Erinnerung: RGSep - Relation: $\bar{x}.\ A \rightsquigarrow B$

Proposition:

(1) sem_stable $(A, \forall \bar{v}.\ B \rightsquigarrow C)$ gdw. $[(B[\bar{x}/\bar{v}] \multimap A) * C[\bar{x}/\bar{v}]] \Rightarrow A$ für alle $\bar{v}$.

(2) sem_stable $(A, (R_1 \cup R_2)^*)$ gdw. sem_stable (A, R_1) und sem_stable (A, R_2).

Bemerkung:

(1) Wir entfernen von einem A-Stück einen Teil, der B erfüllt.
Den Teil ersetzen wir durch einen C-Anteil.
Das Resultat soll wieder A erfüllen.

(2) Sofern es keine B-Teile gibt, die entfernt werden können,
ist $B[\bar{x}/\bar{v}] \multimap A$ äquivalent zu False.
Dann gilt die Implikation trivialerweise.
$\Rightarrow$ Das ist das Szenario, in dem die Relation nicht ausgeführt werden kann.

(3) Eine Assertion ist stabil unter einer Menge von Relationen
gdw. sie unter jeder Relation der Menge stabil ist.

Beispiel:

Betrachte $M, N.\ x \mapsto M \rightsquigarrow x \mapsto N$.

Prüfe Stabilität von $\exists n.\ x \mapsto n$.

Es gilt:

$$\forall M, N.\ [\underbrace{(x \mapsto M \multimap \exists n.\ x \mapsto n)}_{} * x \mapsto N \Rightarrow \exists n.\ x \mapsto n]$$

$$\Leftrightarrow \forall M, N.\ [\exists n.\ \underbrace{(x \mapsto M \multimap x \mapsto n)}_{} * x \mapsto N \Rightarrow \exists n.\ x \mapsto n]$$

$$\Leftrightarrow \forall M, N.\ [\exists n.\ (emp \wedge M = n) * x \mapsto N \Rightarrow \exists n.\ x \mapsto n]$$

$\Leftrightarrow \forall N.\ [\ \underbrace{emp * x \mapsto N}\ \Rightarrow\ \exists n.\ x \mapsto n\ ]$

$\Leftrightarrow \forall N.\ [\ \underbrace{x \mapsto N\ \Rightarrow\ \exists n.\ x \mapsto n}\ ]$

$\Leftrightarrow \forall N.\ true$

$\Leftrightarrow true.$

□

Wir haben sem_stable nun über SL-Aussagen definiert.
Wir benötigen Stabilität aber für RGSep-Aussagen.

Idee: Nur der shared State kann von der Interferenz betroffen sein.
Der shared State ist durch die Boxed Aussagen gegeben.

Definition:

$stable(\boxed{A}, R) := true$

$stable(\boxed{A}, R) := sem\_stable(A, R)$

$stable(P_1 \overset{*}{\wedge} P_2, R) := stable(P_1, R) \wedge stable(P_2, R)$

$stable(\overset{\forall}{\exists} x.\ P, R) := stable(P, R).$

Beachte: Bei $*, \wedge, \vee, \exists x, \forall x$ sind wir eher streng.
Warum: Funktioniert, lässt sich automatisieren.
Aber: Damit gilt in folgendem Lemma nur eine Richtung.

Lemma:

Falls $stable(P, R)$ und $[\![P]\!](s, h_L, h_S)$ und $(s, h_S, h_S') \in R$,
dann $[\![P]\!](s, h_L, h_S')$.

Ein Gegenbeispiel für die Rückrichtung lautet wie folgt.

Beispiel:

- Wir hatten oben gesehen, dass

$$P = \boxed{\exists n.\ x \mapsto n}$$

stabil ist unter

$$R = M.N.\ x \mapsto M \rightsquigarrow x \mapsto N.$$

- Betrachte $P_1 = \boxed{\exists n.\ x \mapsto n \wedge n \leq 0}$

$$P_2 = \boxed{\exists n.\ x \mapsto n \wedge n > 0}.$$

- Es gilt $P \iff P_1 \vee P_2$.

Also obwohl $\text{stable}(P, R) = \text{true}$.

gilt:

$$\begin{aligned}
&\text{stable}(P_1 \vee P_2, R) \\
&= \text{stable}(P_1, R) \wedge \text{stable}(P_2, R) \\
&= \text{sem_stable}(\exists n.\ x \mapsto n \wedge n \leq 0, R) \\
&\quad \wedge \text{sem_stable}(\exists n.\ x \mapsto n \wedge n > 0, R) \\
&= \text{false} \wedge \text{false} \\
&= \text{false}.
\end{aligned}$$

9.5 Spezifikationen und Beweisregeln

Ziel: Definiere RGSep-Spezifikationen und Regeln der Programmlogik.

Notation: $C : (P, R, G, Q)$ — Vorcondition, RGSep-Prozess.

- Programm
- Precondition als RGSep-Prozess
- Rely als Menge von Relationen
- Guarantee auch Menge von Relationen

Semantik: Wie zu erwarten + non-abortiving.

→