si *star* lo está, indica la subcadena reconocida desde la raíz hasta justo antes del nodo pie de un auxiliar adjuntado en el nodo *star*.

El reconocedor empieza aplicando **INI**, que deduce los ítems que cumplen:

INI $[R^a, la, 0, 0, \_, \_, \_, \_, \_\,]$ $\forall a \in I$

La palabra es aceptada si se deduce un ítem de la forma **ACC**.

ACC $[R^a, ra, 0, n, \_, \_, \_, \_, \_\,]$ para algún $a \in I$

La exploración de un terminal de la entrada se realiza mediante **SC1**:

2. *El reconocedor de Schabes usando árboles con puntos*

La presentación original del reconocedor de Schabes consiste en un algoritmo basado en técnicas de programación dinámica que usa la noción de árbol con punto como base para la descripción de los ítems. Un **árbol con punto** es un árbol γ con un solo nodo A^γ con punto. Un **nodo con punto** es un nodo decorado con un punto situado en una de las cuatro posiciones del conjunto $\{la, lb, ra, rb\}$, cuyos significados son: (la) left-above $^\bullet A^\gamma$, (lb) left-below $_\bullet A^\gamma$, (ra) right-above $A^{\gamma\,\bullet}$ y (rb) right-below $A^\gamma_\bullet$. El autor define un recorrido del punto a través de los nodos de un árbol elemental (figura 1) de modo que sean capturadas todas las posibles adjunciones en dicho árbol.

La transformación del algoritmo original de Schabes en un sistema deductivo es prácticamente inmediata y puede verse en **[DCT98]**, por tanto, pasaremos directamente a su descripción. Dada una gramática TAG y una

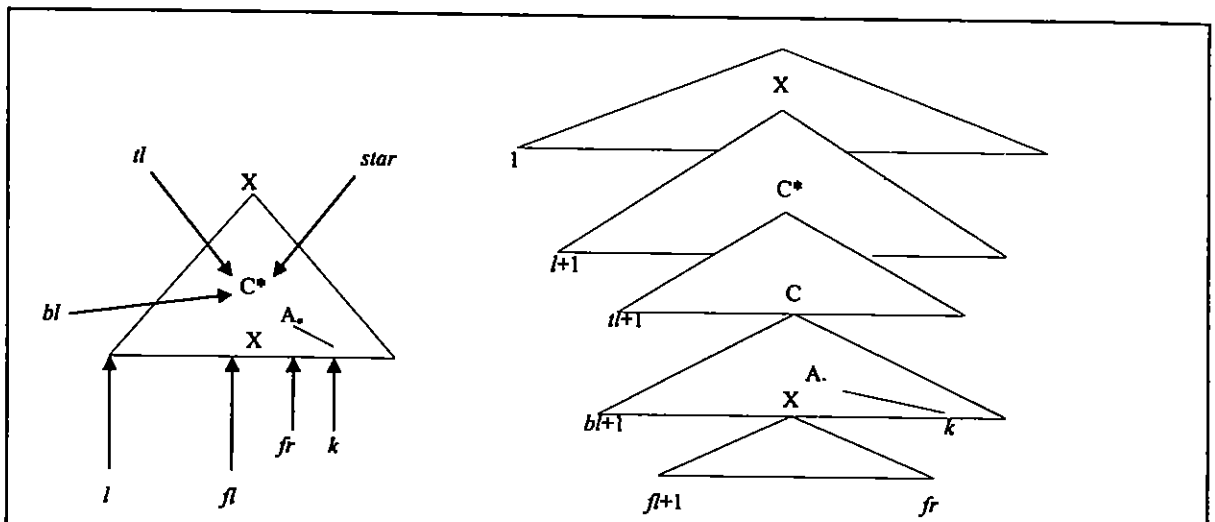

Figura 2. Items del reconocedor de Schabes

$$\textbf{SC1}\quad \frac{[\,A^\gamma,\ la,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[\,A^\gamma,\ ra,\ l,\ k+1,\ fl,\ fr,\ star,\ tl,\ bl\,]}\quad \textbf{Etiq}(A^\gamma)= a_{k+1}$$

La palabra vacía se reconoce con la regla **SC2**:

$$\textbf{SC2}\quad \frac{[\,A^\gamma,\ la,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[A^\gamma,\ ra,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}\quad \textbf{Etiq}(A^\gamma)= \varepsilon$$

La regla **MDD** desciende el punto desde un nodo padre A^γ hacia su nodo hijo más a la izquierda B^γ, si existe. Es decir, si se cumple el predicado **PrimerHijo**(A^γ,B^γ):

$$\textbf{MDD}\quad \frac{[\,A^\gamma,\ lb,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[\,B^\gamma,\ la,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}\quad \textbf{PrimerHijo}(A^\gamma,B^\gamma)$$

La regla **MDU1** mueve el punto desde un nodo A^γ hacia su nodo hermano más cercano por la derecha B^γ, si existe. Es decir, si se cumple el predicado **Hermano**(A^γ,B^γ):

$$\textbf{MDU1}\quad \frac{[\,A^\gamma,\ ra,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[\,B^\gamma,\ la,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}\quad \textbf{Hermano}(A^\gamma,B^\gamma)$$

La regla **MDU2** desplaza el punto desde el último nodo hijo A^γ hacia su nodo padre B^γ, si existe. Es decir, si se cumple el predicado **PadreUltimoHijo**(A^γ,B^γ):

$$\textbf{MDU2}\quad \frac{[\,A^\gamma,\ ra,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[\,B^\gamma,\ rb,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}\quad \textbf{PadreUltimoHijo}(A^\gamma,B^\gamma)$$

La regla **LP1** se aplica si es posible una adjunción de β sobre A^γ, lo que propicia empezar el reconocimiento de una nueva ocurrencia de β:

$$\textbf{LP1}\quad \frac{[\,A^\gamma,\ la,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[\,R^\beta,\ la,\ k,\ k,\ \_,\ \_,\ \_,\ \_,\ \_\,]}\quad \textbf{Adj}(\beta,A^\gamma)$$

Las reglas **LP21** y **LP22** ignoran la posibilidad de adjunción sobre A^γ. Cuando dicho nodo no es el pie de algún auxiliar se aplica **LP21**:

$$\textbf{LP21}\quad \frac{[\,A^\gamma,\ la,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}{[\,A^\gamma,\ lb,\ l,\ k,\ fl,\ fr,\ star,\ tl,\ bl\,]}\quad \textbf{Etiq}(A^\gamma) \in N \wedge \textbf{NoPie}(A^\gamma)$$

mientras que si el nodo es un nodo pie se aplica **LP22**:

$$\textbf{LP22}\quad \frac{[\,F^\beta,\ la,\ l,\ k,\ \_,\ \_,\ star,\ tl,\ bl\,]}{[\,F^\beta,\ lb,\ l,\ k,\ k,\ \_,\ star,\ tl,\ bl\,]}$$

Las reglas **LC1** y **LC2** se aplican cuando ha concluido el reconocimiento de un auxiliar β hasta su nodo pie, pasándose a reconocer el subárbol escindido por la adjunción. **LC1** se aplica si el nodo sobre el que se adjuntó era a su vez el nodo pie de un auxiliar β':

$$\textbf{LC1}\quad \frac{[\,F^\beta,\ lb,\ l,\ k,\ k,\ \_,\ star,\ tl,\ bl\,]\quad [\,F^{\beta'},\ la,\ l',\ l,\_,\_,\ star',\ tl',\ bl'\,]}{[\,F^{\beta'},\ lb,\ l',\ k,\ k,\_,\ F^\beta,\ l,\ k\,]}\quad \textbf{Adj}(\beta,F^{\beta'})$$

mientras que **LC2** se aplica si el nodo A^γ donde se adjuntó no era nodo pie:

$[F^{\beta}, lb, l, k, k, \_, star, tl, bl]$
$[A^{\gamma}, la, l', l, fl, fr, star', tl', bl']$

LC2 ----------

$[A^{\gamma}, lb, l', k, fl, fr, A^{\gamma}, l, k]$

$\mathbf{Adj}(\beta, A^{\gamma}) \wedge \mathbf{NoPie}(A^{\gamma})$

La regla **RP1** se aplica si finalizó el reconocimiento de un subárbol dominado por A^{γ} sobre el que se adjuntó un auxiliar β. A continuación se debe reconocer el resto de β:

$[A^{\gamma}, rb, l, k, fl, fr, A^{\gamma}, tl, bl]$
$[F^{\beta}, lb, tl, bl, bl, \_, star', tl', bl']$

RP1 ----------

$[F^{\beta}, rb, tl, k, bl, k, star', tl', bl']$

$\mathbf{Adj}(\beta, A^{\gamma})$

La regla **RP2** se aplica si ha concluido el reconocimiento de un subárbol sobre cuyo nodo raíz A^{γ} no se efectuó ninguna adjunción:

$[A^{\gamma}, rb, l, k, fl, fr, star, tl, bl]$

RP2 ----------

$[A^{\gamma}, ra, l, k, fl, fr, star, tl, bl]$

$star \neq A^{\gamma}$

La regla **RC** se aplica cuando se ha reconocido todo el árbol β, pasando a reconocer el resto del árbol a partir del nodo A^{γ} donde se efectuó la adjunción:

$[R^{\beta}, ra, l, k, fl, fr, \_, \_, \_]$
$[A^{\gamma}, la, l', l, fl', fr', star, tl, bl]$
$[A^{\gamma}, rb, l', fr, fl'^{*}, fr'^{*}, A^{\gamma}, l, fl]$

RC ----------

$[A^{\gamma}, ra, l', k, fl'^{*}, fr'^{*}, star, tl, bl]$

$\mathbf{Adj}(\beta, A^{\gamma})$

donde f^{*} coincide con f si f está instanciado, y puede tener cualquier valor en otro caso.

3. *El reconocedor de Schabes usando reglas CFG con puntos*

Pasaremos a adaptar el reconocedor original de Schabes mediante reglas CFG con punto usando la notación presentada en **[Ned97]**. Para cada árbol γ incluiremos un nuevo nodo, denotado mediante T^{γ}, cuyo único hijo será su raíz R^{γ}. Para cada auxiliar β incluiremos un nuevo nodo, denotado $\perp^{\beta}$, único hijo del nodo pie.

A todo nodo interior A^{γ}, cuyos n hijos vienen dados por la secuencia ordenada $A^{\gamma}_1 \ldots A^{\gamma}_n$, le asociaremos la regla CFG: $A^{\gamma} \rightarrow A^{\gamma}_1 \ldots A^{\gamma}_n$. Definiremos las reglas $T^{\alpha} \rightarrow R^{\alpha}$, $T^{\beta} \rightarrow R^{\beta}$ y $F^{\beta} \rightarrow \perp^{\beta}$ para cada inicial α y auxiliar β. De esta forma, una gramática TAG puede ser interpretada como un conjunto de reglas CFG resultado de la unión de todas las reglas CFG obtenidas a partir de cada árbol elemental.

El nuevo sistema deductivo utiliza ítems de la forma:

$[R, l, k, fl, fr, star, tl, bl]$

donde mantenemos los componentes *l, k, fl, fr, tl, bl* y *star* presentes en los ítems originales y desaparecen los componentes relacionados con el nodo con punto, que son sustituidos por una regla CFG con punto R de la forma: $B^{\gamma} \rightarrow \nu \bullet \omega$ con $\nu, \omega \in (\Sigma \cup N)^{*}$.

Los puntos en las reglas CFG deberán dar cuenta de la posición $\{la, lb, ra, rb\}$ en la que está situado el punto en el nodo. Para preservar el significado del punto en los nodos definimos las siguientes correspondencias: $A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\omega$ será equivalente a ${}^{\bullet}B^{\gamma}$ y $B^{\gamma} \rightarrow \delta \bullet$ ($\delta \in (\Sigma \cup N)^{*}$) lo será a $B^{\gamma}_{\bullet}$, la regla $F^{\beta} \rightarrow \bullet \perp^{\beta}$ será equivalente a ${}_{\bullet}F^{\beta}$ y la regla $F^{\beta} \rightarrow \perp^{\beta} \bullet$ lo será a $F^{\beta}_{\bullet}$. Interpretaremos, además, la regla $T^{\gamma} \rightarrow R^{\gamma} \bullet$ como $R^{\gamma\bullet}$ y la regla $T^{\gamma} \rightarrow \bullet R^{\gamma}$ como ${}^{\bullet}R^{\gamma}$.

Podemos observar que las reglas **MDD**, **MDU1** y **MDU2** no modifican los índices del ítem, limitándose a garantizar el correcto desplazamiento del punto sobre los nodos de los árboles. Estas reglas, al contrario que las otras, no aportan información significativa para el proceso de reconocimiento. Si se construye un grafo cuyos nodos sean las reglas originales de Schabes y cuyas transiciones sean los posibles movimientos entre reglas, podemos establecer recorridos a través de dicho grafo que obvien aquellas transiciones en las que intervienen las reglas **MDD**, **MDU1** y **MDU2**. La adaptación que proponemos se basa precisamente en concentrar en una sola regla aquel conjunto de transiciones significativas que incluyan la aplicación de dichas reglas.

El reconocimiento empezará con ítems de la forma $[T^{\alpha} \rightarrow \bullet R^{\alpha}, 0, 0, \_, \_, \_, \_, \_]$ para cada $\alpha \in I$. La aceptación se producirá cuando, para algún inicial α, se derive un ítem cuya forma sea $[T^{\alpha} \rightarrow R^{\alpha} \bullet, 0, n, \_, \_, \_, \_, \_]$. Estas reglas se corresponden respectivamente con las reglas **INI** y **ACC** anteriores.

Para representar la equivalencia entre reglas usaremos expresiones regulares de la forma: $u + v$ (se aplica u o v), $[u]$ (la aplicación de u es opcional), y $u \cdot v$ (v se aplicará tras u), donde u, v son reglas o expresiones regulares sobre reglas con o sin paréntesis.

- Exploración de terminal

SCa = SC1· (MDU1+MDU2)

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l, k, fl, fr, star, tl, bl]$

Sca ----------

$[A^{\gamma} \rightarrow \nu B^{\gamma} \bullet\ \omega, l, k+1, fl, fr, star, tl, bl]$

Etiq$(B^{\gamma}) = a_{k+1}$

- Exploración de palabra vacía

SCε = SC2 · (MDU1+MDU2)

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l, k, fl, fr, star, tl, bl]$

SCε ----------

$[A^{\gamma} \rightarrow \nu B^{\gamma} \bullet\ \omega, l, k, fl, fr, star, tl, bl]$

Etiq$(B^{\gamma}) = \varepsilon$

- Predicción (no se efectúa adjunción)

PRED = (LP21 · MDD) + LP22

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l, k, fl, fr, star, tl, bl]$

PRED ----------

$[B^{\gamma} \rightarrow \bullet\ \delta, l, k, fl, fr, star, tl, bl]$

- Predicción de adjunción

PAD = LP1

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l, k, fl, fr, star, tl, bl]$

PAD ----------

$[T^{\beta} \rightarrow \bullet\ R^{\beta}, k, k, \_, \_, \_, \_, \_]$

Adj(β, B^{γ})

- Predicción del nodo pie

PFT = (LC2 · MDD) + LC1

$[F^{\beta} \rightarrow \bullet\ \perp^{\beta}, l, k, \_, \_, star, tl, bl]$

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l', l, fl, fr, star', tl', bl']$

PFT ----------

$[B^{\gamma} \rightarrow \bullet\ \delta, l', k, fl, fr, B^{\gamma}, l, k]$

Adj(β, B^{γ})

- Completitud (no se ha efectuado adjunción)

CMP = RP2 · [MDU1+MDU2]

$[B^{\gamma} \rightarrow \delta \bullet, l, k, fl^*, fr^*, star, tl, bl]$

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l, k', fl, fr, star, tl, bl]$

CMP ----------

$[A^{\gamma} \rightarrow \nu B^{\gamma} \bullet \omega, l, k, fl^*, fr^*, star, tl, bl]$

- Completitud del nodo pie

CFT = RP1

$[B^{\gamma} \rightarrow \delta \bullet, l, k, fl, fr, B^{\gamma}, tl, bl]$

$[F^{\beta} \rightarrow \bullet\ \perp^{\beta}, tl, bl, \_, \_, star, tl', bl']$

CFT ----------

$[F^{\beta} \rightarrow \perp^{\beta} \bullet, tl, k, bl, k, star, tl', bl']$

Adj(β, B^{γ})

- Completitud del árbol auxiliar

CAD = RC · [MDU1 + MDU2]

$[T^{\beta} \rightarrow R^{\beta} \bullet, l, k, fl, fr, \_, \_, \_]$

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma}\ \omega, l', l, fl', fr', star, tl, bl]$

$[B^{\gamma} \rightarrow \delta \bullet, l', fr, fl'^*, fr'^*, B^{\gamma}, l, fl]$

CAD ----------

$[A^{\gamma} \rightarrow \nu B^{\gamma} \bullet \omega, l', k, fl'^*, fr'^*, star, tl, bl]$

Adj(β, B^{γ})

Se observa que, fruto de la inclusión del nodo $\perp^{\beta}$, el valor de *fl* puede actualizarse en la regla **CFT**, y no en la regla **PRED**, como cabría esperar debido a que ésta incluye la regla **LP22**. Esta excepción no influye en la corrección y evita la necesidad de incluir dos reglas de predicción en función de si el nodo actual es el nodo pie o no.

4. *Estudio de la complejidad*

Informalmente, el coste computacional coincide con $O(n^p)$ donde p es el mayor número de índices repetidos que intervienen en una regla. Por tanto, el coste es $O(n^9)$, ya que la regla **CAD** presenta un total de 9 índices repetidos (*l, l', k, fl, fr, fl'*, fr'*, tl, bl*). La complejidad puede reducirse utilizando un paso intermedio, en vez de aplicar una sola regla **CAD**, similar a los propuestos en **[Ned97]** y **[ACCV99]**. El nuevo reconocedor incluirá todas las reglas anteriores salvo la regla **CAD**, que es sustituida por dos nuevas reglas denominadas **CAD0** y **CAD1**:

$[T^{\beta} \rightarrow R^{\beta} \bullet, l, k, fl, fr, \_, \_, \_]$

$[B^{\gamma} \rightarrow \delta \bullet, l', fr, fl'^*, fr'^*, B^{\gamma}, l, fl]$

CAD0 ----------

$[B^{\gamma} \rightarrow \delta \bullet, l', l, k, fl'^*, fr'^*]$

Adj(β, B^{γ})

$[B^{\gamma} \rightarrow \delta \bullet, l', l, k, fl'^*, fr'^*]$

$[A^{\gamma} \rightarrow \nu \bullet B^{\gamma} \omega, l', l, fl', fr', star, tl, bl]$

CAD1 ----------

$[A^{\gamma} \rightarrow \nu B^{\gamma} \bullet \omega, l', k, fl'^*, fr'^*, star, tl, bl]$

que hacen uso de ítems de la siguiente forma $[R, l, r, k, fl, fr]$ cuyo significado es deducible de los antecedentes de la regla **CAD**. Podemos observar que el mayor número de índices repetidos es ahora siete, correspondiéndose con las reglas **CMP**, **CAD0** y **CAD1**. Por tanto, la complejidad se reduce de $O(n^9)$ a $O(n^7)$.

5. *Conclusiones*

En su reconocedor, Nederhof aplica una mejora similar a la presentada anteriormente que reduce la complejidad de $O(n^7)$ a $O(n^6)$. En

el mismo trabajo sugiere la dificultad implícita de reducir a $O(n^6)$ el reconocedor de Schabes mediante técnicas similares a las utilizadas por él.

La adaptación presentada del reconocedor de Schabes consiste en un sistema deductivo con una correspondencia uno a uno entre las reglas de ambos reconocedores. Por tanto, podemos establecer que la diferencia entre ambos radica en el uso y significado que otorgan a los índices de los ítems.

Los ítems del reconocedor de Nederhof son de la forma

$[R, h, i, j, fl, f2]$

donde R es una regla CFG con punto según la notación anterior, y los índices $h \leq i \leq j$, $fl \leq f2$ son enteros (posiblemente no instanciados) en el rango $[0..n]$ (figura 3).

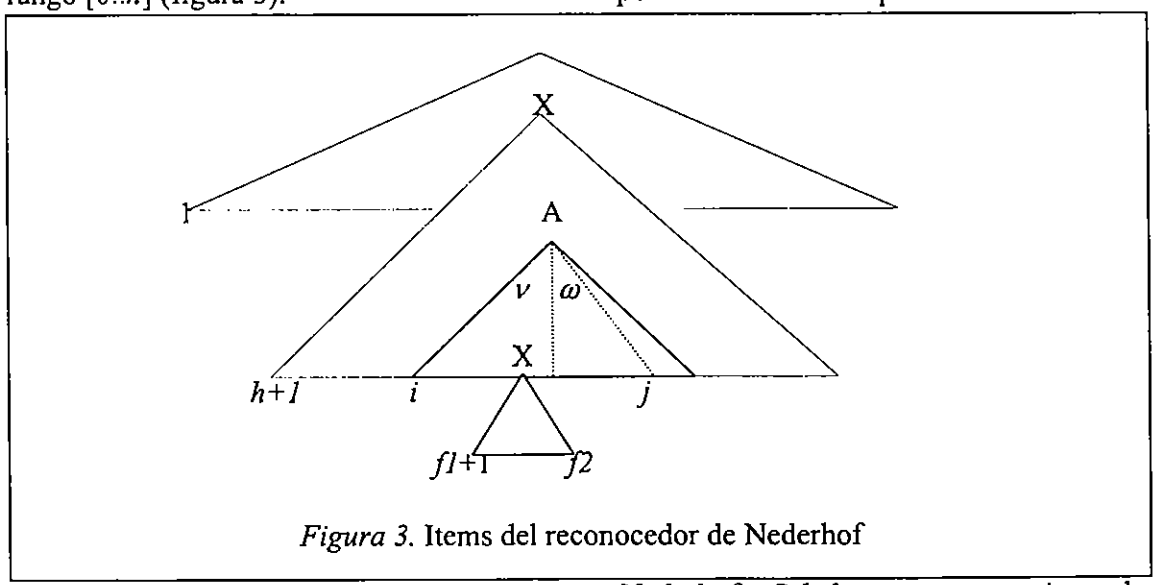

Figura 3. Items del reconocedor de Nederhof

Podemos observar que la semántica de algunos índices es la misma en ambos reconocedores: $h = l$, $j = k$, $fl = fl$ y $f2 = fr$. Nederhof incluye un índice i que no está presente en los ítems de Schabes. Del mismo modo, Schabes incluye los índices tl, bl que no están presentes en los de Nederhof. Puesto que el número de índices juega un papel significativo en el cálculo de la complejidad, Nederhof parte con ventaja al usar un índice menos.

La función del índice i en el reconocedor de Nederhof es indicar la posición de la cadena de entrada donde comenzó el reconocimiento de cada regla CFG. Por otro lado, los índices tl y bl que incluye Schabes tienen como objetivo determinar el contexto izquierdo de un auxiliar, tanto para predecir su parte derecha tras completar el subárbol escindido fruto de la adjunción, como para completar la propia adjunción.

Podemos establecer la relación existente entre ambos reconocedores mediante un nuevo reconocedor que, aunque sin interés práctico, nos establece el nexo entre ambos. Este reconocedor se define mediante ítems de la forma:

$[R, h, i, j, fl, f2, star, tl, bl]$

es decir, sus componentes son la unión de los componentes incluidos en los ítems de los reconocedores de Nederhof y Schabes. Las reglas deductivas del nuevo reconocedor son similares a las propuestas anteriormente en la revisión del reconocedor de Shabes, salvo que ahora reflejan la componente i, cuya semántica es la definida por Nederhof. Según lo expuesto, podemos concluir que los reconocedores de Nederhof y Schabes son *contracciones* de ítems **[Sik97]** del nuevo reconocedor.

Referencias

[ACCV99] Alonso, M.A.; Cabrero, D.; De la Clergerie, E.; Vilares, M. *Tabular Algorithms for TAG Parsing.* EACL'99, Bergen, Noruega (1999)

[DCT98] Díaz, V. J. ; Carrillo, V. ; Toro, M. *Analisis sintáctico de TAGs usando Analizadores Deductivos.* SEPLN'98 Alicante, España, (1998)

[JLT75] Joshi, Aravind K.; Levy, Leon S.; Takahashi, M. *Tree Adjunct Grammars.* Journal of Computer and System Sciences, 10(1): 136-162 (1975)

[Ned97] Nederhof, M. *Solving the Correct-Prefix Property for TAGs.* 5th Meeting of Mathematics of Language. Schloss Dagsthl

(Germany) 1997. Accepted for publication in Computational Linguistics Journal.

[Sch90] Schabes, Y. *Mathematical and Computational Aspects of Lexicalized Grammars*. Ph. D. University of Pennsylvania. Philadelphia. (1990)

[Sik97] Sikkel, K. *Parsing Schemata*. Springer-Verlag. (1997)

[SSP95] Shieber, S; Schabes, Y; Pereira, F. *Principles and Implementation of Deductive Parsing*. Journal of Logic and Computation, 24(1&2) pp 3-36 (1995)