Validation d'algorithmes

Chapitre VIII

Vérification formelle : Logique de Floyd-Hoare

Raphaël Charrondière raphael.charrondiere@inria.fr

On se donne :

- des entiers relatifs $\mathbb{N}$ notés k
- des identificateurs de variables ident notés X, Y, Z
- des indices Ind notés i

Rappel : les expressions arithmétiques s'écrivent
$k|X|a_1 + a_2|a_1 \times a_2|\cdots$
On va maintenant rajouter des indices, c'est-à-dire des inconnues, les expressions arithmétiques s'écrivent alors
$k|X|a_1 + a_2|a_1 \times a_2|\cdots|i$
On définit alors les assertions par $\mathbb{1}|A_1 \wedge A_2|\neg A|a_1 < a_2|\exists i\ A$

- $\exists i\ (Y = X * i \wedge i \geq 1)$ traduit Y est un mutiple positif de X
- La définition peut sembler pauvre mais $\forall i\ A \equiv \neg(\exists i\ \neg A)$
- De même on peut exprimer $a_1 = a_2$, $a_1 > a_2$, $a_1 \leqslant a_2$, $a_1 \geqslant a_2$, $a_1 \neq a_2$, $A_1 \vee A_2$

- Les valuations notées V, à l'image des états mémoire, attribuent une valeur aux indices $V : \mathrm{V} \rightarrow \mathbb{N}$.
- On réutilise la notation $V[k/i] = \begin{array}{ll} j \mapsto V(j) & si\ i \neq j \\ i \mapsto k & \end{array}$
- Les règles d'évaluation arithmétiques sont étendues avec la notation $\sigma, a_1 \hookrightarrow^V a_2$, on ajoute juste la règle $\dfrac{V(i) = k}{\sigma, i \hookrightarrow^V k}$

On peut maintenant définir les règles d'inférence pour l'évaluation des assertions.

$$\frac{}{\sigma, \mathbb{1} \looparrowright^{V} \mathbb{1}}$$

$$\frac{}{\sigma, \mathbb{1} \hookrightarrow^V \mathbb{1}} \qquad \frac{\sigma, A \hookrightarrow^V v'}{\sigma, \neg A \hookrightarrow^V v}\ v' \equiv \mathbb{1} - v$$

$$\frac{}{\sigma, \mathbb{1} \hookrightarrow^V \mathbb{1}} \qquad \frac{\sigma, A \hookrightarrow^V v'}{\sigma, \neg A \hookrightarrow^V v}\ v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \hookrightarrow^V v_1 \qquad \sigma, A_2 \hookrightarrow^V v_2}{\sigma, A_1 \wedge A_2 \hookrightarrow^V v}\ v_1 \wedge v_2 \equiv v$$

$$\frac{}{\sigma, \mathbb{1} \looparrowright^{V} \mathbb{1}} \qquad \frac{\sigma, A \looparrowright^{V} v'}{\sigma, \neg A \looparrowright^{V} v}\ v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \looparrowright^{V} v_1 \qquad \sigma, A_2 \looparrowright^{V} v_2}{\sigma, A_1 \wedge A_2 \looparrowright^{V} v}\ v_1 \wedge v_2 \equiv v$$

$$\frac{\sigma, a_1 \looparrowright^{V} k_1 \qquad \sigma, a_2 \looparrowright^{V} k_2}{\sigma, a_1 < a_2 \looparrowright^{V} \mathbb{1}}\ k_1 < k_2$$

$$\frac{}{\sigma, \mathbb{1} \hookrightarrow^{V} \mathbb{1}} \qquad \frac{\sigma, A \hookrightarrow^{V} v'}{\sigma, \neg A \hookrightarrow^{V} v}\ v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \hookrightarrow^{V} v_1 \qquad \sigma, A_2 \hookrightarrow^{V} v_2}{\sigma, A_1 \wedge A_2 \hookrightarrow^{V} v}\ v_1 \wedge v_2 \equiv v$$

$$\frac{\sigma, a_1 \hookrightarrow^{V} k_1 \qquad \sigma, a_2 \hookrightarrow^{V} k_2}{\sigma, a_1 < a_2 \hookrightarrow^{V} \mathbb{1}}\ k_1 < k_2$$

$$\frac{\sigma, A \hookrightarrow^{V'} \mathbb{1}}{\sigma, \exists i\ A \hookrightarrow^{V} \mathbb{1}}\ V' = V[k/i]$$

$$\frac{}{\sigma, \mathbb{1} \hookrightarrow^{V} \mathbb{1}} \qquad \frac{\sigma, A \hookrightarrow^{V} v'}{\sigma, \neg A \hookrightarrow^{V} v}\ v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \hookrightarrow^{V} v_1 \qquad \sigma, A_2 \hookrightarrow^{V} v_2}{\sigma, A_1 \wedge A_2 \hookrightarrow^{V} v}\ v_1 \wedge v_2 \equiv v$$

$$\frac{\sigma, a_1 \hookrightarrow^{V} k_1 \qquad \sigma, a_2 \hookrightarrow^{V} k_2}{\sigma, a_1 < a_2 \hookrightarrow^{V} \mathbb{1}}\ k_1 < k_2$$

$$\frac{\sigma, A \hookrightarrow^{V'} \mathbb{1}}{\sigma, \exists i\ A \hookrightarrow^{V} \mathbb{1}}\ V' = V[k/i] \qquad \frac{\sigma, A \hookrightarrow^{V'} \mathbb{0}}{\sigma, \forall i\ A \hookrightarrow^{V} \mathbb{0}}\ V' = V[k/i]$$

$$\frac{}{\sigma, \mathbb{1} \hookrightarrow^{V} \mathbb{1}} \qquad \frac{\sigma, A \hookrightarrow^{V} v'}{\sigma, \neg A \hookrightarrow^{V} v}\; v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \hookrightarrow^{V} v_1 \qquad \sigma, A_2 \hookrightarrow^{V} v_2}{\sigma, A_1 \wedge A_2 \hookrightarrow^{V} v}\; v_1 \wedge v_2 \equiv v$$

$$\frac{\sigma, a_1 \hookrightarrow^{V} k_1 \qquad \sigma, a_2 \hookrightarrow^{V} k_2}{\sigma, a_1 < a_2 \hookrightarrow^{V} \mathbb{1}}\; k_1 < k_2$$

$$\frac{\sigma, A \hookrightarrow^{V'} \mathbb{1}}{\sigma, \exists i\ A \hookrightarrow^{V} \mathbb{1}}\; V' = V[k/i] \qquad \frac{\sigma, A \hookrightarrow^{V'} \mathbb{0}}{\sigma, \forall i\ A \hookrightarrow^{V} \mathbb{0}}\; V' = V[k/i]$$

Attention

$$\frac{}{\sigma, \mathbb{1} \looparrowright^{V} \mathbb{1}} \qquad \frac{\sigma, A \looparrowright^{V} v'}{\sigma, \neg A \looparrowright^{V} v} \; v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \looparrowright^{V} v_1 \qquad \sigma, A_2 \looparrowright^{V} v_2}{\sigma, A_1 \wedge A_2 \looparrowright^{V} v} \; v_1 \wedge v_2 \equiv v$$

$$\frac{\sigma, a_1 \looparrowright^{V} k_1 \qquad \sigma, a_2 \looparrowright^{V} k_2}{\sigma, a_1 < a_2 \looparrowright^{V} \mathbb{1}} \; k_1 < k_2$$

$$\frac{\sigma, A \looparrowright^{V'} \mathbb{1}}{\sigma, \exists i \; A \looparrowright^{V} \mathbb{1}} \; V' = V[k/i] \qquad \frac{\sigma, A \looparrowright^{V'} \mathbb{0}}{\sigma, \forall i \; A \looparrowright^{V} \mathbb{0}} \; V' = V[k/i]$$

Attention $\dfrac{\sigma, A \looparrowright^{V} \mathbb{0}}{\sigma, \exists i \; A \looparrowright^{V} \mathbb{0}}$

$$\frac{}{\sigma, \mathbb{1} \looparrowright^{V} \mathbb{1}} \qquad \frac{\sigma, A \looparrowright^{V} v'}{\sigma, \neg A \looparrowright^{V} v}\ v' \equiv \mathbb{1} - v$$

$$\frac{\sigma, A_1 \looparrowright^{V} v_1 \qquad \sigma, A_2 \looparrowright^{V} v_2}{\sigma, A_1 \wedge A_2 \looparrowright^{V} v}\ v_1 \wedge v_2 \equiv v$$

$$\frac{\sigma, a_1 \looparrowright^{V} k_1 \qquad \sigma, a_2 \looparrowright^{V} k_2}{\sigma, a_1 < a_2 \looparrowright^{V} \mathbb{1}}\ k_1 < k_2$$

$$\frac{\sigma, A \looparrowright^{V'} \mathbb{1}}{\sigma, \exists i\ A \looparrowright^{V} \mathbb{1}}\ V' = V[k/i] \qquad \frac{\sigma, A \looparrowright^{V'} \mathbb{0}}{\sigma, \forall i\ A \looparrowright^{V} \mathbb{0}}\ V' = V[k/i]$$

Attention $\quad \dfrac{\sigma, A \looparrowright^{V} \mathbb{0}}{\sigma, \exists i\ A \looparrowright^{V} \mathbb{0}} \qquad \dfrac{\sigma, A \looparrowright^{V} \mathbb{1}}{\sigma, \forall i\ A \looparrowright^{V} \mathbb{1}}$

Définition

- On note $\sigma \models^V A$ « σ satisfait A dans la valuation V » si $\sigma, A \looparrowright^V \mathbb{1}$
- Et $\sigma \models A$ si $\forall V\ \sigma, A \looparrowright^V \mathbb{1}$ « A est satisfiable »
- On dit que A est **valide** noté $\models A$ ssi $\forall\sigma\ \sigma \models A$

- Un triplet de Hoare $\{A\}c\{A'\}$ est consititué de deux prédicats $A,\ A'$ et une commande IMP c.
- Pour un état mémoire σ et une valuation V on dit que « σ satisfait $\{A\}c\{A'\}$ dans V $\sigma \models^V \{A\}c\{A'\}$ » ssi
  $\sigma \models^V A \wedge \sigma, c \Downarrow \sigma' \implies \sigma' \models^V A'$
- On dit que le triplet est **valide** $\models \{A\}c\{A'\}$ ssi
  $\forall \sigma\ \forall V\ \sigma \models^V \{A\}c\{A'\}$

On définit par induction un jugement $\vdash \{A\}c\{A'\}$

$$\frac{}{\vdash \{A\}\textbf{skip}\{A\}} \qquad \frac{\vdash \{A\}c_1\{A'\} \qquad \vdash \{A'\}c_2\{A''\}}{\vdash \{A\}c_1;\ c_2\{A''\}}$$

$$\frac{\vdash \{A \wedge b\}c_1\{A'\} \qquad \vdash \{A \wedge \neg b\}c_2\{A'\}}{\vdash \{A\}\textbf{if}\ \mathrm{b}\ \textbf{then}\ c_1\ \textbf{else}\ c_2\{A'\}}$$

$$\frac{\vdash \{A \wedge b\}c\{A\}}{\vdash \{A\}\textbf{while}\ \mathrm{b}\ \textbf{do}\ c\{A \wedge \neg b\}}$$

$$\frac{}{\vdash \{A[a/X]\}X := a\{A\}}$$ Par exemple $$\frac{}{\vdash \{X+1=3\}X := X+1\{X=3\}}$$

Règle de la conséquence :

$$\frac{\models A_1 \implies A_1' \qquad \vdash \{A_1'\}c\{A_2'\} \qquad \models A_2' \implies A_2}{\vdash \{A_1\}c\{A_2\}}$$

$$\dfrac{\dfrac{}{\vdash \{3 = 3 \wedge 3+1 = 4\} X := 3 \{X = 3 \wedge X+1 = 4\}} \qquad \dfrac{}{\vdash \{X = 3 \wedge X+1 = 4\} Y := X+1 \{X = 3 \wedge Y = 4\}}}{\vdash \{\mathbb{1}\} X := 3;\ Y := X+1 \{X = 3 \wedge Y = 4\}}$$

Note : On a masqué la règle de la conséquence, mais de manière évidente $\mathbb{1} \equiv 3 = 3 \wedge 3+1 = 4$.

Pour faciliter, on annotera directement le code :

$$\begin{array}{l}
\{3 = 3 \wedge 3+1 = 4\} \\
X := 3 \\
\{X = 3 \wedge X+1 = 4\} \\
Y := X+1 \\
\{X = 3 \wedge Y = 4\}
\end{array}$$

Exemple 2

$\{X \geqslant 0 \wedge Y \geqslant 0\} \implies \{X = 0Y + X \wedge X \geqslant 0\}$

$A := 0$

$\{X = AY + X \wedge X \geqslant 0\}$

$B := X$

$\{X = AY + B \wedge B \geqslant 0\}$

while$(B \geqslant Y)$

$\quad \{X = AY + B \wedge B \geqslant Y \wedge B \geqslant 0\} \implies \{X = (A+1)Y + B - Y \wedge B - Y \geqslant 0\}$

$\quad B := B - Y$

$\quad \{X = (A+1)Y + B \wedge B \geqslant 0\}$

$\quad A := A + 1$

$\quad \{X = AY + B \wedge B \geqslant 0\}$

$\{X = AY + B \wedge B \geqslant 0 \wedge B < Y\}$

Retour Boite blanche, trouver des DT en fonction de chemin

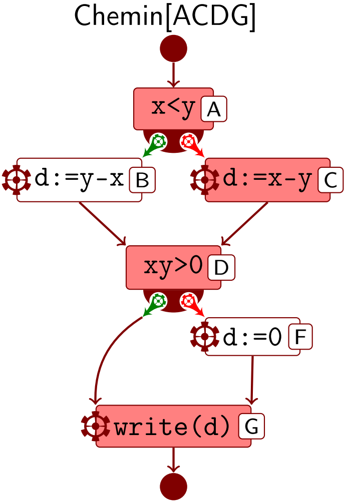
Définition

On suppose d'avoir un chemin, et on veut retrouver des données de test activant ce chemin.

Principe : On propage une condition la plus générale possible le long du chemin voulu. Soit on part de la fin et on remonte le chemin, soit on part du début.

- Au début, aucune condition n'est connue, et l'état mémoire est à définir. On notera par convention la valeur initiale d'une variable par sa majuscule ($\rho : x \mapsto X, y \mapsto Y, \cdots$)
- Cette condition sera transformée via les commandes et conditions
- On notera alors {Condition,ρ},commande,{Condition',ρ'}, et plus simplement on annotera les chemins comme pour la logique de Hoare.
- Il faudra trouver une solution à la condition finale.

- **Affectation** $\{C, \rho\}\texttt{x:=a}\{(\rho, a \looparrowright k), \rho[k/x]\}$ où k est une variable fraîche (non utilisée jusque lors).
- **Conditionnelle**, chemin vrai $\{C, \rho\}\texttt{b}\{(\rho, b \looparrowright \mathbb{1}) \wedge C, \rho\}$
- **Conditionnelle**, chemin faux $\{C, \rho\}\texttt{b}\{(\rho, b \looparrowright \mathbb{0}) \wedge C, \rho\}$
- **Écriture** On ignore ces instructions qui ne modifient pas l'état mémoire et n'influencent pas les choix de chemins
- **Lecture** ... Cas plus compliqué. Peut-on l'ajouter sans modifier le programme ?



$$\mathbb{1}, [x \mapsto X, y \mapsto Y, d \mapsto D]$$

x<y is false

$$\neg X < Y, [x \mapsto X, y \mapsto Y, d \mapsto D]$$

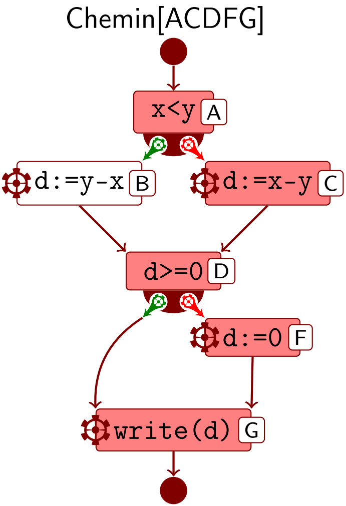
d:=x-y

$$\neg X < Y \wedge k = X - Y, [x \mapsto X, y \mapsto Y, d \mapsto k]$$

xy>0 is true

$$\neg X < Y \wedge k = X - Y \wedge XY > 0, [x \mapsto X, y \mapsto Y, d \mapsto k]$$

- Une solution est $Y = -3, X = -2$



$$\mathbb{1}, [x \mapsto X, y \mapsto Y, d \mapsto D]$$

x<y is false

$$\neg X < Y, [x \mapsto X, y \mapsto Y, d \mapsto D]$$

d:=x-y

$$\neg X < Y \wedge k = X - Y, [x \mapsto X, y \mapsto Y, d \mapsto k]$$

d>=0 is true

$$\neg X < Y \wedge k = X - Y \wedge \neg k \geqslant 0, [x \mapsto X, y \mapsto Y, d \mapsto k]$$

d:=0

$$\neg X < Y \wedge k = X - Y \wedge \neg k \geqslant 0 \wedge l = 0, [x \mapsto X, y \mapsto Y, d \mapsto l]$$

- Il n'y a pas de solution !
- L'ancienne valeur de d est conservée dans les conditions