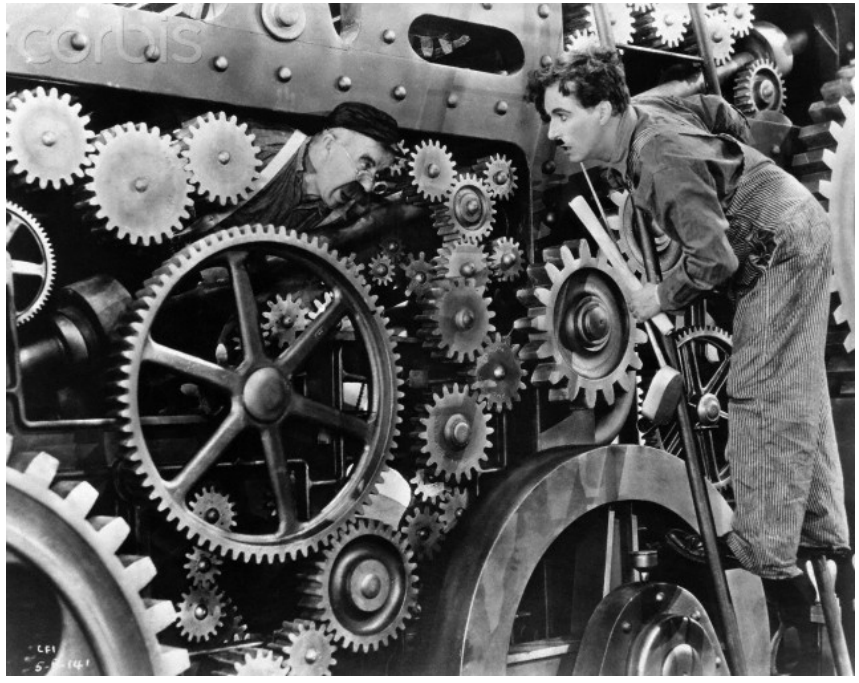
Más sobre Semántica Operacional

Análisis de Lenguajes de Programación

Mauro Jaskelioff 01/09/2017



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:

$$
\begin{aligned}
intexp ::= \; & nat \\
\mid \; & var \\
\mid \; & -_u \; intexp \\
\mid \; & intexp \; + intexp \\
\mid \; & intexp \; -_b \; intexp \\
\mid \; & intexp \; \times \; intexp \\
\mid \; & intexp \; \div \; intexp
\end{aligned}
$$

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

Sintaxis

Los expresiones booleanas son:

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

- 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:

$$
\begin{aligned}
\mathit{comm} ::= {} & \texttt{skip} \\
\mid {} & \mathit{var} := \mathit{intexp} \\
\mid {} & \mathit{comm}; \mathit{comm} \\
\mid {} & \texttt{if}\ \ \mathit{boolexp}\ \ \texttt{then}\ \ \mathit{comm}\ \ \texttt{else}\ \ \mathit{comm} \\
\mid {} & \texttt{repeat}\ \ \mathit{comm}\ \ \mathit{boolexp}
\end{aligned}
$$

- 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.

$$\Sigma = var \to nv$$

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

$$\begin{array}{lcl} \Downarrow_{int} & \subseteq & (intexp \times \Sigma) \times nv \\ \Downarrow_{bool} & \subseteq & (boolexp \times \Sigma) \times bv \\ \rightsquigarrow & \subseteq & (comm \times \Sigma) \times (comm \times \Sigma) \end{array}$$

Reglas de expresiones enteras

- La relación de evaluación de expresiones enteras

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

- La regla para valores

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

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

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

Reglas de expresiones booleanas

- La relación de evaluación para expresiones booleanas

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

- La extensión es similar al caso de las expresiones enteras

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

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

Reglas para comandos (1)

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

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

- Asignación

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

- Toda ejecución que termina lo hace en $(\texttt{skip}, \sigma)$.
- Condicionales

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

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

Reglas para comandos (2)

- Secuenciación

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

- Repeat

$$(\texttt{repeat}\ c\ \texttt{until}\ e, \sigma) \rightsquigarrow (c; \texttt{if}\ e\ \texttt{then skip else repeat}\ c\ \texttt{until}\ e, \sigma)$$

Considerando errores

- Ahora las expresiones enteras pueden fallar con un error.
- Agregamos un valor adicional $\mathbf{err}_i$.
- La relación de evaluación de expresiones enteras es ahora

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

- Podemos agregar la regla

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

- Tambien 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

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

- Ejemplos de reglas adicionales para expresiones booleanas

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

Propagando errores (2)

- La relación de evaluación de comandos es ahora

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

- 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:

$$\begin{aligned} comm ::= & \ \ldots \\ & \mid \ \texttt{input}\ var \\ & \mid \ \texttt{print}\ var \end{aligned}$$

- `input` v lee un entero (por ejemplo del teclado) y asigna ese valor a la variable v.
- `print` v imprime (or 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

$$\frac{}{(\texttt{input}\ v, \sigma) \overset{?n}{\rightsquigarrow} (\texttt{skip}, [\sigma \mid v : n])}$$

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

$$\frac{(e, \sigma) \Downarrow_{int} n}{(\texttt{print}\ e, \sigma) \overset{!n}{\rightsquigarrow} (\texttt{skip}, \sigma)}$$

Semántica de entrada/salida (2)

- La relación de evaluación de comandos es ahora

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

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 $x \overset{\tau}{\rightsquigarrow} y$.
- Debemos reescribir las reglas:

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

Resumen

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

Referencias:

- Theories of Programming languages . John Reynolds. Capítulo 6