Semantica Operazionale Strutturata

Semantica per riscrittura o riduzione o semplificazione basata sulla struttura dei termini che voglio semplificare

Si individuano dei termini (del linguaggio) che rappresentano direttamente dei valori, cioè che non possono essere ulteriormente semplificati.

Si danno delle regole di semplificazione in modo tale che ogni termine corretto si riduca ad uno e uno solo di quei termini che rappresentano valori

In realtà come nel caso denotazionale abbiamo bisogno di ambienti e stati per poter eseguire le semplificazioni.
Quindi la semplificazione agisce su configurazioni, che consistono di un termine e del contesto in cui si vuole valutare il termine.

SOS a grandi passi per LW

Molto simile alla semantica denotazionale

Gli stessi della denotazionale tranne che cambia Value

Come configurazioni usiamo

$$Conf = L \times Env \times States$$

$$L = \bigcup_{s \in N} L_s(LW) \cup \{\tau\} \cup \{*\}$$

$[\![T]\!]$ è l'insieme dei valori sintattici di tipo T: $[\![\mathbf{int}]\!] = \{0,1,2,..-1,-2....\}$, $[\![\mathbf{bool}]\!] = \{true,false\}$

Definiamo la semplificazione come relazione binaria sulle configurazioni:

$$\to \;\subseteq Conf \times Conf$$

Siccome la maggior parte delle semplificazioni non modifica gli ambienti introduciamo una notazione semplificata:

$$(t,s) \to_r (t',s') \text{ sse } (t,r,s) \to (t',r,s')$$

Una configurazione (t,r,s) è finale se

$t = \tau$ (ho rielaborato tutto il termine in esame usandolo per modificare ambiente e/o stato) o

$t \in Value^+$ (ho raggiunto un termine che rappresenta un valore)

A grandi passi vuol dire che se derivo $(t,r,s) \to (t',r,s')$, allora (t',r,s') è finale (non vado mai in uno stato intermedio, ma direttamente alla fine)

Espressioni

Denotazionale | **SOS a grandi passi**

$$\frac{[\![e]\!](r,s) = (v,s') \qquad [\![es]\!](r,s') = (lv,s'')}{[\![e\ es]\!](r,s) = (v\ lv,s'')}$$

$$\frac{(e,s) \longrightarrow_r (v,s') \qquad (es,s') \longrightarrow_r (lv,s'')}{(e\ es,\ s) \longrightarrow_r (v\ lv,s'')}$$

$$r(x) \in Loc_T \quad \frac{}{[\![x]\!](r,s) = (s(r(x)),s)}$$

$$r(x) \in Loc_T \quad \frac{}{(x,s) \longrightarrow_r (s(r(x)),s)}$$

$$\frac{[\![e]\!](r,s) = (v\ ,s')}{[\![x = e]\!](r,s) = (v,s'[v/r(x)])}$$

$$\frac{(e,s) \longrightarrow_r (v,s')}{(x{=}e,s) \longrightarrow_r (v,s'[v/r(x)]))}$$

$$\frac{[\![es]\!](r,s) = (lv\ ,s')}{[\![f(es)]\!](r,s) = r(f)(lv,s')}$$

$$\frac{(es,s) \longrightarrow_r (lv,s')}{(f(es),s) \longrightarrow_r r(f)(lv,s')}$$

a è la rappresentazione sintattica del valore ottenuto sommando i valori rappresentati da v e v’

$$\frac{[\![e]\!](r,s) = (v,s') \quad [\![e']\!](r,s') = (v',s'')}{[\![e + e']\!](r,s) = (a,s'')}\ v + v' = a$$

$$\frac{(e,s) \longrightarrow_r (v,s') \quad (e',s') \longrightarrow_r (v',s'')}{(e{+}e',s) \longrightarrow_r (a,s'')}\ \mathbb{v} + \mathbb{v}' = \mathbb{a}$$

Espressioni completamento

In realtà bisogna anche aggiungere tutti i controlli di tipo

Alcuni controlli si possono omettere in quanto derivano dalla grammatica CF

$$\frac{(e,s) \rightarrow_r (v,s') \qquad (es,s') \rightarrow_r (lv,s'')}{(e\ es,\ s) \rightarrow_r (v\ lv,s'')} \quad e \in L_{Exp}(LW),\ \boxed{v \in Value},\ es \in L_{Exps}(LW),\ \boxed{lv \in Value^{+}}$$

$$\frac{r(x) \in Loc_T \qquad x \in L_{Id}(LW)}{(x,s) \rightarrow_r (s(r(x)),s)}$$

Altri esprimono proprio il fatto che stiamo dando una semantica a grandi passi

$$\frac{(e,s) \rightarrow_r (v,s')}{(x=e,s) \rightarrow_r (v,s'[v/r(x)]))} \quad \cancel{e \in L_{Exp}(LW)},\ \boxed{v \in Value},\ \cancel{x \in L_{Id}(LW)}$$

$$\frac{(es,s) \rightarrow_r (lv,s')}{(f(es),s) \rightarrow_r r(f)(lv,s')} \quad \cancel{f \in L_{Id}(LW)},\ \cancel{es \in L_{Exps}(LW)},\ \boxed{lv \in Value^{+}}$$

Dichiarazioni di variabili

Denotazionale

$$\frac{[\![d]\!]\,(r,s) = (r',s') \qquad [\![ds]\!]\,(r',s') = (r'',s'')}{[\![d\ ds\ (r,s)]\!] = (r'',s'')}$$

SOS a grandi passi

$$\begin{array}{l} d \in L_{Dec}(LW) \\ ds \in L_{Decs}(LW) \end{array}\ \frac{(d,r,s) \longrightarrow (\square,r',s') \quad (ds,r',s') \longrightarrow (\square,r'',s'')}{(d\ ds,r,s) \longrightarrow (\square,r'',s'')}$$

$$\text{Nuova}(l,r,s)\ \frac{}{[\![t\ x]\!](r,s) = (r[l/x],s[\square/l])}$$

$$\begin{array}{l} \text{Nuova}(l,r,s) \\ x \in L_{Id}(LW) \\ t \in L_{Type}(LW) \end{array}\ \frac{}{(t\ x,r,s) \longrightarrow (\square, r[l/x],s[\square/l])}$$

$$\text{Nuova}(l,r,s')\ \frac{[\![e]\!](r,s) = (v\ ,s')}{[\![t\ x = e]\!](r,s) = (r[l/x],s'[v/l])}$$

$$\begin{array}{l} \text{Nuova}(l,r,s') \\ e \in L_{Exp}(LW) \\ x \in L_{Id}(LW) \\ t \in L_{Type}(LW) \end{array}\ \frac{(e,s) \longrightarrow_{r} (v,s')}{(t\ x = e,r,s) \longrightarrow (\square, r[l/x],s'[v/l])}$$

Dichiarazioni di funzioni ricorsive: void

$$\frac{}{[\![\text{void } f(T\ x)\ \{sts\}]\!](r,s) = (r[F/f],s)} \qquad \frac{}{(\text{void } f(T\ x)\ \{sts\},r,s) \ \square\ (\square, r[F/f],s)}$$

Dove $F : [\![T]\!] \ \square\ States \ \square\ \{*\} \ \square\ States$
è definita induttivamente da tutte le regole della semantica più le seguenti due:

$$\frac{[\![sts]\!]\,(r[\square/f,l/x],s_c[v/l]) = s'}{F(v,s_c) = (*,s'-\{l\})} \quad Nuova(l,r,s_c) \quad \frac{(sts,\ s_c[v/l]) \ \square_{\ r[\square/f,l/x]}\ (\square,\ s')}{F(v,s_c) = (*,s'-\{l\})}$$

$$\frac{[\![e]\!](r[\square/f],s_1) = (u\ ,s') \qquad F(u,s') = (*,s'')}{[\![f(e)]\!]\,(r[\square/f],s_1) = (*,s'')} \qquad \frac{(e,\ s_1) \ \square_{\ r[\square/f]}\ (u,\ s') \qquad F(u,s') = (*,s'')}{(f(e),\ s_1) \ \square_{\ r[\square/f]}\ (*,s'')}$$

Dichiarazioni di funzioni ricorsive: BType

$$\frac{}{[\![\text{RT f(T x) \{sts } \textbf{result} \text{ e\}}]\!](r,s) = (r[F/f],s)} \qquad \frac{}{(\text{RT f(T x) \{sts } \textbf{result} \text{ e\}},r,s) \ \square\ (\square, r[F/f],s)}$$

Dove $F : [\![T]\!] \ \square\ \text{States} \ \square\ [\![RT]\!] \ \square\ \text{States}$

è definita induttivamente da tutte le regole della semantica più le seguenti due:

$$\frac{[\![sts]\!]\,(r[\square/f,l/x],s_c[v/l]) = s' \quad [\![e]\!](r[\square/f,l/x], s') = (a\ ,s'')}{F(v,s_c) = (a,s''-\{l\})} \ \text{Nuova}(l,r,s_c)$$

$$\frac{(sts,\ s_c[v/l]) \ \square_{\ r[\square/f,l/x]}\ (\square,\ s') \quad (e,\ s') \ \square_{\ r[\square/f,l/x]}\ (a,\ s'')}{F(v,s_c) = (a,s''-\{l\})} \ \text{Nuova}(l,r,s_c)$$

$$\frac{[\![e]\!](r[\square/f],s_1) = (u\ ,s') \quad F(u,s') = (a\ ,s'')}{[\![f(e)]\!]\,(r[\square/f],s_1) = (a\ ,s'')} \qquad \frac{(e,\ s_1) \ \square_{\ r[\square/f]}\ (u,\ s') \quad F(u,s') = (a,s'')}{(f(e),\ s_1) \ \square_{\ r[\square/f]}\ (a,\ s'')}$$

Statements 1

$$\frac{[\![st]\!](r,s) = s' \qquad [\![sts]\!](r,s') = s''}{[\![st\ sts]\!](r,s) = s''} \qquad st \square L_{Stat}(LW),\ sts \square L_{stats}(LW)\ \frac{(st, s) \square_r (\square, s')(sts, s') \square_r (\square, s'')}{(st\ sts, s) \square_r (\square, s'')}$$

$$\frac{[\![e]\!](r,s) = (v, s')}{[\![e;]\!](r,s) = s'} \qquad e \square L_{Exp}(LW)\ \frac{(e, s) \square_r (v, s')}{(e;, s) \square_r (\square, s')}$$

$$\frac{[\![e]\!](r,s) = (tt, s') \qquad [\![sts]\!](r,s') = s''}{[\![\textbf{if}\ (e)\ \{sts\}\ \textbf{else}\ \{sts'\}]\!](r,s) = s''} \qquad \frac{(e, s) \square_r (true, s') \quad (sts, s') \square_r (\square, s'')}{(\textbf{if}\ (e)\ \{sts\}\ \textbf{else}\ \{sts'\}, s) \square_r (\square, s'')}$$

$$\frac{[\![e]\!](r,s) = (ff, s') \qquad [\![sts']\!](r,s') = s''}{[\![\textbf{if}\ (e)\ \{sts\}\ \textbf{else}\ \{sts'\}]\!](r,s) = s''} \qquad \frac{(e, s) \square_r (false, s') \quad (sts', s') \square_r (\square, s'')}{(\textbf{if}\ (e)\ \{sts\}\ \textbf{else}\ \{sts'\}, s) \square_r (\square, s'')}$$

Statements 2: while

$$\frac{[\![e]\!](r,s) = (ff, s')}{[\![\textbf{while}\ (e)\ \{sts\}]\!](r,s) = s'}$$

$$\frac{(e, s) \square_r (false, s')}{(\textbf{while}\ (e)\ \{sts\}, s) \square_r (\square, s')}$$

$$\frac{[\![e]\!](r,s) = (tt, s') \quad [\![sts]\!](r,s') = s'' \quad [\![\textbf{while}\ (e)\ \{sts\}]\!](r,s'') = s_0}{[\![\textbf{while}\ (e)\ \{sts\}]\!](r,s) = s_0}$$

$$\frac{(e, s) \square_r (true, s') \quad (sts, s') \square_r (\square, s'') \quad (\textbf{while}\ (e)\ \{sts\}, s'') \square_r (\square, s_0)}{(\textbf{while}\ (e)\ \{sts\}, s) \square_r (\square, s_0)}$$

Problema dei valori

Consideriamo la semantica dell'espressione 3 + 2.
Per valutarne la semantica possiamo solo usare la regola della somma

$$\frac{\dfrac{?\quad ?\quad ?}{(3,s) \,\square_{r}\, (v,s')} \qquad \dfrac{?\quad ?\quad ?}{(2,s') \,\square_{r}\, (v',s'')}}{(3+2,s) \,\square_{r}\, (5,s'')} \quad v + v' = 5$$

Non ci sono regole per completare alberi di questa forma.

In realtà in tutte le regole (che ce l'hanno) la valutazione di un'espressione dovrebbe essere opzionale, perché se l'espressione è già un valore non va ulteriormente elaborata.

Quindi, ad esempio, ci vorranno regole tipo

$$\frac{(e,s) \,\square_{r}\, (v,s')}{(e+v',s) \,\square_{r}\, (a,s')} \; v + v' = a \qquad \frac{(e',s) \,\square_{r}\, (v',s')}{(v+e',s) \,\square_{r}\, (a,s')} \; v + v' = a \qquad \frac{}{(v+v',s) \,\square_{r}\, (a,s)} \; v + v' = a$$

$$\frac{}{(\textbf{while}\ (\text{false})\ \{sts\}, s) \,\square_{r}\, (\square, s)}$$

$$\frac{}{(v;, s) \,\square_{r}\, (\square, s)}$$

$$\frac{}{(t\ x = v,r,s) \,\square\, (r[l/x],s[v/l])} \quad \begin{array}{l} \text{Nuova}(l,r,s') \\ e\square L_{Exp}(LW) \\ x\square L_{Id}(LW) \\ t\square L_{Type}(LW) \end{array}$$

Relazione con la semantica denotazionale

Le due semantiche coincidono, fatti i dovuti aggiustamenti tecnici, cioè adeguate le due notazioni:

$[\![d]\!]_{Decs}(r,s)=(r',s')$ se e solo se $(d,r,s) \square\ (\square, r', s')$

$[\![st]\!]_{Stats}\,(r,s)=s'$ se e solo se $(st,s) \square_{r} (\square, s')$

$[\![e]\!]_{Exps}(r,s)=(v,s')$ se e solo se $(e,s) \square_{r} (v, s')$

e modulo una traduzione fra i valori (usati nella semantica denotazionale) e le loro descrizioni sintattiche (usate nella semantica operazionale)

Non sono necessarie prove, o meglio la prova è banale, perché le due semantiche sono descritte da regole “isomorfe” (a volta partizionate in modo diverso, a seconda che le espressioni coinvolte già rappresentino valori)

Quindi i due approcci hanno gli stessi limiti.

In particolare, non danno informazioni in caso di non terminazione.

Questo può essere riduttivo in caso di sistemi che sono progettati in modo non terminante (e.g. sistemi operativi)