Universidad Nacional de Rosario

Facultad de Ciencias Exactas, Ingeniería y Agrimensura

Especificación del Lenguaje Imperativo Simple

Autores:
Caporalini, Joaquín
Arroyo, Joaquín

19 de septiembre de 2022

Índice

1. Sintáxis

1.1. Sintáxis abstracta

```
intexp  ::= nat
          | var
          | −u intexp
          | intexp + intexp
          | intexp −b intexp
          | intexp ∗ intexp
          | intexp / intexp
          | boolexpatom ? intexp : intexp
boolexpatom ::= true
              | false
              | ¬boolexpatom
              | boolexp
boolexp ::= intexp == intexp
          | intexp != intexp
          | intexp < intexp
          | intexp > intexp
          | intexp ∧ intexp
          | intexp ∨ intexp
          | boolexpatom
comm    ::=  skip
          | var = intexp
          | comm; comm
          | if boolexp then comm else comm
          | while boolexp do comm
```

1.2. Sintáxis concreta

```
digit   ::= '0' | '1' | ... | '9'
letter  ::= 'a' | ... | 'z'
nat     ::= digit | digit nat
var     ::= letter | letter var
intexp  ::= nat
          | var
          | '−' intexp
          | intexp '+' intexp
          | intexp '−' intexp
          | intexp '∗' intexp
          | intexp '/' intexp
          | boolexpatom '?' intexp ':' intexp
          | '(' intexp ')'
boolexpatom ::= true
              | false
              | '!' boolexpatom
              | '(' boolexp ')'
boolexp ::= intexp '==' intexp
          | intexp '!=' intexp
          | intexp '<' intexp
          | intexp '>' intexp
          | intexp '&&' intexp
          | intexp '||' intexp
          | boolexpatom
```

```
comm    ::=   'skip'
            | var '=' intexp
            | comm ';' comm
            | 'if' boolexp '{' comm '}'
            | 'if' boolexp '{' comm '}' 'else' '{' comm '}'
            | 'while' boolexp '{' comm '}'
```

2. Semántica

2.1. Semántica de expresiones

```
nv ::= int
bv ::= true | false
```

El significado de cada expresion depende de un estado que le asigna un valor (entero) a sus variables. Llamamos Σ al conjunto de estados que le atribuye a cada variable un valor entero.

$$\frac{}{\langle nv,\sigma\rangle \Downarrow_{exp} nv}\ \text{NVal}$$

$$\frac{}{\langle x,\sigma\rangle \Downarrow_{exp} x}\ \text{Var}$$

$$\frac{\langle e,\sigma\rangle \Downarrow_{exp} n}{\langle -_u e,\sigma\rangle \Downarrow_{exp} -n}\ \text{UMinus}$$

$$\frac{\langle e_0,\sigma\rangle \Downarrow_{exp} n_0 \quad \langle e_1,\sigma\rangle \Downarrow_{exp} n_1}{\langle e_0 + e_1,\sigma\rangle \Downarrow_{exp} \mathrm{n}_0 + n_1}\ \text{Plus}$$

(Análogo para - y $\times$)

$$\frac{\langle e_0,\sigma\rangle \Downarrow_{exp} n_0 \quad \langle e_1,\sigma\rangle \Downarrow_{exp} n_1 \quad n1 \neq 0}{\langle e_0/e_1,\sigma\rangle \Downarrow_{exp} \mathrm{n}_0/n_1}\ \text{Div}$$

$$\frac{\langle e_0,\sigma\rangle \Downarrow_{exp} n_0 \quad \langle e_1,\sigma\rangle \Downarrow_{exp} n1}{\langle e_0 == e_1,\sigma\rangle \Downarrow_{exp} \mathrm{n}_0 = n_1}\ \text{Eq}$$

(Análogo para $>$, $<$ y $\neq$)

$$\frac{}{\langle bv,\sigma\rangle \Downarrow_{exp} bv}\ \text{BVal}$$

$$\frac{\langle p,\sigma\rangle \Downarrow_{exp} b}{\langle \neg p,\sigma\rangle \Downarrow_{exp} \neg b}\ \text{Not}$$

$$\frac{\langle p_0,\sigma\rangle \Downarrow_{exp} b_0 \quad \langle p_1,\sigma\rangle \Downarrow_{exp} b_1}{\langle p_0 \wedge \mathrm{p}_1,\sigma\rangle \Downarrow_{exp} b_0 \wedge \mathrm{b}_1}\ \text{And}$$

(Análogo para $\vee$)

$$\frac{\langle p_0,\sigma\rangle \Downarrow_{exp} \textbf{true} \quad \langle e_0,\sigma\rangle \Downarrow_{exp} n_0}{\langle p_0 ? e_0 : e_1,\sigma\rangle \Downarrow_{exp} \mathrm{n}_0}\ \text{?True}$$

$$\frac{\langle p_0,\sigma\rangle \Downarrow_{exp} \textbf{false} \quad \langle e_1,\sigma\rangle \Downarrow_{exp} n_1}{\langle p_0 ? e_0 : e_1,\sigma\rangle \Downarrow_{exp} n_1}\ \text{?False}$$

2.2. Semántica de comandos

La semántica operacional de LIS se describe en términos de:
$\Gamma = \text{comm} \times \Sigma$, el conjunto de todas las configuraciones,

$\rightsquigarrow$ la relación de transición de Γ a Γ,

$\rightsquigarrow^*$ la clausura transitiva de, donde a $\rightsquigarrow^*$ b si existe una ejecución finita que comienza en a y termina en b.

Notar que toda ejecución que termina lo hace en $\langle skip, \sigma\rangle$ para algún estado σ.

Se utilizan reglas de inferencia para describir la relación de transición, utilizando la semántica operacional de la sección anterior para las expresiones. Una ejecución es válida si y sólo si puede probarse como consecuencia de las siguientes reglas de inferencia:

$$\frac{\langle e,\sigma\rangle \Downarrow_{exp} n}{\langle v \ = \ e \, , \, \sigma\rangle \rightsquigarrow \langle \textbf{skip}, [\sigma' | v : n]\rangle} \text{ Ass}$$

$$\frac{}{\langle \textbf{skip}; \, c_1, \, \sigma\rangle \rightsquigarrow \langle c_1, \sigma\rangle} \text{ Seq1}$$

$$\frac{\langle c_0, \sigma_0\rangle \rightsquigarrow \langle c_1, \sigma_1\rangle}{\langle c_0; \, c_2, \, \sigma_0\rangle \rightsquigarrow \langle c_1; \, c_2, \, \sigma_1\rangle} \text{ Seq2}$$

$$\frac{\langle b,\sigma\rangle \Downarrow_{exp} true}{\langle \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2, \, \sigma\rangle \rightsquigarrow \langle c_0, \, \sigma\rangle} \text{ If1}$$

$$\frac{\langle b,\sigma\rangle \Downarrow_{exp} false}{\langle \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2, \, \sigma\rangle \rightsquigarrow \langle c_1, \, \sigma\rangle} \text{ If2}$$

$$\frac{\langle b,\sigma\rangle \Downarrow_{exp} false}{\langle \textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2, \, \sigma\rangle \rightsquigarrow \langle c_1, \, \sigma\rangle} \text{ If2}$$

$$\frac{\langle b,\sigma\rangle \Downarrow_{exp} false}{\langle \textbf{while } b \textbf{ do } c, \, \sigma\rangle \rightsquigarrow \langle c; \textbf{while } b \textbf{ do } c, \, \sigma\rangle} \text{ While1}$$

$$\frac{\langle b,\sigma\rangle \Downarrow_{exp} false}{\langle \textbf{while } b \textbf{ do } c, \, \sigma\rangle \rightsquigarrow \langle \textbf{skip}, \, \sigma\rangle} \text{ While2}$$