*Estudo comparativo dos formalismos gramaticais DCG e LFG**

Thiago Alexandre Salgueiro Pardo

Sumário

A abordagem da pesquisa em andamento visa à exploração comparativa de alguns formalismos gramaticais para o processamento de línguas naturais.

Neste relatório, são apresentados e comparados os formalismos gramaticais Gramática de Cláusulas Definidas e Gramática Léxico-Funcional, para avaliação da possibilidade de redução entre eles. Além disso, é mostrada uma interface computacional desenvolvida para redução entre DCG e LFG.

* Trabalho realizado sob a orientação da Prof. Lucia Helena Machado Rino.

1. Introdução

Neste relatório, são explorados os formalismos gramaticais Gramática de Cláusulas Definidas (Definite-Clause Grammar – DCG) (Araribóia, 1989; Clocksin and Mellish, 1981; Shieber, 1986) e Gramática Léxico-Funcional (Lexical-Functional Grammar – LFG) (Kaplan and Bresnan, 1982; Shieber, 1986; Silva, 1996), os quais são linguagens artificiais para a representação dos recursos de uma língua. Esses formalismos são usados, em geral, para a modelagem computacional de processos de análise e síntese de línguas.

O objetivo da pesquisa relatada aqui foi comparar os formalismos citados, assim como verificar possíveis reduções entre eles, ou seja, dada uma especificação em DCG, deseja-se saber se é possível reescrevê-la como uma especificação em LFG (Whitelock et al., 1987). Para isso, foram explorados recursos computacionais para implementação de cada formalismo em separado, mediante simulação de construções lingüísticas particulares.

Nas seções que se seguem são apresentados os formalismos DCG (Seção 2) e LFG (Seção 3), a comparação entre suas propriedades e a avaliação da possibilidade de redução entre os mesmos (Seção 4). Uma interface computacional para redução entre DCG e LFG é apresentada na Seção 5. A conclusão desse estudo é apresentada na Seção 6. O Apêndice 1 contém as noções gerais do Cálculo de Predicados (Clocksin and Mellish, 1981), essencial para a especificação e manipulação computacional da DCG no ambiente Prolog, enquanto que o Apêndice 2 exemplifica a relação existente entre a DCG e o Prolog.

2. Gramática de Cláusulas Definidas (Definite-Clause Grammar – DCG)

2.1. Conceitos básicos

Criada por Pereira e Warren, a DCG foi desenvolvida como uma ferramenta de modelagem lingüística, sendo incorporada diretamente ao Prolog. Seu formalismo é flexível e permite a representação de qualquer estrutura sentencial que possa também ser representada por uma gramática livre de contexto (GLC)

A Figura 1 ilustra as regras gramaticais para a análise ou síntese de sentenças S-V-O (Sujeito-Verbo-Objeto) ou S-V (Sujeito-Verbo intransitivo). Formalmente, cada regra da DCG tem a forma de uma regra de produção, isto é, "X→Y", onde X é um símbolo não terminal e Y é uma forma sentencial. Componentes sentenciais conjugados são separados por vírgulas, como em:

sentença → sintagma_nominal, sintagma_verbal.

enquanto que componentes sentenciais disjuntos são representados por cláusulas distintas, como em:

sintagma_verbal → verbo.
sintagma_verbal → verbo, sintagma_nominal.

Regras da forma "X→[Y]" indicam a correspondência de um símbolo não-terminal X com um símbolo do vocabulário, ou item lexical Y. Símbolos não-terminais que introduzem itens lexicais são também chamados de símbolos pré-terminais.

sentença → sintagma_nominal, sintagma_verbal.
sintagma_nominal → determinante, substantivo.
sintagma_verbal → verbo.
sintagma_verbal → verbo, sintagma_nominal.
determinante → [o].
substantivo → [homem].
substantivo → [lápis].
verbo → [perdeu].
verbo → [canta].

Figura 1

Assim, o termo sentença é o símbolo inicial da gramática e, segundo as quatro primeiras regras da Figura 1, genéricas para a língua portuguesa, uma sentença pode assumir a forma de um sintagma_nominal seguido de um sintagma_verbal, um sintagma_nominal pode assumir a forma de um determinante seguido de um substantivo e um sintagma_verbal pode assumir tanto a forma de um verbo intransitivo, quanto de um verbo seguido de um sintagma_nominal, ou seja, um verbo transitivo direto.

Dessa forma, sentenças como “o homem perdeu o lápis”, cuja representação hierárquica correspondente é apresentada na Figura 2, poderiam ser analisadas ou geradas pela gramática ilustrada na Figura 1.

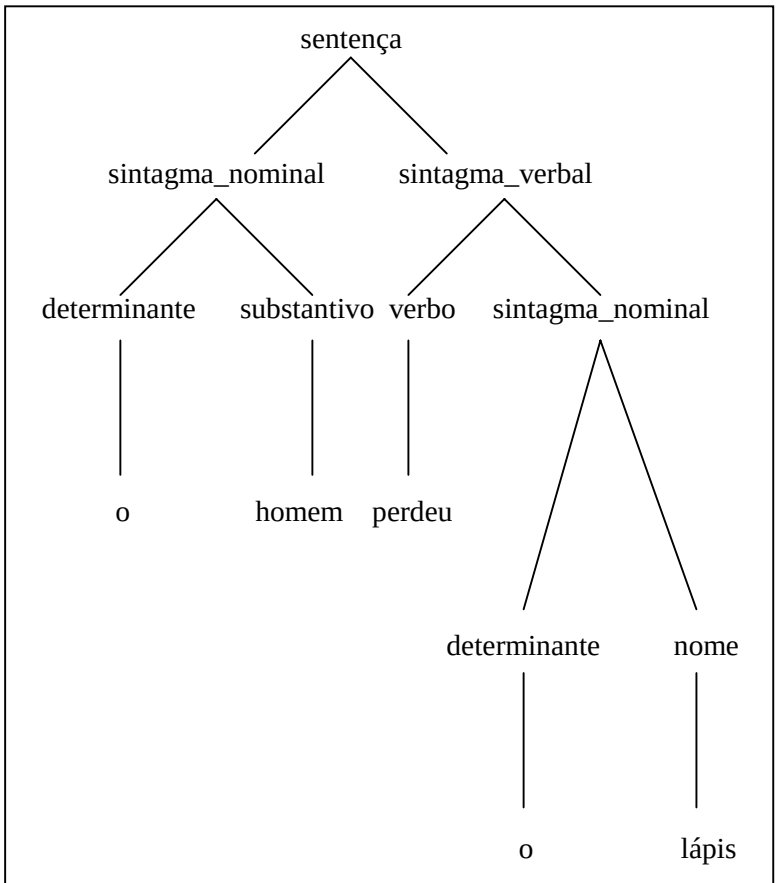


Figura 2

Programas de computador que constroem esse tipo de representação para sentenças de uma língua, conhecida como árvore de análise, são chamados de analisadores. Assim, o problema de construir uma árvore de análise pode ser chamado de problema de análise. A DCG permite resolver problemas de análise, indicando se sentenças de uma certa língua são corretamente estruturadas ou não. Para a resolução de problemas de síntese, isto é, de geração de sentenças, a mesma DCG pode ser utilizada. No entanto, será necessário incluir em suas regras restrições de seleção semântica, para que as sentenças geradas tenham sentido.

DCGs podem representar cláusulas variadas do Cálculo de Predicados , desde que sejam circunscritas a cláusulas de Horn, ou seja, as cláusulas do Prolog puro. Assim, elas possuem poder similar ao poder de representação de construções lógicas no Cálculo de Predicados (ver Apêndice 1 para detalhes). Por essa razão, DCGs podem ser diretamente incorporadas a um ambiente Prolog, transformando-se em programas. Por exemplo, a regra de produção genérica X→Y que, logicamente, pode ser interpretada por "X se Y", é reescrita em Prolog de forma simplificada, como X:-Y. Na verdade, as regras de reescrita de DCGs em programas Prolog envolvem, ainda, a inclusão de termos relacionados a cada símbolo não-terminal, sendo que estes serão símbolos de predicados em Prolog. No Apêndice 2, é apresentado um exemplo de conversão DCG-Prolog.

Justamente por serem fundamentadas no mesmo modelo teórico do Prolog, isto é, no Cálculo de Predicados, é que DCGs podem ser expressas diretamente como programas em Prolog e, portanto, podem passar a ser objetos do processamento lógico nesse ambiente. Assim, os processos de análise e síntese de sentenças passam a ser processos de inferência:

- na análise, espera-se que o mecanismo de inferência do Prolog determine a validade de uma estrutura (ou seja, se uma estrutura é gramatical ou não);
- na síntese, o mecanismo de inferência irá buscar combinações de itens lexicais, para produzir sentenças gramaticais.

2.2. O aumento da complexidade com o uso de argumentos extras

As regras gramaticais vistas até então são bastante simplificadas e genéricas. É possível ter extensões úteis que permitem às regras terem argumentos extras para, por exemplo, tratar restrições de gênero e número existentes entre os constituintes de uma sentença, o que é chamado de dependência contextual. Assim, sentenças como:

* O homem cantam.
* Os homens canta.

não são gramaticais devido à discordância de número entre sujeito e verbo. Poder-se-ia expressar este fato em regras gramaticais representando-se dois tipos de sentenças (no singular e no plural). Uma sentença no singular deve começar com um sintagma nominal no singular, o qual deve ter um substantivo no singular, e assim por diante. Deste modo, o conjunto de regras pode ser expresso como na Figura 3.

sentença → sentença_singular.
sentença → sentença_plural.
sintagma_nominal → sintagma_nominal_singular.
sintagma_nominal → sintagma_nominal_plural.
sentença_singular → sintagma_nominal_singular, sintagma_verbal_singular.
sintagma_nominal_singular → determinante_singular, substantivo_singular.
sintagma_verbal_singular → verbo_singular.
sintagma_verbal_singular → verbo_singular, sintagma_nominal.
determinante_singular → [o].
substantivo_singular → [homem].
substantivo_singular → [lápis].
verbo_singular → [perdeu].
verbo_singular → [canta].

Figura 3

Um conjunto de regras similar seria definido para sentenças no plural. Esta não é uma forma elegante, e além do mais, ignora o fato de que sentenças no singular e no plural têm muitas estruturas em comum. Para se restringir o número de regras, um modo melhor é associar argumentos extras com os termos da sentença, indicando se eles estão no singular ou no plural. Assim, sentença(singular) representa um termo que é uma sentença no singular e, no geral, sentença(X) uma sentença de número X. As regras, então, apresentam consistência sobre os valores desses argumentos, e deste modo, a dependência contextual adequada é inserida na gramática. A concordância de número entre um sintagma nominal na função de sujeito e o sintagma verbal deve ser observada. De modo similar, a DCG permite restringir gramáticas pela inclusão de termos de diversas naturezas. A redefinição da gramática apresentada na Figura 3, considerando a inclusão de termos restritivos, é apresentada na Figura 4.

sentença → sentença(X).
sentença(X) → sintagma_nominal(X), sintagma_verbal(X).
sintagma_nominal(X) → determinante(X), substantivo(X).
sintagma_verbal(X) → verbo(X).
sintagma_verbal(X) → verbo(X), sintagma_nominal(_).
determinante(singular) → [o].
substantivo(singular) → [homem].
* substantivo(_) → [lápis].
verbo(singular) → [perdeu].
verbo(singular) → [canta].

Figura 4

O símbolo ‘_’ em (*) indica que o número do termo pode ser tanto singular como plural. Assim, o substantivo lápis pode assumir as duas formas.

Os argumentos podem ser usados não somente para descrever a concordância de número entre os termos de uma sentença, mas também para outros tipos de relação que se queira estabelecer. Pode-se ter, por exemplo, argumentos que guardem um registro dos constituintes que tenham sido encontrados durante a análise de uma sentença, retornando a árvore de análise da mesma, como mostra a Figura 5.

```
sentença(sentença(SN,SV)) → sintagma_nominal(SN), sintagma_verbal(SV).
sintagma_nominal(sintagma_nominal(D,N)) → determinante(D), substantivo(N).
sintagma_verbal(sintagma_verbal(V)) → verbo(V).
sintagma_verbal(sintagma_verbal(V,SN))→ verbo(V), sintagma_nominal(SN).
determinante(determinante(o)) → [o].
substantivo(nome(homem)) → [homem].
substantivo(nome(lápis)) → [lápis].
verbo(verbo(perdeu)) → [perdeu].
verbo(verbo(canta)) → [canta].
```

Figura 5

Desse modo, a análise da sentença "o homem perdeu o lápis" permitirá que a estrutura sentença(SN, SV) na primeira regra gramatical, seja associada à estrutura ilustrada na Figura 6, que é a representação hierárquica da sentença em questão.

```
sentença(
        sintagma_nominal(
                        determinante(o),
                        substantivo(homem)
                        ),
        sintagma_verbal(
                        verbo(perdeu),
                        sintagma_nominal(
                                        determinante(o),
                                        substantivo(lápis)
                                        )
                        )
        )
```

Figura 6

Com a introdução dos argumentos extras nas regras não se pode garantir que a linguagem definida pela gramática ainda seja livre de contexto, porém ela normalmente é. (Clocksin and Mellish, 1981).

2.3. O léxico na DCG

Os exemplos ilustrados até agora admitem regras do tipo "X→[Y]", sendo Y um item lexical, para a representação de itens do vocabulário terminal. Essa não é uma boa forma de se representar, pois a variação do domínio implica a reescrita de todas as regras em um processo exaustivo, fazendo com que qualquer gramática de uma língua apresente um grau de complexidade alto e um número excessivo de regras gramaticais.

Uma maneira mais interessante seria expressar a informação comum sobre todos os itens lexicais por meio de processos genéricos, isolando-os de processos específicos para itens lexicais particulares. É possível fazer isso incluindo diretamente nas regras gramaticais instruções em Prolog. Por exemplo, o reconhecimento de um substantivo N em determinado instante da análise, cujo resultado é a estrutura subst(N), como termo do não-terminal "substantivo" na regra abaixo, é verdadeiro se for comprovado que N é substantivo. Esta condição é expressa pelo programa {é_substantivo(N,S)} na mesma regra, o qual pode abranger um número indeterminado de itens de vocabulário.

substantivo(S,subst(N)) → [N], {é_substantivo(N,S)}.

é_substantivo(homem,singular).
é_substantivo(banana,singular).
é_substantivo(bananas,plural).

As chaves, neste caso, são usadas para indicar programas diretamente escritos em Prolog e, portanto, indicam a forma como se faz a compilação de DCG em Prolog (Apêndice 2).

Essa regra diz que um termo do tipo substantivo pode tomar a forma de qualquer palavra N que esteja na coleção "é_substantivo" com alguma definição de número S (singular ou plural). Essa mudança ainda não é muito elegante, pois são necessários dois predicados "é_substantivo" para cada novo item lexical que exige flexão de número. Isso é desnecessário para a maioria dos substantivos que permitem variação regular de número, que pode ser expressa por uma simples regra do tipo:

"Se X é a forma no singular de um substantivo, então a palavra formada pela adição de um 's' no fim de X é a forma no plural daquele substantivo."

Com esta nova regra, um novo conjunto de condições será acrescentado, as quais um substantivo que observa tais restrições deve satisfazer. Como a representação das palavras está na forma de átomos em Prolog, devem ser feitas considerações sobre como as palavras decompõem-se em letras. Isso é necessário para que se possa isolar a letra 's' no final de uma palavra para posterior remoção, resultando na forma singular da palavra. A decomposição será feita pelo predicado "name", pré-definido do Prolog. O predicado "append", também pré-definido, serve para dividir uma lista em duas listas que juntas formam a lista original. Assim, uma lista com a palavra sem o 's' final é gerada. A nova regra é:

```
substantivo(plural, subst(RaizN)) → [N],
                                    {( name(N, PlSubstantivo),
                                    append(SingSubstantivo, "s", PlSubstantivo),
                                    name(RaizN, SingSubstantivo),
                                    é_substantivo(RaizN, singular) )}.
```

Essa regra expressa uma forma geral dos plurais. As exceções devem ser listadas uma a uma ou através de outras regras.

2.4. A representação lógica na DCG

A Figura 7 apresenta uma gramática cujas regras são usadas para se obter uma estrutura lógica da sentença, designada por P. Essa representação pretende indicar o raciocínio lógico e, assim, deve corresponder a fórmulas lógicas do Cálculo de Predicados, como especificado no Apêndice 1.

```
sentença(P) → sintagma_nominal(X,P1,P), sintagma_verbal(X,P1).
sintagma_nominal(X,P1,P) → determinante(X,P2,P1,P), substantivo(X,P3).
sintagma_verbal(X,P) → verbo(X,Y,P1), sintagma_nominal(Y,P1,P).
determinante(X,P1,P2,todo(X,(P1->P2))) → [todo].
determinante(X,P1,P2,existe(X,(P1&P2))) → [uma].
substantivo(X,homem(X)) → [homem].
substantivo(X,mulher(X)) → [mulher].
verbo(X,Y,ama(X,Y)) → [ama].
```

Figura 7

Neste caso, a quantificação universal, como na sentença “todo homem ama uma mulher”, será expressa pela seguinte estrutura:

todo(X,(homem(X) -> existe(Y,(mulher(Y)&ama(X,Y)))))

2.5. Outros tipos de estruturação na DCG

Um caso muito interessante de estruturação de regras gramaticais surge ao se analisar o caso das sentenças imperativas. Supondo que se queira analisar a sentença “coma seu jantar”, ela é interpretada como tendo o sujeito omitido, ou seja, sua forma explícita seria “você coma seu jantar”. Para que esse fenômeno seja representado por uma DCG, é necessário acrescentar as seguintes regras:

sentença → imperativa, sintagma_nominal, sintagma_verbal.
imperativa → [ ].
imperativa, [você] → [ ].

Assim sendo, a terceira regra “consome” o item lexical “você”, produzindo a forma imperativa correta. Essa forma de estruturação de regras é permitida, ou seja, a aplicação de uma regra, em determinado instante da análise, sem que qualquer símbolo da cadeia de entrada seja consumido. Deste modo, a gramática acima deixa de ser livre de contexto, mas nada impede que ela seja reescrita para representar este caso. Logicamente, para se ter uma sentença imperativa correta também é necessário que o verbo esteja flexionado corretamente, o que é possível adicionando-se argumentos a essas regras.

3. Gramática Léxico-Funcional (Lexical-Functional Grammar – LFG)

3.1. Conceitos básicos

Desenvolvida por Kaplan e Bresnan (1982) a partir da teoria lingüística léxico-funcional, a LFG enfatiza a representação mental dos constituintes gramaticais e as restrições universais das línguas naturais, orientando a análise às funções gramaticais e ao léxico, daí o nome de gramática léxico-funcional.

Segundo Allen (1987), as teorias lingüísticas estão principalmente interessadas em produzir uma especificação formal da estrutura lingüística, ou seja, elas devem caracterizar os princípios de organização geral que servem de base para todas as línguas humanas. Portanto, sendo a LFG desenvolvida a partir da teoria lingüística léxico-funcional, ela deve ser capaz de representar a maioria das características de uma língua. A LFG é um formalismo elegante (Shieber, 1986) por apresentar regras gramaticais simples e incorporar os aspectos complexos à representação do léxico.

3.2. A gramática e o léxico

A partir da especificação de uma gramática e de um léxico para uma língua, a LFG atribui duas estruturas sintáticas a cada sentença analisada desta língua. A primeira estrutura produzida diz respeito à configuração estrutural da sentença e é chamada de estrutura de constituintes (estrutura-c), sendo esta derivada diretamente das regras gramaticais. Analisando a estrutura-c, as descrições funcionais da sentença são produzidas, as quais terão papéis determinantes para a construção da segunda estrutura, a estrutura-funcional (estrutura-f), que por sua vez representa as funções gramaticais presentes na sentença e suas relações. Todas as passagens necessárias para a análise de uma sentença são ilustradas pela Figura 8.

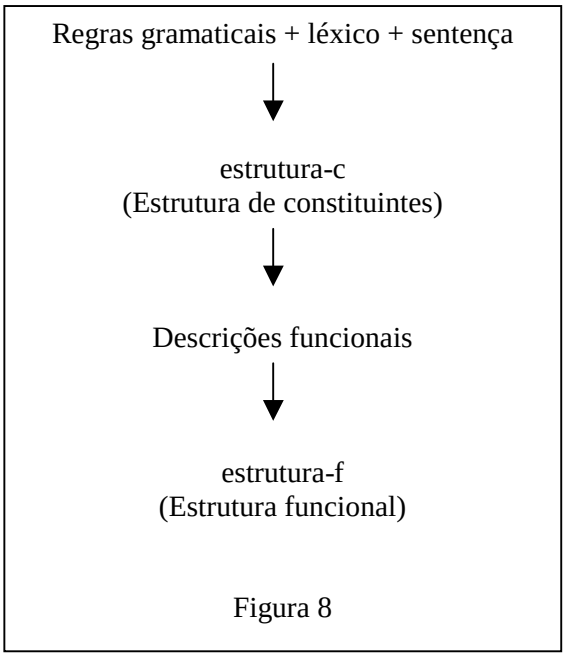


Figura 8

As regras gramaticais são expressas por regras livres de contexto, com algumas modificações, a saber:

- a inclusão obrigatória de esquemas funcionais associados aos termos não terminais das regras gramaticais, que servem para especificar a construção das estruturas sintáticas;
- a possibilidade de inclusão de condições e/ou restrições aos termos não terminais das regras gramaticais.

Por exemplo, na regra gramatical a seguir, os esquemas funcionais são representados por (↑Sujeito=↓) e (↑=↓), enquanto que a condição é representada por (↓Pro caso)=reto e a restrição por $(\uparrow \text{flexionado})_{\mathbf{r}}$.

sentença	→	sintagma_nominal	sintagma_verbal
		(↑Sujeito)=↓	(↑=↓)
		(↓Pro caso)=reto	$(\uparrow \text{flexionado})_{\mathbf{r}}$

As setas ↑ e ↓ são chamadas de metavariáveis de dominância imediata, pois são variáveis intermediárias entre as regras gramaticais e as estruturas produzidas e indicam a hierarquia entre os constituintes da sentença. Considerando o esquema (↑Sujeito)=↓ associado ao sintagma nominal, ele indica que o conteúdo do sintagma nominal (representado por ↓) é imediatamente dominado pelo termo sentença (representado por ↑), além de especificar que o sintagma nominal constitui a estrutura sujeito na sentença. O esquema (↑=↓) associado ao sintagma verbal indica que, da mesma forma que o esquema associado ao sintagma nominal, o conteúdo do sintagma verbal (representado por ↓) é imediatamente dominado pelo termo sentença (representado por ↑).

Na LFG, a diferença fundamental entre condição e restrição é que a condição pode não ser satisfeita, enquanto que a restrição deve ser satisfeita. A restrição $(\uparrow \text{flexionado})_{\mathbf{r}}$ (diferenciada de uma condição pela presença do 'r' associado) especifica que o verbo presente no sintagma verbal deve estar flexionado, enquanto que a condição (↓Pro caso)=reto especifica que, se o conteúdo do sintagma nominal for um pronome (indicado por Pro), ele deve ser do caso reto. A seta ↓ se associa ao léxico para garantir que o caso do pronome seja reto.

Enquanto que as regras gramaticais indicam a formação permitida aos constituintes de uma sentença, o léxico da LFG representa as palavras próprias de uma língua, juntamente com

suas categorias gramaticais e todas os traços desejados, como número, gênero, pessoa, tempo verbal e transitividade. A especificação do item lexical "comprou" a seguir considera as características número, pessoa, tempo e transitividade:

comprou: verbo, (↑número)=singular
(↑pessoa)=terceira
(↑tempo)=passado
(↑pred)='comprar<(↑Sujeito)(↑Objeto)>'

A transitividade do verbo "comprar" é indicada pelo valor associado ao esquema (↑pred), o qual especifica que o verbo, cuja forma infinitiva é "comprar", exige uma estrutura Sujeito e uma Objeto, sendo assim, transitivo direto.

O esquema (↑pred) representado em especificações lexicais e incorporado às estruturas sintáticas, pode ser utilizado em uma posterior análise semântica, por algum modelo semântico adequado. Já as setas ↑ dos esquemas indicam que o item lexical ao qual elas pertencem poderá se associar a um símbolo pré-terminal das regras gramaticais.

A Figura 9 ilustra um exemplo simples de gramática, podendo-se acrescentar a essa gramática a condição de que, se o sujeito for um pronome, ele deverá ser do caso reto (Figura 10), e/ou a restrição que o verbo deve estar flexionado (Figura 11).

S → SN SV
(↑Suj=↓) (↑=↓)

SN → (Det) Substantivo

SV → Verbo SN
(↑Obj=↓)

Figura 9

S → SN SV
(↑Suj=↓) (↑=↓)
(↓Pro caso)=reto

SN → (Det) Substantivo

SV → Verbo SN
(↑Obj=↓)

Figura 10

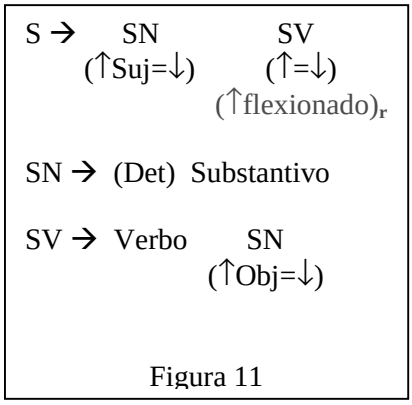


Figura 11

Os parênteses que envolvem o termo Det indicam que esse termo é opcional.

A Figura 12 ilustra um exemplo de léxico para a sentença “o homem ganhou o prêmio”.

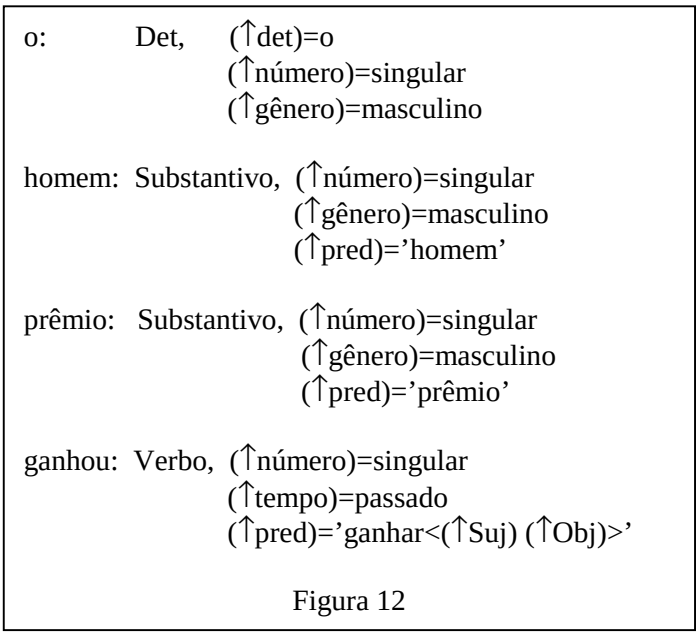


Figura 12

3.3. A estrutura de constituintes

A estrutura de constituintes, ou estrutura-c, pode ser representada por uma árvore de análise hierárquica. Essa estrutura pode ser determinada pela análise de uma sentença, com base nas regras léxico-funcionais. Considerando a gramática da Figura 9 e o léxico da Figura 12, a estrutura-c resultante da análise da sentença “o homem ganhou o prêmio” é ilustrada na Figura 13.

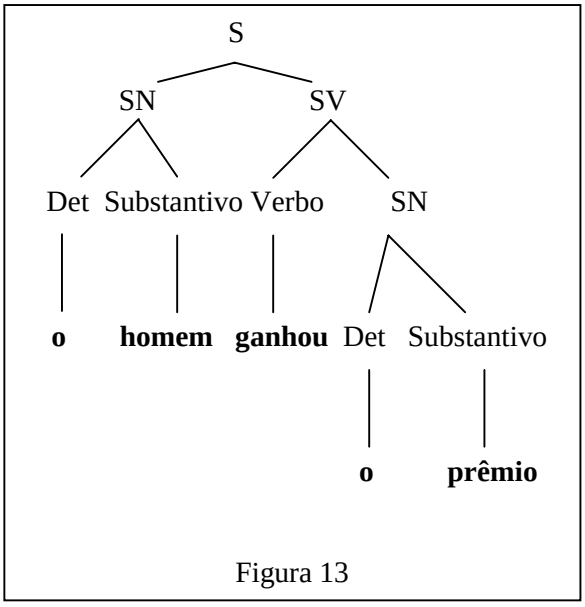


Figura 13

3.4. A descrição funcional

As descrições funcionais permitem construir um passo intermediário entre a estrutura de constituintes e a estrutura funcional de uma sentença. Elas servem para analisar se a estrutura funcional de uma sentença possui todas as propriedades requeridas pela gramática à qual a sentença pertence e são derivadas da estrutura-c seguindo os três passos seguintes:

- esquemas funcionais são associados a cada termo da estrutura-c segundo a gramática e o léxico especificados;
- variáveis únicas são associadas aos símbolos não-terminais presentes na estrutura-c;
- as metavariáveis dos esquemas funcionais são substituídas pelas variáveis associadas aos termos no passo anterior, formando as descrições funcionais.

Assim, por exemplo, considerando a estrutura-c da Figura 13, o primeiro passo associa os esquemas presentes na gramática da Figura 9 a seus respectivos símbolos não-terminais e as especificações lexicais presentes na Figura 12 aos símbolos pré-terminais, resultando na estrutura da Figura 14.

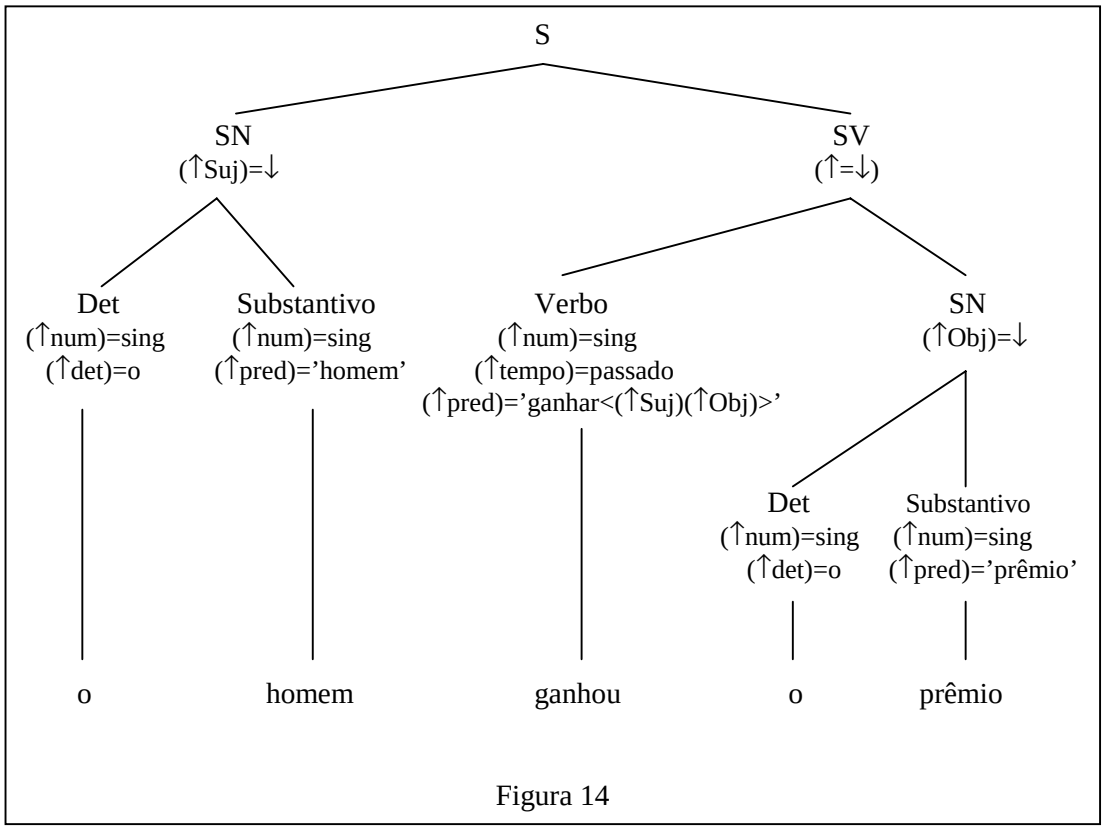


Figura 14

A Figura 15 ilustra a associação de variáveis (f_i) aos símbolos não-terminais. Assim, o símbolo raiz S e os símbolos que possuem a metavariável ↓ associada a eles na estrutura-c acima recebem um identifcador único.

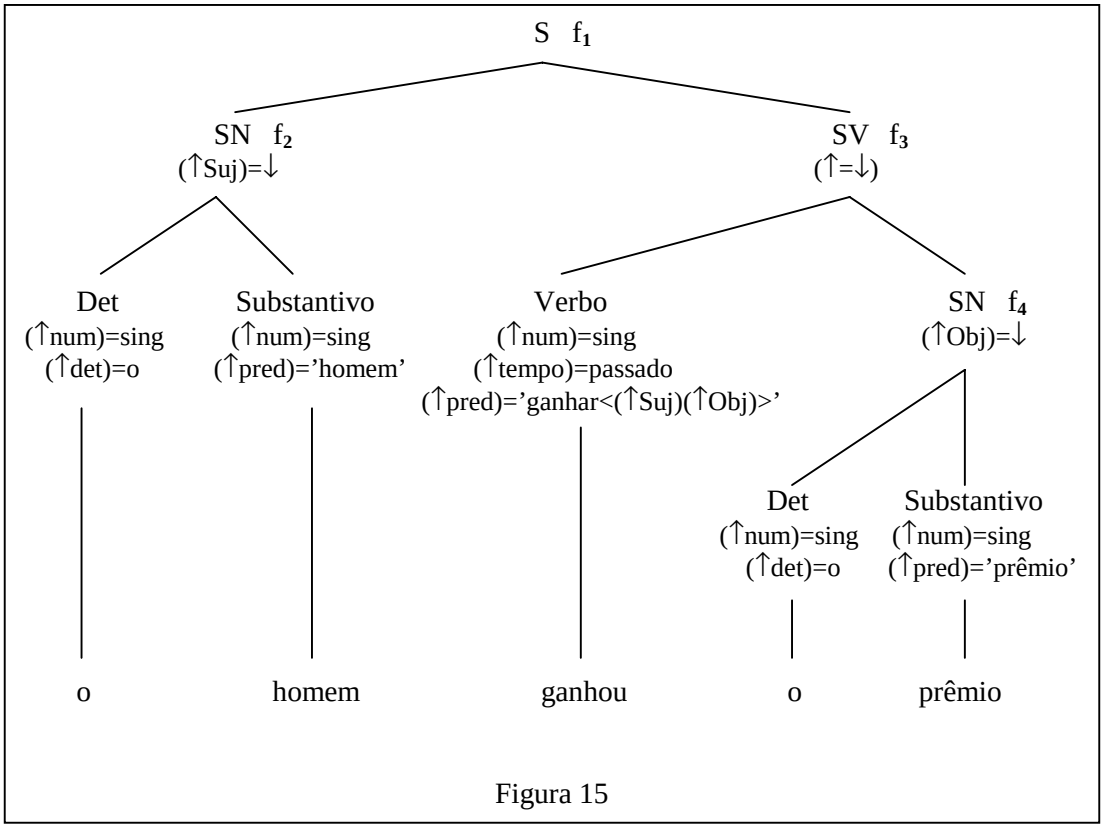


Figura 15

O terceiro passo coleta os esquemas da estrutura-c da figura acima e substitui cada metavariável indicada por ↑ pela variável associada ao termo que imediatamente domina o termo a que ela pertence, enquanto que as metavariáveis indicadas por ↓ são substituídas pela variável do próprio termo a que elas pertencem. Desse modo as descrições funcionais produzidas são:

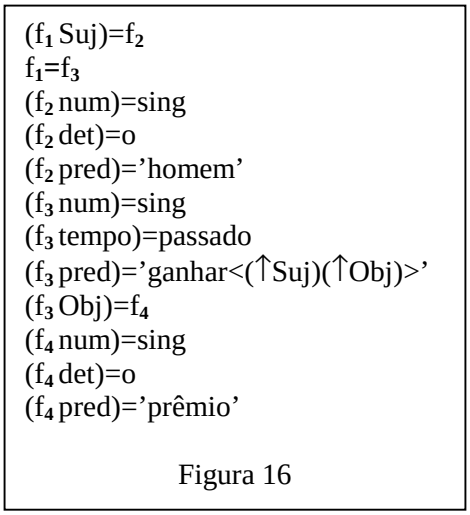


Figura 16

As descrições funcionais produzidas devem ter valores únicos, ou seja, para uma sentença ser gramatical não podem ocorrer, por exemplo, as descrições (f_2 num)=sing e (f_2 num)=plural, as quais definem números diferentes para a estrutura f_2 que representa o sujeito da sentença. É desse modo, portanto, que uma LFG permite estabelecer a concordância de número.

3.4. A estrutura funcional

A estrutura funcional, ou estrutura-f, de uma sentença é construída a partir de sua estrutura-c e de suas descrições funcionais. Assim, por exemplo, a partir da estrutura-c da Figura 13 e das descrições funcionais da Figura 16, é possível construir a estrutura funcional da sentença "o homem ganhou o prêmio", como é ilustrada pela Figura 17.

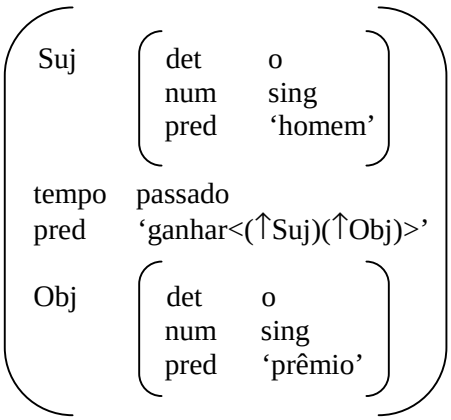


Figura 17

3.5. Outros níveis de representação

A LFG também permite expressar disjunção, repetição e conjuntos de constituintes. Por exemplo, um sintagma nominal, que pode ser constituído por um determinante opcional seguido por um substantivo ou somente por um substantivo próprio, terá como regra gramatical associada a expressão:

$$SN \rightarrow \left\{ \begin{array}{l} (Det)\ \ Substantivo \\ SubstantivoPróprio \end{array} \right\}$$

onde as chaves {} representam a disjunção. Já na presença de vários sintagmas preposicionais, a regra gramatical que expressa essa particularidade é, por exemplo:

$$\begin{array}{lccc} SV \rightarrow & Verbo & SN & SP^* \\ & & (\uparrow Obj)=\downarrow & \downarrow E(\uparrow Adjuntos) \end{array}$$

sendo que * indica a repetição, incluindo a possibilidade de zero ocorrências de SP. Para o caso de conjuntos de constituintes, a representação na LFG é dada, por exemplo, pelo esquema $\downarrow E(\uparrow Adjuntos)$ associado ao termo SP da regra acima, indicando que todo SP será incluído em um conjunto chamado Adjuntos. Para esses níves de representação, a partir da aplicação das regras léxico-gramaticais correspondentes, sentenças como “o homem ganhou o medalhão de ouro do João” seriam analisadas, resultando na estrutura da Figura 18.

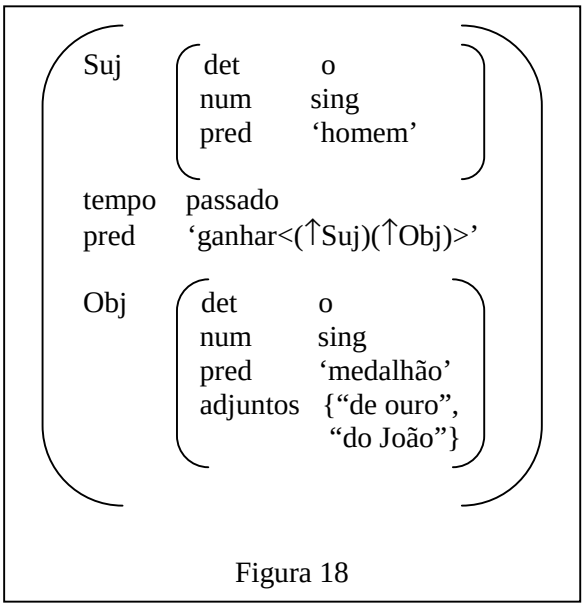


Figura 18

De forma similar, as regras gramaticais ilustradas na Figura 19, em especial a regra para sv, admitem construções complexas, como por exemplo, a ocorrência de transitividade verbal ou não, ou de sitagmas preposicionais. A opcionalidade desses componentes sentenciais é expressa na regra de sv pelos parênteses e chaves.

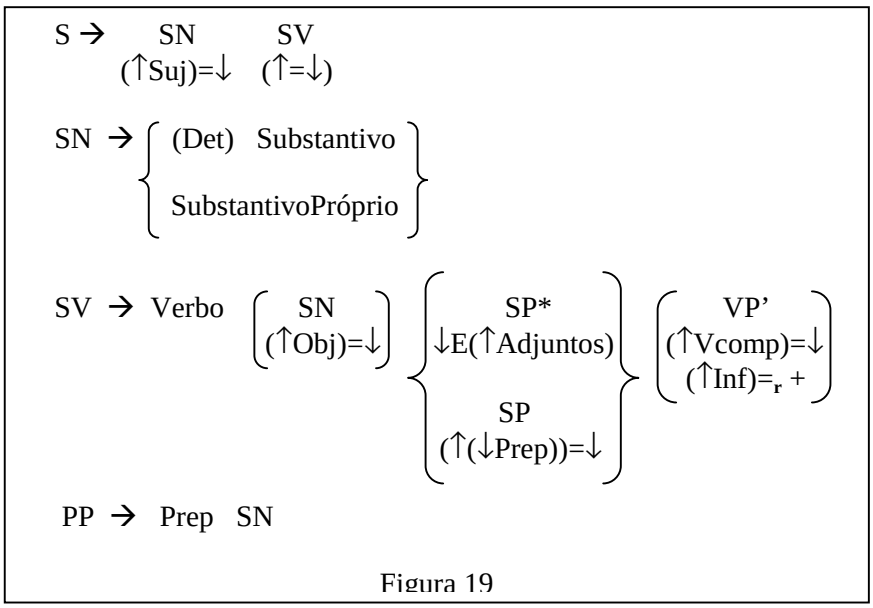


Figura 19

Além disso, a regra permite a possibilidade de um complemento para formar uma oração mais complexa, como "o homem foi ver a exposição", cuja estruturas-f é ilustrada pela Figura 20.

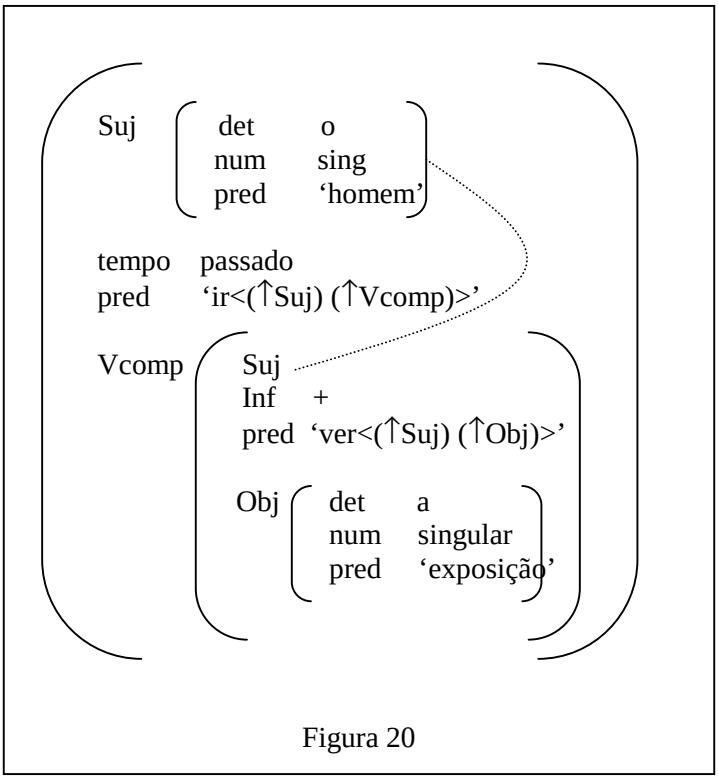


Figura 20

Na Figura 20, a linha tracejada indica repetição de estruturas, ou seja, o sujeito do verbo "ver" é o mesmo sujeito do verbo "ir", estando declarado na própria especificação lexical do verbo 'ir' por Vcomp. Esse tipo de dependência é chamada de controle funcional. Já o verbo 'ver' aparece em sua forma infinitiva devido a restrição (↑Inf)=$_{\mathbf{r}}$ + estar presente na gramática .

O léxico necessário para formar a sentença acima é ilustrado na Figura 21.

o: Det, (↑det)=o
(↑número)=singular

a: Det, (↑det)=a
(↑número)=singular

homem: Substantivo, (↑número)=singular
(↑pred)='homem'

exposição: Substantivo, (↑número)=singular
(↑pred)='exposição'

ir: Verbo, (↑número)=singular
(↑tempo)=passado
(↑pred)='ir<(↑Suj)(↑Vcomp)>'
(↑Vcomp Suj)=(↑Suj)

ver: Verbo, (↑número)=singular
(↑tempo)=passado
(↑pred)='ver<(↑Suj)(↑Obj)>'

Figura 21

3.6. As dependências de longa distância na LFG

Uma dependência de longa distância abrange, por exemplo, o caso de sentenças interrogativas e de orações relativas, em que um dos constituintes está extraposto à oração, ou seja, o caso em que um elemento preenche o papel gramatical de outro ausente. Por exemplo, para o caso de orações relativas, pode-se ter:

"A garota descobriu [quem] viu o bebê"
"A garota descobriu quem o bebê viu [ ]"
"O brinquedo que a garota deu [ ] ao bebê era grande"

Na primeira sentença, o pronome exerce o papel de sujeito, enquanto que nas duas últimas, o papel de objeto direto (funções indicadas superficialmente por [ ]). Esse tipo de dependência é chamado de controle de constituintes.

Na LFG, a solução para a questão de depêndencia de longa distância é o uso das metavariáveis de dominância limitada, representadas por ⇑ e ⇓, onde ⇓ é associada ao controlador da relação de dependência, ou seja, o elemento que preeenche o papel do elemento ausente, e ⇑ é associada ao elemento controlado, ou seja, o elemento ausente. Enquanto o par ↑-↓ está atado a nós de dominância imediata, o par ⇑-⇓ pode ser atado a nós separados em uma árvore de análise por um caminho mais longo.

Por exemplo, na gramática da Figura 22, na regra que expande S', o esquema marca o SN como o foco da questão, ou seja, o controlador, pois está associado à metavariável ⇓. No

caso das duas primeiras sentenças apresentadas, o SN corresponde ao pronome "quem". Já para simbolizar o elemento ausente, usa-se o símbolo "e" associado à metavariável ⇑, sendo assim, o elemento controlado.

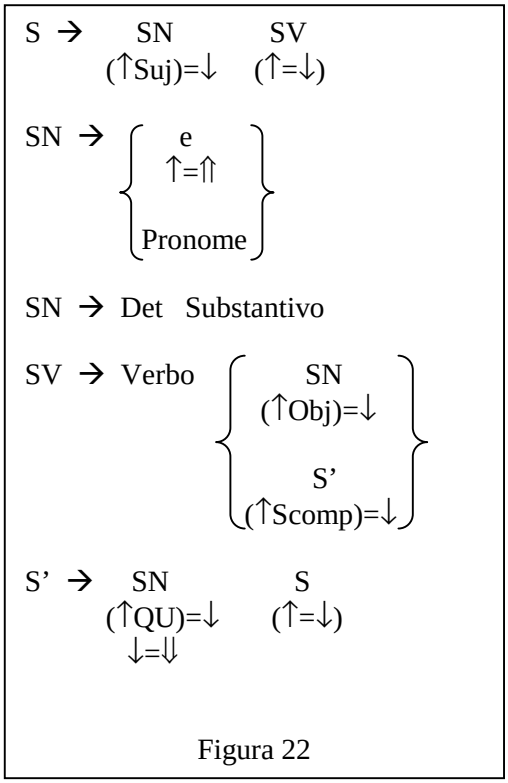


Figura 22

Considerando a sentença "a garota descobriu quem o bebê viu", a estrutura-c produzida a partir da gramática acima é ilustrada na Figura 23.

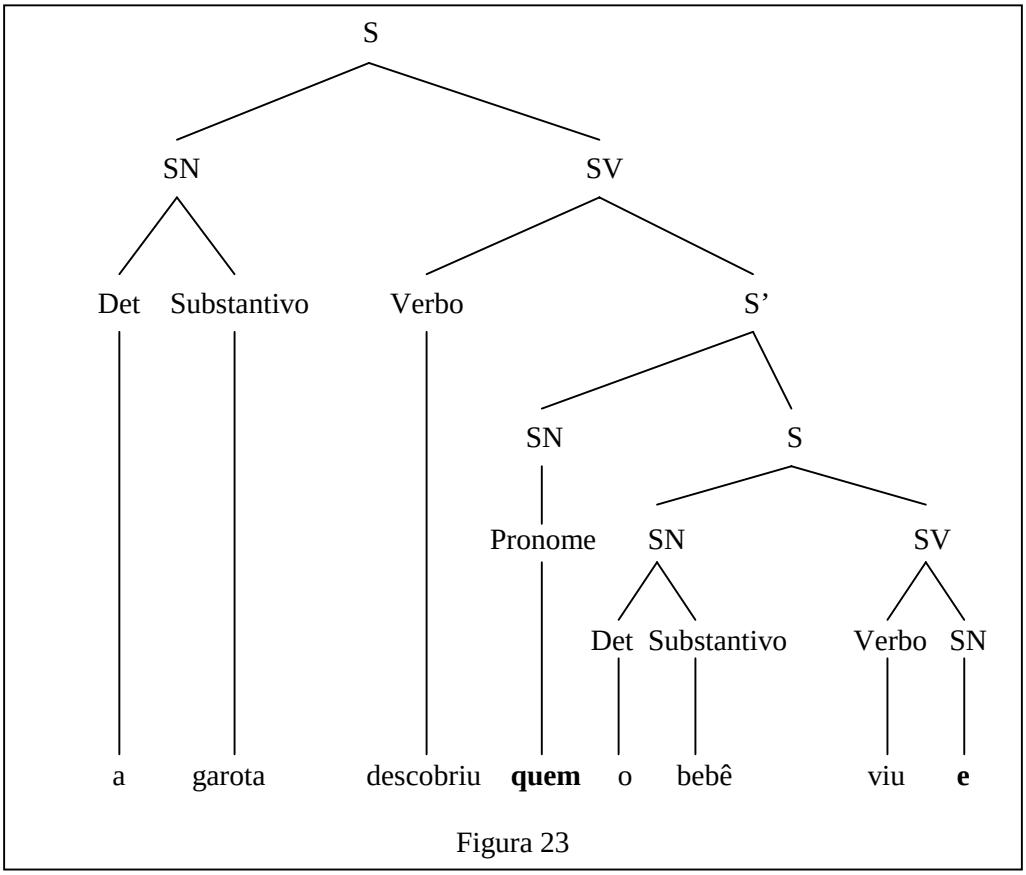


Figura 23

A associação de variáveis e esquemas aos nós da estrutura-c acima é feita como foi descrito anteriormente, porém, associando-se uma nova variável a cada metavariável ⇓ ou ⇑ presentes na estrutura-c. Após os passos necessários para a produção das descrições funcionais da sentença "a garota descobriu quem o bebê viu", a estrutura-c ilustrada na Figura 24 não apresenta esquemas lexicais para melhor vizualisação.

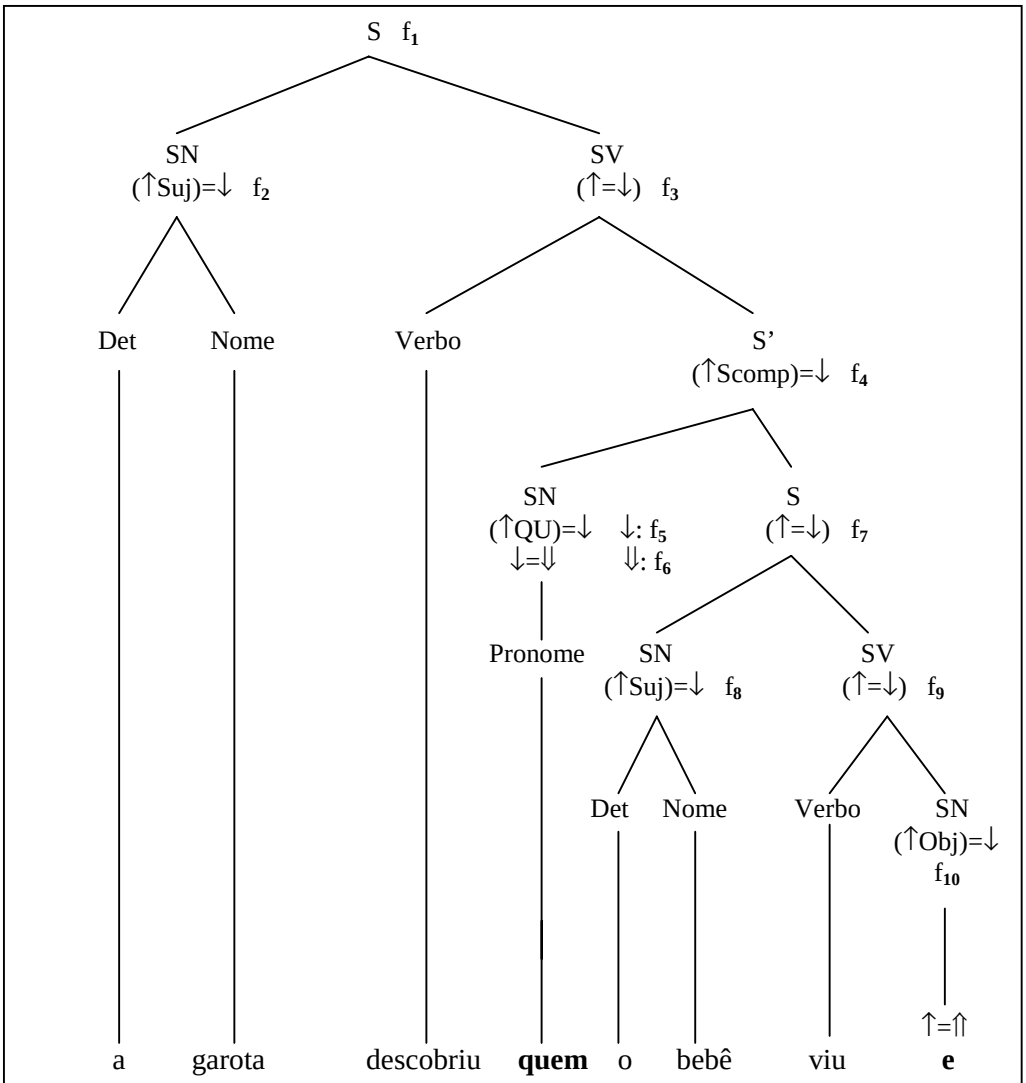


Figura 24

As descrições funcionais produzidas são ilustradas na Figura 25, na qual

$(f_1\ \text{Suj})=f_2$
$f_1=f_3$
$(f_3\ \text{Scomp})=f_4$
$(f_4\ \text{QU})=f_5$
$f_5=f_6$
$f_4=f_7$
$(f_7\ \text{Suj})=f_8$
$f_7=f_9$
$(f_9\ \text{Obj})=f_{10}$
$f_{10}=f_6$

Figura 25

a última descrição funcional (f_{10}=f_6) deriva do esquema ↑=⇑ do elemento controlado, o qual indica que a variável que imediatamente o domina (f_{10}) é igual à variável de seu controlador (f_6). Dessa forma, a estrutura funcional resultante é mostrada na Figura 26.

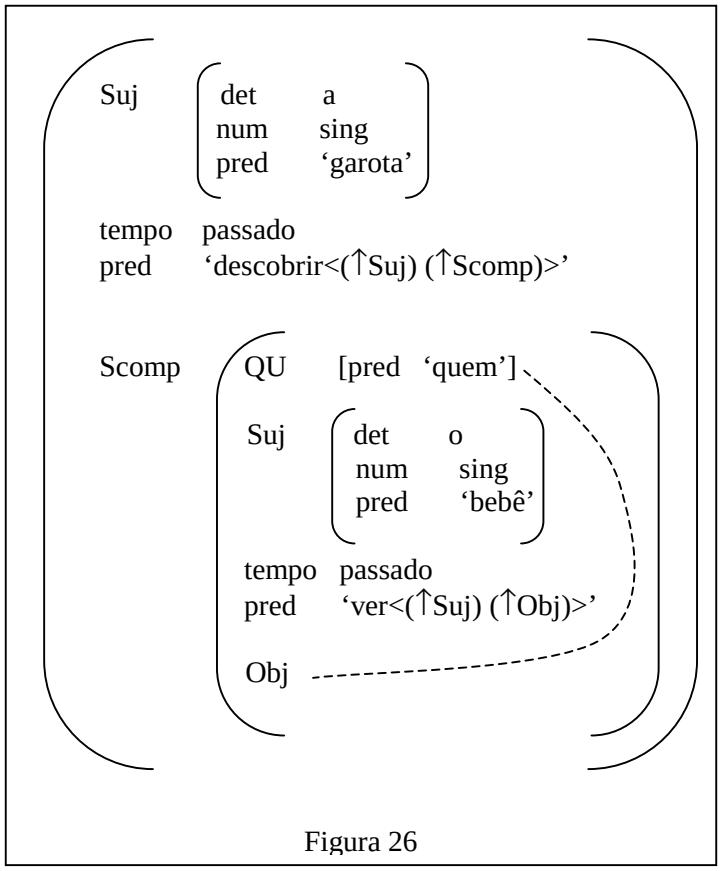


Figura 26

Para se representar sentenças interrogativas, a LFG assume o mesmo princípio de elementos controladores e controlados, usando as metavariáveis de dominância limitada ⇑ e ⇓, com a exceção de que se especifica algumas condições extras para maior segurança, como por exemplo, garantir que a o pronome interrogativo esteja presente na sentença. Por exemplo, na regra:

$$S' \rightarrow \underset{(\uparrow QU)=\Downarrow_{[+QU]}}{NP_{SN}} \quad \underset{(\uparrow=\downarrow)}{S} \qquad \downarrow=\Downarrow^{S}_{SN}$$

a primeira metavariável de dominância limitada diz que o conteúdo da estrutura QU deve ser o pronome interrogativo [+QU], o qual será pesquisado no léxico. Além disso, indica-se que ele terá o papel de SN (sintagma nominal) na sentença. Já a segunda metavariável diz que o elemento controlado tem papel de SN na sentença e é imediatamente dominado pelo nó denominado de S. Enquanto a primeira metavariável se associa ao léxico, a segunda se associa à correspondente na regra. Dessa forma, pode-se alterar a entrada lexical para que ela se associe à primeira metavariável, como por exemplo, na especificação do pronome "quem":

$$\text{quem: pronome interrogativo, } \underset{\uparrow=\Uparrow_{[+QU]}}{(\uparrow \text{pred})=\text{'quem'}}$$

Apesar dessas alterações, a forma como a produção das estruturas sintáticas é feita não é alterada.

3.7. As condições de boa formação de sentenças na LFG

Para uma estrutura funcional ser bem formada e, portanto, aceita pela gramática, ela deve ser completa e coerente. Ela é completa se e somente se ela contém todas as funções gramaticais exigidas pelas formas semânticas (indicadas por "pred"). Como exemplo, a estrutura da Figura 26 é completa pois apresenta as funções que os verbos exigem em suas formas semânticas.

Por outro lado, uma estrutura funcional é coerente se todas as funções gramaticais que ela apresenta são exigidas pelas formas semânticas que ela contém. A estrutura da Figura 26 está de acordo com a definição, logo é coerente também.

4. Comparação e Avaliação da Possibilidade de Redução entre DCG e LFG

Nesta seção são apresentadas as propriedades principais dos formalismos DCG e LFG, assim como suas vantagens e desvantagens, para uma avaliação da possibilidade de redução entre eles, ou seja, dada uma especificação em DCG, deseja-se saber se é possível reescrevê-la como uma especificação em LFG.

4.1. Concepção

A DCG surgiu baseada no conceito de ferramenta linguística, ou seja, uma representação computacional para as línguas naturais, enquanto que a LFG foi desenvolvida a partir da teoria léxico-funcional, a qual procura caracterizar as restrições e os princípios universais da linguagem.

4.2. Restritividade

Um formalismo é restrito se ele possui domínio e forma bem definidos, podendo ser, portanto, bem representado e analisado.

Tanto a DCG quanto a LFG são formalismos restritos, pois são baseados na gramática livre de contexto, a qual é uma representação restrita.

4.3. Expressividade

Expressividade é a facilidade que um formalismo tem para expressar as propriedades e os aspectos existentes em uma língua.

Devido ao fato de poder apresentar condições e restrições em suas regras, a LFG é mais expressiva que a DCG, pois consegue representar mais facilmente as relações e generalizações existentes em uma determinada língua.

4.4. Simplicidade

A representação da gramática de uma língua em LFG se faz de forma mais simples que em DCG, pelo fato da LFG representar toda a complexidade em seu léxico. Já a DCG, que o faz em suas próprias regras gramaticais através da adição de argumentos, pode-se tornar bastante complexa dependendo do tamanho e do nível de dependência contextual entre os constituintes da gramática representada.

4.5. Capacidade de Geração

A capacidade de geração está relacionada à abrangência de línguas que um formalismo pode representar. Neste caso, tanto a DCG quanto a LFG possuem capacidade de geração similar, correspondendo ao mesmo grupo de línguas representáveis por uma GLC.

4.6. Vantagens e Desvantagens

Pelo fato de a DCG poder ser direta e automaticamente codificada em Prolog, e logo, transformar-se em um programa compilado, além de ser veloz e eficiente, ela apresenta maior facilidade de manuseio, podendo ser facilmente integrada a ambientes Prolog, fazendo uso de todo o potencial de representação lógica da linguagem. Entretanto, apesar de se mostrar bastante flexível para representação de praticamente qualquer estrutura sentencial, a DCG se torna árdua para a representação de gramáticas de grande porte, pois a ordem dos argumentos presentes em um termo e sua aridade (número de argumentos) devem ser pré-especificados.

Já a LFG apresenta grande facilidade para lidar com concordâncias e dependências contextuais, pois o léxico se responsabiliza pelas devidas instanciações. Além disso, o fato das estruturas sintáticas produzidas carregarem o componente semântico (“pred”), pode facilitar a interpretação por algum modelo semântico adequado.

4.7. Redução

Apesar de possuírem concepções e bases teóricas diferentes, os formalismos DCG e LFG podem ser reduzidos um ao outro sem perda significante de expressividade e capacidade de geração. Isto se deve ao fato de ambos serem baseados em gramáticas livres de contexto e apresentarem mecanismos suficientes, como por exemplo, argumentos no caso da DCG e condições, restrições e esquemas no caso da LFG, para poderem representar a mesma gramática e seu nível de dependência contextual.

Assim, por exemplo, a gramática DCG da Figura 27 pode ser reduzida para a representação em LFG da Figura 28.

sentença → sn(Número, Gênero), sv(Número).
sn(Número, Gênero) → determinante(Número, Gênero), substantivo(Número, Gênero).
sn(singular,masculino) → nomepróprio.
sv(Número) → verbo(Número), sn(_,_), sp.
sp → preposição, sn(_,_).
determinante(singular, masculino) → [o].
determinante(plural, feminino) → [as].
substantivo(singular, masculino) → [homem].
substantivo(plural, feminino) → [flores].
nomepróprio → [maria].
verbo(singular) → [deu].
preposição → [a].

Figura 27

REGRAS:

sentença → sn sv
(↑Suj=↓) (↑=↓)

sn → { determinante substantivo
nomepróprio }

sv → verbo sn sp
(↑ObjD=↓) (↑(↓prep))=↓

sp → preposição sn
(↑ObjI)=↓

LÉXICO:

o: determinante, (↑Número)=singular
(↑Gênero)=masculino
(↑Det)=o

as: determinante, (↑Número)=plural
(↑Gênero)=feminino
(↑Det)=as

flores: substantivo, (↑Número)=plural
(↑Gênero)=feminino
(↑Pred)=’flores’

homem: nome, (↑Número)=singular
(↑Gênero)=masculino
(↑Pred)=’homem’

maria: nomepróprio, (↑Pred)=’maria’

deu: verbo, (↑Número)=singular
(↑Pred)=’deu<(↑Suj)(↑ObjD)(↑Prep ObjI)>’

a: preposição, (↑caso)=a

Figura 28

A redução da gramática em LFG (Figura 28) para a gramática em DCG (Figura 27) também é possível. Essa gramática permite reconhecer como válidas sentenças como “o homem deu as flores a maria”.

Os princípios de redução entre DCG e LFG são:

- os itens lexicais presentes na gramática DCG formam o léxico da gramática LFG, por exemplo:

  DCG: substantivo(singular,masculino) → [homem].

  LFG: homem: substantivo, (↑Número)=singular
  (↑Gênero)=masculino
  (↑pred)=’homem’

- os argumentos presentes na gramática DCG que expressam dependência contextual entre os constituintes de uma sentença são associados às entradas lexicais da LFG, por exemplo, como já mostrado no item anterior, as palavras Número e Gênero;

- os argumentos presentes na gramática DCG que não expressam dependência contextual entre os constituintes de uma sentença podem ser mapeados para esquemas, condições e/ou restrições associados aos termos das regras da LFG;

- as regras gramaticais que não introduzem itens lexicais e que não apresentam disjunções na gramática DCG são mapeadas uma a uma para as regras gramáticais da LFG, por exemplo:

  DCG: sn → determinante, substantivo.
  sv → verbo.

  LFG: sn → determinante substantivo
  sv → verbo

- as regras gramaticas que não introduzem itens lexicais e que apresentam disjunções na gramática DCG são mapeadas para uma única regra em LFG, a qual pode expandir em alguns termos opcionais pelo uso de parênteses e/ou em várias regras disjuntas, utilizando-se chaves, por exemplo:

  DCG: sn → determinante, substantivo.
  sn → nomepróprio.

  LFG: sn → $\left\{ \begin{array}{ll} \text{determinante} & \text{substantivo} \\ \text{nomepróprio} & \end{array} \right\}$

- a todo termo não-terminal das regras gramaticais em LFG, excluindo os termos pré-terminais, é associado um esquema funcional.

Os princípios de redução de LFG em DCG são similares aos citados, porém, de forma reversa.

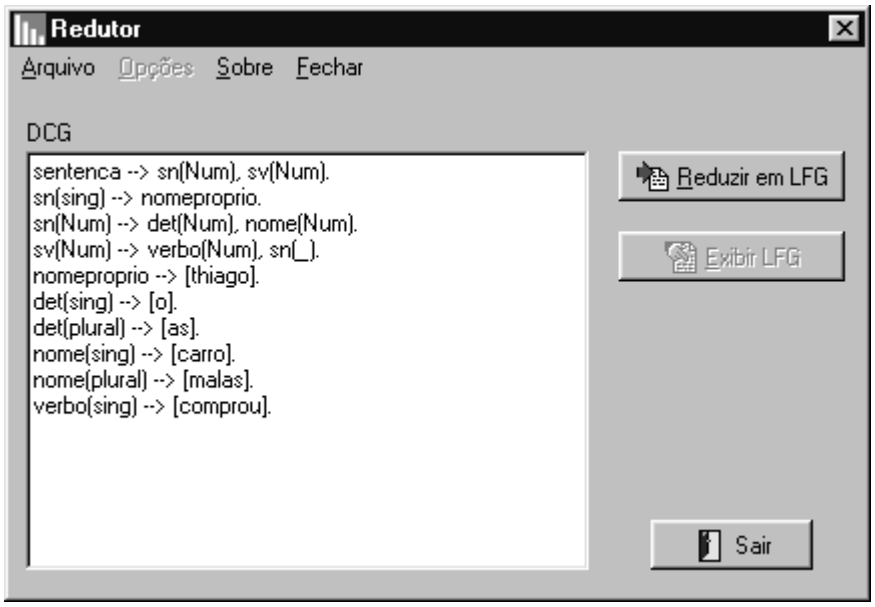
5. Uma interface para redução DCG-LFG

Para investigar a questão de redução de gramáticas DCG para gramáticas LFG, um sistema que realiza a redução de forma automática foi desenvolvido.

Em seu estágio atual de desenvolvimento, esse sistema reduz corretamente gramáticas simples em DCG, isto é, aquelas que não envolvem a associação de funções próprias do Prolog ou aquelas que não tratam da extraposição de constituintes.

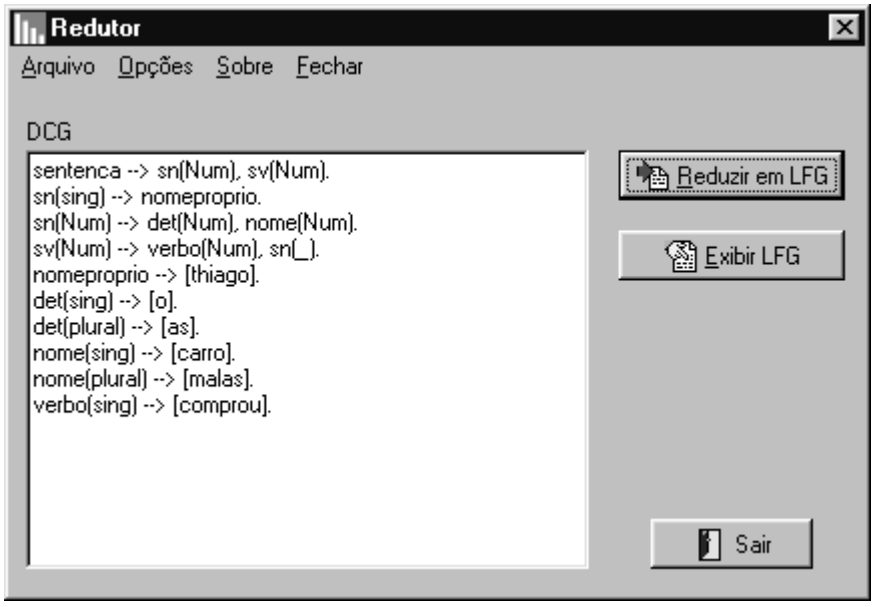
Com uma interface amigável, o sistema aceita como entrada para o processamento uma especificação gramatical em DCG, que em seguida é compilada a fim de extrair todas as informações possíveis da DCG. Como resultado da compilação, tem-se estruturas de dados que servem de base para a construção da representação apropriada em LFG. Após este processo ter sido realizado, tanto a DCG e a LFG, quanto as estruturas de dados produzidas são armazenadas em arquivos, estando à disposição do usuário do sistema para serem comparados. Um exemplo detalhado da execução do sistema é apresentado a seguir, mediante *dumpings* de tela.

Primeiro passo: especificação da DCG.



Janela 1

Segundo passo: ao se selecionar o botão “Reduzir em LFG”, se a gramática em DCG estiver especificada de forma correta, o botão “Exibir LFG” e o menu suspenso “Opções” são habilitados.

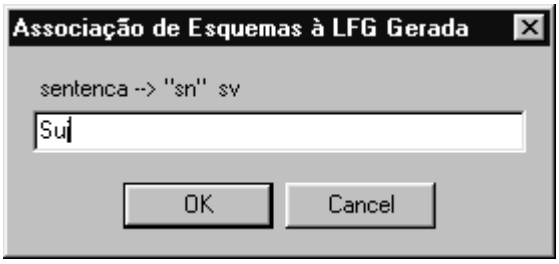


Janela 2

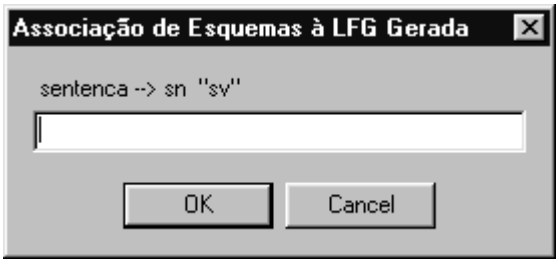
Terceiro passo: o programa associa automaticamente esquemas funcionais padrões a seus respectivos termos durante a montagem da LFG, porém, caso se queira alterar esses esquemas, basta selecionar o menu suspenso “Opções”. A partir daí, o sistema exibe um esquema por vez indicando a regra e o termo correspondente da LFG aos quais o esquema está associado. Por exemplo, para a regra indicada na Janela 3 abaixo, o programa indica que esq1 é a estrutura padrão associada ao termo entre aspas (sn) da regra introduzida por sentença.

Janela 3

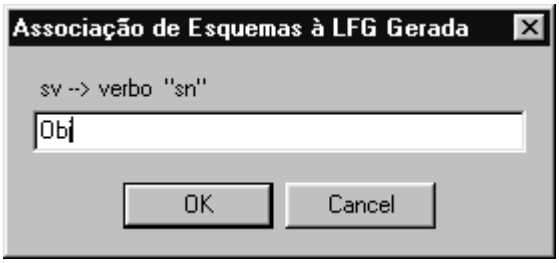
Para alterar tal esquema, basta introduzir o novo esquema, como indica a Janela 4, e assim, o esquema asssociado a sn passa a ser Suj. A seguir deseja-se que o sv seja inserido diretamente na estrutura como um todo. Neste caso, não é associado um esquema a ele, como indica a Janela 5. O usuário pode indicar, assim, o esquema associado a sv, isto é, Obj, como mostra a Janela 6.



Janela 4



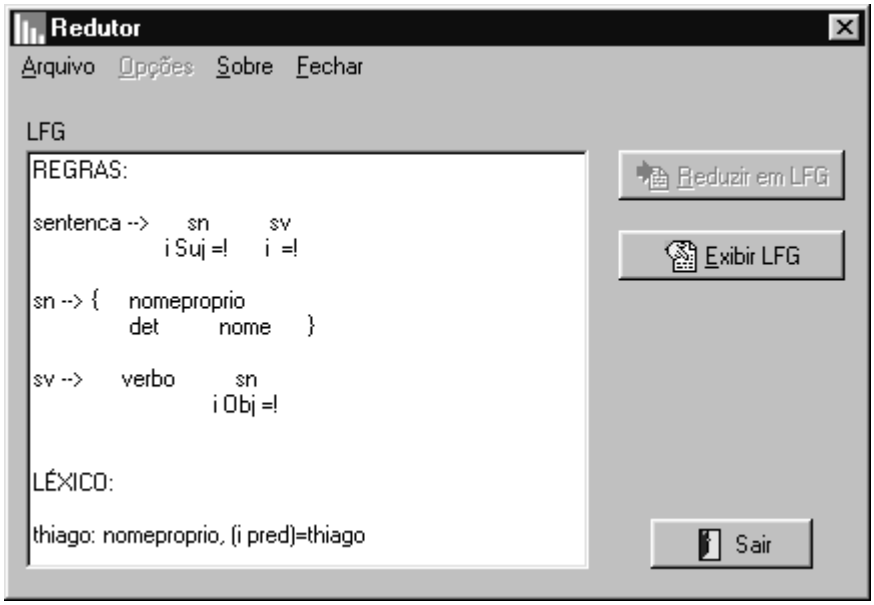
Janela 5



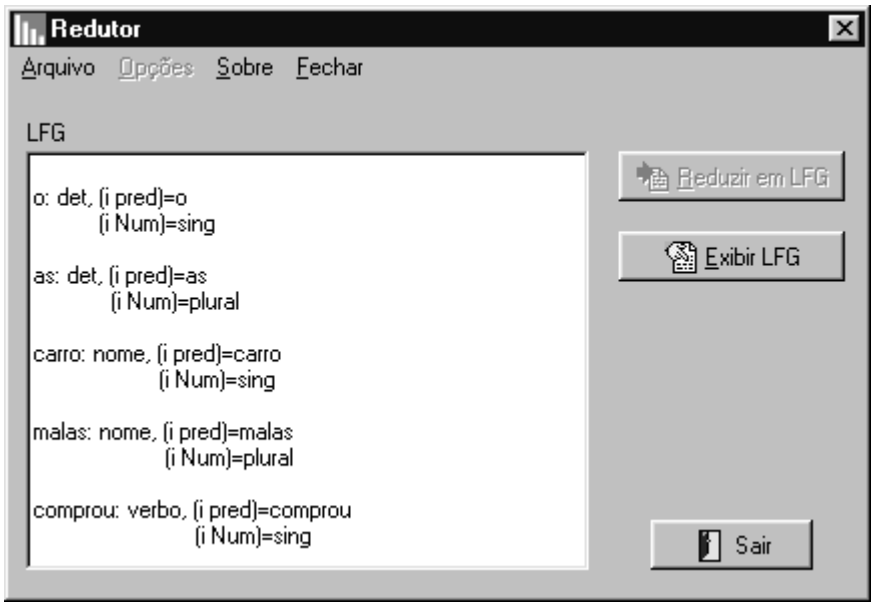
Janela 6

A geração da gramática em LFG correspondente à DCG inicial (Janela 1) pode ser realizada, procedendo-se ao passo 4.

Quarto passo: ao se selecionar o botão “Exibir LFG”, a LFG gerada é exibida na tela. O conteúdo desta tela (Janelas 7 e 8) pode ser maior do que o exibido e, portanto, a tela pode ser rolada para baixo, utilizando-se o cursor. Pelo fato da linguagem de programação, no caso, Delphi 3.0 Client/Server, não prover os símbolos ↑ e ↓, eles foram substituídos por, respectivamente, i e !.



Janela 7



Janela 8

6. Conclusão

Foram apresentados neste relatório caracterizações detalhadas dos formalismos gramaticais DCG e LFG, com o objetivo de estabelecer suas bases teóricas e conceituais, procedendo, então, à comparação e avaliação da possibilidade de redução entre eles. Foi construída, ainda, uma interface computacional para redução automática de uma especificação gramatical em DCG para sua correspondente em LFG.

A principal conclusão deste trabalho é que a redução entre os formalismos é viável, se considerarmos construções livres de contexto. Construções que introduzem dependências contextuais podem ser resolvidas, dentro de certos limites, em ambos os formalismos, pelo uso de argumentos em DCG e esquemas, condições e/ou restrições em LFG. Casos complexos tais como dependências de longa distância (ou extraposições complexas) não foram, porém, averiguados.

Referências Bibliográficas

Allen, J. (1987), *Natural Language Understand.* The Benjamim/Cummings Publ. Co., Inc., USA.

Araribóia, G. (1989), *Inteligência Artificial: Um Curso Prático*. Livros Técnicos e Científicos Editora Ltda, ILTC. Rio de Janeiro – RJ.

Arity (1988). The Arity Prolog Language Reference Manual. Arity Corporation, Massachusetts, USA.

Clocksin, W. and Mellish, C. (1981), *Programming in Prolog*. Springer-Verlag, Germany.

Kaplan, Ronald M. and Bresnan, Joan (1982). Lexical-Functional Grammar: A Formal System for Grammatical Representation. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations*. Cambridge, MA, MIT Press.

Rich, E. and Knight, K. (1991). *Artificial Intelligence*. McGraw-Hill, Inc. (2nd Edition).

Shieber, S. M. (1986), *An Introduction to Unification-Based Approaches to Grammar*. University of Chicago Press, USA.

Silva, B.C.D.S. (1996*). A Face Tecnológica dos Estudos da Linguagem: O Processamento Automático das Línguas Naturais*. Tese de Doutorado. UNESP, Araraquara.

Whitelock, P., Wood, M. M., Somers H. L., Johnson R. and Bennett P. (1987), *Linguistic Theory & Computer Applications*. Centre for Computational Linguistics UMIST, Manchester, UK. St Edmundsbury Press Limited, Bury St Edmunds, Suffolk.

Apêndice 1 – Cálculo de Predicados

Neste apêndice são apresentadas as noções gerais do Cálculo de Predicados. Para estudo mais aprofundado, refira-se a (Clocksin and Mellish, 1981).

"Cálculo de Predicados" é um formalismo através do qual se expressam relações lógicas, ou seja, proposições verdadeiras ou falsas que podem ser relacionadas entre si e sobre as quais se pode inferir outras proposições.

Para se expressar proposições sobre o mundo, deve-se ser possível descrever os objetos que estão envolvidos em uma proposição. Em "Cálculo de Predicados", os objetos são representados por termos. Assim sendo, um termo pode ser de uma das seguintes formas:

- uma constante - representa um único indivíduo ou conceito;
- uma variável - pode representar indivíduos diferentes em momentos diferentes;
- um termo composto – consiste de uma função junto com um conjunto ordenado de argumentos.

Para se expressar proposições sobre objetos, deve-se ser posssível expressar relações entre objetos. Isso é feito através de predicados. Uma proposição atômica consiste de um predicado junto com um conjunto ordenado de termos como seus argumentos. Como exemplo, as seguintes proposições são atômicas:

```
humano(João)
gosta(homem,vinho)
```

onde está declarado que João é humano e que homem gosta de vinho.

Já uma proposição composta pode ser feita de várias maneiras. Pode-se usar os conectivos lógicos indicados na tabela abaixo juntamente com seus significados:

Conectivo	**Sintaxe**	**Significado**
Negação	~ a	"não a"
Conjunção	a & b	"a e b"
Disjunção	a # b	"a ou b"
Implicação	a -> b	"a implica em b"
Equivalência	a <-> b	"a é equivalente a b"

Então, por exemplo:

```
homem(João) # mulher(João)
```

poderia ser usado para representar a proposição de que João é homem ou João é mulher. No exemplo:

```
homem(João) -> humano(João)
```

poderia representar a proposição que João sendo homem implica que João é humano.

Já no caso de aparecimento de variáveis em proposições, o significado é somente definido quando tais variáveis são introduzidas por quantificadores. Quantificadores permitem a representação de conjunto de indivíduos e o que é verdade sobre eles. Este modelo provê dois quantificadores. Se v representa qualquer variável e p qualquer proposição, pode-se resumi-los da seguinte forma:

Sintaxe	**Significado**
todo(v, p)	"p é verdade para qualquer valor de v"
existe(v, p)	"há algum valor de v que torne p verdade"

O primeiro deles é chamado de quantificador universal devido ao fato dele afirmar sobre todos os objetos no Universo. O segundo é chamado de quantificador existencial devido ao fato dele afirmar sobre a existência de algum objeto. No exemplo a seguir:

todo(X, homem(X) -> humano(X))

significa que para qualquer X, se X é um homem, então X é humano. Já o caso:

existe(X, pai(João, Z) & mulher(Z))

significa que existe um Z, tal que, João é pai de Z e Z é mulher.

Também é possível o uso dos dois quantificadores em uma mesma proposição, como no exemplo abaixo:

todo(X, animal(X) -> existe (Y, mãe(X, Y)))

que diz que para todo X, se X é animal, então existe algum Y, tal que, Y é mãe de X.

Apêndice 2 – Compilação da DCG em Prolog

Neste apêndice é apresentada a forma como o Prolog compila uma especificação gramatical em DCG. Para estudo mais aprofundado, refira-se a (Clocksin and Mellish, 1981).

A estrutura principal que o problema de análise envolve é a seqüência de palavras que deve ser analisada. Para isso, isolam-se subseqüências dessa estrutura como sendo os vários termos aceitos pela gramática, e ao final da análise, mostra-se que a sentença inteira é aceita como um termo do tipo "sentença". Desde que o modo padrão de se representar uma seqüência é em forma de lista, a entrada para o analisador será representada como uma lista do Prolog. Deste modo, cada palavra será representada como átomos do Prolog.

Para fazer a análise, o Prolog compila cada regra da gramática para o Prolog, adicionando dois argumentos a cada termo da gramática. O primeiro argumento especifica a seqüência de entrada que será analisada, enquanto o segundo argumento especifica a seqüência que deve ser analisada no passo seguinte. A adição dos dois argumentos extras não altera o comportamento dos argumentos previamente definidos nos termos da DCG.

A gramática da Figura 1, transcrita abaixo na Figura A1, é compilada em Prolog como ilustra a Figura A2.

```
sentença → sintagma_nominal, sintagma_verbal.
sintagma_nominal → determinante, substantivo.
sintagma_verbal → verbo.
sintagma_verbal → verbo, sintagma_nominal.
determinante → [o].
substantivo → [homem].
substantivo → [lápis].
verbo → [perdeu].
verbo → [canta].
```

Figura A1

```
sentença(S0,S) :- sintagma_nominal(S0,S1), sintagma_verbal(S1,S).
sintagma_nominal(S0,S) :- determinante(S0,S1), substantivo(S1,S).
sintagma_verbal(S0,S) :- verbo(S0,S).
sintagma_verbal(S0,S) :- verbo(S0,S1), sintagma_nominal(S1,S).
determinante([o|S],S).
substantivo([homem|S],S).
substantivo([lápis|S],S).
verbo([perdeu|S],S).
verbo([canta|S],S).
```

Figura A2

A interpretação das regras da Figura A2 pode ser expressa da seguinte forma:

"sintagma_nominal(S0,S1) é verdadeiro se: existe uma sub-cadeia inicial na seqüência S0 que corresponde a um sintagma nominal. O que não for reconhecido como sintagma nominal é armazenado em S1, para análise subseqüente."

"sintagma_verbal(S1,S) é verdadeiro se: : existe uma sub-cadeia inicial na seqüência S1 que corresponde a um sintagma verbal. O que não for reconhecido como sintagma verbal é armazenado em S, para análise subseqüente."

"sentença(S0,S) é verdadeiro se: existe uma cadeia inicial na seqüência S0 que corresponde a uma sentença. O que não for reconhecido como sentença é armazenado em S."

"o determinante([o|S],S) é verdadeiro se: analisando o primeiro elemento da lista [o|S] (que é o item lexical 'o', enquanto que o restante da lista é representado pelo símbolo S) é deixado o resto da seqüência, ou seja, S, para análise subseqüente."

Na regra inicial da gramática, isto é, a regra indicada pelo símbolo inicial "sentença", deseja-se que S0 seja toda a seqüência que constitui uma sentença, enquanto que S seja a lista vazia deixada após a análise da sentença. Inicialmente, um teorema de provas em Prolog indicará a lista de constituintes da sentença, como mostra a meta:

?- sentença([o,homem,perdeu,o,lápis],[ ]).

Caso a gramática admita regras para analisar sentenças como a proposta na meta, a prova é bem sucedida.

No caso das especificações lexicais, ou seja, regras da forma X→[Y], se fosse desejado acrescentar o novo substantivo "banana" ao léxico, por exemplo, a seguinte regra teria que ser acrescentada à gramática:

substantivo(singular, substantivo(banana)) → [banana].

que é compilada para

substantivo(singular, substantivo(banana), [banana|S], S).

São muitos argumentos sem utilidades para se especificar somente um substantivo. Portanto, usando-se {}, impede-se que a compilação ocorra, tornando assim, o "programa" mais simples e eficiente.