Processamento de Linguagens
Semântica das linguagens de programação

João Reis

Universidade da Beira Interior

2018/2019

(re)lease

REliable And SEcure Computation Group

Significado

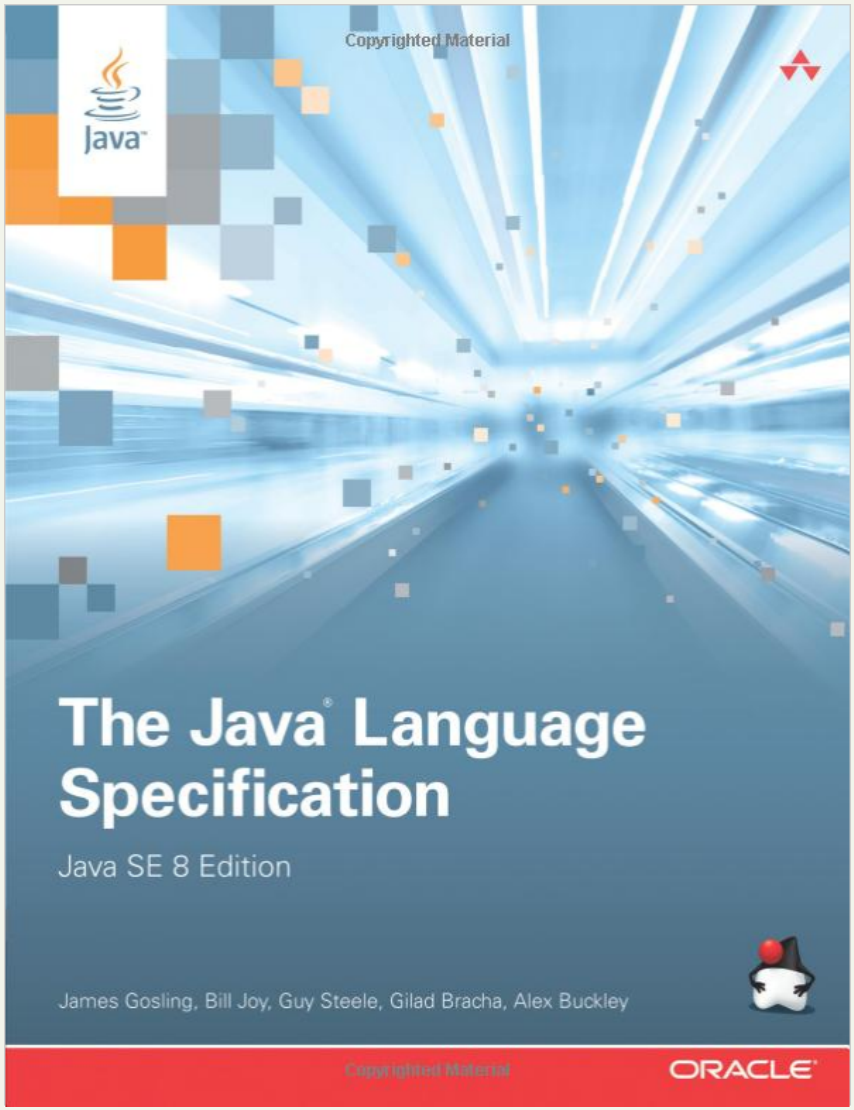
Como podemos definir o significado de um programa escrito numa determinada linguagem?

Na maioria das vezes, aceitamos uma descrição informal, em linguagem natural (norma ISO, *standard*, manual de referência, etc.).

É algo pouco satisfatório na verdade, porque muitas vezes é impreciso e até mesmo ambíguo.

Semântica informal

Exemplo



The Java programming language guarantees that the operands of operators appear to be evaluated in a specific evaluation order, namely, from left to right.
It is recommended that code not rely crucially on this specification.

Semântica formal

A **semântica formal** carateriza matematicamente os cálculos descritos num programa.

Útil para o desenvolvimento do ferramentas (interpretadores, compiladores, etc.).

Indispensável para raciocinar sobre programas.

Uma outra questão

Mas o que é um programa?

Como objeto sintático (conjunto de caracteres), é muito difícil de trabalhar.

É preferível usar a **sintaxe abstrata**.

Semântica

É sobre a sintaxe abstrata que definimos a semântica.

Existem diversas abordagens:

- semântica axiomática
- semântica denotacional
- semântica por tradução
- semântica operacional

Semântica axiomática

Também conhecida como **lógica de Hoare**
Carateriza os programas por intermédio de propriedades cumpridas pelas variáveis; introduzimos o triplo

$$\{P\}\ i\ \{Q\}$$

que significa "se a fórmula P é verdadeira previamente à execução da instrução i, então a fórmula Q será verdadeira após a execução desta instrução".

Exemplo:

$$\{x \geq 0\}\ x := x + 1\ \{x > 0\}$$

Exemplo de regra:

$$\{P[x \leftarrow E]\}\ x := E\ \{P(x)\}$$

Semântica denotacional

A **semântica denotacional** associa a cada expressão e a denotação $[\![e]\!]$, que é um objeto matemático que representa o cálculo nomeado por e.

Exemplo: expressões aritméticas com uma só variável x

$$e ::= x \mid n \mid e + e \mid e * e \mid \ldots$$

A denotação pode ser uma função que associa ao valor de x o valor da expressão:

$$\begin{aligned}
[\![x]\!] &= x \mapsto x \\
[\![n]\!] &= x \mapsto n \\
[\![e_1 + e_2]\!] &= x \mapsto [\![e_1]\!](x) + [\![e_2]\!](x) \\
[\![e_1 * e_2]\!] &= x \mapsto [\![e_1]\!](x) \times [\![e_2]\!](x)
\end{aligned}$$

Semântica por tradução

(Também conhecida como semântica denotacional *à la Strachey*).

Podemos definir a semântica de uma linguagem traduzindo-a para uma outra cuja semântica já é conhecida.

Semântica operacional

A **semântica operacional** descreve o sequência de cálculos elementares que levam uma expressão ao seu resultado (ao seu valor).

Esta opera diretamente sobre os objetos sintáticos (sintaxe abstrata).

Duas formas de semântica operacional:

- "semântica natural" (ou "*big-steps semantics*")

$$e \twoheadrightarrow v$$

- "semântica por reduções" (ou "*small-steps semantics*")

$$e \to e1 \to e2 \to \cdots \to v$$

Linguagem WHILE

Definimos a semântica operacional sobre esta linguagem minimal.

e	$::=$		**expressão**
	$\mid$	c	constante
	$\mid$	x	variável
	$\mid$	$e\ op\ e$	operação binária ($+$, $<$, ...)

c	$::=$		**constante**
	$\mid$	n	constante inteira (-17, 42, ...)
	$\mid$	b	constante booleana (`true`, `false`)

Linguagem WHILE

s	$::=$		**instrução**
	$\mid$	$x \leftarrow e$	atribuição
	$\mid$	`if` e `then` s `else` s	condicional
	$\mid$	`while` e `do` s	ciclo
	$\mid$	s; s	sequência
	$\mid$	`skip`	não fazer nada

Exemplo

$$
\begin{array}{l}
a \leftarrow 0; \\
b \leftarrow 1; \\
\texttt{while } b < 100 \texttt{ do} \\
\quad b \leftarrow a + b; \\
\quad a \leftarrow b - a
\end{array}
$$

Semântica operacional *big-steps* do `WHILE`

Procuramos definir uma relação entre uma expressão e e um **valor** v

$$e \twoheadrightarrow v$$

Os valores são aqui reduzidos aos inteiros e aos booleanos

$$\begin{array}{rcll} v & ::= & & \textbf{valor} \\ & & \mid\ n & \text{valor inteiro} \\ & & \mid\ b & \text{valor booleano} \end{array}$$

(de uma maneira geral, os valores não coincidem necessariamente com as constantes)

Valor de uma variável

O valor de uma variável é dado por um **ambiente** E (uma função de variáveis para valores).

Definimos uma relação na forma

$$E, e \twoheadrightarrow v$$

que se lê como "num ambiente E, a expressão e tem o valor v"

Exemplo

No ambiente

$$E = \{a \mapsto 34, b \mapsto 55\}$$

a expressão

$$a + b$$

tem o valor

$$89$$

que denotamos como

$$E, a + b \twoheadrightarrow 89$$

Regras de inferência

Uma relação poderá definir-se como a **menor relação** que satisfaz um conjunto de regras sem premissas (axiomas) na forma

$$\frac{}{P}$$

e um conjunto de regras com premissas na forma

$$\frac{P_1 \;\; P_2 \;\; \cdots \;\; P_n}{P}$$

Estas são chamadas **regras de inferência**.

Regras de inferência

Exemplo: podemos definir a relação Even(n) com as duas regras

$$\frac{}{\text{Even}(0)} \quad e \quad \frac{\text{Even}(n)}{\text{Even}(n+2)}$$

que se devem ler como

em primeiro lugar	$\text{Even}(0)$
em segundo lugar	$\forall n.\ \text{Even}(n) \Rightarrow \text{Even}(n+2)$

A menor relação que satisfaz estas duas propriedades coincide com a propriedade "n é um inteiro par".

Árvore de derivação

Uma **derivação** é uma árvore cujos nós correspondem às regras com premissas e as folhas correspondem aos axiomas; exemplo:

$$\dfrac{\dfrac{\dfrac{}{\mathsf{Even}(0)}}{\mathsf{Even}(2)}}{\mathsf{Even}(4)}$$

O conjunto de derivações possíveis caraterizam exatamente a menor relação que satisfaz as regras de inferência.

Semântica das expressões

A relação $E, e \twoheadrightarrow v$ é definida pelas regras de inferência seguintes:

$$\frac{}{E, n \twoheadrightarrow n} \qquad \frac{}{E, b \twoheadrightarrow b}$$

$$\frac{}{E, x \twoheadrightarrow E(x)}$$

$$\frac{E, e_1 \twoheadrightarrow n_1 \quad E, e_2 \twoheadrightarrow n_2 \quad n = n_1 + n_2}{E, e_1 + e_2 \twoheadrightarrow n} \quad \text{etc.}$$

Semântica das instruções

Uma instrução pode modificar o valor de certas variáveis (atribuição).

Para darmos a semântica de uma instrução s, introduzimos a relação

$$E, s \twoheadrightarrow E'$$

que se lê "no ambiente E, a avaliação da instrução s termina e leva ao ambiente E'"

Semântica das instruções

$$\frac{}{E, \texttt{skip} \twoheadrightarrow E} \qquad \frac{E, s_1 \twoheadrightarrow E_1 \quad E_1, s_2 \twoheadrightarrow E_2}{E, s_1; s_2 \twoheadrightarrow E_2}$$

$$\frac{E, e \twoheadrightarrow v}{E, x \leftarrow e \twoheadrightarrow E\{x \mapsto v\}}$$

$$\frac{E, e \twoheadrightarrow \texttt{true} \quad E, s_1 \twoheadrightarrow E_1}{E, \ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2 \twoheadrightarrow E_1} \qquad \frac{E, e \twoheadrightarrow \texttt{false} \quad E, s_2 \twoheadrightarrow E_2}{E, \ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2 \twoheadrightarrow E_2}$$

$$\frac{E, e \twoheadrightarrow \texttt{true} \quad E, s \twoheadrightarrow E_1 \quad E_1, \texttt{while}\ e\ \texttt{do}\ s \twoheadrightarrow E_2}{E, \ \texttt{while}\ e\ \texttt{do}\ s \twoheadrightarrow E_2}$$

$$\frac{E, e \twoheadrightarrow \texttt{false}}{E, \ \texttt{while}\ e\ \texttt{do}\ s \twoheadrightarrow E}$$

Exemplo de derivação

Consideremos o ambiente $E = \{a \mapsto 21\}$

$$\dfrac{\dfrac{E, a \twoheadrightarrow 21 \quad E, 0 \twoheadrightarrow 0}{E, a > 0 \twoheadrightarrow \texttt{true}} \quad \dfrac{\dfrac{E, 2 \twoheadrightarrow 2 \quad E, a \twoheadrightarrow 21}{E, 2 \times a \twoheadrightarrow 42}}{E, a \leftarrow 2 \times a \twoheadrightarrow \{a \mapsto 42\}}}{E, \ \texttt{if}\ a > 0\ \texttt{then}\ a \leftarrow 2 \times a\ \texttt{else}\ \texttt{skip} \twoheadrightarrow \{a \mapsto 42\}}$$

Expressões sem valor

Existem expressões e para as quais não existe um valor v tal que $E, e \twoheadrightarrow v$
Exemplo: $1 + \texttt{true}$

Da mesma forma, existem instruções s para as quais não existe uma avaliação $E, s \twoheadrightarrow E'$
Exemplo: $\texttt{while}\ 0 = 0\ \texttt{do skip}$

Recorrência na derivação

Para se estabelecer uma propriedade de uma relação definida por um conjunto de regras de inferência, podemos raciocinar por **indução estrutural** na derivação *i.e.* podemos aplicar a hipótese de indução a qualquer sub-derivação.

De maneira equivalente, podemos dizer que raciocinamos por indução sobre a altura da derivação.

Exemplo

Proposição (determinismo da avaliação)

Se $E, e \twoheadrightarrow v$ e $E, e \twoheadrightarrow v'$ então $v = v'$.

Por recorrência sobre as derivações de $E, e \twoheadrightarrow v$ e de $E, e \twoheadrightarrow v'$
Caso de uma adição $e = e_1 + e_2$

$$\frac{\begin{array}{cc}(D_1) & (D_2) \\ \vdots & \vdots \\ E, e_1 \twoheadrightarrow n_1 & E, e_2 \twoheadrightarrow n_2\end{array}}{E, e_1 + e_2 \twoheadrightarrow v} \qquad \frac{\begin{array}{cc}(D_1') & (D_2') \\ \vdots & \vdots \\ E, e_1 \twoheadrightarrow n_1' & E, e_2 \twoheadrightarrow n_2'\end{array}}{E, e_1 + e_2 \twoheadrightarrow v'}$$

Com $v = n_1 + n_2$ e $v' = n_1' + n_2'$
Por HI temos que $n_1 = n_1'$ e $n_2 = n_2'$, então $v = v'$

(Os outros casos são idênticos ou também muito simples).

Exemplo

Proposição (determinismo da avaliação)

Se $E, s \twoheadrightarrow E'$ e $E, s \twoheadrightarrow E''$ então $E' = E''$.

Exercício: fazer a prova.
No caso da regra

$$\frac{E, e \twoheadrightarrow \texttt{true} \quad E, s \twoheadrightarrow E_1 \quad E_1, \texttt{while } e \texttt{ do } s \twoheadrightarrow E_2}{E, \texttt{ while } e \texttt{ do } s \twoheadrightarrow E_2}$$

vemos que a recorrência é feita sobre da derivação e não sobre da instrução (que não diminui).

Determinismo

Nota: a relação de avaliação não é necessariamente determinista.

Exemplo: juntemos uma primitiva `random` para tirar ao azar um inteiro 0 ou 1 e, portanto, a regra

$$\frac{0 \leq n < 2}{E,\ \texttt{random} \twoheadrightarrow n}$$

Temos então que $E,\ \texttt{random} \twoheadrightarrow 0$ bem como $E,\ \texttt{random} \twoheadrightarrow 1$.

Defeitos da semântica natural

A semântica natural não permite distinguir os programas cujos cálculos falhem, como

$$1 + \texttt{true}$$

assim como programas cuja avaliação não termine, como

```
while true do skip
```

Semântica operacional *small-steps*

A semântica operacional **small-steps** ultrapassa os limites mencionados anteriormente introduzindo uma noção de etapa elementar de cálculo $E_1, s_1 \to E_2, s_2$, que vamos iterar.

Podemos então distinguir três situações

1. a iteração termina (leva a um valor)

$$E, s \to E_1, s_1 \to E_2, s_2 \to \cdots \to E', \texttt{skip}$$

2. a iteração bloqueia sobre E_n, s_n (irredutível)

$$E, s \to E_1, s_1 \to E_2, s_2 \to \cdots E_n, s_n$$

3. a iteração não termina

$$E, s \to E_1, s_1 \to E_2, s_2 \to \cdots$$

Semântica operacional *small-steps* para WHILE

$$\frac{E, e \twoheadrightarrow v}{E, x \leftarrow e \rightarrow E\{x \mapsto v\}, \texttt{skip}}$$

$$\frac{}{E, \texttt{skip}; s \rightarrow E, s} \qquad \frac{E, s_1 \rightarrow E_1, s_1'}{E, s_1; s_2 \rightarrow E_1, s_1'; s_2}$$

$$\frac{E, e \twoheadrightarrow \texttt{true}}{E, \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2 \rightarrow E, s_1} \qquad \frac{E, e \twoheadrightarrow \texttt{false}}{E, \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2 \rightarrow E, s_2}$$

$$\frac{E, e \twoheadrightarrow \texttt{true}}{E, \texttt{while } e \texttt{ do } s \rightarrow E, s; \texttt{while } e \texttt{ do } s}$$

$$\frac{E, e \twoheadrightarrow \texttt{false}}{E, \texttt{while } e \texttt{ do } s \rightarrow E, \texttt{skip}}$$

Nota

Mantemos a semântica *big-steps* para as expressões visto que estas sempre terminam.

Equivalência

Proposição (equivalência de duas semânticas)

As duas semânticas operacionais são equivalentes para os programas cuja avaliação termina, *i.e.*

$$E, s \twoheadrightarrow E' \quad \text{se e apenas se} \quad E, s \rightarrow^* E', \texttt{skip}$$

($\rightarrow^*$ é o fecho reflexivo transitivo de $\rightarrow$)

Prova

Proposição (*big steps* implica *small steps*)

Se $E, s \twoheadrightarrow E'$, então $E, s \rightarrow^* E', \texttt{skip}$.

Por recorrência sobre a derivação $E, s \twoheadrightarrow E'$ e por caso sobre a última regra utilizada

- caso de $s_1; s_2$

$$\frac{E, s_1 \twoheadrightarrow E_1 \quad E_1; s_2 \twoheadrightarrow E_2}{E; s_1; s2 \twoheadrightarrow E_2}$$

Prova

Proposição (*big steps* implica *small steps*)

Se $E, s \twoheadrightarrow E'$, então $E, s \rightarrow^* E', \texttt{skip}$.

Por recorrência sobre a derivação $E, s \twoheadrightarrow E'$ e por caso sobre a última regra utilizada

- caso de $s_1; s_2$

$$\frac{E, s_1 \twoheadrightarrow E_1 \quad E_1; s_2 \twoheadrightarrow E_2}{E; s_1; s2 \twoheadrightarrow E_2}$$

então $E, s_1 \rightarrow^* E_1, \texttt{skip}$ por HI
portanto,

$$\begin{array}{lll} E, s_1; s_2 & \rightarrow^* & E_1, \texttt{skip}; s_2 \quad (\textit{small steps} \text{ para } ;) \\ & \rightarrow & E_1, s_2 \\ & \rightarrow^* & E_2, \texttt{skip} \quad (\textit{HI}) \end{array}$$

s

Prova

- caso de `while` e `do` s
  se

$$\frac{E, e \twoheadrightarrow \texttt{true} \quad E, s \twoheadrightarrow E_1 \quad E_1, \texttt{while } e \texttt{ do } s \twoheadrightarrow E_2}{E, \texttt{while } e \texttt{ do } s \twoheadrightarrow E_2}$$

Prova

- caso de `while` e `do` s
  se

$$\frac{E, e \twoheadrightarrow \texttt{true} \quad E, s \twoheadrightarrow E_1 \quad E_1, \texttt{while } e \texttt{ do } s \twoheadrightarrow E_2}{E, \texttt{while } e \texttt{ do } s \twoheadrightarrow E_2}$$

  então

$$\begin{array}{lll} E, \texttt{while } e \texttt{ do } s & \rightarrow & E, s; \texttt{while } e \texttt{ do } s \\ & \rightarrow^* & E_1, \texttt{skip}; \texttt{while } e \texttt{ do } s \quad (\text{HI} + \text{regra ;}) \\ & \rightarrow & E_1, \texttt{while } e \texttt{ do } s \\ & \rightarrow^* & E_2, \texttt{skip} \quad (\text{HI}) \end{array}$$

Exercício: tratar os outros casos

O outro significado

Lema

Se $E_1, s_1 \to E_2, s_2 \twoheadrightarrow E'$, então $E_1, s_1 \twoheadrightarrow E'$.

Por recorrência sobre a derivação $\twoheadrightarrow$

- caso $s_1 = u_1; v_1$
  - caso $u_1 = skip$
    Temos $E_1, \texttt{skip}; v_1 \to E_1, v_1 \twoheadrightarrow v_1 \twoheadrightarrow E'$ e assim
    $$\frac{E_1, \texttt{skip} \twoheadrightarrow E_1 \quad E_1, v_1 \twoheadrightarrow E'}{E_1, \texttt{skip}; v_1 \twoheadrightarrow E'}$$
  - caso $u_1 \neq \texttt{skip}$
    Temos $E_1, u_1; v_1 \to E_2, u_2; v_1 \twoheadrightarrow E'$, isto é, $E_1, u_1 \to E_2, u_2$ e
    $$\frac{E_2, u_2 \twoheadrightarrow E_2', v_1 \twoheadrightarrow E'}{E_2, u_2; v_1 \twoheadrightarrow E'}$$
    Por HI deduzimos
    $$\frac{E_1, u_1 \twoheadrightarrow E_2' \quad E_2', v_1 \twoheadrightarrow E'}{E_1, u_1; v_1 \twoheadrightarrow E'}$$

(Tratar os outros casos)

O outro significado

Deduzimos

Proposição (*small steps* implica *big steps*)

Se $E, s \to^* E', \texttt{skip}$, então $E, s \twoheadrightarrow E'$.

Prova: temos

$$E, s \to E_1, s_1 \to E_2, s_2 \to \cdots \to E_n, s_n \to E', \texttt{skip}$$

Sabemos que $E', \texttt{skip} \twoheadrightarrow E'$ logo $E_s, s_n \twoheadrightarrow$ pelo lema precedente.
De forma igual, $E_{n-1}, s_{n-1} \twoheadrightarrow E'$ pelo lema precedente, e por aí em diante, até que atingimos a forma $E, s \twoheadrightarrow E'$ (recorrência sobre o nome das etapas).