Absztrakt program tulajdonságai, megoldás fogalma.

Az előző tételben pusztán kikötéseket tettünk a feladatot majdan megoldó program viselkedésére. Most előbb megvizsgáljuk egy adott program tulajdonságait annak "szövegéből", majd megmondjuk megfelel-e az ilyen tulajdonságokkal rendelkező program egy adott feladatspecifikációnak. A jelölések nagyon hasonlóak lesznek mint az előzőkben, de míg ott egyszerűen jeleket vezettünk be különféle elvárásaink leírására, most pontosan definiált, kiszámítható műveleteink lesznek tényszerű tulajdonságok leírására

Absztrakt program tulajdonságai

Legyen s egy utasítás [feltételes szimultán értékadás], R pedig egy logikai függvény $[A \to \mathbb{L}]$. Az R utófeltétel s utasításra vonatkozó leggyengébb előfeltétele egy $lf(s,R) : A \to \mathbb{L}$ alakú logikai függvény lesz, amit az igazsághalmazával adunk meg:

$$\lceil lf(s,R) \rceil := \{a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil R \rceil\}$$

A Q előfeltétel s utasításra vett legszigorúbb utófeltétele pedig egy $sp(s,Q) : A \to \mathbb{L}$ alakú logikai függvény, melyre:

$$\lceil sp(s,Q) \rceil := sp(s)(\lceil Q \rceil)$$

Az lf igazsághalmaza azt a legbővebb állapottér-részhalmazt jelöli ki, aminek bármely eleméből s egyszeri végrehajtása biztosan R igazsághalmazába visz. Tehát nem száll el és nem vihet olyan helyre (is), ahol R nem igaz. sp pedig az állapottér azon legszűkebb részhalmazát adja meg, ahova s-vel biztosan eljuthatunk, ha Q-t kielégítő állapotból indultunk.

Az lf-et absztrakt párhuzamos programra általánosíthatjuk oly módon, hogy nem csak egy utasítására vizsgáljuk, hogy vajon "R-be visz-e", hanem azt nézzük, az összes S értékadás-halmazbeli értékadására külön-külön igaz-e a feltétel, azaz:

$$lf(S,R) := \forall s \in S : lf(s,R)$$

Ez azért van így, mert nem tudjuk melyik utasítás fog kiválasztódni, ezért ha biztosra akarunk menni, akkor mindegyikre meg kell vizsgálni a feltételt.

Megoldás

Jellemzően az lf-et $Q \Rightarrow lf(S,R)$ alakú bizonyítások ellenőrzésekor használjuk. Ekkor a fenti definíció teljesülésének ellenőrzésével ekvivalens a $Q \Rightarrow \bigwedge_{i=1}^{n} lf(s_i,R)$ vagy $\forall s \in S : Q \Rightarrow lf(s,Q)$ ellenőrzése, de ezeket egy kicsit könnyebb kiszámolni. (Ez a $\forall$ és az $\wedge$ közötti viszony miatt van így, az $\exists$ és a $\vee$ esetén ez a három kifejezés nem ugyanazt jelentené!)

A következő 4 tulajdonság az s-t S-re cserélve a párhuzamos lf-re is igaz. OK az s-re való teljesülés mellett az $\wedge$ asszociatív és kommutatív tulajdonsága.

1. Csoda kizárásának elve: $lf(s,\downarrow) = \downarrow$

   Azaz, "sehonnan nem juthatunk el valahova". Def. szerint $\lceil lf(s,\downarrow)\rceil := \{a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil\downarrow\rceil\}$, de mivel "hamis" igazsághalmaza üres, csak úgy képezhet a programfüggvény ebbe bele, ha a programfüggvény képe egyenlő az üres halmazzal. Azaz, ha nem jutunk el sehova.

2. Monoton tulajdonság: $(P \Rightarrow Q) \rightarrow (lf(s,P) \Rightarrow lf(s,Q))$

   $lf(s,P) = \{a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil P\rceil\}$, de mivel $P \Rightarrow Q$ def. szerint azt jelenti, hogy $\lceil P\rceil \subseteq \lceil Q\rceil$, ezért $\{a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q\rceil\}$ is igaz, ami pedig azt jelenti, hogy mindazok az állapotok, amikben $lf(s,P)$ igaz, azokban $lf(s,Q)$ is igaz.

3. Gyenge additív tulajdonság: $lf(s,Q) \vee lf(s,R) \Rightarrow lf(s,Q \vee R)$

   Lássuk be, ha a állapot eleme a baloldalnak, akkor a jobbnak is.
   $a \in \lceil lf(s,Q) \vee lf(s,R)\rceil \longleftrightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q\rceil \vee a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil R\rceil \rightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q \vee R\rceil \vee a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil R \vee Q\rceil \longleftrightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q \vee R\rceil \longleftrightarrow a \in \lceil lf(s,Q \vee R)\rceil$

   A másik irány nem igaz. Egy ellenpélda: ...

4. Multiplikatív tulajdonság: $lf(s,Q) \wedge lf(s,P) = lf(s,P \wedge Q)$

   Vegyünk egy tetszőleges a állapotot, belátjuk, hogy pontosan akkor eleme a bal oldalnak, ha a jobb oldalnak.
   $a \in \lceil lf(s,Q) \wedge lf(s,R)\rceil \longleftrightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q\rceil \wedge a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil R\rceil \longleftrightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q\rceil \wedge p(s)(a) \subseteq \lceil R\rceil \longleftrightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q\rceil \cap \lceil R\rceil \longleftrightarrow a \in \mathcal{D}_{p(s)} : p(s)(a) \subseteq \lceil Q \wedge R\rceil \longleftrightarrow a \in \lceil lf(s,Q \wedge R)\rceil$

Egy biztonságossági tulajdonság egy S programra és P,Q logikai függvényekre (feltételekre), hogy P stabil feltéve, hogy nem Q. Jelölése: $P \triangleright_S Q$ avagy relációként: $(\lceil P\rceil, \lceil Q\rceil) \in \triangleright_S$

Definíció: $\triangleright_S \subseteq \wp(A) \times \wp(A), \triangleright_S ::= \{(\lceil P\rceil, \lceil Q\rceil) | P \wedge \neg Q \Rightarrow lf(S, P \vee Q)\}$ (tehát olyan párok vannak relációban, akikre igaz az a híres feltétel)

P-t stabilnak hívjuk, ha $P \triangleright_S \downarrow$. Ez, ha felírjuk a definíciót, azt jelenti, hogy ha egyszer P igaz lett, igaz is marad örökre. Egy haladási tulajdonság egy S programra és P,Q logikai függvényekre, hogy P biztosítja Q-t. Jelölése $P \mapsto_S Q$, relációként: $(\lceil P\rceil, \lceil Q\rceil) \in \mapsto_S$. Definíció: $\mapsto_S \subseteq \wp(A) \times \wp(A), \mapsto_S ::= \{(\lceil P\rceil, \lceil Q\rceil) | (\lceil P\rceil, \lceil Q\rceil) \in \triangleright_S \wedge \exists s \in S : P \wedge \neg Q \Rightarrow lf(s,Q)\}$ azaz, ha van olyan s utasítás, ami átvisz Q-ba, akkor a pártatlan ütemezés miatt ki is választódik, és mivel $\triangleright_S$ is igaz, ezért át is fog vinni Q-ba, mivel egészen s kiválasztódásáig $P \wedge \neg Q$-ban vagyunk.

Még egy haladási tulajdonság egy S programra és P,Q logikai függvényekre, hogy P-ből elkerülhetetlen Q. Jelölése $P \hookrightarrow_S$, relációként: $(\lceil P\rceil, \lceil Q\rceil) \in \hookrightarrow_S$.

Definíció: $\hookrightarrow_S \subseteq \wp(A) \times \wp(A), \hookrightarrow_S ::= \mapsto_S^{tdl}$, azaz a biztosítja reláció tranzitív, diszjunktív lezártja.

Részletesebben:

1. $P \mapsto_S Q \longrightarrow P \hookrightarrow_S Q$ — ha biztosítja, akkor elkerülhetetlen
2. $P \hookrightarrow_S Q \wedge Q \hookrightarrow_S R \longrightarrow P \hookrightarrow_S R$ — tranzitivitás
3. Tetszőleges W megszámlálható halmazra $\forall m \in W : P(m) \hookrightarrow_S Q \longrightarrow (\exists m \in W : P(m)) \hookrightarrow_S Q$ — diszjunktivitás.

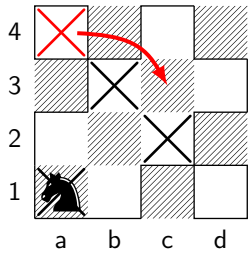
   Érthetőbben: $P \hookrightarrow_S Q \wedge R \hookrightarrow_S Q \rightarrow P \vee R \hookrightarrow_S Q$

Ez a reláció egyértelműen létezik, és egyébként azonos a $\rightsquigarrow$-val, ami meg a feltétlen pártatlan ütemezés mellett generált fa lehetséges útjait jelenti. Magyarán szólva $\hookrightarrow_S$ azt mutatja meg, hova lehet egy adott pontból eljutni.

A program invariánsait az $inv_s(Q)$ halmaz fogja jelenteni. Ebben a halmazban olyan logikai függvények igazsághalmazai (tehát állapot-halmazok) vannak, amelyekre igaz, hogy "kezdetben" is igazak, valamint hogy ha egyszer igazak voltak, akkor nem romolhatnak már el sohse. A zárójelben szereplő Q előfeltételt jelent. Az, hogy "kezdetben igaz", az azt jelenti, hogy ha Q-ból kiindulva lefuttatom az s_0-t (mert annak lefuttatásával kezd a program mindenképp) akkor lesz igaz. Az invariánsság második feltétele pedig a már korábban megismert stabilitási feltétel, tehát: $inv_S : \wp(A) \mapsto \wp(\wp(A))$, azaz $inv_S(\lceil Q \rceil) \subseteq \wp(A), inv_S(\lceil Q \rceil) ::= \{\lceil P \rceil : sp(s_0, Q) \Rightarrow P \wedge P \Rightarrow lf(S, P)\}$ Akkor van "jóság", ha minden elkerülendő állapot az invariánson kívül van.

Az azonosan igaz függvény minden programra invariáns, ezért biztosan van mindig invariáns. Jelölje $INV_S(Q)$ az összes invariánsok konjunkcióját, a legszigorúbb invariánst. Mivel van invariáns, ezért legszigorúbb is van. És persze két tetszőleges invariáns konjunkciója invariáns az $\wedge, \Rightarrow, lf$ tulajdonságai miatt. A legszigorúbb invariáns igazsághalmaza adja meg az elérhető állapotok halmazát. Mivel benne van a definíciójában, ezért kimondható: minden invariáns egyben stabil tulajdonság is. Programot úgy érdemes csinálni, hogy könnyű legyen benne nagyon szűk invariánst számolni. Minél szigorúbb invariánsok igazak egy programra, annál közelebb vagyunk az ideálishoz.

Mindig igaz tulajdonságnak nevezzük a legszigorúbb invariáns következményeit: $\overline{true_S} : \wp(A) \mapsto \wp(\wp(A))$, azaz $true_S(\lceil Q \rceil) \subseteq \wp(A), true_S(\lceil Q \rceil) ::= \{\lceil P \rceil : INV_S(Q) \Rightarrow P\}$. Persze az azonosan igaz a mindig igaz tulajdonságok között is ott van, szóval ez a halmaz se üres. Az invariáns szigorúbb dolog, mint a mindig igaz. Az invariánsban (nem a legszigorúbban) lehetnek nem elérhető állapotok is, de azt várjuk el, hogy ott is "jól viselkedik" a program. A mindig igazaknál csak az elérhető állapotokra fókuszálunk. Egy egy komponensből álló program esetén elég a mindig igazakat vizsgálni. Akkor lehet ugyanis baj, ha egy másik komponens elvisz egy olyan állapotba, ahova egyedül nem tudtunk volna eljutni. Az ilyen állapotokban az invariánsokra támaszkodhatunk, a mindig igazakra nem. Azonos invariánsoknak megfelelő párhuzamos komponenseket nyugodt szívvel össze lehet ereszteni. Ha az csak mindig igaz tulajdonság, akkor ott még további vizsgálatok lehetnek szükségesek.



A fenti ábra segítségével szemléltethető, hogy van olyan mindig igaz állítás, amelyik nem invariáns. Tegyük fel, hogy a bal alsó mezőből indul egy huszár, amely lólépésben haladhat. Az állapottér csak az a1, a2, a4, b1, b2, b3, c2, c3, c4, d1, d3 mezőkből áll, a hiányos "sakktábláról" lelépni nem szabad — az a3, b4, c1, d2, d4 mezők nem részei az állapottér mezőinek. Könnyen ellenőrizhetjük, hogy a huszár mindig "x"-szel jelölt mezőkön marad. Mégsem invariáns tulajdonsága a huszárnak az, hogy "x"-szel jelölt mezőn áll, mert van olyan "x"-szel jelölt mező amelyről jelöletlen mezőre is léphet. Ilyen a bal felső sarokban lévő mező. Ez a mező a huszár számára nem elérhető, ezért sohasem tapasztaljuk az invariáns sérülését. Megállapíthatjuk, hogy a mindig igaz és az invariáns állítások között a lényegi különbség a nem elérehető állapotok esetén jelentkezik. Egy invariáns állítás még nem elérhető állapotokból is megmarad, egy mindig igaz állítás csak az elérhető állapotok felett teljesül. Az invariáns állítások azért fontosak, mert két komponens együttműködése könnyen eredményezheti azt, hogy korábban el nem érhető állapotok elérhetővé válnak. Több komponensből álló elosztott programok esetén tehát csak az invariáns tulajdonságokra támaszkodhatunk

Az invariánsok mindig igazak is egyben, valamint egy invariáns és egy mindig igaz konjunkciója mindig igaz lesz. A legszigorúbb mindig igaz nem más, mint a legszigorúbb invariáns.

Fixpontba akkor jut a program, ha olyan állapotban van, hogy abban minden értékadása olyan, hogy nem teljesül a feltétele (azaz skipként fut le, ha kiválasztódik), vagy teljesül ugyan, de épp azt adja értékül az állapottérnek, ami a jelenlegi értéke. Akkor van egy program fixpontban, ha már biztosan nem változik az állapot, amiben van. A fixpontok halmazát $fixpont_S$-vel vagy φ_S-vel jelöljük. $\varphi_S ::= (\underset{j\in J, i\in[1..n]}{\wedge}(\neg\pi_{j_i} \vee (\pi_{j_{id}} \wedge v_i = F_{j_i}(v_1, \ldots, v_n))))$ ahol $\pi_{j_{id}}$ jelentése, hogy az F_{j_i} determinisztikus. Ha mindegyik az, akkor a fixpontok halmaza így számolható: $\varphi_S ::= (\underset{j\in J, i\in[1..n]}{\wedge}(\pi_{j_i} \rightarrow v_i = F_{j_i}(a)))$. Fixpont tulajdonságnak azokat a logikai függvényeket nevezzük, amelyek fixpontban igazak: $FP_S ::= \{R : A \mapsto \mathbb{L} | \varphi_S \Rightarrow R\}$. A fixpontok halmaza lehet üres, ekkor minden logikai függvény fixpont tulajdonságú. A fixpontok halmazában lehet nem elérhető állapot is. A fixpont tulajdonság gyengíthető. Azaz egy fixpont tulajdonságú állításból következő állítás is fixpont tulajdonságú lesz. A biztosan fixpontba jut tulajdonságú logikai függvények azok, amelyekből kiindulva biz-

tosan fixpontba jutunk... $TERM_S ::= \{Q : A \mapsto \mathbb{L} | Q \hookrightarrow_S fixpont_S\}$

Viselkedési relációnak a $(\triangleright_S, \mapsto_S, \hookrightarrow_S, FP_S, inv_S, TERM_S)$ hatost nevezzük, $p(S)$-vel jelöljük.

Megoldás

Amit a megrendelő kért, azt a $\triangleright_h$ és hasonló alakú kikötések formájában kaptuk, ami meg van, azok így néznek ki: $\triangleright_S$. Ezeket kéne valahogyan összehozni. Elég valami invariáns mellett nézni, hiszen az invariánson kívüliek nem elérhető állapotok. Ha két programkomponens ugyanazon invariáns mellett megfelel a feladatnak, akkor együtt is jók lesznek. Más invariánsok esetén összekeveredhetnek. Minél szigorúbb az invariáns, annál biztosabb, hogy a program mellette meg fog felelni a feladatnak. Cáfolni a legszigorúbb invariáns mellett lehet, de nehéz.

Megoldást bizonyítani invariáns nélkül is lehet, mert az olyan, mintha az azonosan igaz függvényt éselném oda, az pedig invariáns. Ez persze nagyon "drága" programokat eredményez.

Tehát akkor felel meg az S program az F feladatnak, ha minden specifikációs feltételnek megfelel K invariáns mellett, azaz:
ha $\exists K \in inv_S(\bigwedge_{Q \in INIT_h} Q)$

1. $inv_h P$ feltételre: $P \wedge K \in inv_S(\bigwedge_{Q \in INIT_h} Q)$
2. $P \triangleright_h Q$ feltételre: $P \wedge K \triangleright_S Q \wedge K$
3. $P \mapsto_h Q$ feltételre: $P \wedge K \mapsto_S Q \wedge K$
4. $P \hookrightarrow_h Q$ feltételre: $P \wedge K \hookrightarrow_S Q \wedge K$
5. $P \hookrightarrow_h FP_h$ feltételre: $(sp(s_0, P) \wedge K) \in TERM_S$
6. $FP_h \Rightarrow R$ feltételre: $\varphi_S \wedge K \Rightarrow R$

Feladatot és programot is finomítunk. A finomítás tulajdonképpen szigorítás. Feladat finomítása annyit jelent, "elcserélem" a feladatot egy olyanra, amire igaz az, hogy minden program, ami ezt megoldja, az eredetit is. Program finomítása pedig olyan program, ami megfelel az eredeti kikötéseinek is. Mivel több megkötésünk lesz – és a megkötések alapján építjük a programot – a finomítások során vesztünk a hatékonyságból, cserébe a program kézenfekvőbb, illetve könnyebben kitalálható, átlátható lesz.

Akadnak finomítási tételek, amik ezt a programfejlesztési technikát segítik. Például a görbe nyíl haladási feltétel finomítási tétele így szól:

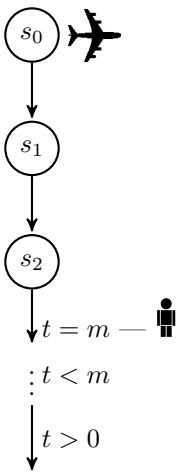
- Ha S megfelel $(P \mapsto_h Q)$-nak, akkor $(P \hookrightarrow_h Q)$-nak is.
- Ha S megfelel $(P \hookrightarrow_h Q)$-nak és $(Q \hookrightarrow_h R)$-nak is, akkor $(P \hookrightarrow_h R)$-nak is.
- $\forall m \in W$ Ha S megfelel $(P(m) \hookrightarrow_h Q)$-nak, akkor $(\exists m \in W : P(m) \hookrightarrow_h R$-nek is.

Ezek a szabályok persze ismételten alkalmazhatók. Vegyük pl. az elsőt, az azt mondja ki, hogy ha szeretnék olyan programot írni, ami megfelel a görbe nyílnak, akkor ezt hagyjam, mert ilyet nehéz csinálni, csináljak helyette inkább olyat, ami megfelel az egyenes nyílnak, és akkor emiatt a tétel miatt a görbének is meg fog. És így tovább. Ezek elég erős megkötést jelentenek. Görbe nyíl bizonyítása igencsak bajos, de itt van nekünk a variánsfüggvény-tétel amely segítő kezet nyújt ilyen helyzetekben.

Maga a variánsfüggvény ezt jelenti: egy $t : A \mapsto \mathbb{Z}$ állapotokhoz egész számokat rendelő függvény, melyre készítünk két logikai függvényt minden m egész számra: $t = m$ jelentse azt, hogy t függvény értéke éppen m, és $t > m$ jelentse azt, hogy nagyobb, mint m. Formálisan $t = m : A \mapsto \mathbb{L}$, úgy hogy $\lceil t = m \rceil = \{a \in A | t(a) = m\}$. Na, ezzel felvértezve kétségbeesésünk hamar tovaillan, a helyességbizonyítás Szküllája és Karübdisze közt sikerrel navigálunk el a variánsfüggvény-tétel segítségével, mely így szól:

Legyen $P, Q : A \mapsto \mathbb{L}$ logikai függvény és $t : A \mapsto \mathbb{Z}$ egy olyan variánsfüggvény, melyre teljesül, hogy $P \wedge \neg Q \Rightarrow t > 0$. Ha $\forall m \in \mathbb{N} : (P \wedge \neq Q \wedge t = m) \hookrightarrow_S ((P \wedge t < m) \vee Q)$ akkor $P \hookrightarrow_S Q$.

Itt a végtelen leszállás nevű ötletet alkalmazzuk. Tudjuk, hogy ha nem teljesül Q, akkor $t > 0$.De azt is tudjuk, hogy vagy teljesül Q, vagy csökken a t. Na de mivel $t \in \mathbb{N}$ nem csökkenhet a végtelenségig, akkor meg előbb-utóbb Q-nak igaznak kell lennie. Amúgy mivel görbe nyíl van, nem azt várjuk el, hogy t szigorúan monoton csökkenjen, csak azt hogy minden számra legyen igaz, hogy ha t épp annyi, akkor előbb-utóbb, akár ideiglenes növekedések után, az alá a szám alá jusson. Vagy Q-ba. De az meg jó. Szemléltetésül itt van ez a szuper ábra ("alulról korlátos függvény csökkenése nem lehet végtelen folyamat")



A másik nagy harcostársunk a fixpontfeltétel finomítása névre hallgat. S így szól: Ha S megfelel az $inv_h P, FP_h \Rightarrow R$ specifikációs feltételeknek és $P \wedge$

$R \Rightarrow Q$, akkor megfelel az $FP_h \Rightarrow Q$ specifikációs feltételnek is. Azaz (ezt a tételt fordítva alkalmazzuk), ha van egy feladatunk, hogy $FP_h \Rightarrow Q$-nek való megfelelést bizonyítsuk be, akkor inkább cseréljük el egy egyszerűbb R-re, meg vezessünk be egy P invariánst, amikből ő már következik és ezt a kettőt lássuk be.

Bizonyítása (nem (így) van a könyvben, de ez egyszerű): Tudjuk, S program megfelel $inv_h P$-nek. Megoldás definíciója + legszűkebb invariánsos lemmából következik: $P \wedge INV_S(Q') \in invs_S(Q')$. Tudjuk azt is, hogy S megfelel $FP_h \Rightarrow R$-nek. Megoldás definíciója + legszűkebb invariánsos lemmából következik: $\varphi_S \wedge INV_S(Q') \Rightarrow R$. Az kéne hasonló okokból, hogy $\varphi_S \wedge INV_S(Q') \Rightarrow Q$. A harmadik feltevésünk miatt az elég, hogy $\varphi_S \wedge INV_S(Q') \Rightarrow P \wedge R$. Az R az következik, az imént láttuk, P-t pedig úgy tudjuk kihozni, hogy ezt a dolgot metsszük el vele, P invariáns, azzal lehet metszeni büntetlenül: $\varphi_S \wedge INV_S(Q') \wedge P \Rightarrow R \wedge P$. Ez igaz. De mivel a legszűkebb invariáns bármivel metszve önmaga lesz, ezért ez a formula ekvivalens ezzel: $\varphi_S \wedge INV_S(Q') \Rightarrow R \wedge P$. És ezt kellett belátni.