UN ANALIZADOR SINTÁCTICO PARA GRAMÁTICAS ASOCIATIVAS POR LA IZQUIERDA

José Miguel Gómez
Dep. Matemática Aplicada a las TT.II.
E.T.S.I. Telecomunicación - U.P.M.
Ciudad Universitaria, Madrid

José Miguel Goñi
Dep. Matemática Aplicada a las TT.II.
E.T.S.I. Telecomunicación - U.P.M.
Ciudad Universitaria, Madrid
jmg@mat.upm.es

José Carlos González
Dep. Ingeniería de Sistemas Telemáticos
E.T.S.I. Telecomunicación - U.P.M.
Ciudad Universitaria, Madrid
jgonzalez@dit.upm.es

Madrid, Junio de 1994

Resumen

Esta comunicación presenta las posibilidades de las Gramáticas Asociativas por la Izquierda (LAG, *Left Associative Grammar*) para describir un lenguaje natural, en particular el castellano. Este formalismo gramatical surgió como una alternativa a las fórmulas tradicionales de enfocar el problema de la descripción de la sintaxis, como las Gramáticas de Estructura de Frase, o las Gramáticas Categoriales. Estos paradigmas tradicionales no recogen el hecho de que el lenguaje es inherentemente lineal en el tiempo. En consecuencia, el análisis de cadenas utilizando estas descripciones gramaticales necesita del concurso de grandes y complejas estructuras intermedias que lo faciliten, como *charts*, tablas LR, etc, al no ser eficiente el análisis directo.

Las Gramáticas Asociativas por la Izquierda permiten describir un lenguaje de una forma lineal (de izquierda a derecha), facilitando así el que el dispositivo computacional de análisis trabaje eficientemente -palabra por palabra, de izquierda a derecha- interpretando directamente las reglas de la gramática sin recurrir a la elaboración de estructuras intermedias. Se ha realizado una implementación de un analizador para gramáticas de este tipo, con una filosofía modular que permite su integración con otras aplicaciones. Simultáneamente se ha escrito una gramática del español que ha servido como prototipo para evaluar las posibilidades de este formalismo para describir algunos fenómenos lingüísticos de nuestro idioma.

1. Introducción

Las Gramáticas Asociativas por la Izquierda (LAG, *Left Associative Grammar*) fueron propuestas por

Reglas de Estructura de Frase (PSG) o de cancelación categorial. En ellos, la estructura de constituyentes se

organiza en forma de árbol, poniendo de manifiesto las relaciones de dominancia. Así, en un formalismo PSG

las reglas de reescritura son de la forma:

$$A \rightarrow B\ C$$

Mediante la sustitución (reescritura) del símbolo A por B y C, esta regla genera una estructura en árbol

en la que A domina a B y C, y conceptualmente representa una derivación que se realiza de forma *descendente* para formar la estructura de constituyentes.

Las reglas de cancelación categórica son de la forma (Zeevat, 1988):

$$\alpha_{(Y/X)} \odot \beta_{(X)} \rightarrow \alpha\beta_{(Y)}$$

$$\alpha_{(X)} \odot \beta_{(Y\backslash X)} \rightarrow \alpha\beta_{(Y)}$$

Estos esquemas de reglas combinan α y β en $\alpha\beta$ mediante la cancelación de X en la categoría de una de ellas con la correspondiente categoría de la otra. El resultado es también una estructura en árbol donde la categoría Y de $\alpha\beta$ domina a las de α y β. En este caso, el orden de derivación es conceptualmente *ascendente*.

En ninguno de estos casos se considera el hecho de que el lenguaje es inherentemente lineal en el tiempo: un oyente entiende el principio de una oración sin saber cómo va a seguir y, análogamente, un hablante puede decidir en medio de una oración cómo continuarla. Por este motivo, las gramáticas tradicionales no son adecuadas para el análisis directo de oraciones, por lo que éste ha de realizarse con el concurso de grandes estructuras intermedias que nada tienen que ver con la gramática: *charts* (Ritchie & Thompson, 1984), tablas LR (Tomita, 1986), etc., y que son consideradas el aspecto *procedimental* del análisis (Hausser, 1989).

Las gramáticas LAG presentan una serie de propiedades que las hacen especialmente atractivas para su tratamiento computacional:

1. Todas las propiedades importantes de una derivación están recogidas declarativamente en las reglas de la gramática, al tiempo que el aspecto *procedimental* del análisis no es más que una sencilla aplicación de dichas reglas (*type transparency*).
2. Están psicológicamente bien fundamentadas, al permitir únicamente derivaciones lineales en el tiempo, de izquierda a derecha y de abajo hacia arriba.
3. Son inherentemente decidibles, admitiendo además una implementación computacional eficiente.

2. Definición de LAG

Una Gramática Asociativa por la Izquierda puede definirse de manera formal (Hausser, 1989). Esto

permite derivar formalmente una serie de propiedades sobre el conjunto de lenguajes que estas gramáticas pueden describir y sobre su decibilidad: véanse (Hausser, 1991) y (Hausser, 1989). Aquí definiremos informalmente una LAG como un conjunto finito de reglas de la forma:

$$r_i : [C_1\ C_2] \rightarrow [rp_i\ C_3]$$

Esta regla, a partir de un principio de cadena (representado por su categoría C_1) y un nuevo elemento (representado por su categoría C_2), obtiene un nuevo principio de cadena (al que se le asigna la categoría C_3) y un estado representado por un paquete de reglas rp_i. Éste representa las posibles reglas que son aplicables en la siguiente composición asociativa por la izquierda: si la operación categórica de la regla r_i es satisfactoria, las reglas incluidas en el paquete de reglas rp_i pueden ser aplicadas al par formado por la categoría del nuevo principio de cadena (C_3), y la categoría del siguiente elemento. Así, un principio de cadena y un siguiente elemento se pueden combinar solo si (i) hay una regla que acepta sus respectivas categorías, y (ii) esta regla está contenida en el paquete de reglas "activado" por la última composición. El paquete de reglas representa por tanto una estructura de control que guía el proceso de derivación. La categoría derivada C_3 se obtiene a partir de C_1 y C_2 mediante la aplicación de una función recursiva[1].

Para poder realizar este proceso se cuenta con un diccionario de símbolos terminales del lenguaje, que tienen asignados una categoría formada por una secuencia de *segmentos categóricos*. Para poder iniciar una derivación es necesario especificar un conjunto finito de estados iniciales, cada uno compuesto de una categoría (que indica la de los posibles comienzos) y un conjunto de reglas aplicables para la misma. Una derivación es correcta si se consumen todos los símbolos terminales de la cadena de entrada y se llega a un estado final incluido en un conjunto finito de posibles estados finales -cada uno de ellos compuesto por una categoría y un paquete de reglas.

3. Un ejemplo

Este ejemplo muestra una gramática asociativa por la izquierda para el lenguaje formado por las

[1] Posteriormente se verá, a través de ejemplos, cómo puede especificarse esta función dentro de la regla.

cadenas de tipo $a^n b^n c^n$ (por ejemplo *abc*, *aabbcc*, *aaabbbccc*, etc). Se define como sigue:

LX = {["a" (b c)], ["b" (b)], ["c" (c)]}

STS = {[{r-1, r-2} (b c)]}

r-1: [(X) (b c)] => [{r-1, r-2} (b X c)]

r-2: [(b X c) (b)] => [{r-2, r-3} (X c)]

r-3: [(c X) (c)] => [{r-3} (X)]

STF = {[rp-3 ()]}

Los símbolos terminales del lenguaje son "a", "b" y "c", y los posibles segmentos categóricos son **b** y **c**. El léxico queda definido mediante el conjunto LX, con tres elementos formados por un símbolo terminal y una categoría definida como una lista de segmentos categóricos. El único estado inicial de STS especifica que el primer elemento debe ser de categoría **(b c)**, es decir, una "a". Además indica que las únicas reglas que pueden intentarse aplicar al empezar son **r-1** y **r-2**. Las reglas especifican las operaciones categóricas en términos de la variable conjunto[2] **X**, que puede contener cero o más segmentos categóricos, y los segmentos categóricos **b** y **c**. La regla **r-1** toma un principio de cadena de cualquier categoría (representada por **(X)**), y un siguiente elemento de categoría **(b c)** y deriva la categoría de salida añadiendo los segmentos **b** al principio y **c** al final de la categoría original de principio de cadena.

La regla **r-2** toma un principio de cadena tal que su categoría empiece por el segmento **b** y termine por el **c**, y un siguiente elemento de categoría **(b)** y deriva una categoría de salida suprimiendo el primer segmento de la categoría del principio de cadena original. De manera similar se comporta la regla **r-3**. Cada regla lleva asociada un paquete de reglas en su parte derecha: **rp-1 = {r-1, r-2}**, **rp-2 = {r-2, r-3}** y **rp-3 = {r-3}**, que representan las reglas aplicables a partir de ese momento. El único estado final posible (en STF) es una categoría vacía tras aplicar alguna de las reglas en **rp-3**.

3.1. Ejemplo de derivación

Si analizamos la cadena *aaabbbccc* con la gramática anterior, obtenemos la siguiente derivación:

1 *COMIENZO

(b c) a

[2] Esta variable representa cualquier secuencia de segmentos categóricos.

(b c) a

2 *r-1

(b b c c) a a

(b c) a

3 *r-1

(b b b c c c) a a a

(b) b

4 *r-2

(b b c c c) a a a b

(b) b

5 *r-2

(b c c c) a a a b b

(b) b

6 *r-2

(c c c) a a a b b b

(c) c

7 *r-3

(c c) a a a b b b c

(c) c

8 *r-3

(c) a a a b b b c c

(c) c

9 *r-3

() a a a b b b c c c

Cada composición asociativa por la izquierda se ha indicado mediante un número de orden, seguido del identificador de la regla aplicada. Cada categoría derivada a partir de las dos anteriores sirve como punto de partida para la siguiente composición asociativa. Junto a las categorías de principio de cadena se ha indicado en cada caso la porción de cadena analizada hasta ese momento.

En el comienzo del análisis se toma el primer elemento de la cadena, en este caso el elemento "a" de categoría (b c), y se comprueba si verifica alguno de los estados iniciales (sólo uno en este ejemplo). La categoría del elemento se compara con la del estado inicial, obtiéndose el primer "principio de oración" formado únicamente por ese elemento de la cadena. El paquete de reglas del estado inicial indica que se pueden probar las reglas **r-1** y **r-2**. El segundo elemento de la cadena, otra "a" de categoría (b c), hace el papel de "siguiente elemento". La única regla del paquete que podemos aplicar es **r-1**, tras lo cual obtenemos la categoría (b b c c) asociada a la porción de cadena analizada ("aa").

Este proceso sigue hasta que se terminan los elementos de la cadena. Si la categoría resultante es compatible con alguno de los estados finales definidos en STF (en el ejemplo sólo contiene la categoría vacía, que es la que se ha obtenido en el proceso), la cadena pertenece al lenguaje definido por la gramática. Como la derivación procede linealmente de izquierda a derecha, el árbol sintáctico que la derivación asigna a la cadena es directamente la secuencia de composiciones asociativas realizadas.

4. Gramáticas LAG para lenguajes naturales

Veamos cómo pueden usarse gramáticas LAG para describir lenguajes naturales, en particular el tratamiento de tres fenómenos básicos: valencia, concordancia y orden de las palabras.

Para escribir una gramática LAG del castellano hemos procedido en tres pasos:

1. Traducción de las propiedades morfosintácticas de cada palabra en una categorización adecuada.
2. Codificación de las composiciones locales (de principios de oración y palabras siguientes) en operaciones categóricas basadas en propiedades de concordancia y valencia.
3. Organización de las operaciones categóricas en una estructura de control simple, basado en reglas que invocan paquetes de reglas.

4.1. Concordancia

Se dice que dos expresiones en lenguaje natural están en concordancia si tienen la misma propiedad categórica en alguno de sus rasgos comunes, por ejemplo, concuerdan en género, o en número, o en persona, etc. La concordancia puede tener lugar junto con la cancelación de una posición de valencia, como veremos más adelante.

En LAG, los rasgos de concordancia están codificados en la categoría, usando para ello segmentos góricos específicos. Si, por ejemplo, consideramos que ese papel lo juega el segmento **k**, los esquemas de las para el manejo de la concordancia son:

$$r_i : [(X\ k\ Y)\ (k')] \rightarrow [rp_i\ (X\ k\ Y)]$$

$$r_i : [(k')\ (X\ k\ Y)] \rightarrow [rp_i\ (X\ k\ Y)]$$

El segmento **k** restringe la combinación a palabras con categoría **k'**. Como puede observarse, dicho mento no se cancela a la salida, con lo que se pueden añadir más palabras con el rasgo de concordancia **k'**. el primer esquema, el modificador sigue al modificado mientras que en el segundo el modificador precede modificado. En el siguiente ejemplo, tanto el determinante como el adjetivo deben coincidir en género y nero con el núcleo del sintagma nominal, el nombre:

el (DET M S) perro (NC M S) gordo (ADJ M S)

Los rasgos de concordancia se codifican mediante los segmentos categóricos: de género (**M**, masculino) úmero (**S**, singular), que coinciden con los que va marcado el nombre. Los adjetivos invariantes en género n un caso particular de esta norma:

el (DET M S) pescado (NC M S) azul (ADJ N S)

la (DET F S) pelota (NC F S) azul (ADJ N S)

Donde vemos que el segmento categórico de género **N** permite combinar el adjetivo tanto con un mbre masculino como femenino, siempre que alguna regla LAG lo valide. De manera similar puede recogerse concordancia de persona y número entre sujeto y verbo.

4.2. Valencia

Una expresión es transportadora de valencia si tiene una o más "posiciones" que se tienen que rellenar n cierto tipo de expresiones para estar completas. Las propiedades de valencia de una expresión son presadas en las LAG por medio de la categorización: se utilizan unos segmentos categóricos para indicar que a posición debe ser "rellenada" por una expresión de determinado tipo. Ello constituye una forma de ncordancia, aunque la diferenciamos de ésta en que sólo puede haber un elemento para rellenar cada posición valencia. La relación entre transportadores de valencia y sus "rellenadores" es manejada mediante reglas AG con la siguiente estructura:

$$r_i : [(X\ k\ Y)\ (k')] \rightarrow [rp_i\ (X\ Y)]$$

$$r_i : [(k')\ (X\ k\ Y)] \rightarrow [rp_i\ (X\ Y)]$$

Es decir, la posición de valencia k está restringida a elementos de categoría k'. La adición de este elemento tiene como resultado la cancelación de la posición de valencia. En el primer esquema, el transportador de valencia precede al elemento, mientras que en el segundo, lo antecede. Como ejemplo de valencia en castellano se puede citar el caso de los verbos transitivos, que necesitan obligatoriamente un complemento directo. éste puede estar situado antes o después del verbo:

lo (PRON_C M S CD) compro (V N S 1 _CD)

comieron (V N P 3 _CD) carne (NCI F S)

En el primer ejemplo, el "rellenador" precede al portador de la posición de valencia (indicado por _CD, complemento directo). En este caso, se utiliza el segmento CD para cancelar el _CD. En el segundo caso, el complemento directo que necesitaba "comieron", lo rellena "carne". Aquí no se emplea ningún segmento categórico especial para realizar la cancelación de _CD, sino que esta acción la llevará a cabo determinada regla de la gramática.

Si ocurriera que la oración terminase, y no se hubiera rellenado la posición de valencia, no se alcanzaría un estado final y, por tanto, la oración no se consideraría correcta. En castellano, el verbo no necesita al sujeto para formar una expresión completa bien formada, mientras que en inglés el verbo debe llevar una posición de valencia que se cancela con el correspondiente sujeto (que generalmente precede al verbo).

4.3. Orden de las palabras

El gran número de diferentes esquemas para las operaciones categóricas es una de las razones del poder descriptivo de las LAG. Cada lenguaje, sin embargo, utiliza sólo cierto tipos de operaciones categóricas. Si consideramos el ejemplo del inglés, la combinación sujeto-verbo debe producirse en ese orden, por lo que la regla que añada el verbo será del tipo:

$$r_i : [(k') (X\ k\ Y)] \rightarrow [rp_i (X\ Y)]$$

Es decir, el sujeto (el "rellenador") siempre precede al verbo (el portador de la posición de valencia). En el caso del verbo-complemento directo, ocurría al contrario. En castellano, hay mucha más libertad en cuanto a la posición de las palabras dentro de una oración. Por eso las reglas que manejan la valencia, deben ser capaces de anularla si el "rellenador" precede al portador de la posición, o de indicar que debe cubrirse más adelante.

Otro factor que influye en el orden de las palabras es la estructura de control que forman las reglas.

Si, por ejemplo, queremos obligar a que el determinante preceda al nombre, tenemos que incluir la regla que acepta el nombre en el paquete de reglas asociado a la regla en la que se acepta el determinante. Un esquema de reglas que muestra un orden rígido de las palabras es el siguiente:

$$r_1 : [(X)\ (a)] \rightarrow [\{r_2\}\ (X)]$$

$$r_2 : [(X)\ (b)] \rightarrow [\{r_3\}\ (X)]$$

$$r_3 : [(X)\ (c)] \rightarrow [rp_3\ (X)]$$

Este esquema asegura que un elemento de categoría (a) sea seguido por uno de categoría (b) y éste por uno de categoría (c), sin otra opción posible. Este esquema se ha utilizado en la gramática para el castellano cuando se utiliza el verbo "haber" (como auxiliar para las formas compuestas de los verbos): la única regla que se puede aplicar después es la que acepta como palabra siguiente una con categoría verbal.

5. Ejemplo de análisis en castellano

Para ilustrar todo lo comentado anteriormente, veremos a continuación un ejemplo de análisis de una sencilla oración en castellano. Para ello utilizaremos parte de la gramática desarrollada. La oración a analizar es la siguiente:

el perro come carne .

El proceso de análisis nos muestra la siguiente derivación:

1 *COMIENZO

(DET M S) el

(NCI M S) perro

2 *yNC

(NC M S 3) el perro

(V N S 3 _CD) come

3 *SUJyVERB

(V _CD) el perro come

(NCI F S) carne

4 *VERByCD

(V NCI F S) el perro come carne

(.) .

5 *yFIN

() el perro come carne .

La primera palabra de la oración es el determinante "el", definida en el diccionario mediante la entrada ("el", (DET M S)). El único estado inicial posible en el que encaja su categoría es:

[{DETyPOA, yNC, yADJ} (DET X1)]

Podremos aplicar, por tanto, cualquiera de las reglas indicadas en ese paquete. De entre ellas DETyPOA necesita un postartículo (segmento POA), yNC necesita un nombre común (segmentos NC, NCI o NCH), mientras que yADJ necesita un adjetivo (segmento ADJ), por lo que solamente se puede aplicar la regla etiquetada yNC al considerar que en la categoría de la siguiente palabra ("perro") está el segmento NCI que indica que se trata de un nombre común. Para ilustrarlo se lista a continuación dicha regla[3]:

```
DEF_REGLA
yNC: [(X1 seg6 seg1 seg2 X2) (seg7 seg3 seg4)]
     =>
     [{SUJyVERB, SUJyHABER, yADJ, yPRON_C, yREFL, NCySPREP,
          yREL_CD, yREL_SUJ, yCONJ, yFIN, VERByCD, VERByCI, VERBySPREP}
     (X3 NC seg5 seg2 3)]
COND_yNC = ((((seg1)==(N) @AND @NOT (seg3)==(seg1)
                                   @AND @IF ((seg1)==(N) @AND @NOT (seg3)==(seg1))
                                        seg5=(seg3))
          @OR ((seg3)==(N) @AND @NOT (seg3)==(seg1)
                                   @AND @IF ((seg3)==(N) @AND @NOT (seg3)==(seg1))
                                        seg5=(seg1))
          @OR ((seg1)==(seg3) @AND @IF (seg1)==(seg3)
                                        seg5=(seg1)))
     @AND (seg2)==(seg4)
```

[3] DEF_REGLA es una palabra clave que indica que comienza una definición de regla LAG. COND_yNC indica una serie de condiciones que deben cumplirse entre las variables que aparecen en las categorías de la regla y permiten el cómputo de la categoría de la parte derecha de la regla.

@AND (((seg7) @PER (NC NCI) @AND @NOT (_H) @PER (X1))

@OR (seg7)==(NCH))

@AND X3=((X1)-(_H))

@AND (((seg6)==(PRA) @AND (X2)==(_DE))

@OR (seg6) @PER (POA DET IND)))

Esta regla es aplicable a un comienzo de oración (categoría C_1) y un siguiente elemento (categoría C_2) siempre que C_1 contenga, tras un comienzo cualquiera (X1) un segmento (seg6) que sea PRA (cuando simultáneamente C_1 acabe con el segmento _DE) o bien POA, DET o IND -es decir, el nombre común debe seguir a un preartículo, postartículo, determinante o indefinido-. En nuestro ejemplo anterior, se trata del segmento DET.

C_1 contiene dos segmentos -que siguen a seg6- indicados por las variables segmento seg1 y seg2, que se refieren a los rasgos de concordancia de género y número. En nuestro caso estos segmentos toman los valores M y S respectivamente.

C_2 debe constar de tres segmentos. El primero denotado por seg7, deberá ser NC[4] o NCI[5] (en cualquiera de estos dos casos se requiere además que C_1 no requiera un nombre con el rasgo [+humano], lo que se indica con el segmento _H en la secuencia X1), o bien un segmento NCH[6]. Los otros dos segmentos de C_2 (seg3 y seg4) se refieren a los rasgos de concordancia (género y número). Las ecuaciones de la regla relativas a estos segmentos indican compatibilidad siempre que coincidan los segmentos de género o bien que uno de ellos esté sin especificar (valor N para el segmento de género), calculando el segmento de género para la categoría de la parte derecha (C_3). Una de las condiciones de la regla impone la igualdad entre los segmentos que codifican el rasgo número en las categorías C_1 y C_2, trasmitiéndola a la categoría C_3.

En la categoría C_3, la variable X3 será instanciada por la secuencia de los mismos segmentos que tuviera X1, excepto _H, que se elimina en caso de existir. El segmento NC queda incluido incondicionalmente

[4] Nombre común.

[5] Nombre común que puede ir sin artículo cuando funciona como complemento.

[6] Nombre común humano.

y a continuación los ya mencionados de género y número. Después se añade el segmento **3**, que codifica el rasgo **[tercera-persona]**, necesario para la concordancia con el verbo.

Retomando la derivación de nuestro ejemplo, y tras verificar cómo se obtiene la categoría **(NC M S 3)** para la porción de frase "el perro", se observa que la única regla aplicable del paquete de reglas asociado a la última aplicada es **SUJyVERB**. La siguiente palabra a considerar es "come" que aparece en el diccionario con la entrada: **("come" (V N S 3 _CD))**. La aplicación de la regla es similar a la anterior, con la particularidad de que el verbo "comer" es transitivo, con lo que se crea una posición de valencia para el complemento directo marcada con el segmento **_CD**. Así, la categoría de salida es **(V _CD)**.

Del paquete de reglas de **SUJyVERB**, únicamente se puede aplicar **VERByCD**. Aplicada con éxito esta regla, queda rellenada la posición de valencia del complemento directo, con lo que la categoría de salida es (V NCI F S). A continuación viene el signo de puntuación definido en el diccionario como **(".", (.))**, con lo que es aplicable la regla **yFIN**, que es usada en la gramática para validar una oración, comprobando que hay verbo, que las posiciones de valencia de los complementos directos, indirectos y atributos no están "pendientes de rellenarse", que las oraciones subordinadas relativas están completas, etcétera. Tras comprobar todos estos extremos se obtiene la categoría vacía () que, al estar incluida en el conjunto de posibles estados finales, indica que la oración es sintácticamente correcta.

Este ejemplo es una muestra de un análisis sencillo. En la mayoría de los casos, desde cada estado puede ser posible aplicar varias reglas, y además con varias categorías para una misma palabra, o incluso una misma regla de varias formas posibles. En la sección siguiente describiremos cómo se han abordado esas cuestiones en la implementación realizada.

6. Algunos detalles de implementación

La implementación del analizador se ha realizado siguiendo una metodología modular que facilita su integración con otras aplicaciones, así como su crecimiento y su mantenimiento. Se ha codificado en lenguaje C, definiéndose una interfaz de programación para las funcionalidades básicas que ofrece los módulos centrales del analizador (inicialización, carga de la gramática y del léxico, análisis, acceso a la estructura sintáctica resultante y funciones auxiliares de gestión). De esta manera se facilita su integración con otras funcionalidades de una manera rápida y sencilla.

En la implementación actual no se ha considerado análisis morfológico, para centrarnos en el sintáctico. De momento se necesitan en el diccionario todas las formas flexivas para un paradigma dado, cada una de ellas con sus rasgos pertinentes, aunque según lo expuesto anteriormente la integración de un analizador morfológico sería sencilla. El algoritmo del analizador se ha realizado como un intérprete directo de las reglas LAG, debido a la propiedad de *type transparency* ya citada. En cada instante el analizador tiene un estado definido por la categoría de comienzo de cadena y el paquete de reglas aplicables. A partir de ese momento cualquiera de estas reglas es aplicable para ese estado y la categoría o categorías[7] de la palabra siguiente. Ello hace que tengamos varias posibilidades a considerar, aunque algunas de ellas puedan llevar al analizador a una vía muerta. El análisis se desarrolla mediante un proceso de búsqueda en profundidad[8], ya que la profundidad máxima de la búsqueda es el número de palabras que forman la frase, no habiendo posibilidad de que el análisis no finalice. Es por ello que una pila y un evaluador de operaciones entre categorías constituyen la base del algoritmo.

Se ha visto en un ejemplo anterior cómo se especifican en las reglas las operaciones categóricas: las tres categorías que aparecen en una regla pueden representarse mediante una secuencia de variables y/o segmentos categóricos. Cada regla contiene restricciones que estos elementos deben cumplir para que la regla se pueda aplicar. Estas restricciones se expresan mediante una notación especial que permite:

- Asignar valores a las variables no instanciadas de la categoría de la parte derecha de la regla.
- Operar con secuencias de segmentos categóricos: unión, intersección, concatenación y eliminación selectiva de algún segmento.
- Realizar comprobaciones sobre los valores de los segmentos categóricos (comprobación de igualdad) o de secuencias de los mismos (comprobación de inclusión y de igualdad).
- Coordinar estas comprobaciones mediante operaciones lógicas (AND, OR NOT) y la estructura de decisión básica (IF-THEN-ELSE).

Se ha especificado una pequeña gramática para el español que ha servido para evaluar el analizador implementado y como prototipo para evaluar las posibilidades de las LAG para describir algunos fenómenos lingüísticos que aparecen en español. Se ha abordado con éxito, entre otros, las conjunciones de sintagmas nominales, de adjetivos, el anidamiento de oraciones subordinadas de relativo, las formas compuestas de los

---

[7] Pueden ser varias si la palabra presenta ambigüedad léxica.

[8] Se elige una de las posibilidades y se explora hasta sus últimas consecuencias, volviendo después para explorar los demás. En la literatura anglosajona se la conoce como *depth-first search*.

verbos y la elisión del sujeto, además de los fenómenos ya citados de concordancia, valencia y orden.

7. Conclusiones

La conveniencia de las gramáticas LA para la descripción de lenguaje natural se ha mostrado en análisis extensos de la sintaxis del Inglés y del Alemán, así como en pequeños sistemas de Francés, Latín clásico, Polaco y Japonés (Hausser, 1989). Estas aplicaciones demostraron que las LAG manejan diferentes tipos de estructuras características del lenguaje natural (orden de las palabras, concordancia, dependencias a larga distancia, etc) de forma simple. Sirva este trabajo para añadir, modestamente, el castellano a esta lista. En lo relativo a su procesamiento por ordenador, las gramáticas LA son fáciles de analizar porque las reglas de la gramática se usan directamente. Las gramáticas LA para lenguajes naturales (y para los formales también) son sencillas de escribir (aunque a primera vista puedan resultar aparatosas), de ampliar y de depurar, debido a que el análisis de las oraciones está basado en el principio de las posibles continuaciones. La gramática del castellano comenzó con unas pocas reglas básicas y se fue ampliando poco a poco. Las ampliaciones consistieron en añadir algunas reglas y retocar detalles de aquellas otras que se vieron afectadas. Finalmente, la propiedad fundamental de las LAG es su aproximación al lenguaje como un fenómeno estrictamente lineal en el tiempo. Dado que el estudio de las posibles continuaciones está lingüística y psicológicamente justificado, las excelentes propiedades en cuanto a procesamiento con ordenador de las LAG, proporciona incentivos adicionales para exploraciones más profundas de este nuevo enfoque para los lenguajes naturales y formales.

Referencias

[1] Gómez Apesteguía, José Miguel (1993). Análisis Sintáctico de Lenguaje Natural: Realización de un Analizador para Gramáticas Asociativas por la Izquierda. *Escuela Técnica Superior de Ingenieros de Telecomunicación, Universidad Politécnica de Madrid.*

[2] Hausser, Roland (1985). Left Associative Grammar and the Parser NEWCAT. *IN-CLSI-85-5. Center for the Study of Language and Information. Stanford University.*

[3] Hausser, Roland (1987). Left Associative Grammar: Theory and Implementation. *Technical Report, CMU-CMT-87-104. Center for Machine Translation. Carnegie Mellon University.*

[4] Hausser, Roland (1989). Computation of Language. An Essay on Syntax, Semantics and Pragmatics

in Natural Man-Machine Communication. *Springer-Verlag. Berlin-New York.*

[5] Hausser, Roland (1991). Complexity in Left-Associative Grammar. *Technical Report. Friedrich-Alexander Universität.*

[6] Ritchie, Graeme and Thompson, Henry (1984). Implementing Natural Language Parsers. *In Artificial Intelligence: Tools, Techniques and Applications, ch. 9, pp. 245-300. Harper and Row.*

[7] Tomita, Masaru (1986). Efficient Parsing for Natural Language. *Kluwer Academic Publishers.*

[8] Zeevat, Henk (1988). Combining Categorial Grammar and Unification. *In Reyle, U. and Rohrer, C., eds.: Natural Language Parsing and Linguistic Theories, pp. 202--229. Reidel.*