Engenharia de Linguagens
Engenharia Gramatical

Universidade do Minho, MEI

Ano lectivo 2010/2011

Trabalho Prático

José Pedro Silva - pg17628 Mário Ulisses Costa - pg15817 Pedro Faria - pg17684

20 de Novembro de 2010

Resumo

Descrição da solução proposta para os problemas do dia 08-11-2010

Conteúdo

1 Problema

Considere a linguagem de programação Lisp na qual um programa é uma Symbolic-Expression (SExp). Sabe-se que uma SExp é um valor atómico (palavra ou numero) ou é uma lista e que cada elemento da lista é uma SExp.

A GIC abaixo define formalmente a linguagem Lisp, de acordo com a descrição acima:

```
T = { num, pal, "(", ")" }
N = { Lisp, SExp, SExplist }
S = Lisp
P = {
      p1: Lisp     --> SExp
      p2: SExp     --> num
      p3: SExp     --> pal
      p4: SExp     --> "(" SExplist ")"
      p5: SExplist --> SExp SExplist
      p6: SExplist --> &
    }
```

Pede-se então que escreva uma Gramática de Atributos, GA, para

- a) calcular o nível de aninhamento de cada lista (comece em 1 para a lista principal).
- b) construir uma lista com todos os ”operadores”presentes (considera-se ”operador”o 1ºelemento de cada lista).
- c) verificar se todos os ”operadores”usados pertencem ao conjunto de operadores válidos de acordo com uma ”biblioteca”fornecida no inicio.
- d) verificar se o numero de argumentos (”operandos”) com que cada operador é invocado condiz com a cardinalidade associada aos operadores válidos na referida ”biblioteca”.

2 Solução

As subsecções que se seguem referem-se à resolução de cada alínea do capítulo anterior.

2.1 Alínea a)

A solução para a primeira alínea é a criação de um atributo herdado *alinh* do tipo *Integer* que ocorrerá nos Não Terminais *SExp* e *SExpList*, conforme se poderá ver no código seguinte:

Listing 1: Code to a Contextual Condition for Symbol Transition

```
inherited attribute alinh :: Integer;
attribute alinh occurs on SExp, SExpList;
```

O valor será incrementado na produção `SExp -> "("SExplist ")"` e adicionado ao atributo sintetizado *output* para posteriormente ser imprimido para o ecrã (código em 2).

Listing 2: Code to a Contextual Condition for Symbol Transition

```
concrete production sExp3
se::SExp ::= '(' sel::SExpList ')'
{se.output="["++ sel.output ++ ":"  ++ toString(se.alinh) ++ "]";
sel.alinh = se.alinh + 1;
}
```

2.2 Alínea b)

Para esta alínea, foram precisos dois atributos para a sua resolução. O primeiro, *ishlista*, é um valor herdado booleano que será preciso para verificar se o valor a retirar é o primeiro da lista ou não.

Pela Figura 1 podemos ver como esse atributo (que ocorre nos Não Terminais *SExp* e *SExpList*) será tratado. A figura representa a árvore de derivação (incompleta) para o exemplo `(add (add 1 2) (mul 3 4))`.

O segundo valor, *operandos*, é um atributo sintetizado do tipo *[String]*. Nas produções `SExp -> num` e `SExp -> pal` será confirmado se se trata do primeiro elemento de uma lista. Caso seja, será retornado uma lista com o valor correspondente, caso contrário, será retornado uma lista vazia, para posteriormente ser concatenado com os outros valores lidos (código em 3).

Listing 3: Parte do código onde se verifica se é o primeiro elemento da lista

```
se::SExp ::= pl::Pal
{se.output="palavra";
se.operandos = if se.ishlista then [pl.lexeme] else [] ;
}
```

2.3 Alínea c) e d)

Estas duas alíneas ainda não foram devidamente resolvidas por falta de exploração da ferramenta *Silver*. Mas, do ponto visto teórico, a resolução passava por criar um atributo herdado que tivesse a estrutura de uma lista (para a alínea *c)*).

Listing 4: Parte do código onde se define o atributo biblioteca

```
synthesized attribute biblioteca :: [String];
attribute biblioteca occurs on Lisp,SExp,SExpList;
```

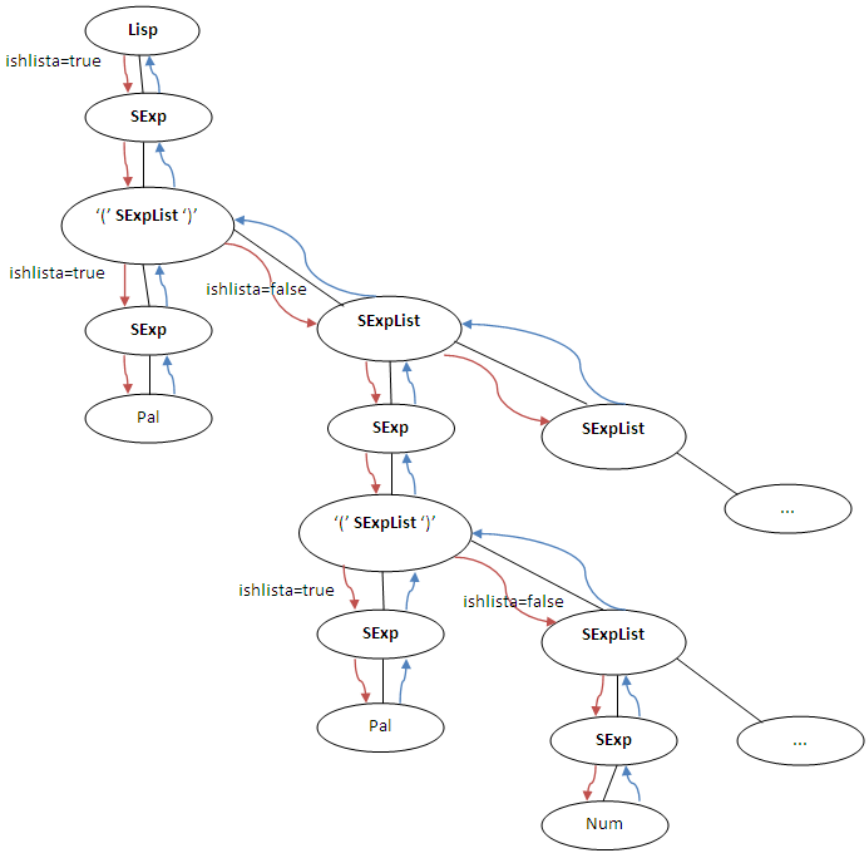

Figura 1: Árvore de derivação da linguagem Lisp

Este atributo seria a biblioteca de operandos, criado na produção inicial (como se pode ver em 5), e deveria descer na árvore de derivação até à leitura dos operandos para se verificar se o valor seria correcto ou não. A ferramenta *Silver* dispõe de várias funções para trabalhar com listas. A função *listContains* seria a ideal para resolver esta alínea, mas infelizmente não se a conseguiu utilizar.

Listing 5: Parte do código para a criação da biblioteca

```
concrete production program
ls::Lisp ::= se::SExp
{
ls.biblioteca = ["add"] ++ ["mul"] ++ ["let"];
se.biblioteca = ls.biblioteca;
}
```

Para a alínea *d)*, a solução passaria por criar uma estrutura mais complexa da biblioteca que indicasse a cada operador o número correspondente de argumentos.