Perfil de PLC - Processamento de Linguagens e de Conhecimento (MiEI-MEI)

Resolução do Exercício 1 da Ficha de Exercícios 1 de
GCS – Gramáticas na Compreensão de Software

Ano Letivo 2017/18

1 Exercício 1: Listas Mistas sensíveis ao contexto

1.1 Definição Sintática

Desenvolva uma GIC para definir uma linguagem que permita escrever listas mistas de números e palavras, de tal forma que as 3 frases abaixo sejam frases válidas dessa linguagem.

```
LISTA 1 .
Lista aaa .
Lista 1, 2, agora, 3, 4, fim, 7, 8.
```

Resolução:

Para resolver este exercício, relembremos o conceito de GIC:

Uma ***Gramática Independente de Contexto (GIC)*** *define-se como sendo um tuplo*

$$GIC = <T,N,S,P>$$

onde
T é o conjunto dos ***símbolos terminais*** *da linguagem (o alfabeto ou vocabulário).*
N é o conjunto dos ***símbolos não-terminais*** *da gramática.*
S ∈ N é o ***símbolo inicial ou axioma*** *da gramática.*
P é o conjunto de ***produções ou regras de derivação*** *da gramática.*

Cada produção p ∈ P é uma regra da forma

$$p : X0 \rightarrow X1 \ldots Xi \ldots Xn$$

em que p é o identificador da regra, → é o operador derivação, X0 ∈ N e Xi ∈ (N ∪ T), 0 ≤ i ≤ n.
Numa produção com etiqueta p o lado esquerdo do operador de derivação, sempre um não-terminal, denotase por ***LHS****(p)*[a] *e o lado direito do operador de derivação, uma sequência de símbolos terminais ou não-terminais, denota-se por* ***RHS****(p)*[b]*.*
O conjunto T dos símbolos terminais divide-se em 3 subconjuntos disjuntos — T = PR∪Sin∪TV — das ***Palavras-Reservadas****, dos* ***Sinais*** *e dos* ***Terminais-Varíaveis****.* [1]

[a] Do inglês, Left Hand Side.
[b] Do inglês, Right Hand Side.

Numa primeira fase, vamos definir uma linguagem em Notação Matemática, para dar resposta aos requisitos do problema (Listing 1).

```
Frase  ->  Lista   Elems '.'
Elems  ->  Elem
         |  Elems '.' Elem

Elem   ->  num
         |  pal
```

Listing 1: Notação Matemática

Como o objetivo deste ano letivo é usarmos Gramáticas de Atributos (GA) e o gerador AnTLR[1] (ANother Tool for Language Recognition), vamos ter de transformar a Notação Matemática anterior, escrita na Notação BNF-puro, para a GIC na Notação BNF-estendido.

Para o fazer relembremos o conceito de BNF:

*A notação **BNF (de Backus-Naur Form)** é uma notação textual, formal, para representar gramáticas independentes de contexto.*
Em BNF cada produção ou regra de derivação da gramática é vista como um triplo cujo elemento central é o operador de derivação. O operando do lado esquerdo desse operador é um símbolo não-terminal; o do lado direito é sua expansão, que pode conter zero ou mais símbolos terminais e não-terminais.[2]
Os meta-símbolos utilizados na notação BNF são [2]:

- *::= – representa o operador "deriva em" ou "definido como";*
- *| – indica um operando direito alternativo para o mesmo operando esquerdo;*
- *< > – delimita o identificador de cada símbolo gramatical.*

*A notação **EBNF** extende a notação BNF com os seguinte meta-símbolos [2]:*

- ***** *(ou { }) – indica uma parte que se pode repetir 0 ou mais vezes;*
- *+ – indica uma parte que se pode repetir 1 ou mais vezes;*
- *? (ou [ ]) – indica uma parte opcional;*
- *( ) – indica precedências dentro da regra;*
- *" " – indica um carácter a tratar como terminal e.g., "<".*

Com esta transformação, vamos obter uma Gramática Independente de Contexto escrita em notação BNF-estendido do AnTLR (Listing 2).

```
/*
 * Linguagem: "Lista Mista"
 * Processador: Gramatica Independente de Contexto que permite escrever listas mistas de numeros e
    palavras
 * PRH 2017.09.25
 */

grammar gcs17F1Ex1_GIC;


lista : 'LISTA' elems  '.'
      ;

elems : elem  (',' elem)*
      ;
```

[1] AnTLR é um poderoso gerador de compiladores para reconhecimento (parsing) e processamento de frases da linguagem definida pela gramática que lhe é fornecida como entrada. A partir de uma gramática independente de contexto, tradutora ou de atributos, o ANTLR gera um parser, um construtor/ navegador na árvore de parsing e um tradutor. In: `http://www.antlr.org/`

```
15
16 elem  : NUMERO
17       | PAL
18       ;
19
20
21 /* Definicao do Analisador LEXICO */
22
23 PAL:      [a-zA-Z][-a-zA-Z_0-9]* ;
24
25 NUMERO:  '0'..'9'+ ; // [0-9]+
26
27 Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 2: Notação BNF-estendido do AnTLR – Gramática Independente de Contexto (GIC)

Note no exemplo da Listing 2 que, em AnTLR, toda a gramática abre com um preâmbulo que contém uma ou mais secções com informações gerais para o gerador; essas secções são, em geral, auto-explicativas: a primeira, que deve estar sempre presente, é o tipo de gramática e seu nome único (neste caso, '`grammar gcs17F1Ex1_GIC`'); as restantes serão explicadas na próxima alínea. Apenas um detalhe deve ser marcado: o nome da gramática tem de ser precisamente o nome do ficheiro que contém a gramática. O nome do ficheiro também deve ter uma extensão '.g4'; assim sendo, neste exemplo o ficheiro de entrada tem de ser denominado '`gcs17F1Ex1_GIC.g4`'.

1.2 Definição Semântica

Transforme a GIC numa GA de modo calcular os resultados pedidos nas alíneas seguintes. Comece por definir uma GT recorrendo a variáveis globais e ações semânticas para resolver a alínea a).

Resolução:

Para resolver esta parte relembremos os conceitos de GT:

Uma ***Gramática Tradutora (GT)*** *é uma extensão à Gramática Independente de Contexto para definir (semi-formalmente) como processar as frases da linguagem gerada pela GIC. Assim acrescenta-se à GIC um Conjunto de símbolos semânticos AS (identificadores de Acções Semânticas) e modificam-se as produções de P, passando-se a ter o seguinte tuplo*

$$GT =< T,N,S,AS,Px >$$

onde
Cada produção $p \in Px$ *passa agora a ser uma regra da forma*

$$p \ : \ X_0 \ \rightarrow \ X_1 \ ... \ X_i \ ... \ X_n$$

as em que $as \in AS$ *indica a Acção Semântica a executar*[a] *após reconhecer todos os símbolos no lado direito da produção. Para especificar a semântica da linguagem usa-se uma Gramática de Atributos, cuja definição se segue.* [1]

[a]Pelo facto de apenas indicar o nome da acção e não descrever de qualquer forma essa acção, é que se afirma que é uma especificação semi-formal.

a) contar o comprimento da lista e a quantidade de números.

Resolução:

Para responder à questão vamos definir uma GT a partir da GIC anterior. Nesta nova gramática vamos acrescentar a cada produção da GIC original uma ação semântica para ser possível efectuar os cálculos necessários para obter o *comprimento da lista* e a *quantidade de números* (Listing 3).

```
1 /*
2  * Linguagem: "Lista Mista"
```

```
3  * Processador: Contador do Comprimento da lista e quantidade de nums  contidos na lista
4  * Uso de gramatica tradutora
5  * PRH 2017.09.25
6  */
7
8
9  grammar gcs17F1Ex1_GT;
10
11 @header{
12         import java.util.*;
13 }
14
15 @members {
16           int contador, contaN, soma;
17 }
18 lista
19 @init{
20       contador=contaN=0;
21       }
22 @after{
23           System.out.println("Comprimento da lista: "+contador+" sendo numeros: "+contaN);
24        }
25       : 'LISTA' elems  '.'
26       ;
27
28 elems : elem  {contador=1;} (',' elem {contador++;})*
29       ;
30
31 elem  : NUMERO {contaN++;}
32       | PAL
33       ;
34
35
36 /* Definicao do Analisador LEXICO */
37
38 PAL:     [a-zA-Z][-a-zA-Z_0-9]* ;
39
40 NUMERO: '0'..'9'+ ; // [0-9]+
41
42 Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 3: Notação BNF-estendido do AnTLR – Gramática Tradutora (GT)

Com foi dito anteriormente, em Antlr, cada gramática abre com um preâmbulo que contém uma ou mais secções com informações gerais para o gerador; essas secções são: a primeira, como já dito, define o tipo de gramática e o seu nome único (`grammar gcs17F1Ex1_GT`); a segunda, **@header**, é usada para identificar bibliotecas Java que devem ser importadas (neste exemplo, `import java.util.*`); a terceira, **@members**, é usada para declarar variáveis globais, classes e funções que serão usadas nas ações semânticas (neste exemplo, `int contador, contaN, soma`).

Neste caso utilizou-se o bloco opcional `@init{}` associado ao não terminal `lista` para inicializar o contador `contaN` a zero, antes de reconhecer o lado direito da respetiva produção. Também se fez uso do bloco opcional `@after{}`, que é executado após o reconhecimento com sucesso do não terminal `lista`, para imprimir os resultados do processamento.

Resolvido o pedido desta alínea através da escrita da GT definida anteriormente, vamos abandonar daqui para a frente esta abordagem com recurso a variáveis globais e passar a fazer uso de uma especificação baseada no uso de informação local a cada produção (atributos associados ao seu símbolo à esquerda ou aos símbolos à direita). Assim, vamos definir, agora e na resolução dos exercícios daqui para a frente, uma GA para especificar a semântica da linguagem.

Para resolver esta parte relembremos os conceitos de GA:

Uma `Gramática de Atributos (GA)` *define-se como sendo um tuplo*

$$GA =< GIC,A,RC,CC,RT >$$

onde
$A = SA(X), \forall\ X \in (N \cup T)$ *é o conjunto dos* ***atributos*** *de todos os símbolos da gramática.*
$RC = SRC(p), \forall\ p \in P$ *é o conjunto das* ***regras de cálculo dos atributos*** *em todas as produções da gramática.*
$CC = SCC(p), \forall\ p \in P$ *é o conjunto das* ***condições de contexto*** *em todas as produções da gramática.*
$RT = SRT(p), \forall\ p \in P$ *é o conjunto das* ***regras de tradução*** *em todas as produções da gramática.*

Os atributos A(X) de cada símbolo dividem-se em dois subconjuntos disjuntos

$$A(X) = AI(X) \cup AS(X),\ AI(X) \cap AS(X) = \emptyset$$

em que
AI(X) é o conjunto dos ***atributos herdados*** *do símbolo X, ou seja, dos atributos que transportam a informação do contexto esquerdo ou direito pela subárvore abaixo.*
AS(X) é o conjunto dos ***atributos sintetizados*** *do símbolo X, ou seja, dos atributos que sintetizam a informação a partir das folhas e a transportam pela árvore acima.*

Do que foi dito conclui-se que os atributos herdados de X_0 têm de ser calculados fora da produção `p`[a] *e o seu valor pode ser usado para cálculos nessa produção.*
Deduz-se também que os atributos sintetizados de X_0 terão de ser calculados nessa produção.

Chama-se ***conjunto de saída*** *da produção* `p` *e representa-se por A_{out} ao conjunto formado pelos atributos sintetizados de X_0 e pelos atributos herdados de X_i.*
Chama-se ***conjunto de entrada*** *da produção* `p` *e representa-se por A_{in} ao conjunto formado pelos atributos herdados de X_0 e pelos atributos sintetizados de X_i.*

O conjunto A(X) caracteriza completamente, do ponto de vista semântico, cada símbolo $X \in N \cup T$. Numa frase concreta cada instância do símbolo X na árvore de derivação ficará associada a um conjunto de ***propriedades*** *–* ***atributos com os seus valores concretos*** *que descrevem completamente o seu significado. Uma árvore de derivação assim preenchida com as propriedades de cada símbolo, ou seja carregando o significado de casa nodo, chama-se uma* ***árvore de derivação decorada****.*
As regras de cáculo indicam precisamente como valorar os atributos, face ao contexto concreto de X, para obter o dito significado. A partir do significado de cada símbolo constrói-se o significado da frase, que se considera como sendo dado pelas propriedades do axioma da gramática (precisamente a raíz da árvore decorada). Tendo o significado da frase, e de cada símbolo, é possivel transformar, ou traduzir, a frase para obter o resultado pretendido – para definir rigorosamente essa transformação usam-se as regras de tradução.
Porém em uma frase só pode ser processada se for sintacticamente válida e se estiver correcta semanticamente. A correcção sintática é assegurada pelas produções da ***GIC****; a validade semântica é estabelecida pelo conjunto de condições de contexto da* ***GA****, as quais explicitam, ao nível de cada produção, as restrições que os valores concretos dos atributos terão de respeitar para a* ***frase fazer sentido*** [1].

[a] `p` da forma $p\ :\ X_0\ \rightarrow\ X_1\ ...\ X_i\ ...\ X_n$.

Para facilitar a compreensão dos conceitos definidos acima e melhor ilustrar as relações entre eles, esquematiza-se na Figura 1 as árvores correspondentes a duas produções `p` e `p'` associando-se a cada nodo da árvore o conjunto dos seus atributos herdados (ilustrados a cor azul) e sintetizados (ilustrados a cor vermelha). Por cada atributo do conjunto de saída A_{out}, cujo valor tem de ser calculado no contexto da produção `p`, é mostrada uma função de cálculo f_i, que usa os valores dos atributos em A_{in} ou contantes para determinar o valor desse atributo. Note-se que os atributos A_{out} são denotados por triângulos que apontam para fora da produção `p` (para a árvore do progenitor ou para a as subárvores dos filhos); os atributos A_{in} são por sua vez, representados por triângulos que apontam para dentro da produção `p`.

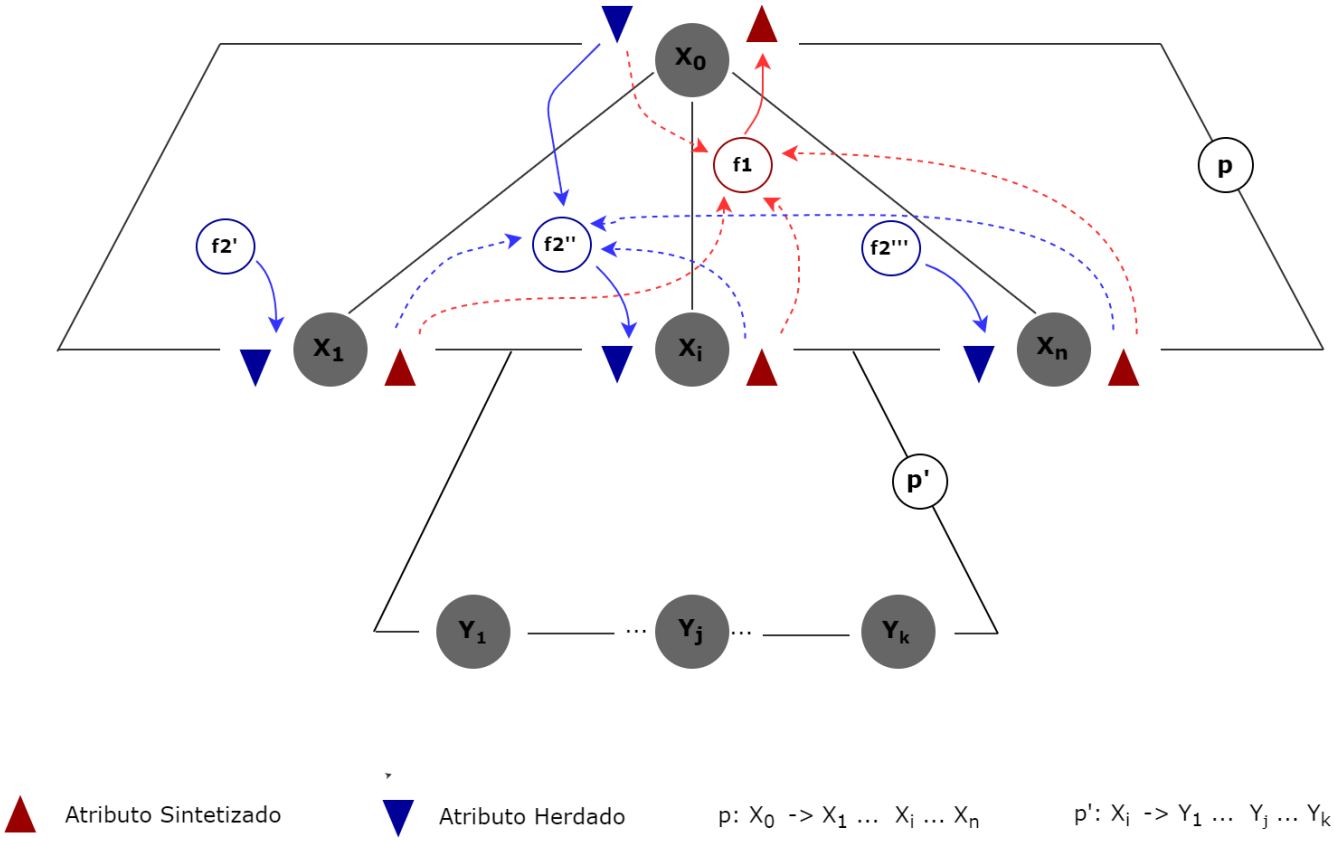

Figura 1: Esquema atributivo associado a uma produção da GA

Nesta GA vamos adicionar atributos aos símbolos da gramática, regras de cálculo dos atributos e regras de tradução em todas as produções da gramática, de modo a ser possível realizar os cálculos necessários para obter *comprimento da lista* e *quantidade de números* (Listing 4).

```
/*
 * Linguagem: "Lista Mista"
 * Processador: Contador do Comprimento da lista e quantidade de nums  contidos na lista
 * Uso de gramatica de atributos: apenas atributos sintetizados
 * PRH 2017.10.02
 */

grammar gcs17F1Ex1a_GA;

lista returns[int comp, int contaN]
@after{
        System.out.println("Comprimento da lista: "+ $lista.comp );
        System.out.println("Quantidade de Numeros: "+ $lista.contaN );
      }
      : 'LISTA' elems  '.'
        {$lista.comp=$elems.comp; $lista.contaN=$elems.contaN;}
      ;

elems returns[int comp, int contaN]
      : elem      {$elems.comp=1; $elems.contaN=$elem.num;}
       (',' elem {$elems.comp++; $elems.contaN=$elems.contaN + $elem.num;})*
      ;

elem returns[int num]
     : NUMERO {$elem.num=1; }
     | PAL    {$elem.num=0; }
     ;
```

```
28
29
30 /* Definicao do Analisador LEXICO */
31
32 PAL:      [a-zA-Z][-a-zA-Z_0-9]* ;
33
34 NUMERO: '0'..'9'+ ; // [0-9]+
35
36 Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 4: Notação BNF-estendido do AnTLR – Gramática Atributos (GA_a)

Note-se que os atributos que usamos são associados ao símbolo através da palavra reservada `returns` que precede a lista entre `[ ]`. Estes atributos, designam-se **atributos sintetizados** (sintetizam a informação a partir das folhas e transportam-na pela árvore acima).

Para calcular o comprimento da lista e a quantidade de números usamos **regras de cálculo dos atributos**, como por exemplo, '`$elems.contaN=$elems.contaN + $elem.num;`'.
Também recorremos a **regras de tradução** para obtermos o resultado final dos cálculos, como por exemplo, '`System.out.println("Comprimento da lista:  "+ $lista.comp);`'.

b) acrescente aos resultados anteriores a média[2] de todos os números que apareçam na lista.

Resolução:

Para solucionar o problema vamos acrescentar à gramática de atributos anterior o cálculo do somatório dos elementos numéricos da lista; a partir do somatório e do contador de números é imediato e simples calcular o valor do atributo média da lista.

Tal como na gramática anterior também iremos recorrer a atributos nos símbolos da gramática, regras de cálculo dos atributos e regras de tradução em todas as produções da gramática, para realizar os cálculos necessários.

Para enriquecer o exercício, vamos acrescentar uma restrição ao problema: *uma lista só é válida se o seu comprimento for maior ou igual a 5 e se for menor ou igual a 10; fora desse intervalo de comprimento a lista é considerada errada*. Neste caso, para verificar a restrição semântica, necessitamos de usar uma condição de contexto que vai testar o valor do atributo comprimento da lista, como se vê na Listing 5.

```
1  /*
2   * Linguagem: "Lista Mista"
3   * Processador: Contador do Comprimento da lista e Quantidade de nums;
4   *              Calculador da Media dos numeros  contidos na lista
5   *              Verificador da Restricao Semantica: "A lista TEM DE TER Comprimento [5..10]"
6   * Uso de gramatica de atributos: apenas atributos sintetizados e condicoes de contexto
7   * PRH 2017.10.02
8   */
9
10 grammar gcs17F1Ex1b_GA;
11
12 lista returns[int comp, int contaN, float media]
13 @after{
14        //condicao de contexto
15        if (($lista.comp >= 5) && ($lista.comp <= 10)){
16             System.out.println("Comprimento da lista: "+ $lista.comp );
17             System.out.println("Quantidade de Numeros: "+ $lista.contaN );
18             System.out.println("Media de Numeros: "+ $lista.media );}
19          else{System.out.println("Erro semantico: fora do intervalo permitido");}
20       }
21       : 'LISTA' elems  '.'
22         {$lista.comp=$elems.comp; $lista.contaN=$elems.contaN;
23          $lista.media=(float)$elems.soma/$elems.contaN;
24         }
```

[2] lembre-se que é o quociente de uma divisão e portanto um *número real*.

```
25        ;
26
27  elems returns[int comp, int contaN, int soma]
28        : elem      {$elems.comp=1; $elems.contaN=$elem.num; $elems.soma=$elem.val;}
29         (',' elem {$elems.comp++; $elems.contaN=$elems.contaN + $elem.num;
30                    $elems.soma=$elems.soma + $elem.val;})*
31        ;
32
33  elem returns[int num, int val]
34       : NUMERO {$elem.num=1; $elem.val=$NUMERO.int;}
35       | PAL    {$elem.num=0; $elem.val=0;}
36       ;
37
38
39  /* Definicao do Analisador LEXICO */
40
41  PAL:      [a-zA-Z][-a-zA-Z_0-9]* ;
42
43  NUMERO: '0'..'9'+ ; // [0-9]+
44
45  Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 5: Notação BNF-estendido do AnTLR – Gramática Atributos (GA_b)

Tal como no exercício anterior também recorremos a atributos sintetizados para realizar os cálculos necessários. Esses atributos são declarados numa lista entre parêntesis retos escrita à frente do símbolo do lado esquerdo da produção, sendo usada a palavra-reservada `returns` antes da lista de atributos.

Como foi dito anteriormente, recorremos às condições de contexto para verificar a restrição semântica (A lista tem de ter comprimento entre 5 e 10, caso contrário imprime um erro semântico). Esta restrição semântica foi introduzida no bloco `@after{}` associado ao axioma da gramática, `lista`, onde já se tinham incluido as regras de tradução: `'if(($lista.comp >= 5 ) && ($lista.comp <= 10)) {...} else{System.out.println ("Erro semantico: fora do intervalo permitido");}'`.

c) obrigar a que quantidade de palavras seja igual à quantidade de números.

Resolução:

Para resolver a questão é necessário usar atributos nos símbolos da gramática, regras de cálculo dos atributos e regras de tradução em todas as produções da gramática, de forma a calcular a quantidade de palavras e de números contidos na lista. Por fim, recorremos novamente a condições de contexto para verificar se o valor do atributo quantidade de palavras é igual ao valor do atributo quantidade de números (Listing 6).

```
1   /*
2    * Linguagem: "Lista Mista"
3    * Processador: Verificador/Contador da quantidade de nums  e da quantidade das palavras
       contidos na lista
4    * Uso de gramatica de atributos: apenas atributos sintetizados e condicoes de contexto
5    * PRH 2017.10.02
6    */
7
8   grammar gcs17F1Ex1c_GA;
9
10  lista returns[int contaP, int contaN]
11  @after{
12         //condicao de contexto
13         if ($lista.contaP != $lista.contaN){
14              System.out.println("Erro semantico: Quantidade Numeros e Palavras nao e IGUAL!");}
15        }
16        : 'LISTA' elems  '.'
17          {$lista.contaP=$elems.contaP; $lista.contaN=$elems.contaN;
18          }
19        ;
20
21  elems returns[int contaP, int contaN]
```

```
      : elem      {$elems.contaP = $elem.palav; $elems.contaN=$elem.num;}
       (',' elem {$elems.contaN=$elems.contaN + $elem.num;
                  $elems.contaP=$elems.contaP + $elem.palav;})*
      ;

elem returns[int num, int palav]
     : NUMERO {$elem.num=1; $elem.palav=0;}
     | PAL    {$elem.num=0; $elem.palav=1;}
     ;


/* Definicao do Analisador LEXICO */

PAL:     [a-zA-Z][-a-zA-Z_0-9]* ;

NUMERO: '0'..'9'+ ; // [0-9]+

Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 6: Notação BNF-estendido do AnTLR – Gramática Atributos (GA_c)

Esta restrição semântica também foi introduzida no bloco `@after{}` associado ao axioma da gramática, `lista`. Neste caso, a restrição semântica testa se o atributo quantidade de palavras é diferente do valor do atributo quantidade de números. Caso isso se verifique imprime um erro semântico (`'if($lista.contaP != $lista.contaN) {System.out.println("Erro semantico:  Quantidade Numeros e Palavras nao e IGUAL!");}'`).

d) calcular o máximo dos números.

Resolução:

Para solucionar o problema, mais uma vez, é necessário usar atributos sintetizados nos símbolos da gramática, regras de cálculo dos atributos e regras de tradução em todas as produções da gramática, de forma a identificar o número máximo contido na lista (Listing 7). Neste caso, para além dos atributos sintetizados também é necessário utilizar **atributos herdados** porque vai ser preciso passar informação do contexto esquerdo (o máximo encontrado no prefixo da lista já percorrida) para o elemento que está a ser reconhecido de modo a compará-lo com esse máximo anterior para se poder então decidir qual será o máximo seguinte (se se mantém o herdado, o anterior, ou se passa a ser o novo elemento).

```
/*
 * Linguagem: "Lista Mista"
 * Processador: Calculo do Maior inteiro
 * PRH 2017.09.29
 */

grammar gcs17F1Ex1d_GA;

@header{
        import java.util.*;
}
lista
     returns [int maximo]
@after{
         System.out.println("Maior Numero da lista de inteiros: "+$lista.maximo);
       }
      : 'LISTA' elems  '.'
       { $lista.maximo=$elems.max; }
      ;
elems
     returns [int max]
      : e1=elem[Integer.MIN_VALUE]    { $elems.max=$e1.max; }
        (',' e2=elem[$elems.max]      { $elems.max=$e2.max; })*
      ;
elem [int maxAnt]
     returns [int max]
```

```
27      : NUMERO { if ($NUMERO.int > $elem.maxAnt) $elem.max=$NUMERO.int; else $elem.max = $elem.
        maxAnt; }
28      | PAL    { $elem.max = $elem.maxAnt; }
29      ;
30
31
32  /* Definicao do Analisador LEXICO */
33
34  PAL:     [a-zA-Z][-a-zA-Z_0-9]* ;
35
36  NUMERO: '0'..'9'+ ; // [0-9]+
37
38  Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 7: Notação BNF-estendido do AnTLR – Gramática Atributos (GA_d)

Na linha 22, utilizamos a constante `Integer.MIN_VALUE` para inicializar o valor do atributo herdado `maxAnt` do símbolo `elem` com o menor valor inteiro possível (de modo a atualizar o máximo logo na primeira comparação, pois qualquer que seja o número reconhecido esse número é garantidamente maior ou igual ao menor valor do sistema). Uma outra possibilidade seria inicializar esse atributo herdado `maxAnt` do símbolo `elem` a zero, mas essa hipótese só estaria correto se os números fossem todos maiores ou igual a zero.

Note-se que na linha 25 (Listing 7) utilizamos ***atributos herdados***. Os atributos herdados estão associados ao símbolo `elem` devendo (na metanotação do AnTLR) ser declarados (*tipo* do atributo e **nome** do atributo) numa lista entre `[ ]` colocada imediatamente a seguir ao nome do símbolo e antes de `return`. Estes designam-se de herdados porque transportam a informação do contexto esquerdo ou direito, ou mesmo do pai do símbolo, pela subárvore abaixo, o que significa que o seu valor tem obrigatoriamente que ser passado numa lista entre `[ ]` quando o respetivo símbolo não terminal surge no lado direito de uma produção; note-se ainda que este, ou estes, valores passados no ato de reconhecer o símbolo podem ser *constantes* do tipo declarado, podem ser *variáveis* (incluindo o nome do mesmo ou outro atributo do símbolo em causa ou do seu pai ou irmãos), ou podem ser *expressões*.
No exemplo da listagem 7 o símbolo não terminal `elem` recebe apenas um atributo herdado `maxAnt` na produção que vai da linha 26 à 33, podendo o seu valor ser usado nas regras de cálculo de outros atributos dessa mesma produção (verificar as linhas 26 a 33 onde o dito atributo é usado). O valor desse atributo de desse símbolo tem de receber obrigatoriamente um valor quando o símbolo em causa (`elem`) é reconhecido — ver linhas 23, primeira utilização, e 24, subsequentes utilizações 0 ou mais).

e) calcular o somatório apenas dos números contidos entre 'start' e 'stop'.

Resolução:

Para resolver este problema temos de realizar um somatório condicionado da lista, ou seja, apenas vamos somar os números quando surgir a palavra *'start'* e até surgir a palavra *'stop'*, ou se atingir o fim da lista; se na mesma lista voltar a surgir *'start'* o somatório deve ser retomado até uma das condições de fim aparecer. Deste forma, é necessário usar atributos sintetizados e herdados associados aos símbolos da gramática, e então definir as respetivas regras de cálculo dos atributos e regras de tradução às produções da gramática onde tal faça sentido, de forma a realizar o somatório condicionado (Listing 8). É importante notar que o atributo herdado que aqui se vai usar, `'startIn'`, deve passar a informação de que já foi encontrado numa dado momento a palavra *'start'*, ou a palavra *'stop'*, o que se reflete nos restantes elementos que surjam a seguir.

```
1  /*
2   * Linguagem: "Lista Mista"
3   * Processador: Calculo da Soma Condicionada as Palavras "start" / "stop"
4   * PRH+CEA 2017.10.10
5   */
6
7  grammar gcs17F1Ex1e_GA;
8
9  @header{
10         import java.util.*;
11 }
12 lista
```

```
13      returns [int soma]
14  @after{
15            System.out.println("Somatorio condicionado da lista = "+$lista.soma);
16          }
17        : 'LISTA' elems  '.'
18          { $lista.soma=$elems.soma; }
19        ;
20  elems
21      returns [int soma]
22  @init { boolean aux=false; }
23        : e1=elem[aux]    { $elems.soma = $e1.val; aux = $e1.startOut; }
24          (',' e2=elem[aux]      { $elems.soma += $e2.val; aux = $e2.startOut; })*
25        ;
26  elem [boolean startIn]
27      returns [int val, boolean startOut]
28  @init { $elem.startOut = $elem.startIn; }
29      : NUMERO { if ($elem.startIn) $elem.val = $NUMERO.int; else $elem.val = 0; }
30      | PAL    { if ($PAL.text.equals("start")) { $elem.startOut = true; }
31                 if ($PAL.text.equals("stop"))  { $elem.startOut = false;}
32                 $elem.val = 0; }
33      ;
34
35
36  /* Definicao do Analisador LEXICO */
37
38  PAL:      [a-zA-Z][-a-zA-Z_0-9]* ;
39
40  NUMERO:  '0'..'9'+ ; // [0-9]+
41
42  Sep: ('\r'? '\n' | ' ' | '\t')+  -> skip;
```

Listing 8: Notação BNF-estendido do AnTLR – Gramática Atributos (GA_e)

Neste exercício recorremos a atributos `'startIn'` (herdado) e `'startOut'` (sintetizado) do tipo booleano — `boolean` (que é um tipo de dados com apenas dois valores possíveis: *verdadeiro* ou *falso*) — para controlar o somatório. Na linha 22 inicializamos a variável `'aux'` a falso para a passar como valor ao atributo `'startIn'`, a primeira vez que um elemento é reconhecido (linha 23) obrigatoriamente, pois o primeiro elemento nunca poderá ser somado[3]. Nas vezes seguintes (zero ou mais) o valor de `'aux'`, que é usado para atribuir um valor ao atributo herdado `'startIn'` de `elem` nas ocorrências seguintes deste símbolo (linha 24), é definido como sendo o valor do respetivo atributo sintetizado `'startOut'` da ocorrência anterior de `elem` (ver atribuições nas linhas 23 e 24).

Referências

[1] Pedro Manuel Rangel Santos Henriques. Brincando às Linguagens com rigor: Engenharia Gramatical. Technical report, Universidade do Minho, November 2013.

[2] Ivan Ricarte. *Introdução à Compilação*. Elsevier Brasil, 2012.

[3] Se for um número como é o primeiro nunca está depois do *'start'* e portanto deve ser ignorado; se for uma palavra deve também ser ignorado, a menos que seja *'start'* pois nesse caso deve dar início ao somatório.