Questões em Tradução de Linguagens

Capítulos 3 do Pratt e do Sebesta

- Sintaxe de uma LP é a forma de suas expressões, instruções e unidades de programa:
  - descreve a sequência de símbolos que tornam válido um programa,
  - em uma linguagem natural é o arranjo de palavras como elementos em uma sequência para mostrar o relacionamento entre elas.

  $\Rightarrow$ sintaxe é o conjunto de regras que determinam quando uma sentença é bem formada.
- Semântica de uma LP é o significado de suas expressões, instruções e unidades de programa.

Programas exemplos feitos pelo Prof. Wagner Teixeira

Introdução

- Quem deve usar as definições de uma LP:
  - outros projetistas de linguagens,
  - implementadores,
  - programadores (usuários da linguagem)
- definições básicas:
  - sentença: string de caracteres sobre um alfabeto.
  - linguagem: conjunto de sentenças.
  - lexema: unidade de menor nível sintático de uma linguagem (p.ex.: *, for, begin).
  - token: categoria de lexemas (p.ex.: identificador).
  - símbolos (tokens) são combinados em sentenças para formar um programa.

Questões em Tradução de Linguagens

Regras sintáticas são suficientes?

- Anos 60: para projetar LP só era necessário uma sintaxe formal, tipo BNF (Backus-Naur Form).

  <inteiro> ::= <sinal><digito> | <inteiro><digito>

  - Em expressões ambíguas, as regras sintáticas **não** são suficientes para determinar as interpretações corretas:
    - associativa: 1-2-3 = -4 ou 2 ?
    - precedência: 2+3*4 = 20 ou 14 ?
    - tipos: x = 2.45+3.67 = 5, se x e + são inteiros
      = 6, se x inteiro e + real
      = 6.12, se x e + são reais

- Declarações, sequência de controle, operações, ambiente de referência, etc, são necessários.

inteiro é produzido por sinal seguido de dígito ou inteiro seguido de dígito

Sintaxe de LP

Critérios gerais sintáticos

- Propósito principal da sintaxe:
  - prover notação para comunicação entre o programador e o processador da linguagem.
- Propósitos secundários da sintaxe:
  - legibilidade: ler e entender facilmente,
  - capacidade de escrita: facilitar a programação,
  - facilidade de verificação: facilitar o exame da exatidão do código,
  - facilidade de tradução: facilitar trabalho do tradutor,
  - ausência de ambiguidade.

Propósitos secundários da sintaxe

- Legibilidade
  - Pela inspeção do programa, as estruturas dos algoritmos e dos dados DEVEM ficar aparentes, sem necessitar consultar documentação adicional:
    - comandos estruturados palavras chaves comentários
      declarar dados op. Mnemônicos campo-livre
      variedade de construções sintáticas

Exemplo: Cobol (auto-documentado)

Contra exemplo: programas Lisp ou Mumps

Legibilidade

Exemplo: Programa Cobol


```
IDENTIFICATION DIVISION.
PROGRAM-ID. SUM-OF-PRICES.
AUTHOR. T-PRATT.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. SUN.
OBJECT-COMPUTER. SUN.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT ARQ-ENTRADA  ASSIGN TO INPUT.
    SELECT ARQ-SAIDA  ASSIGN TO OUTPUT.
DATA DIVISION.
FILE SECTION.
FD ARQ-ENTRADA  LABEL RECORD IS OMITTED.
01 ITEM-PRECO.
      02  ITEM      PICTURE   X(30).
      02  PRECO  PIC           9999V99.
FD ARQ-SAIDA.
 01 LINHA-IMPRESSORA  PIC X(80).
WORKING-STORAGE  SECTION.
77 TOTAL          PIC 9999V99, VALUE 0, USAGE IS
     COMPUTATIONAL.
77 CONTADOR   PIC  9999, VALUE 0, USAGE IS
     COMPUTATIONAL.
01 LINHA-TOTAL.
     02 FILLER VALUE  'SOMA  = ' PIC X(12).
     02 TOTAL-SOMA PIC   $$,$$$,$$9.99.
     02 FILLER   PIC  X(15), VALUE  '  TOTAL DE ITENS   '.
     02 TOTAL-CONTADOR  PIC  ZZZ9.
01 LINHA-SAIDA.
    02 CONTADOR  PIC  Z,ZZ9.
    02 ITEM           PIC X(30).
    02 FILLER         PIC X(05), VALUE SPACES.
    02 PRECO          PIC $,$$9.99.
```

Legibilidade


```
PROCEDURE DIVISION.
INICIA.
        OPEN INPUT ARQ-ENTRADA  AND OUTPUT ARQ-SAIDA.
LEDADOS.
         READ ARQ-ENTRADA AT END GO TO FINALIZA.
         ADD PRECO  OF ITEM-PRECO TO  TOTAL.
         ADD 1 TO  CONTADOR.
         MOVE CORRESPONDING  ITEM-PRECO     TO   LINHA-SAIDA.
         MOVE CONTADOR           TO  CONTADOR  OF  LINHA-SAIDA.
         WRITE LINHA-IMPRESSORA FROM LINHA-SAIDA.
         GO TO LEDADOS.
FINALIZA.
        MOVE   TOTAL  TO   TOTAL-SOMA.
        MOVE   CONTADOR TOTAL-CONTADOR.
        WRITE  LINHA-IMPRESSORA FROM LINHA-TOTAL.
        CLOSE  ARQ-ENTRADA  AND  ARQ-SAIDA.
        STOP RUN.
```

Legibilidade

Contra exemplo Programa Lisp


```
; ARQ-ENTRADA COM ZERO OU MAIS REGISTROS (LISTA), CADA QUAL CONTENDO ITEM E PRECO.
; LE ARQ-ENTRADA E GRAVA SEUS REGISTROS NO ARQ-SAIDA, TOTALIZANDO OS VALORES.
; ULTIMO REG: TOTAL DE REGISTROS GRAVADOS E TOTAL DE VALORES ACUMULADOS

(DEFUN PROCEDIMENTO(ARQ-ENTRADA  ARQ-SAIDA)

   ((PROBE-FILE ARQ-ENTRADA)
        (OPEN-INPUT-FILE       ARQ-ENTRADA)
         (OPEN-OUTPUT-FILE   ARQ-SAIDA)
         (SETQ CONTADOR  0   TOTAL  0)
          (LOOP
             (TERPRI)                                          ; MUDA DE LINHA
             ((NULL (SETQ ITEM-PRECO (READ)))                  ; EOF
                   (PRINT  (PACK*  "QTE DE ITENS =  " CONTADOR "     VALOR TOTAL = " TOTAL)) )
             (INCQ CONTADOR 1)                                 ;  INCREMENTA CONTADOR ITENS LIDOS
             (INCQ TOTAL (NTH 1 ITEM-PRECO))                   ; ADICIONA O VALOR DO ITEM AO TOTAL
             (PRINT (PACK* "ORDEM " CONTADOR  "   ITEM= " (CAR ITEM-PRECO)
                           "    VALOR =  "  (CDR ITEM-PRECO)))  )
          (CLOSE-OUTPUT-FILE          ARQ-SAIDA)
          (CLOSE-INPUT-FILE           ARQ-ENTRADA) ))
```

Propósitos secundários da sintaxe

- **Facilidade de escrever**
  - Características sintáticas que facilitam escrever um programa:
    - Enfatizar estruturas sintáticas concisas e regulares

      i++;

      for (<exp1>;<exp2>;<exp3>) <comando>;
    - Convenções sintáticas implícitas
      - declaração implícita de inteiro (I-N) e real em Fortran.
      - regras de associatividade e prioridade de avaliação implícitas de operadores (+-/*)
    - Comandos estruturados, operadores mnemônicos, campo-livre, tamanho livre de identificadores, etc.

Exemplo: Fortran, C, C++

Contra exemplo: Pascal, Cobol

Facilidade de escrever

Exemplo: Programa Fortran


```
      PROGRAM MAIN
C   DADO DUAS MATRIZES QUADRADAS A E B, OBTER  C = A + B
        PARAMETER (MAX=99)
        INTEGER T
        REAL  A (MAX,MAX), B (MAX,MAX), C (MAX,MAX)
10      WRITE(6, 100)  MAX
100     FORMAT(" ENTRE COM A DIMENSAO DAS MATRIZES. O MAXIMO EH = ", I5)
        READ (5, 200) T
200     FORMAT(I5)
        IF (T.LE.0.OR.T.GT.MAX) GO TO 500
        PRINT *, "ENTRE COM OS VALORES  DA  MATRIZ  A"
        READ *, (A(L,K) , L=1,T, K=1,T)
        PRINT *, "ENTRE COM OS VALORES  DA  MATRIZ  B"
        READ *, (B(L,K),L=1,T, K=1,T)
        DO 400 K=1,T
          DO 300 L=1,T
300           C(L,K) = A(L,K) + B(L,K)
400     CONTINUE
        PRINT *,  (C(L,K), L=1,T,K=1,T)
        GO TO 800
500     WRITE(6, 600) MAX
600     FORMAT( "DIMENSAO ERRADA. MENOR QUE ZERO OU MAIOR QUE ", I5)
        GO TO 10
800     STOP
        END
```

Facilidade de escrever

Contra exemplo: Programa Pascal


```
Program somatrizes (input,output,infile);
  const max=99;
  type mat_real = array [1..max, 1..max]
        of real;
  var infile: text;     a,b,c: mat_real;
        l,k,t: integer;
  begin
     writeln ('Entre com a dimensão
        das matrizes quadradas. Valor
        máximo é ', max:5);
     repeat   readln (t);
        if (t <=0) or (t>max)
        writeln ('valor da dimensão
        invalido');
     until (t >0) and (t < max);
     writeln ('entre com o valores da
        matriz A, por linha');
     for l:=1 to t do          { lê a matriz A}
        for k:=1 to t do  read (a[l,k]) ;
     for l:=1 to t do         {lê a matriz B}
        for k:=1 to t do
           begin read (b[l,k]) ;
                 c[l,k] := a[l,k]+b[l,k]
            end;
     for l:=1 to k do
        begin writeln;
              for k:=1 to k do
                 write (c[l,k]:10:2);
         end;
end. { fim do programa}
```

Propósitos secundários da sintaxe

- Facilidade de Verificação
  - As estruturas sintáticas da LP devem facilitar o exame da exatidão do código gerado:
    - envolve aspectos sintáticos e semânticos.
    - entender automaticamente cada comando é fácil.
    - o processo de criar um programa correto é extremamente difícil.
    - há necessidade de técnicas para provar matematicamente a corretude de um programa.

Exemplo: linguagens declarativas puras.

Contra exemplo: linguagens imperativas.

Propósitos secundários da sintaxe

- Facilidade de tradução
  - A construção de tradutores é facilitada pela:
    - regularidade das estruturas sintáticas.
    - pequena variedade de estruturas.

  Exemplos: Lisp, Haskell, Hugs, ML

  pela simplicidade de suas estruturas são ruins de ler e escrever mais fáceis de traduzir.

  Contra exemplos: Cobol

  semântica simples, fácil de ler, ruim para escrever e difícil de traduzir devido a variedade de estruturas (comandos e declarações).

Propósitos secundários da sintaxe

- Ausência de ambiguidade
  - Idealmente, cada construção sintática deve ter uma única interpretação:
    - nem sempre acontece nas LP, pois uma estrutura ambígua permite duas ou mais interpretações diferentes.
  - A interpretação de uma estrutura sintática isoladamente não traz problemas.
  - A ambiguidade aparece quando são consideradas combinações entre as diversas estruturas sintáticas permitidas pelas regras da LP.

Ausência de ambiguidade

- Exemplos:

a) chamada de funções e referência a arrays em Fortran:

A(I,J) é chamada de função ou referência ao elemento A_{ij} do array A?

b) aninhamento de if

if <bexp> then <comando$_1$> else <comando$_2$> (ok)

if <bexp> then <comando$_1$> (ok)

if <be$_1$> then if <be$_2$> then <comando$_1$> else comando$_2$; (?)

- em Algol? R.: uso de begin e end.
- em C e Pascal? R.: regra arbitrária: else se refere ao then mais próximo.
- em Ada? R.: uso de endif.

Elementos Sintáticos de uma LP

- Conjunto de caracteres
  - Ao projetar a sintaxe de uma LP, a primeira escolha é o conjunto de caracteres. (**tendência é usar Unicode?**)
- Identificadores
  - Uma cadeia de letras e dígitos, começando com uma letra, é largamente aceito.
- Símbolos de operadores
  - Usual adotar uma combinação de caracteres especiais para alguns operadores e identificadores para outros.
- Palavra-chave
  - identificador usado como uma parte fixa de um comando.
  - Palavra-reservada não pode ser usada pelo programador.
  - Palavras opcionais (noise): melhorar a legibilidade.

Elementos Sintáticos de uma LP

- Comentários
  - Texto inserido no programa com propósito de documenta-lo.
    - Linha de comentário (toda a linha), com campo fixo.

      Fortran — C (na coluna 1) seguido do comentário.
    - Delimitado por caracteres especiais (mais de uma linha).

      C — /* comentário sem limites de linhas */

      Pascal — (* comentário *) ou { comentário }
    - Inicia em qualquer posição, indo até o final da linha.

      LISP — ; seguido do comentário

      ADA — - seguido do comentário

      C++ — // seguido do comentário

      Fortran 90 — ! seguido do comentário

- Espaço em branco (tem papel sintático)
  - Usado como separador, exceto se em uma string.

Elementos Sintáticos de uma LP

- Exemplos de uso de espaço em branco
  - Cobol: move x to y ≡ move x to y
  - Pascal
    a) while b < c do b:=b+1;
    b) st := `João e Maria` + ` ` + `são casados`;
  - Lisp
    (defun mdc(a b)(cond ((= b 0) a) (T (mdc b (mod a b)))))
- Delimitador e agrupamento (brackets)
  - Marca o início ou fim de unidade sintática
    comando, expressão, etc
  - Aumentam a legibilidade, facilitam a análise sintática e removem ambigüidades.
  - Chaveamento (brackets) são pares de delimitadores.
    (...), begin...end

Elementos Sintáticos de uma LP

- Sintaxe de campo fixo (estritamente)

  Cada elemento do comando precisa aparecer numa dada posição da linha de entrada.

  Ex.: versões antigas da linguagem Assembler, JCL, etc.

- Sintaxe de campo fixo (parcialmente)

  Alguns elementos do comando tem posição fixa outros são livres.

  Ex.: Fortran, Cobol

- Sintaxe de campo livre

  Os elementos do comando podem começar em qualquer lugar da linha de entrada e os elementos na sequência podem ser separados por um ou mais espaços.

  Ex.: Algol, Pascal, C, etc.

Elementos Sintáticos de uma LP

- Expressões
  - **São funções que acessam os objetos de dados e retornam algum valor.**
  - São os blocos básico de construções de comandos.
  - Em linguagens imperativas, em combinação com o comando de atribuição, permitem alterar o estado da máquina.
    Ex.: A = cos(x)+y^2;
  - Em linguagens funcionais, o fluxo de controle é feito pela avaliação de expressões (funções).
    Ex. Em Lisp, um programa é uma expressão simbólica
    (mapcar '(lambda(x) (* x x)) '(1 2 3 4 5 6 7 8))

Elementos Sintáticos de uma LP

- Comandos
  - É o principal elemento sintático das linguagens imperativas.
  - As linguagens funcionais puras não possuem comandos; elas são declarativas.
  - Os comandos podem ser simples ou compostos (estruturados ou aninhados)
  - A sintaxe dos comandos influi na ortogonalidade, legibilidade e facilidade de escrita de uma linguagem.
  - Cobol tem uma sintaxe de comandos prolixa, muito específica para cada tipo de comando.