LA GRAMÁTICA FUNCIONAL DE DIK COMO MÓDULO DE GENERACIÓN SINTÁCTICA

J. Carlos Ruiz Antón

This paper explores the possibility of using Simon Dik's Functional Grammar (FG) as the base formalism for a computer-oriented sentence generator (much in the sense the term has, for example, in Machine Translation). In contrast to other implementations of FG, our approach aims at establishing a modular and declarative system, based on the well-established procedure of unification.

En este artículo se exploran las posibilidades de utilizar las ideas de la Gramática Funcional (GF) de S. Dik (1989) como formalismo básico de un generador oracional (en el sentido que tiene el término, por ejemplo, en traducción automática): un sistema computacional que acepta como entrada una representación semántica de cierto tipo, y produce como resultado una expresión gramaticalmente correcta en una lengua dada).

En este momento existe una primera versión, aun en fase de desarrollo. Está escrita en lenguaje PROLOG, y funciona tanto en PC (WINDOWS) como UNIX.[1]

En el diseño de este sistema se han pretendido alcanzar los siguientes objetivos:

(a) obtener una adaptación lo más fiel posible del espíritu de la teoría de la Gramática Funcional. Como consecuencia, se utiliza el mismo tipo de representación y de reglas que en el modelo estándar de la teoría (Dik 1989).

(b) implementar un sistema computacional *modular*, *declarativo* (separando de manera tajante los algoritmos y los datos en componentes distintos del programa) y *monotónico* (en el que los resultados que se obtienen del cálculo son independientes del orden en que se aplican las reglas). Hay que señalar que las implementaciones anteriores de la GF (en concreto el sistema ProfGlot descrito en Dik 1992) no satisfacen casi ninguno de estos requisitos.

(c) producir un sistema con el que llevar a cabo una comparación tipológica entre los diferentes procedimientos sintácticos de los que se sirven diferentes lenguas para expresar significados.[2]

1. El modelo de la GF

La GF es una teoría funcional de la sintaxis y la semántica de las lenguas naturales. En esta sección describiremos únicamente los aspectos del modelo que conciernen a la

[1] Se utiliza SWI-Prolog (versión 2.7.14).

[2] Como banco de pruebas se está experimentando con la elaboración de gramáticas para dos lenguas: farsi (una lengua indoeuropea del tronco iranio, con interesantes peculiaridades sintácticas), y maya (una lengua amerindia que exhibe ergatividad parcial).

generación de oraciones; para un conocimiento más completo el lector deberá consultar l bibliografía especializada.[3]

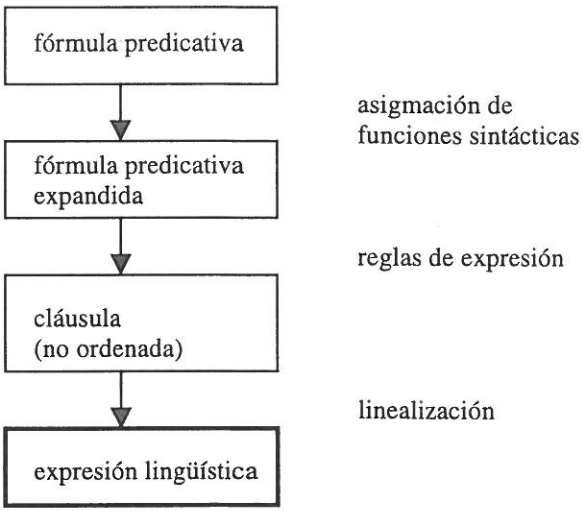
La GF representa la semántica de las oraciones mediante fórmulas predicativas bastant abstractas, donde los participantes en la acción (argumentos y adjuntos) están señalados pc su función semántica y pragmática. Por ejemplo, la representación de la frase *el hombr mató las serpientes con un palo* sería:

$$[\text{Pasado Pos } e_i \text{ matar}_V (\text{d } 1 : \text{hombre}_N(x_1))_{\text{AgTema}} (\text{d m } x_2 : \text{serpiente}_N(x_2))_{\text{GoFoc}} (\text{i } 1\ x_3 : \text{palo}_N(x_3))_{\text{Instr}} (e_i)\ ]$$

Esta predicación describe un acontecimiento e_i situado en tiempo pasado (Pasado) positivo (i.e. no negado). Este acontecimiento es una acción descrita por el predicado verb *matar*, con un agente (Ag) y un objetivo (Go). El agente es un individuo x_1 definido (d perteneciente a la clase nominal *hombre*, mientras que el objetivo es un conjunto múltipl (m) y definido de individuos de la clase nominal *serpiente*. El instrumento es un individu indefinido (i) de la clase *palo*. Obsérvese que la fórmula especifica también la funció pragmática de los términos: el agente es también el tema, y el objetivo está marcado com foco.[4]

Las fórmulas predicativas se componen fundamentalmente de predicados (*planta*, *agricultor*, *patata*) y de operadores (temporales, aspectuales, cuantificadores, de definitud de polaridad, etc.).

Según la teoría estándar de la GF, la proyección entre estas fórmulas predicativas y l forma superficial de las oraciones tiene lugar en una serie de etapas (véase gráfico):



[3] La obra de referencia fundamental es Dik (1989: especialmente cap.3). Dik (1991) es una introducción breve per muy comprensiva, mientras que Siewierska (1991) es un interesante manual crítico.

[4]. El tema se refiere a la entidad de la que se predica algo en una fórmula. El foco corresponde a aquella informació que es la más importante o destacada en un contexto cimunicativo dado (Dik 1989, §13.3).

- Asignación de funciones sintácticas (sujeto y objeto) a una parte de los constituyentes de la fórmula.
- Ordenación de los constituyentes. Esta operación se lleva a cabo aplicando un conjunto de **reglas de colocación** (*placement rules*) que asignan a los constituyentes una cierta posición dentro de un esquema funcional de ordenamiento (*functional pattern*).
- Aplicación de una serie de **reglas de expresión**, que especifican las formas que adoptan los constituyentes, con arreglo a su función sintáctica y a los operadores presentes en la fórmula. En la práctica estas reglas se ocupan de insertar elementos léxicos que corresponden a operadores, y de tratar morfológicamente las palabras de la oración.
- Procedimientos morfológicos de *sandhi*, que dependen del orden relativo de los elementos de una oración. (por ejemplo, el que en inglés selecciona la forma *an* del artículo indeterminado cuando la palabra siguiente comienza por vocal).

2. Arquitectura del generador

Esta sección describe con algo más de detalle los módulos del generador que ejecutan cada una de las tareas que se acaban de mencionar.

2.1 *Representaciones de entrada*

Las fórmulas predicativas que proporcionan la entrada (*input*) al generador se representan internamente como estructuras de rasgos, similares a las que se emplean en los formalismos de unificación como LFG (v. Bresnan 1982) o FUG (v. Kay 1984).

Las estructuras de rasgos son conjuntos de pares atributo-valor (*rasgos*), donde la posibilidad de disponer de valores complejos permite prescindir (con ventaja) de los clásicos árboles de estructura de frase. Además, resultan muy fáciles de implementar en PROLOG mediante listas (o alternativamente, mediante estructuras tipificadas; cf. el artículo de T. Badia en este mismo volumen). La utilización de estructuras de rasgos también permite usar el procedimiento de unificación durante las etapas sucesivas de la generación.

Véase, como ejemplo, la estructura funcional correspondiente a la fórmula predicativa (1):

(2) *El hombre mató las serpientes con un palo*

```
[op(tiempo)=pasado,
 op(polaridad)=positiva,
 ref=x1,
 pred=mortigi,
 arg1=[op(def)=d,op(cuant)=1,ref=x2,
pred=viro,fnc(pragm)=tema],
 arg2=[op(def)=d,op(cuant)=m,ref=x3,
pred=serpento,fnc(pragm)=foco],
 instr=[op(def)=i,op(cuant)=1,ref=x4, pred=bastono]
]
```

Los operadores están representados aquí como rasgos simples de tipo `op(X)`, donde es el tipo de operador (definitud, tiempo, polaridad, cuantificación, etc.). Los términos de predicación aparecen a su vez como rasgos complejos. En lugar de los papeles semántic características de la GF (agente, objetivo, destinatario, etc) se utilizan etiquetas semántic más abstractas: primer argumento (ARG_1) y segundo argumento (ARG_2), ya que much reglas gramaticales se pueden formular de manera más simple en términos de est funciones abstractas.[5] Hemos preferido, además, representar los predicados por medio lexemas de esperanto (*viro*, *mortigi*, *serpento*, etc.), una lengua que, entre otras ventaja resulta muy adecuada como *interlingua* semántica por su falta casi total de polisemia (c Hutchins y Somers 1992: § 17.2).[6]

2.2 *Componentes del sistema*

El generador está formado por una serie de componentes, que se aplican en el orden qu se establece a continuación:

1. ASIGNACIÓN DE FUNCIONES SINTÁCTICAS
2. INSERCIÓN LÉXICA
3. LINEALIZACIÓN
4. EXPRESIÓN MORFOLÓGICA

La principal diferencia con la arquitectura del modelo estándar de GF (Dik 1989) es qu las reglas de expresión originales se han dividido en dos tipos: **reglas de inserción léxic** (que se ocupan fundamentalmente de insertar los elementos léxicos que corresponden operadores en la fórmula subyacente, y que se deben aplicar necesariamente antes de l reglas de colocación de constituyentes) y **reglas morfológicas** (que se ocupan sobre todo c la flexión de las palabras (aplicadas después de linealizar la fórmula). De esta forma l reglas flexivas y las reglas de *sandhi* se pueden agrupar, de forma más natural, en el mism módulo morfológico.

2.3 *Un vistazo a los tipos de reglas*

Nuestra implementación separa de manera rigurosa las reglas que describen hech gramaticales de las reglas que describen operaciones computacionales. Por esta razón, l parte puramente algorítmica del sistema (los módulos denominados PLAC, para la sintaxi y MORPH, para morfología) pueden utilizarse, en principio, con datos gramaticales d lenguas de muy diverso tipo.

Al margen de las reglas morfológicas (que son poco pertinentes para la discusión y qu no abordaremos aquí), las reglas descritas tienen la forma siguiente:

A. Reglas de asignación de función sintáctica. Son reglas que determinan lo constituyentes que ocupan las funciones sintácticas de sujeto y objeto en cada cláusula. As la regla [3a] selecciona el segundo argumento (ARG_2) como sujeto si la fórmula predicativ

---

[5]. Véase también Perlmutter (1981) y Dik (1989, §5.3.1).

[6]. Por ejemplo, a la palabra *banco* le corresponden en esperanto los siguientes lexemas: *benko* (asiento), *rifo* (en dominio de la naútica), *banko* (empresa), *fisxarego* (banco de peces) y *stablo* (banco de carpintero).

no tiene primer argumento especificado (como en *la pintura ha sido restaurada con todo detalle*). En caso contrario, la regla [3b] opta por el primer argumento (ARG_1).[7]

(3a)
```
fsint(clausula, suj, X, [suj = A2,op(voz)=pasiva|X2]) :-
        unify_constraints([arg2,not(arg1)],X,X1),
        select(X1, arg2 = A2, X2).
```

(3b)
```
fsint(clausula, suj, X, [suj = A1|X1]) :-
        select(X, arg1 = A1, X1).
```

(3c)
```
fsint(clausula, obj, X, [obj = A1|X1]) :-
        select(X, arg2 = A1, X1).
```

B. Reglas de inserción léxica. El cometido de estas reglas es añadir elementos léxicos dada la presencia de determinados operadores semánticos o gramaticales en la fórmula predicativa (Algunos operadores están presentes en la fórmula inicial; otros operadores se introducen como resultado de alguna regla previa, como es el caso de [3a], que introduce el operador de voz pasiva). Un ejemplo (procedente de nuestra gramática de farsi) es la regla que introduce el determinante *yek* 'uno' si el término tiene el operador indefinido y es singular:

(4)
```
insert(termino, op(def)=i,_,[det=lex(num,yek,[])]) :-
        unificacion(op(cuant)=1,M).
```

Otras reglas introducen afijos y clíticos, que se adjuntan a la correspondiente base léxica durante la fase de generación morfológica. Así, la regla [5] introduce el afijo nominal *–hâ* cuando el término es plural (lo que corresponde al operador de cuantificación *m* = 'multiple'):

(5)
```
insert(termino, op(cuant)=m,_,[ plural=afijo(n,'hâ') ]).
```

Por último, otras reglas de inserción no lexicalizan operadores, sino que introducen algún elemento léxico en función de ciertos aspectos de la estructura de la fórmula. Un caso es la regla que inserta el verbo copulativo cuando la predicación tiene tiempo y el predicado es nominal o adjetivo:

(6)
```
insert( clausula, pred=P, M, [cop=lex(v,'COP',[]) ] ) :-
        unificacion( op(tiempo)= _,M),
        ( P = lex(adj,_,_); P = lex(n,_,_)).
```

Otras reglas específicas de farsi son la que introduce la posposición *râ* en la estructura del objeto si éste es un término definido:

(7)
```
insert(clausula, obj=term:DF,_,[]/[rel=lex(post,'râ',[])]) :-
        unificacion(op(def)=d,DF).
```

[7] En este y sucesivos ejemplos, las reglas se expresan como cláusulas de PROLOG.

Y la regla que inserta la preposición *ba* como marcador de la función semántica 'instrumento':

(8) `insert(clausula,`
`instr=cons(termino,_),_,[]/[rel=lex(prep,ba,[])]).`

C. Linealización. La linealización de las fórmulas predicativas se lleva a cabo en la GF por la interacción de dos tipos de restricciones: la información que proporcionan los patrones funcionales (*functional patterns*) y la influencia de una serie de tendencias universales de ordenamiento de constituyentes que Dik denominó LIPOC (*Language-Independent Preferred Order of Constituents*), que regula la preferencia que muestran las lenguas a colocar los constituyentes en orden de complejidad creciente.

Los patrones funcionales describen el orden de los elementos funcionales en un determinado dominio. Por ejemplo, el siguiente patrón describe la estructura de la cláusula en farsi:

(9) cláusula: REL P1 SUJ OBJ ADJS ATR $NEG $CONC $TIEMPO PRED SUB

Donde REL es la posición de los relatores y nexos, como *ke* (que intorduce subordinadas completivas); P_1 es la llamada posición inicial, reservada en la GF para constituyentes de especial relevancia pragmática, como por ejemplo, el tema (v. Dik 1989: §17.2); SUJ y OBJ se refieren, respectivamente a las posiciones de sujeto y objeto; ADJS cubre la posición de diversos complementos oblicuos, adjuntos y adverbiales; ATR es la posición del adjetivo atributivo; los elementos marcados con el símbolo '$' son morfológicamente ligados, y se afijan a la base léxica en la última etapa de la generación. En el caso del ejemplo, los afijos de negación, concordancia y tiempo presente se adjuntan a la base verbal siguiente (PRED). Por último SUB es la posición de las cláusulas subordinadas (incluyendo las completivas).

En el esquema, cada posición sólo puede ser ocupada por un constituyente, excepto ADJS, que admite más de un ocupante. El procedimiento de unificación es el encargado de que esta restricción se cumpla.

Los elementos de una fórmula se linealizan de acuerdo con la función que tienen asignada (ya sea léxicamente o como resultado de la aplicación de las reglas de asignación de función sintáctica). Hay un serie de reglas de colocación (*placement rules*) que cambian la función 'primitiva' de algunos elementos, si se satisfacen ciertas condiciones.

Veamos algunos ejemplos de reglas de colocación en nuestra gramática de farsi:

(10a) `coloc(cl,Fn=cons(clausula,_),_,sub).`

(10b) `coloc(cl,Fn=cons(_,C),_,p1) :-`
`unificacion(fnc(pragm)=tema,C).`

(10c) `coloc(cl,instr=_,_,adjs).`

(10d) `coloc(cl,pred=lex(Cat,W,M),_,atr) :- (Cat=adj; Cat=n).`

Las reglas especifican primero el patrón al que corresponden (aquí `cl` = cláusula), el elemento de la fórmula predicativa sobre el que se aplican, y su función resultante en el patrón.

Un importante problema que afecta a la implementación del generador es determinar la secuencia de aplicación de las reglas de ordenamiento. El método preferido en varias de las implementaciones de la GF (Connolly 1983; Bakker 1989; Dik 1992) es una ordenación no estricta siguiendo de izquierda a derecha las expectativas del patrón. De seguir esta estrategia, en nuestro esquema de cláusula se aplicarían primero las reglas que colocan los relatores (REL), luego las reglas que sitúan constituyentes en P_1, y así sucesivamente.

La opción que se adoptado en la implementación que describimos en este artículo es distinta: consiste en aplicar las reglas de colocación a partir de los rasgos, según el orden (en sí irrelevante) en que aparezcan en su estructura funcional. Esta aplicación producirá una estructura de rasgos parcial, que deberá unificar con el producto de la aplicación sucesiva de las reglas de colocación sobre los demás rasgos de la estructura. De esta manera, la unificación impide que una posición pueda estar ocupada por más de un constituyente, con excepción de las posiciones abiertas, como ADJS (v. *supra*).

3. Un ejemplo

Veamos ahora un ejemplo de cómo tiene lugar la generación de una frase en la actual implementación. Tomaremos como punto de partida la estructura de rasgos (2), correspondiente a la fórmula predicativa (1). El contenido de esta fórmula puede parafrasearse como "el hombre mató las serpientes con un palo".

La primera operación consiste en 'traducir' los valores léxicos del rasgo PRED a los correspondientes lexemas en farsi. Una serie de reglas muy simples llevan a cabo la tarea:

```
dicc( mortigi,lex(v,kosht,[])).
dicc(viro,lex(n,mard,[sem=animado(hum)])).
dicc(bastono,lex(n,chub,[sem=inanimado])).
dicc(serpento,lex(n,mâr,[sem=animado(nohum)])).
```

El resultado es la estructura de rasgos [11]:

(11)
```
[op(tiempo)=pasado,
 op(polaridad)=positiva,
 ref=x1,
 pred=lex(v,kosht,[]),
 arg1=[op(def)=d,op(cuant)=1,ref=x2, fnc(pragm)=tema,
       pred=lex(n,mard,[sem=animado(hum)])],
 arg2=[op(def)=d,op(cuant)=m,ref=x3,
pred=lex(n,'mâr',[sem=animado(nohum)]),fnc(pragm)=foco],
```

```
instr=[op(def)=i,op(cuant)=1,ref=x4,
       pred=lex(n,chub,[sem=inanimado])] ]
```

Etapa 1: Asignación de funciones sintácticas. La estructura de rasgos [11] quedaligeramente modificada al asignar la función sintáctica de sujeto (SUJ) al término ARG1 (por la regla 3b) y de objeto (OBJ) al término ARG2 (según la regla 3c):

```
(12) [op(tiempo)=pasado,
     op(polaridad)=pos,
     ref=x1,
     pred=lex(v,kosht,[]),
⇒    subj=[op(def)=d,op(cuant)=1,ref=x2,
           pred=lex(n,mard,[sem=animado(hum)]),fnc(pragm)=tema],
⇒    obj=[op(def)=d,op(cuant)=m,ref=x3,
          pred=lex(n,mâr,[sem=animado(nohum)]),fnc(pragm)=foco],
     instr=[op(def)=i,op(cuant)=1,ref=x4,
            pred=lex(n,chub,[sem=inanimado])] ]
```

Etapa 2: Inserción léxica. Las reglas se inserción se aplican recursivamente: en la estructura principal y en todas las subestructuras subordinadas. En el nivel más alto (cláusula) se inserta la posposición *râ* en el objeto definido (por regla 7), y la preposición *ba*, como marca de la función semántica de instrumento (regla 8). En el nivel de los términos, se introduce el afijo plural *-hâ* en el objeto (regla 5), y el determinante indefinido *yek* en el instrumento (regla 4):

```
(13) [op(tiempo)=pasado,
     op(polaridad)=pos,
     ref=x1,
     pred=lex(v,kosht,[]),
     subj=[op(def)=d,op(cuant)=1,ref=x2,
     pred=lex(n,mard,[sem=animado(hum)]),fnc(pragm)=theme],
⇒    obj=[rel=lex(post,râ,[]), plural=af(n,'hâ'),
          op(def)=d,op(cuant)=m,ref=x3,
          pred=lex(n,mâr,[sem=animado(nohum)]),fnc(pragm)=focus],
⇒   instr=[rel=lex(prep,ba,[]),det=lex(num,yek,[]),
           op(def)=i,op(cuant)=1,ref=x4,
           pred=lex(n,chub,[sem=inanimado])]]
```

Etapa 3: Linealización. La aplicación de las reglas de linealización sitúa a cada uno de los elementos sintagmáticos y léxicos de la estructura en una determinada posición dentro de los patrones. Estas reglas se aplican también de forma recursiva.

Las reglas de colocación sitúan al sujeto temático en la posición P1 (regla 10b) y al instrumento en la posición ADJS (regla 10c). Las funciones OBJ y PRED se sitúan en sus posiciones predefinidas, según el patrón [9].

Para la linealización de los términos P1, OBJ y ADJS se aplica el patrón nominal, que (simplificado) tiene la forma siguiente:

(14) Nominal: REL DET $PL PRED ADJ POST

Donde REL es la posición de los relatores, DET de los determinantes, $PL la del afijo de plural, PRED la del núcleo nominal (predicado), ADJ el adjetivo, y POST la posposición. La tabla siguiente muestra la ordenación de los correspondientes términos de acuerdo con el patrón:

	NEX	DET	$PL	N	ADJ	POST
P1 (SUBJ)				PRED (*mard*)		
OBJ			PLURAL (*hâ-*)	PRED (*mâr*)		REL (*râ*)
INSTR	REL (*ba*)	DET (yek)		PRED (*chub*)		

Por tanto, el resultado de la linealización es la lista de palabras siguiente (simplificada):

```
[ [mard], ['hâ', mâr râ], [ba, yek, chub], kosht]
```

Etapa 4: Expresión morfológica. En esta etapa se genera la forma superficial de los lexemas de la expresión, de acuerdo con las reglas morfológicas de la lengua en cuestión. En el ejemplo únicamente es preciso afijar el sufijo *–hâ* a su base (*mâr*), operación que se lleva a cabo por medio de una operación muy general de afijación en el módulo MORPH. Otras operaciones posibles en esta fase son la adjunción de clíticos (cf. castellano `[dar, me, lo]` → `dármelo`), las contracciones (`[de,el]` → `del`), y los casos de sandhi (por ejemplo en farsi el prefijo negativo *na-* añade uan consonante epentética -y- ante vocal: `[pref(na), avard]` → `nayavard`). Como resultado de esta última fase, el generador produce la expresión final *Mard mârhâ râ ba yek chub kosht.*

4. Conclusión

Esta artículo presenta un nuevo concepto de generación que aprovecha las posibilidades de las reglas de orden de la GF. Este enfoque hace uso de esquemas funcionales de orden y de reglas de colocación, y nos parece preferible (tanto desde un punto de vista teórico como metodológico) a otros tratamientos basados en series de transformaciones (como por ejemplo MacCord 1989), ya que facilita un tratamiento declarativo y fácilmente extensible de las gramáticas de generación.

Referencias

D. Bakker, "A formalism for functional grammar expression rules". En Connolly, J.H. y S Dik (eds.) *Functional grammar and the computer.* (Dordrecht 1989) 45-63.

J. Bresnan (ed.), *The mental representation of grammatical relations.* (Cambridge, Mass. 1982).

J. H. Connolly, "Placement rules and syntactic templates". En S. C. Dik (ed.) *Advances i functional grammar* (Dordrecht 1983) 247-266.

S. Dik, *The theory of functional grammar. part I: the structure of the clause* (Dordrecht 1989).

S. Dik, *Functional grammar in PROLOG* (Berlín/Nueva York 1992).

W. J. Hutchins y H. L. Somers, *An introduction to Machine Translation* (Londres 1992).

M. Kay "Functional unification grammar: a formalism for machine translation", *COLING* 84 (1984) 75-78.

M. C. McCord, Michael "Design of LMT: a Prolog-based machine translation system" *Computational Linguistics* 15.1 (1989) 33-52.

A. Siewierska, *Functional grammar* (Londres 1991).