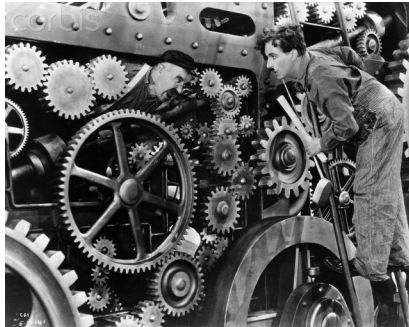


Más sobre Semántica Operacional

Análisis de Lenguajes de Programación

Mauro Jaskelioff 30/08/2019



Semántica de un lenguaje imperativo

- ▶ ¿Cómo expresar la semántica operacional de un lenguaje con efectos laterales?
 - ▶ estado
 - ▶ errores
 - ▶ entrada/salida
- ▶ Lo veremos con un lenguaje imperativo simple.

Sintaxis

Los expresiones enteras son:

$$\begin{array}{l} \textit{intexp} ::= \textit{nv} \\ \quad | \textit{var} \\ \quad | -_u \textit{intexp} \\ \quad | \textit{intexp} + \textit{intexp} \\ \quad | \textit{intexp} -_b \textit{intexp} \\ \quad | \textit{intexp} \times \textit{intexp} \\ \quad | \textit{intexp} \div \textit{intexp} \end{array}$$

- ▶ Los valores numéricos \textit{nv} son los enteros.
- ▶ El valor de una expresión entera depende de un estado global (por \textit{var}).

Sintaxis

Los expresiones booleanas son:

$$\begin{aligned} \text{boolexp} ::= & \text{true} \mid \text{false} \\ & \mid \text{intexp} = \text{intexp} \\ & \mid \text{intexp} < \text{intexp} \\ & \mid \text{intexp} > \text{intexp} \\ & \mid \text{boolexp} \ \&\& \ \text{boolexp} \\ & \mid \text{boolexp} \ || \ \text{boolexp} \\ & \mid \text{not} \ \text{boolexp} \end{aligned}$$

- ▶ Los valores booleanos *bv* son true y false.
- ▶ El valor de una expresión booleana depende de un estado global (por contener expresiones enteras).

Sintaxis

Los comandos son:

$$\begin{array}{l} comm ::= \text{skip} \\ \quad | \quad var := intexp \\ \quad | \quad comm; comm \\ \quad | \quad \text{if } boolexp \text{ then } comm \text{ else } comm \\ \quad | \quad \text{repeat } comm \text{ } boolexp \end{array}$$

- ▶ Los comandos leen y modifican el estado global.
- ▶ Los comandos no tienen valores en sí mismo, excepto por su efecto sobre el estado.

Representando efectos laterales en la evaluación

- ▶ Este lenguaje tiene efectos laterales
 - ▶ Modificación del estado global
- ▶ Modelamos el estado global Σ como una función de identificadores en valores enteros.

$$\Sigma = var \rightarrow nv$$

- ▶ Las relaciones de evaluación deben tener en cuenta el estado global:

$$\Downarrow_{int} \subseteq (intexp \times \Sigma) \times nv$$

$$\Downarrow_{bool} \subseteq (boolexp \times \Sigma) \times bv$$

$$\rightsquigarrow \subseteq (comm \times \Sigma) \times (comm \times \Sigma)$$

Reglas de expresiones enteras

- ▶ La relación de evaluación de expresiones enteras

$$\Downarrow_{int} \subseteq (intexp \times \Sigma) \times nv$$

- ▶ La regla para valores

$$\frac{}{(v, \sigma) \Downarrow_{int} \sigma \ v}$$

- ▶ La extensión de otras reglas al uso de estado es simple.

$$\frac{e_0 \Downarrow_{int} n_0 \quad e_1 \Downarrow_{int} n_1}{e_0 + e_1 \Downarrow_{int} n_0 + n_1} \Rightarrow \frac{(e_0, \sigma) \Downarrow_{int} n_0 \quad (e_1, \sigma) \Downarrow_{int} n_1}{(e_0 + e_1, \sigma) \Downarrow_{int} n_0 + n_1}$$

Reglas de expresiones booleanas

- La relación de evaluación para expresiones booleanas

$$\Downarrow_{bool} \subseteq (boolexp \times \Sigma) \times bv$$

- La extensión es similar al caso de las expresiones enteras

$$\frac{(e_0, \sigma) \Downarrow_{int} n_0 \quad (e_1, \sigma) \Downarrow_{int} n_1}{(e_0 = e_1, \sigma) \Downarrow_{bool} n_0 = n_1}$$

$$\frac{(p, \sigma) \Downarrow_{bool} b}{(\text{not } p, \sigma) \Downarrow_{bool} \neg b}$$

Reglas para comandos (1)

- La relación de evaluación (de paso chico) para comandos

$$\rightsquigarrow \subseteq (comm \times \Sigma) \times (comm \times \Sigma)$$

- Asignación

$$\frac{(e, \sigma) \Downarrow_{int} n}{(v := e, \sigma) \rightsquigarrow (\text{skip}, [\sigma \mid v : n])}$$

- Toda ejecución que termina lo hace en (skip, σ) .
- Condicionales

$$\frac{(e, \sigma) \Downarrow_{bool} \text{true}}{(\text{if } e \text{ then } c_0 \text{ else } c_1, \sigma) \rightsquigarrow (c_0, \sigma)}$$

$$\frac{(e, \sigma) \Downarrow_{bool} \text{false}}{(\text{if } e \text{ then } c_0 \text{ else } c_1, \sigma) \rightsquigarrow (c_1, \sigma)}$$

Reglas para comandos (2)

► Secuenciación

$$\frac{(c_0, \sigma) \rightsquigarrow (c'_0, \sigma')}{(c_0; c_1, \sigma) \rightsquigarrow (c'_0; c_1, \sigma')} \qquad \frac{}{(\text{skip}; c_1, \sigma) \rightsquigarrow (c_1, \sigma)}$$

► Repeat

$$\begin{array}{l} (\text{repeat } c \text{ until } e, \sigma) \rightsquigarrow \\ (c; \text{if } e \text{ then skip else repeat } c \text{ until } e, \sigma) \end{array}$$

Considerando errores

- ▶ Ahora las expresiones enteras pueden fallar con un error.
- ▶ Agregamos un valor adicional **err**_{*i*}.
- ▶ La relación de evaluación de expresiones enteras es ahora

$$\Downarrow_{int} \subseteq (intexp \times \Sigma) \times (nv \cup \{\mathbf{err}_i\})$$

- ▶ Podemos agregar la regla

$$\frac{(e_1, \sigma) \Downarrow_{int} 0}{(e_0 \div e_1, \sigma) \Downarrow_{int} \mathbf{err}_i}$$

- ▶ También deberemos agregar reglas propagando el error

Propagando errores (1)

- ▶ Ejemplos de reglas adicionales para expresiones enteras

$$\frac{(e_0, \sigma) \Downarrow_{int} \mathbf{err}_i}{(e_0 + e_1, \sigma) \Downarrow_{int} \mathbf{err}_i} \qquad \frac{(e_0, \sigma) \Downarrow_{int} n \quad (e_1, \sigma) \Downarrow_{int} \mathbf{err}_i}{(e_0 + e_1, \sigma) \Downarrow_{int} \mathbf{err}_i}$$

- ▶ La relación de evaluación para expr. booleanas es ahora

$$\Downarrow_{bool} \subseteq (boolexp \times \Sigma) \times (bv \cup \{\mathbf{err}_b\})$$

- ▶ Ejemplos de reglas adicionales para expresiones booleanas

$$\frac{(e_0, \sigma) \Downarrow_{int} \mathbf{err}_i}{(e_0 = e_1, \sigma) \Downarrow_{bool} \mathbf{err}_b} \qquad \frac{(p, \sigma) \Downarrow_{bool} \mathbf{err}_b}{(\text{not } p, \sigma) \Downarrow_{bool} \mathbf{err}_b}$$

Propagando errores (2)

- La relación de evaluación de comandos es ahora

$$\rightsquigarrow \subseteq (comm \times \Sigma) \times ((comm \cup \{\mathbf{err}_c\}) \times \Sigma)$$

- Ejemplo de reglas adicionales para comandos

$$\frac{(e, s) \Downarrow_{int} \mathbf{err}_i}{(v := e, \sigma) \rightsquigarrow (\mathbf{err}_c, \sigma)} \qquad \frac{(c_0, \sigma) \rightsquigarrow \mathbf{err}_c}{(c_0; c_1, \sigma) \rightsquigarrow (\mathbf{err}_c, \sigma)}$$

Agregando Entrada/Salida

- ▶ Ahora queremos manejar entrada/salida.
- ▶ Extendemos la sintaxis de comandos:

$$\begin{array}{l} comm ::= \dots \\ \quad | \text{ input } var \\ \quad | \text{ print } var \end{array}$$

- ▶ `input v` lee un entero (por ejemplo del teclado) y asigna ese valor a la variable `v`.
- ▶ `print v` imprime (por ejemplo en pantalla) el valor de `v`.

Semántica de entrada/salida (1)

- Indicamos la entra/salida **etiquetando las transiciones**.
- La etiqueta $n?$ indica una entrada n

$$\frac{}{(\text{input } v, \sigma) \xrightarrow{n?} (\text{skip}, [\sigma \mid v : n])}$$

- La etiqueta $n!$ indica una salida n .

$$\frac{(e, \sigma) \Downarrow_{int} n}{(\text{print } e, \sigma) \xrightarrow{n!} (\text{skip}, \sigma)}$$

Semántica de entrada/salida (2)

- La relación de evaluación de comandos es ahora

$$\rightsquigarrow \subseteq (comm \times \Sigma) \times L \times ((comm \cup \{\mathbf{err}_c\}) \times \Sigma)$$

donde las etiquetas L son:

$$L ::= nv? \mid nv! \mid \tau$$

- La etiqueta τ representa una transición **silenciosa**.
 - Usualmente escribimos $x \rightsquigarrow y$ en lugar de $x \xrightarrow{\tau} y$.
- Debemos reescribir las reglas:

$$\frac{(c_0, \sigma) \xrightarrow{\alpha} (c'_0, \sigma')}{(c_0; c_1, \sigma) \rightsquigarrow (c'_0; c_1, \sigma')} \qquad \frac{(c_0, \sigma) \xrightarrow{\alpha} \mathbf{err}_c}{(c_0; c_1, \sigma) \rightsquigarrow (\mathbf{err}_c, \sigma)}$$

- ▶ Al agregar efectos laterales las relaciones de evaluación se van haciendo más complejas.
- ▶ Al agregar un efecto, además de nuevas reglas deben describirse las reglas anteriores.
- ▶ La especificación del lenguaje sigue siendo intuitiva.

Referencias:

- ▶ Theories of Programming languages . John Reynolds.
Capítulo 6