Programkonstrukciók és tulajdonságaik

A cél valami olyasmi, hogy ha már a komponensekre tudjuk a tulajdonságokat, akkor ezekből könnyedén lehessen a konstrukciókra is számolni, ne kelljen mindent újra.

Unió

Adottak S_1 és S_2 programok. Legyen az uniójuk $S_1 \cup S_2$, amit így adhatunk meg:

$$S_1 \cup S_2 = (s_{1,0} \parallel s_{2,0}, S_1 \cup S_2)$$

Tehát akkezdeti értékadás a két résztvevő program kezdeti értékadásának párhuzamos futtatása, az értékadáshalmaz pedig a két értékadáshalmaz uniója.

A két program nem szükséges hogy azonos állapottéren fusson, az unió állapottere olyan, aminek altere mindkét résztvevő program állapottere, és ekkor a két programot ki is terjesztjük [az valami olyasmit jelent, hogy mindent megtartunk ami van, az új állapottér-komponeneskre meg minden utasítás skip] erre a közös állapottérre. Ami még nagyon fontos kikötés, és ez az egész uniódolog csak akkor valósítható meg, ha a két progi eredeti állapotterének legnagyobb közös alterére nézve az $s_{1,0}$ és $s_{2,0}$ kezdeti értékadások ugyanazt adják. Értelmesebben fogalmazva: ha a közös változókat nem próbáljuk két különböző értékre is inicializálni. Ha az egyik kezdeti értékadás rendel egy közös változóhoz valamit, a másik nem, az OK, meg az is OK, ha egyik se rendel semmit. Az nem OK, ha két különböző értéket.

Unió viselkedési relációja

Legyen $S := S_1 \cup S_2$. ekkor:

$$\triangleright_s = \triangleright_{S_1} \cap \triangleright_{S_2}$$

Ez azért van így, mert a dolog mélyén az lf van. Ott meg minden értékadáshalmazbeli értékadásra kell teljesülnie valaminek, az meg most a két értékadáshalmaz uniója, azaz az összes értékadás. Tehát mindre teljesülnie kell. Ezért van az "éselés", azaz a metszet. Itt amúgy felhasználjuk az lf tulajdonságai tételt is.

$$\mapsto_S = \triangleright_{S_1} \cap \triangleright_{S_2} \cap (\mapsto_{S_1} \cup \mapsto_{S_2})$$

Teljesülnie kell a háromszögnek és annak is, hogy akad az unióban olyan utasítás ami átvisz P-ből Q-ba. Nyilván akkor van ilyen ha legalább az egyikben van, ezért van ott unió jel.

$$\hookrightarrow_S = (\triangleright_{S_1} \cap \triangleright_{S_2} \cap (\mapsto_{S_1} \cup \mapsto_{S_2}))^{tdl}$$

Ez színtisztán a definíció. A zárójelben ugyanis az egyenesnyíl van. P-ből Q-ba. Nyilván akkor van ilyen ha legalább az egyikben van, ezért van ott unió jel.

$$\forall Q : sp(s_{1,0} \parallel s_{2,0}, Q) \Rightarrow sp(s_{1,0}, Q) \wedge sp(s_{2,0}, Q) \rightarrow inv_{S_1}(Q) \wedge inv_{S_2}(Q) \subseteq inv_S(Q)$$

A közös invariánsok, a közös programnak is invariánsai.

$$\varphi_S = \varphi_{S_1} \wedge \varphi_{S_2}$$

A fixpont definíciója is egy nagy éselés, innen jön ez.

$$FP_{S_1} \wedge FP_{S_2} \subseteq FP_S$$

A fixpont tulajdonság olyan ami fixpontban igaz. Az előzőben láttuk, hogy fixpont akkor van, amikor mindkettő programra igaz, hogy fixpontban van. Tehát a közös fixpont-tulajdonságok fixpont-tulajdonságai lesznek a közös programnak.

$$TERM_S = ((\triangleright_{S_1} \cap \triangleright_{S_2} \cap (\mapsto_{S_1} \cup \mapsto_{S_2}))^{tdl})^{(-1)}(\varphi_{S_1} \wedge \varphi_{S_2})$$

A zárójel belsejében az az izé a görbenyíl definíciója (azaz hogy "honnan hova tudok eljutni"), ennek az inverzét ha alkalmazom a fixpontokra, akkor megkapom azt, hogy honnan jutok fixpontba. Na, és hát pont ezek az állapotok, ahonnan terminálok. Amúgy szerintem ez minden programra igaz, nem csak az unióra, persze a görbenyíl és fixpont aktuális definíciójával.

Még néhány megjegyzés:

Ez: $\triangleright_S = \triangleright_{S_1} \cap \triangleright_{S_2}$, azt jelenti, hogy ha $P \triangleright_s Q$ igaz akkor $P \triangleright_{s_1} Q$ és $P \triangleright_{s_2} Q$ igaz, valamint ez az implikáció fordítva is teljesül.

Ha $P \mapsto_{S_1} Q$ és s$P \mapsto_{S_2} Q$ is teljesül, akkor $P \mapsto_S Q$ Fordítva nem igaz! Azért nem igaz, mert egyenesnyílhoz az unió értékadáshalamzából kell, hogy egy értékadás elvigyen P-ből Q-ba. Ha az mondjuk S_1-beli, akkor S_2-nek már nem kell ilyen értékadással rendelkeznie, tehát egyáltalán nem biztos, hogy mindkettő programra igaz az egyenes nyíl. Igaz viszont a háromszög, és az egyikre az egyenes nyíl, ezt mutatja a következő összefüggés:

$$(P \mapsto_{S_1} Q \wedge P \mapsto_{S_2} Q) \to P \mapsto_S Q$$

Nem igaz az, hogy: $(P \hookrightarrow_{S_1} Q \wedge P \hookrightarrow_{S_2} Q) \to P \hookrightarrow_S Q$, nézzünk rá egy ellenpéldát ($\hookrightarrow$ és $\rightsquigarrow$ ekvivalenciáját fogjuk használni):

$$\begin{aligned} A &= \underset{x}{\mathbb{Z}} \\ S_1 &= (SKIP, \{x := x + 1\}) \\ S_2 &= (SKIP, \{x := x - 1\}) \\ Q &= (|x| > 5) \\ P &= (|x| \leq 5) \end{aligned}$$

$P \hookrightarrow_{S_1} Q$ igaz, hiszen x egyre csak nő, előbb utóbb nagyobb lesz, mint 5. És $P \hookrightarrow_{S_2} Q$ is igaz, hiszen egyre csak csökken, előbb utóbb kisebb lesz, mint -5.

De nem igaz, hogy eljutok P-ből Q-ba S-sel, tehát nem igaz, hogy $P \rightsquigarrow_S Q$, például az $< s_1, s_2, s_1, s_2, \cdots >$ ütemezés mellett, mert ha x-et mondjuk 0-ról indítom, akkor 0 és 1 között fog váltakozni végig.

Tanulság, hogy egyenes nyilat könnyű számolni unióhoz, de görbe nyilat már csak az így megkapott egyenes nyíl tranzitív, diszjunktív lezárásával, ami meg elég melós dolog.

Szuperpozíció

Behozok új változókat úgy, hogy az eddigi működésen nem változtatok. Egyik lehetőség az, hogy egy új utasítást adok hozzá a halmazhoz, úgy hogy az ne rontsa el az eddigi tulajdonságokat, a másik pedig, hogy az egyik értékadást módosítom úgy, hogy valami új segédváltozónak is értéket adjon szimultán. Mindenesetre az eredeti program változóit maximum lekérdezhetem.

s_1 és s_2 két azonos állapottéren értelmezett értékadás szuperpozícióján az alábbit értjük, amennyiben

$$VL(s_2) \cap V(s_1) = 0 : s_1 \parallel s_2 ::= \parallel_{v_i \notin VL(s_2)} (v_i :\in F_{1_i}(v_1, \ldots, v_n),\ ha\ \pi_{1_i}) \parallel \parallel_{v_i \in VL(s_2)} (v_i :\in F_{2_i}(v_1, \ldots, v_n),\ ha\ \pi_{2_i})$$

Azaz, azokhoz az elemekhez, amikhez s_2 nem rendel semmit, azokhoz s_1 értékadását végezzük el szimultán, amelyikekhez meg igen (és ezekhez feltétel szerint s_1 nem), azokhoz meg s_2-ét. A $\parallel$ jel a baloldalon a szuperpozíciót jelöli, hisz épp most definiáljuk, a jobb oldalon pedig a szimultán értékadást. És akkor most egy párhuzamos program és egy utasítás szuperpozíciója: Legyen $S = (s_0, s_1, \ldots, s_m)$ egy program az A állapottér egy altere felett, s pedig egy feltételes értékadás A felett, úgy hogy azon változók, amelyeknek s értéket ad $(VL(s))$, nem szerepelnek S-ben. Terjesszük ki S-t A-ra, legyen ez $S' = (s'_0, \{s'_1, \ldots, s'_m\})$. Az alábbi két típusú programot hívjuk szuperpozíciónak (vagy szuperponált programnak):

a) $(s'_0, \{s'_1, \ldots, s'_m, s\})$

b) $(s'_0, \{s'_1, \ldots, (s'j \parallel s), \ldots, s'_m\})$

Tehát összesen $m + 1$ féle módon tudunk szuperponálni.

Szuperpozíció viselkedési relációja

Legyen az A' állapottéren adott S' program az A alterén adott S program és az s utasítás egy szuperpozíciója. Jelöljük az A altéren adott P, Q logikai függvények

A'-re való kiterjesztését P', Q'-vel. Ekkor:

$$\begin{aligned} P \triangleright_S Q &\Rightarrow P' \triangleright_{S'} Q' \\ P \mapsto_S Q &\Rightarrow P' \mapsto_{S'} Q' \\ P \hookrightarrow_S Q &\Rightarrow P' \hookrightarrow_{S'} Q' \\ \forall Q : P \in inv_S(Q) &\Rightarrow P' \in inv_{S'}(Q') \\ R \in FP_S &\Rightarrow R' \in FP_{S'} \\ \varphi_{S'} &= \varphi'_S \wedge \varphi_S \end{aligned}$$

ahol φ'_S a φ_S logikai függvény kiterjesztése, a φ_S pedig:

$$\varphi_s ::= \bigwedge_{i \in [1 \ldots n]} (\neg\pi \vee (\pi_{id} \wedge v_i = F_i(v_1, \ldots, v_n)))$$

Szekvencia

????