

# Universidad Nacional de Rosario

FACULTAD DE CIENCIAS EXACTAS, INGENIERÍA Y  
AGRIMENSURA



Análisis de Lenguajes de Programación

R-322

INFORME TRABAJO PRÁCTICO I

Alumnos

Cavigioli, Axel

Rassi, Octavio

Sferco, Martín

17 de Septiembre, 2024

# 1. Ejercicio 1.

## 1.1. Enunciado.

Extienda las sintaxis abstracta y concreta de **LIS** para incluir a los operadores  $++$  y  $--$  que construyen una expresión entera a partir de una variable. La expresión  $v++$  ( $v--$ ) tiene como valor el valor de  $v$  más (menos) uno, y además tiene como efecto secundario actualizar el valor de  $v$  luego de sumarle (restarle) uno.

Estas extensiones permiten simplificar la escritura de algunos patrones frecuentes, por ejemplo:

- La secuencia de comandos  $x = x + 1; y = x$  puede re-escribirse como  $y = x++$ .
- La secuencia de comandos  $x = x - 1; \text{if } x < 0 \{y = 1\}$  puede re-escribirse como  $\text{if } x-- < 0 \{y = 1\}$

## 1.2. Resolución.

### 1.2.1. Extensión de la sintaxis abstracta.

$$\begin{aligned} \text{intexp} ::= & \text{nat} \mid \text{var} \mid -_u \text{intexp} \\ & \mid \text{var} ++ \mid \text{var} -- \\ & \mid \text{intexp} + \text{intexp} \\ & \mid \text{intexp} -_b \text{intexp} \\ & \mid \text{intexp} \times \text{intexp} \\ & \mid \text{intexp} \div \text{intexp} \end{aligned}$$
$$\begin{aligned} \text{boolexp} ::= & \text{true} \mid \text{false} \\ & \mid \text{intexp} == \text{intexp} \\ & \mid \text{intexp} \neq \text{intexp} \\ & \mid \text{intexp} < \text{intexp} \\ & \mid \text{intexp} > \text{intexp} \\ & \mid \text{boolexp} \wedge \text{boolexp} \\ & \mid \text{boolexp} \vee \text{boolexp} \\ & \mid \neg \text{boolexp} \end{aligned}$$
$$\begin{aligned} \text{comm} ::= & \text{skip} \\ & \mid \text{var} = \text{intexp} \\ & \mid \text{comm}; \text{comm} \\ & \mid \text{if } \text{boolexp} \text{ then } \text{comm} \text{ else } \text{comm} \\ & \mid \text{repeat } \text{comm} \text{ until } \text{boolexp} \end{aligned}$$

### 1.2.2. Extensión de la sintaxis concreta.

```
digit ::= '0' | '1' | ... | '9'

letter ::= 'a' | ... | 'z' | 'A' | ... | 'Z'

nat ::= digit | digit nat

var ::= letter | letter var

intexp ::= nat
         | var
         | var '++'
         | var '--'
         | '-' intexp
         | intexp '+' intexp
         | intexp '-' intexp
         | intexp '*' intexp
         | intexp '/' intexp
         | '(' intexp ')'

boolexp ::= 'true' | 'false'
         | intexp '==' intexp
         | intexp '!=' intexp
         | intexp '<' intexp
         | intexp '>' intexp
         | boolexp '&&' boolexp
         | boolexp '||' boolexp
         | '!' boolexp
         | '(' boolexp ')'

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

## 2. Ejercicio 4.

### 2.1. Enunciado.

Modifique la semántica *big-step* de expresiones enteras para incluir a los operadores  $++$  y  $--$  descritos en el Ejercicio 2.2.

Dado que las expresiones podrán modificar el entorno de variables, utilizar la notación  $[f \mid x : e]$  para denotar la función  $f'$ , tal que  $\text{dom } f' = \text{dom } f \cup \{x\}$  y

$$f'(y) = \begin{cases} e & \text{si } y = x \\ f(y) & \text{si } y \neq x \end{cases}$$

### 2.2. Resolución.

Extenderemos la semántica *big-step* para incluir las siguientes reglas

$$\frac{x \in \text{dom } \sigma}{\langle x++; \sigma \rangle \Downarrow_{\text{exp}} \langle \sigma x + 1, [\sigma \mid x : \sigma x + 1] \rangle} \text{VARINC}$$

$$\frac{x \in \text{dom } \sigma}{\langle x--; \sigma \rangle \Downarrow_{\text{exp}} \langle \sigma x - 1, [\sigma \mid x : \sigma x - 1] \rangle} \text{VARDEC}$$

### 3. Ejercicio 5.

#### 3.1. Enunciado.

¿Es la relación de evaluación de un paso  $\rightsquigarrow$  determinista? Si lo es, demostralo (puede suponer que la relación  $\Downarrow_{\text{exp}}$  es determinista). Caso contrario, proveer un contraejemplo.

#### 3.2. Resolución.

Demostraremos que, efectivamente, la evaluación de un paso  $\rightsquigarrow$  es determinista. La idea general de la prueba se repite a lo largo de los distintos casos, y consiste en partir de  $(t, t') \in \rightsquigarrow$  y analizar la última regla que se aplicó sobre  $t \rightsquigarrow t'$  para luego probar que, si elegimos un  $t''$  cualquiera y suponemos que  $t \rightsquigarrow t''$ , entonces la regla que se utilizó en esta última derivación ha de ser la misma. Finalmente, con la información que obtenemos de estas dos proposiciones, demostramos que  $t'$  y  $t''$  son iguales, probando entonces que

$$\forall t'' \in \Gamma \quad t \rightsquigarrow t' \wedge t \rightsquigarrow t'' \implies t' = t''$$

**Demostración.** Veamos que la relación de evaluación en un paso  $\rightsquigarrow$  es determinista.

Es decir, queremos probar que si  $t \rightsquigarrow t'$  y  $t \rightsquigarrow t''$ , entonces  $t' = t''$ .

Lo probaremos por inducción sobre la relación  $\rightsquigarrow$ . Queremos demostrar que la propiedad

$$P(t, t') : \forall t'' \in \Gamma \quad t \rightsquigarrow t'' \implies t' = t''$$

vale para todo par  $(t, t') \in \rightsquigarrow$ .

Separemos entonces por casos sobre el último paso de derivación en  $t \rightsquigarrow t'$ .

**REGLA ASS.** Sea  $t'' \in \Gamma$  cualquiera y supongamos que  $t \rightsquigarrow t'$  se construyó a partir de ASS. Tenemos entonces que

- $t$  es de la forma  $\langle v = e, \sigma \rangle$ .
- $t'$  es de la forma  $\langle \mathbf{skip}, [\sigma' \mid v : n] \rangle$ .
- Vale  $\langle e, \sigma \rangle \Downarrow_{\text{exp}} \langle n, \sigma' \rangle$ .

Notemos que como  $t$  es una asignación, la única regla que es posible aplicar es ASS. Luego, si suponemos que  $t \rightsquigarrow t''$ , necesariamente se habrá aplicado la misma regla. Nuevamente tenemos entonces que

- $t$  es de la forma  $\langle v_1 = e_1, \sigma_1 \rangle$ .
- $t''$  es de la forma  $\langle \mathbf{skip}, [\sigma'_1 \mid v_1 : n_1] \rangle$ .
- Vale  $\langle e_1, \sigma_1 \rangle \Downarrow_{\text{exp}} \langle n_1, \sigma'_1 \rangle$ .

Luego, sigue de las dos definiciones de  $t$  que  $v = v_1$ ,  $e = e_1$ ,  $\sigma = \sigma_1$ . Ahora, como  $\Downarrow_{\text{exp}}$  es determinista y  $e = e_1, \sigma = \sigma_1$ , tenemos que como  $\langle e, \sigma \rangle \Downarrow_{\text{exp}} \langle n, \sigma' \rangle$  y  $\langle e_1, \sigma_1 \rangle \Downarrow_{\text{exp}} \langle n_1, \sigma'_1 \rangle$ , deberá valer que  $n = n_1$  y  $\sigma' = \sigma'_1$ .

Finalmente, de las igualdades anteriores sigue que  $\langle \mathbf{skip}, [\sigma' \mid v : n] \rangle = \langle \mathbf{skip}, [\sigma'_1 \mid v_1 : n_1] \rangle$ . Es decir,  $t' = t''$ .

**REGLA SEQ<sub>1</sub>.** Sea  $t''$  cualquiera y supongamos que  $t \rightsquigarrow t'$  se construyó a partir de SEQ<sub>1</sub>. Tenemos entonces que

- $t$  es de la forma  $\langle \mathbf{skip}; c_1, \sigma \rangle$
- $t'$  es de la forma  $\langle c_1, \sigma \rangle$

Supongamos ahora que  $t \rightsquigarrow t''$ , luego por la forma de  $t$  la última regla con la que se construyó el par necesariamente es SEQ<sub>1</sub>. Esto implica que

- $t$  es de la forma  $\langle \mathbf{skip}; c'_1, \sigma' \rangle$
- $t''$  es de la forma  $\langle c'_1, \sigma' \rangle$

Nuevamente, sigue de comparar ambas definiciones de  $t$  que  $c_1 = c'_1$  y  $\sigma = \sigma'$ . Luego, es claro que  $t' = t''$ .

**REGLA SEQ<sub>2</sub>.** Sea  $t''$  cualquiera y supongamos que  $t \rightsquigarrow t'$  se construyó a partir de SEQ<sub>2</sub>. Tenemos entonces que

- $t$  es de la forma  $\langle c_0; c_1, \sigma \rangle$
- $t'$  es de la forma  $\langle c'_0; c_1, \sigma' \rangle$
- Vale  $\langle c_0, \sigma \rangle \rightsquigarrow \langle c'_0, \sigma' \rangle$

Notemos que  $c_0$  no puede tratarse de un **skip**, pues sabemos que  $\langle c_0, \sigma \rangle \rightsquigarrow \langle c'_0, \sigma' \rangle$  y no existe una regla tal que  $\langle \mathbf{skip}, \sigma \rangle \rightsquigarrow \langle c'_0, \sigma' \rangle$ . Entonces, al quedar descartada esa opción, si suponemos que  $t \rightsquigarrow t''$ , la única regla que puede haber construido al par  $(t, t'')$  es nuevamente SEQ<sub>2</sub>.

Ahora, tenemos que

- $t$  es de la forma  $\langle c_2; c_3, \sigma_1 \rangle$
- $t''$  es de la forma  $\langle c'_2; c_3, \sigma'_1 \rangle$
- Vale  $\langle c_2, \sigma_1 \rangle \rightsquigarrow \langle c'_2, \sigma'_1 \rangle$

Primero observemos que, comparando las definiciones de  $t$ , es claro que

$$c_0 = c_2, \quad c_1 = c_3, \quad \sigma = \sigma_1$$

Pero ademias, como  $\langle c_0, \sigma \rangle \rightsquigarrow \langle c'_0, \sigma' \rangle$  es una derivación previa, podemos aplicar la hipótesis inductiva sobre ella. Esto es, vale que

$$\forall t''' \in \Gamma \quad \langle c_0, \sigma \rangle \rightsquigarrow t''' \implies \langle c'_0, \sigma' \rangle = t'''$$

Siguiendo esta idea, como vimos que vale  $\langle c_2, \sigma_1 \rangle \rightsquigarrow \langle c'_2, \sigma'_1 \rangle$  y que  $c_0 = c_2$  y  $\sigma = \sigma_1$ , tenemos que, para  $t''' = \langle c'_2, \sigma'_1 \rangle$ ,

$$\langle c_0, \sigma \rangle \rightsquigarrow \langle c'_2, \sigma'_1 \rangle \implies \langle c'_0, \sigma' \rangle = \langle c'_2, \sigma'_1 \rangle$$

Es decir,

$$c'_0 = c'_2, \quad \sigma' = \sigma'_1$$

Finalmente, juntando todas las igualdades anteriores, sigue que

$$t' = \langle c'_0; c_1, \sigma' \rangle = \langle c'_2; c_3, \sigma'_1 \rangle = t''$$

como queríamos probar.

**REGLA IF<sub>1</sub>.** Sea  $t''$  cualquiera y supongamos que  $t \rightsquigarrow t'$  se construyó a partir de IF<sub>1</sub>. Tenemos entonces que

- $t$  es de la forma  $\langle \text{if } b \text{ then } c_0 \text{ else } c_1, \sigma \rangle$
- $t'$  es de la forma  $\langle c_0, \sigma' \rangle$
- Vale  $\langle b, \sigma \rangle \Downarrow_{\text{exp}} \langle \text{true}, \sigma' \rangle$

Supongamos ahora que  $t \rightsquigarrow t''$ . Observemos que, basándonos en la forma de  $t$ , existiría la posibilidad de que la última regla utilizada para construir  $t \rightsquigarrow t''$  sea tanto IF<sub>1</sub> como IF<sub>2</sub>.

Sin embargo, si la regla utilizada fuera IF<sub>2</sub>, debería valer que  $\langle b, \sigma \rangle \Downarrow_{\text{exp}} \langle \text{false}, \sigma'_1 \rangle$ , lo que no es posible pues  $\langle b, \sigma \rangle \Downarrow_{\text{exp}} \langle \text{true}, \sigma' \rangle$  y  $\Downarrow_{\text{exp}}$  es determinista. Luego, nuevamente la última regla aplicada en ambos pasos de derivación es la misma, IF<sub>1</sub>. Tenemos entonces que

- $t$  es de la forma  $\langle \text{if } b_1 \text{ then } c_2 \text{ else } c_3, \sigma_1 \rangle$
- $t''$  es de la forma  $\langle c_2, \sigma'_1 \rangle$
- Vale  $\langle b_1, \sigma_1 \rangle \Downarrow_{\text{exp}} \langle \text{true}, \sigma'_1 \rangle$

Sigue que

$$b = b_1, \quad c_0 = c_2, \quad c_1 = c_3, \quad \sigma = \sigma_1$$

Y por el determinismo de  $\Downarrow_{\text{exp}}$ , utilizando las igualdades anteriores, vale  $\sigma' = \sigma'_1$ .

De todo lo anterior podemos concluir que  $\langle c_0, \sigma' \rangle = \langle c_2, \sigma'_1 \rangle$ , es decir,  $t' = t''$ .

**REGLA IF<sub>2</sub>.** Es análoga a la regla IF<sub>2</sub>.

**REGLA REPEAT.** Sea  $t''$  cualquiera y supongamos que  $t \rightsquigarrow t'$  se construyó a partir de REPEAT. Tenemos entonces que

- $t$  es de la forma  $\langle \text{repeat } c \text{ until } b, \sigma \rangle$
- $t'$  es de la forma  $\langle c; \text{if } b \text{ then skip else repeat } c \text{ until } b, \sigma \rangle$

Si suponemos ahora que  $t \rightsquigarrow t''$ , por el argumento usual tenemos que la última regla aplicada en esta derivación ha de ser REPEAT. Le sigue que

- $t$  es de la forma  $\langle \text{repeat } c_1 \text{ until } b_1, \sigma_1 \rangle$
- $t''$  es de la forma  $\langle c_1; \text{if } b_1 \text{ then skip else repeat } c_1 \text{ until } b_1, \sigma_1 \rangle$

donde de la comparación entre las definiciones de  $t$  es claro que

$$c = c_1 \quad b = b_1 \quad \sigma = \sigma_1$$

luego  $t'$  y  $t''$  son iguales.



## 4. Ejercicio 6.

### 4.1. Enunciado.

Pruebe que los siguientes programas son semánticamente equivalentes:

- a)  $x = x + 1;$   
 $y = x$
- b)  $y = x++$

### 4.2. Resolución.

Evaluaremos ambos programas para demostrar que dado un estado inicial  $\sigma \in \Gamma$  cualquiera, ambos llegan a la configuración final  $\langle \mathbf{skip}, [[\sigma \mid x : \sigma x + 1] \mid y : \sigma x + 1] \rangle$ .

Consideremos entonces  $\sigma \in \Sigma$  un estado tal que  $x \in \text{dom } \sigma$ , pues de otra manera no podría aplicar VAR en las derivaciones. Evaluemos los programas.

#### 4.2.1. Programa A.

Podemos escribir al programa A como:

$$\langle x = x + 1; y = x, \sigma \rangle.$$

Probemos primero que  $\langle x = x + 1; y = x, \sigma \rangle \rightsquigarrow \langle \mathbf{skip}; y = x, [\sigma \mid x : \sigma x + 1] \rangle$

$$\frac{\frac{\frac{x \in \text{dom } \sigma}{\langle x, \sigma \rangle \Downarrow_{\text{exp}} \langle \sigma x, \sigma \rangle} \text{VAR} \quad \frac{\langle 1, \sigma \rangle \Downarrow_{\text{exp}} \langle 1, \sigma \rangle}{\text{NVAL}}}{\langle x + 1, \sigma \rangle \Downarrow_{\text{exp}} \langle \sigma x + 1, \sigma \rangle} \text{PLUS} \quad \frac{\langle x + 1, \sigma \rangle \Downarrow_{\text{exp}} \langle \sigma x + 1, \sigma \rangle}{\langle x = x + 1, \sigma \rangle \rightsquigarrow \langle \mathbf{skip}, [\sigma \mid x : \sigma x + 1] \rangle} \text{ASS}}{\langle x = x + 1; y = x, \sigma \rangle \rightsquigarrow \langle \mathbf{skip}; y = x, [\sigma \mid x : \sigma x + 1] \rangle} \text{SEQ}_2$$

Luego,  $\langle \mathbf{skip}; y = x, [\sigma \mid x : \sigma x + 1] \rangle \rightsquigarrow \langle y = x, [\sigma \mid x : \sigma x + 1] \rangle$

$$\frac{\langle \mathbf{skip}; y = x, [\sigma \mid x : \sigma x + 1] \rangle \rightsquigarrow \langle y = x, [\sigma \mid x : \sigma x + 1] \rangle}{\text{SEQ}_1}$$

Y por último,  $\langle y = x, [\sigma | x : \sigma x + 1] \rangle \rightsquigarrow \langle \mathbf{skip}, [[\sigma | x : \sigma x + 1] | y : \sigma x + 1] \rangle$

Observemos que por definición de  $[\sigma | x : \sigma x + 1]$  y como  $x \in \text{dom } \sigma$ , vale  $x \in \text{dom } [\sigma | x : \sigma x + 1]$ .

$$\frac{\frac{x \in \text{dom } [\sigma | x : \sigma x + 1]}{\langle x, [\sigma | x : \sigma x + 1] \rangle \Downarrow_{\text{exp}} \langle \sigma x + 1, [\sigma | x : \sigma x + 1] \rangle} \text{VAR}}{\langle y = x, [\sigma | x : \sigma x + 1] \rangle \rightsquigarrow \langle \mathbf{skip}, [[\sigma | x : \sigma x + 1] | y : \sigma x + 1] \rangle} \text{ASS}$$

(Obs. Aplicamos  $[\sigma | x : \sigma x + 1] x = \sigma x + 1$ )

#### 4.2.2. Programa B.

Probaremos que  $\langle y = x++, \sigma \rangle \rightsquigarrow \langle \mathbf{skip}, [[\sigma | x : \sigma x + 1] | y : \sigma x + 1] \rangle$

$$\frac{\frac{x \in \text{dom } \sigma}{\langle x++, \sigma \rangle \Downarrow_{\text{exp}} \langle \sigma x + 1, [\sigma | x : \sigma x + 1] \rangle} \text{VARINC}}{\langle y = x++, \sigma \rangle \rightsquigarrow \langle \mathbf{skip}, [[\sigma | x : \sigma x + 1] | y : \sigma x + 1] \rangle} \text{ASS}$$