Un parser basado en la sintaxis de dependencias

J. Carlos Ruiz Antón
ISS (Barcelona)

Resumen

Este artículo describe el submódulo de análisis sintáctico-semántico del castellano en el proyecto MMI² de ESPRIT, cuyo objetivo es producir una interficie multimodal -lenguaje natural, gráficos y comandos- con un sistema experto en el diseño de redes locales) (Binot *et al.* 1990).

El analizador consiste en:

(i) una gramática de estructura de frase que, interpretada por un parser ascendente, produce representaciones funcionales de tipo semántico, y

(ii) un módulo que transfiere estas representaciones de dependencia en fórmulas de lógica de primer orden, susceptibles de ser interpretadas pragmáticamente por un gestor de diálogo.

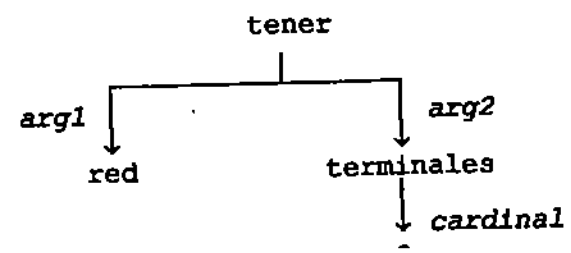
El parser está integrado con un módulo, denominado experto semántico, que contiene información sobre los tipos de objetos y sus relaciones en el dominio.

1. Representación de Dependencia Funcional

La representación funcional subraya las relaciones abstractas de dependencia que se dan entre las formas léxicas de los objetos lingüísticos (Kaplan y Bresnan 1982, Mel'čuk 1988).

Las estructuras de dependencia se representan habitualmente como grafos dirigidos, donde los nudos son elementos léxicos y los arcos indican relaciones funcionales. La figura (1) muestra una posible representación para la frase *la red tiene ocho terminales*:

(1)



(Leídos de arriba abajo, los arcos indican rección, y dependencia en el sentido contrario. Nótese que el grafo debe estar dirigido porque las relaciones sintácticas son antisimétricas.)

En toda teoría de dependencias, los grafos deben respetar una serie de condiciones de buena formación (que varían en los detalles según los autores). Las condiciones siguientes definen específicamente las representaciones usadas en nuestro sistema:

- Los nudos corresponden siempre a formas léxicas, anotadas con listas de rasgos.

- Cada nudo depende únicamente de otro nudo regente, excepto la raíz del grafo, que corresponde al núcleo semántico de la expresión (ie. el nudo *tener* en 1). Esta condición define matemáticamente este tipo de grafos como árboles.

No requeriremos una proyección biunívoca entre palabras de la expresión superficial y nudos de la estructura de dependencias. De acuerdo con este presupuesto, las palabras vacías (cópula), los operadores semánticos (artículo, cuantificadores, verbos modales) y otras palabras pertenecientes a clases léxicas cerradas (preposiciones, conjunciones) no se representarán como nudos, sino como rasgos del correspondiente núcleo.

En nuestro sistema, las representaciones de dependencia aparecen bajo la forma de estructuras complejas de rasgos (ECR). Formalmente, los rasgos son términos de la forma <ATRIBUTO>=<VALOR>, tal como corrientemente se usan en las gramáticas de unificación (Kay 1985, Shieber 1986). Por ejemplo, (2) es una traslación (incompleta) del grafo (1):[1]

(2)

```
[ HEAD = tener
  ARG1 = [ HEAD = red
           +DEF ]
  ARG2 = [ HEAD = terminal
           NUM  = pl
           CARD = 8 ] ]
```

Una ECR reproduce una estructura de dependencias si contiene un rasgo etiquetado HEAD. Los atributos ARG1, ARG2, ARG3 y ADJ contienen la

información de los argumentos y los adjuntos, respectivamente.[2] El resto de los rasgos presentes en la ECR se interpretan como anotaciones del núcleo.

Sin entrar en la discusión teórica sobre la adecuación del modelo de dependencias frente a la representación por constituyentes (sintagmática),[3] la opción que hemos adoptado aquí ofrece algunas ventajas. La más evidente es la facilidad de manejo: las estructuras de dependencias son más planas y mejor organizadas que las estructuras de constituyentes, de forma que es más fácil localizar y modificar la información contenida. Por ejemplo, en los formalismos de base sintagmática hay que propagar los rasgos pertinentes del núcleo a todas sus proyecciones (ie. N^1, SN, etc), para que sean accesibles. Mientras que en una representación de dependencia esto es innecesario, al no existir diferencia entre nudos terminales y nudos sintagmáticos.

2. Subcategorización

Uno de los ejes del presente sistema es la noción de **marco de valencia**. Los marcos son estructuras de datos que describen las restricciones selectivas de los argumentos regidos por un núcleo. Los marcos incluyen varias casillas (una por cada argumento) con información gramatical y semántica sobre los términos que deben ocupar dichas posiciones.[4]

Así, el marco de valencia de un verbo contendrá rasgos sobre la categoría o preposiciones regidas por sus argumentos, así como información semántica sobre el tipo de estos. Esta es una perspectiva lexicista: las entradas léxicas contienen en sí mismas el embrión de la predicación de la que son núcleo. Por ejemplo, la entrada léxica de la forma verbal *cuesta* será semejante a:[5]

---

[2]. Habría que incluir además una etiqueta para los elementos coordinados, si bien este punto aún no ha sido tratado extensamente.

[3]. Hay bastante bibliografía sobre la relativa excelencia de ambos modelos de representación. Véase a este respecto la polémica que enfrentó a Hudson y a Ö. Dahl en las páginas de *Linguistics*: Hudson (1980a), Dahl (1980) y Hudson (1980b).

[4]. Para una panorámica completa del tratamiento de la valencia en lingüística computacional, véase Somers (1987).

(3)
```
HEAD = costar
TIPO = estativo
TEMP = pasado
ARG1 = -prep
       TIPO = ente
       PERS = 3
       NUM  = sing
ARG2 = -prep
       TIPO = cuantia
```

3. Reglas gramaticales

3.1 Preliminares

El parser deduce las relaciones de dependencia a partir de una estructura de constituyentes.

El sistema utiliza básicamente reglas binarias y unarias que definen combinaciones válidas entre categorías. Estas reglas tienen el formato siguiente:

<RESULTADO> --> <CATEGORIA$_1$> [<CATEGORIA$_2$> [SI <ANOTACIONES>]]

Las <ANOTACIONES> son operaciones de manipulación de rasgos que se aplican tras completar la regla. Estas operaciones incluyen

- Unificación de las ECRs asociadas a cada hija
- Adscripción local de una hija
- Adscripción no local de una hija
- Adición de rasgos por defecto a una ECR
- Sobreescritura de rasgos en una ECR
- Evaluación de constricciones sobre los rasgos de una ECR

Estas anotaciones pueden combinarse por medio de los conectores lógicos $\wedge$ (AND) y $\vee$ (OR).

Véanse algunos ejemplos de reglas:

(4) a. n --> art-n SI unifican

b. s --> sn < sv SI attach(arg1 | arg2)

c. sv --> qu > sv SI mover(arg1 | arg2) $\wedge$ def(act=intp)

d. sn --> sn < cr / attach(adj)

e. sv --> sv < scomp / attach(arg2 | arg1)

(las categorías empleadas en las reglas son únicamente claves para el parser; carecen de cualquier significación teórica. En estos ejemplos su significado es: n = nombre; art = artículo; qu = elemento interrogativo; sv = sintagma verbal; sp = sintagma preposicional; sn = sintagma nominal; cr = cláusula relativa; scomp = cláusula completiva.

Una de las anotaciones indica qué tipo de combinación (unificación o adscripción) hay que aplicar entre las ECRs de ambas hijas (v. 3.2 y 3.3).

El sistema usa un algoritmo relativamente simple para la combinación incremental de las subestructuras de rasgos. Alcanza su objetivo cuando se obtiene una ECR única como representación de la expresión analizada.

3.2 Unificación

La unificación es la principal operación constructiva del parser. De forma general, la unificación agrupa toda la información de dos listas de rasgos. La operación falla si entre ellas hay rasgos incompatibles (para una definición mucho más precisa, v. Knight 1989).

El sistema usa una versión constructiva del procedimiento en PROLOG de Eisele y Dörre (1986), al que se ha añadido una extensión para tratar atributos de tipo conjuntivo (ie. aquellos que, con valores distintos, pueden repetirse en una expresión, como por ejemplo, ADJUNTO).

3.3 Adscripción

La operación de **adscripción** hace depender un nudo de otro, ya sea como argumento (cuando unifica con la especificación de alguna de las casillas del marco de valencia de un núcleo), ya sea como adjunto.

Existen dos tipos de adscripción:

- **local** (o **attach**): Uno de los nudos pasa a depender del otro, adoptando una de las funciones semánticas especificadas en la anotación.

- **no local** (o **mover**) hace depender un nudo N_1 de otro nudo N_2 o de alguna predicación dependiente de N_2. Esta operación cubre los casos de dependencia a larga distancia, como cláusulas relativas, interrogativas, topicalizadas o elevación (*raising*).

En esta clase de operaciones, es posible especificar la relación sintáctica (arg o adj) preferida por el nudo adscrito. Así, la anotación a la regla (4e) muestra que la ECR de la cláusula completiva (scomp) debe adscribirse como arg_2 (objeto) o arg_1 (sujeto) del núcleo (sv) (cf. respectivamente las expresiones *dijo que vendría, me sorprendió que viniera*).

Si el parser no puede adscribir un nudo como argumento de otro, tratará de adscribirlo como adjunto, aplicando una serie de restricciones semánticas que evalúan la categoría (por ejemplo, el hecho de que los adverbios no modifican SNs), el tipo semántico, o cualquier otra información pertinente sobre los participantes. Estas restricciones se expresan mediante un tipo especial de reglas declarativas, cuya forma es la siguiente (cf. Moens et al. 1988):

$$\text{<PAPEL>} == \text{<RESTRICCION}_{\text{NUCLEO}}\text{>} \;/\; \text{<RESTRICCION}_{\text{DEPENDIENTE}}\text{>}$$

Por ejemplo:

(5) a. dirección == [type=movimiento] / [type = concreto].
b. instr == [type=acción] / [type /= animado].
c. duración == [type=acción] / [type = medida_temporal].

Es en el léxico donde se determina la gama de papeles temáticos que pueden transmitir las preposiciones. Supongamos un atributo SEMANT con valor disyuntivo. Así, la entrada léxica de la preposición *con* incluirá

(6) PREP = con,
SEMANT = (compañía ; contenido ; modo ; instr)]

3.4 Evaluación de constricciones

Una extensión particularmente útil del mecanismo de evalución de que dispone el parser es un evaluador de constricciones. Permite comprobar si sobre una lista de rasgos dada se verifica una serie de constricciones (de aquí en adelante CC), referidas a la presencia o ausencia en ella de ciertos rasgos.

Se representa <NUDO> tiene <CC>, donde CC es una lista de constricciones.

Las constricciones posibles son:

(a) **Verificación de presencia de rasgos:**

* A=V. Comprobación de valor único. El rasgo especificado debe existir en la ER (V puede ser un símbolo atómico o una variable). Esta constricción admite las variantes +A, -A para rasgos booleanos.

* A=DISY. Comprobación de valor disyuntivo. DISY es una disyunción de la formá (V_1 o V_2 o ... o V_n). El atributo A debe tener uno de los valores de DISY).

(b) **Verificación de ausencia de rasgos:**

* A \= DISY . Esta constricción se cumple si A no tiene ninguno de los valores especificados en la disyunción DISY.

* NOT A . La ECR no contiene ningún rasgo con atributo A.

4. Implementación

LIBER está implementado en BIMprolog como parte del modulo de análisis de castellano del proyecto MMI[2] de ESPRIT. Se complementa con un analizador morfológico, y un submódulo responsable de traducir las representaciones de dependencia a fórmulas de lógica de primer orden reificada (Quine 1985). La traducción es un proceso bastante directo a partir de las ECRs, que incluye un cálculo de alcance de cuantificadores basado en Ioup (1975) y V. Dahl (1981).

REFERENCIAS

Binot, J.L., Falzon, P., Pérez, R., Peroche, B., Sheehy, N., Rouault, J. y Wilson, M. (1990) "Architecture of a multimodal dialogue interface for knowledge-based systems". Aparecerá en las Actas de la ESPRIT Conference, Bruselas (Diciembre 1990).

Dahl, Ö. (1980) "Some Arguments for Higher Nodes in Syntax". *Linguistics*, 18, pp. 485-488.

Dahl, V. (1981) "Translating Spanish into logic through logic". *Computational Linguistics*. 7.3 pp. 149-164.

Eisele, A. y Dörre J. (1986) "A Lexical Functional Grammar System in PROLOG". COLING, Bonn.

Hudson, R. (1980a) "Constituency and Dependency". *Linguistics*, 18, pp. 179-198.

Hudson, R. (1980b) "A Second Attack on Constituency: A Reply to Dahl". *Linguistics*, 18, pp. 489-504.

Hudson, R. (1984) *Word Grammar*. Oxford, Basil Blackwell.

Ioup, G. (1975) "Some Universals for Quantifier Scope". En J. Kimball, ed. *Syntax and Semantics*, vol. 4, Academic Press.

Kay, M. (1985) "Unification in Grammar", en V. Dahl y P. Saint-Dizier (eds.) *Natural Language Understanding and Logic Programming*, vol I, Elsevier.

Knight, K. (1989) "Unification: A multidisciplinary survey". *ACM Computing Surveys*, 21.1, pp. 93-124.

Mel'čuk, I.A. (1988) *Dependency Syntax: Theory and Practice*. New York, State University of New York Press.

Moens, M., Calder, J., Klein, E., Reape y Zeevat, H. (1989) "Expressing generalizations in unification-based grammar formalisms". *4th Conference of the European Chapter of the ACL*.

Quine, W. (1985) "'Events and reification". En LePore, E. y McLaughlin, B. (eds.) *Actions and Events. Perspectives on the Philosophy of Donald Davidson*. Oxford, Blackwell.

Shieber, S. (1986) *An Introduction to Unification-Based Approaches to Grammar*. Stanford, CSLI.

Somers, H. (1987) *Valency and case in Computational Linguistics*. Edinburgh University Press.