Eötvös Loránd Tudományegyetem
Természettudományi Kar
Általános Számítástudományi Tanszék

Horváth Zoltán

A PÁRHUZAMOS PROGRAMOZÁS ALAPJAI

JEGYZET

BUDAPEST, 1994,2001.

HORVÁTH ZOLTÁN

A PÁRHUZAMOS PROGRAMOZÁS ALAPJAI

JEGYZET

BUDAPEST, 1994,2001

Minden észrevételt, amely valamilyen hibára vonatkozik örömmel fogadok.
Budapest, 2001. szeptember 24.

Horváth Zoltán

Eötvös Loránd Tudományegyetem

Általános Számítástudományi Tanszék

HORVÁTH ZOLTÁN

A PÁRHUZAMOS PROGRAMOZÁS ALAPJAI

BUDAPEST 1994

Tartalomjegyzék

Előszó

Ez a jegyzet az ELTE programtervező matematikus hallgatóinak készül, a párhuzamos programozás alapjai című tárgy előadásainak anyagát és az érdeklődő hallgatók számára háttéranyagot tartalmaz. Feltételezzük, hogy az olvasó elsajátította a Bevezetés a programozáshoz ill. Programozás tárgyak tananyagát, rendelkezik megfelelő programozási gyakorlattal, ismeri az Ada programozási nyelvet, írt legalább néhány egyszerű párhuzamos programot, amelyben Ada taszkokat használt.

A következőkben egy olyan matematikai modellt fogalmazunk meg, amely alkalmas arra, hogy párhuzamos programok szintetizálását támogassa.

A bemutatott modell két fontos előzményre épít. Az egyik a nemdeterminisztikus szekvenciális programok reláció alapú modellje [Fót 83], a másik Chandy és Misra párhuzamos programozási módszertana. Mindkettő közös gyökere Dijkstra "programozási diszciplinája".

A párhuzamos programok tervezéséhez készített modell tehát tudatosan vállalja a folytonosságot a szekvenciális programokéval.

Bevezetés

Programozási feladatok specifikációjának és megoldásának alkalmas módszereit keressük többprocesszoros rendszerek esetén.

Párhuzamos programok tervezésének egy matematikai modelljére teszünk javaslatot oly módon, hogy kiterjesztjük a nemdeterminisztikus szekvenciális programok relációs alapú szemantikai modelljét [Fót 83] és a programozási feladatok megfogalmazására és megoldására korábban sikeresen alkalmazott módszereket [Dij 76, Fót Hor 91] párhuzamos programokra is.

Célunk, hogy a modell eszközei segítségével a feladat specifikációját helyettesíteni tudjuk olyan feladatok specifikációival, amely feladatok megoldása esetén a rendelkezésre álló matematikai eszközökkel belátható az eredeti feladat megoldásának helyessége [Var 81, Fót Hor 91, Cha Mis 89].

Arra törekszünk, hogy a megoldás előállításával párhuzamosan a megoldás helyességének bizonyítását is előállítsuk. Nem célunk az automatikus programszintézis [Lav 78],[Eme Sri 88]/4.1.3., és nem akarjuk kész algoritmusok helyességét utólag igazolni [Eme Sri 88]/4.2. Ennek elsősorban az az oka, hogy párhuzamos programokat a legtöbb esetben részben vagy kizárólag azért írunk, hogy a megoldás hatékonyabb legyen a szekvenciális architektúrán elérhető megoldásnál. A hatékonyság lényeges szempont lehet természeténél fogva párhuzamos programmal megoldandó feladatok esetén is, pl. valós idejű alkalmazásoknál, folyamatszabályozó vagy on-line tanácsadó rendszerek esetén [Hor 88]. Talán csak egyes szimulációs feladatok vagy prototípus szoftver fejlesztése során másodlagos a megoldás hatékonysága. Hatékony megoldás előállítására az automatikus programszintézis bonyolultabb feladatok esetén általában nem alkalmas. Az [Eme Sri 88]-ban ismertetett eredmények meggyőzően mutatják azt is, hogy a szintetizáló algoritmusok általában nagyon rossz hatékonyságúak, a megoldás előállításához a specifikáció hosszával exponenciálisan arányos időre van szükség. Hasonló állítások igazak a programbizonyításra is. A programbizonyítási eljárás sikertelensége esetén nincs elegendő támpont a program javításához sem.

A UNITY [Cha Mis 89] lineáris idejű temporális logikára támaszkodó operátorait újrafogalmazzuk a leggyengébb előfeltétel és más predikátumtranszformerek segítségével. Megvizsgáljuk a bevezetett specifikációs módszer kifejezőerejét és az általánosítási lehetőségeket.

Az egyes fejezetekről

A 1. fejezetben megvizsgáljuk, hogy a párhuzamos programok leírására alkotott modellek milyen lényeges tulajdonságokban térnek el egymástól. Megadjuk, hogy az általunk javasolt modell milyen jelenségek leírására alkalmas.

A 3. fejezetben bevezetjük azokat a matematikai eszközöket, amelyeket a későbbiekben felhasználunk, illetve az ismertetett modellek matematikai hátterét tisztázzák. Összefoglaljuk a leképezések fixpontjaira vonatkozó eredményeket és bevezetjük az olvasót a temporális logikák világába. A temporális logikákról szóló 3.1. bekezdésben bevezetett fogalmakra és tételekre a II. részben általában csak egyes megjegyzésekben és lábjegyzetekben utalunk, így a II. rész fejezeteinek megértéséhez ez a bekezdés nem szükséges.

A második részben bevezetjük egy reláció alapú modell alapfogalmait.

A 6. fejezetben megadjuk a relációs modell alapfogalmainak definícióit. A két legfontosabb bevezetett fogalom a feladat és az absztrakt program fogalma.

Az 7. fejezetben adjuk meg, hogy egy absztrakt program mikor old meg egy feladatot. Kimondunk néhány tételt is, amelyek igazolják, hogy a bevezetett megoldásfogalom megfelel elvárásainknak.

A 10. fejezetben megvizsgáljuk a bevezetett modell tulajdonságait, kifejezőerejét.

A 8. fejezetben több hasznos tételt bizonyítunk, amelyek segítségével a későbbiek során könnyebben igazolhatjuk feladatok és programok tulajdonságait.

A harmadik részben programozási tételeket mondunk ki és összetett problémák megoldása során alkalmazható eszközöket vezetünk be.

A 11. fejezetben általánosan megfogalmazott programozási feladatokat oldunk meg. A kapott megoldásokat programozási tételeknek nevezzük, mert széles körben alkalmazhatóak konkrét feladatok megoldása során. Az egyik ilyen alapfeladat asszociatív művelet eredményének párhuzamos kiszámítása. A másik az elemenként feldolgozható, ill. a sorozatokon többszöros függvénykompozícióval definiált függvény értékének kiszámítása. Példát mutatunk csatornaváltozók használatára és adatcsatornás megoldási módszerekre is. Megvizsgáljuk, hogy a kapott megoldások milyen architektúrákon implementálhatók hatékonyan. Olyan megoldásokat dolgozunk ki, amelyek osztott és aszinkron osztott memóriás rendszerekre is könnyen leképezhetőek.

A 9. fejezetben összetett problémák megoldása során alkalmazható eszközöket vezetünk be. A megoldást modulokból állítjuk össze. Az absztrakt programokat egyesítjük és szuperponáljuk, támaszkodunk az ún. nyitott specifikáció fogalmára [Cha Mis 89]. Megvizsgáljuk programok szekvenciális ill. valós párhuzamos kompozíciójának lehetőségét.

A 12. fejezetben finomítjuk az elemi művelet fogalmát és absztrakt architektúrákat definiálunk. Megadjuk, hogy az 6. fejezetben bevezetett absztrakt program implementációja mikor helyes az egyes architektúrákon. Bevezetjük az elemi adat, elemi művelet fogalmát és kísérletet teszünk a típusinvariáns fogalmának [Fót 83] általánosítására.

A 13. fejezetben összefoglaljuk és értékeljük a leírt módszereket.

A függelékben megvizsgáljuk, hogy az absztrakt programokat hogyan kódoljuk Ada `task`-ok segítségével és mellékeljük a tárgy tematikáját és követelményrendszerét.

A könnyebb tájékozódást kereszthivatkozások, tárgymutató és jelölések jegyzéke, a legfontosabb fogalmak definíciói, stb. segíti.

Egyes megjegyzéseket lábjegyzetben helyeztünk el. A lábjegyzetekben általában más modellek rokon fogalmaira utalunk röviden. Ezek a megjegyzések elsősorban azoknak az olvasóknak szólnak, akik ezekben a modellekben járatosak.

Gyakran használt jelölések

$::=$ - definiáló egyenlőség
$A::=\overset{\times}{_{i\in I}} A_i$ - állapottér
$a = (a_1, \ldots, a_n) \in A$ - állapot
$R \subseteq \overset{\times}{_{i\in I}} A_i$ - reláció
$R_n(A)$ az $\overset{\times}{_{i\in[1..n]}} A$ direktszorzat feletti relációk halmaza
$R \subseteq A \times B$ - bináris reláció
$R(a)$ - az R reláció képhalmaza
$\mathcal{D}_R$ - az R reláció értelmezési tartománya
$\alpha = (\alpha_1, \ldots, \alpha_n)$ - véges sorozat
$\alpha = (\alpha_1, \ldots\ldots)$ - végtelen sorozat
A^* : A elemeiből képzett véges sorozatok halmaza
A^∞ : A elemeiből képzett végtelen sorozatok halmaza
$A^{**} ::= A^* \cup A^\infty$
$\mathcal{P}(A)$ - az A hatványhalmaza
$R : A \longrightarrow B$ - parciális függvény A-ról B-re
$R : A \longmapsto B$ - függvény A-ról B-re
$pr_{A_1} : A \longmapsto A_1$ - az A térről az A_1 altérre vetítés
$\mathcal{L}::=\{igaz, hamis\}$ - logikai értékek halmaza
$\uparrow$: $A \longmapsto \mathcal{L}$ - az azonosan igaz,
$\downarrow$: $A \longmapsto \mathcal{L}$ - az azonosan hamis logikai függvény.
$\lceil f \rceil$ - logikai függvény (állítás) igazsághalmaza
$P \Rightarrow Q ::= \lceil P \rceil \subseteq \lceil Q \rceil$
$\lceil P \wedge Q \rceil ::= \lceil P \rceil \cap \lceil Q \rceil$
$\lceil P \vee Q \rceil ::= \lceil P \rceil \cup \lceil Q \rceil$
$\lceil \neg Q \rceil ::= A \setminus \lceil Q \rceil$
$\lceil P \to Q \rceil ::= \lceil \neg P \vee Q \rceil$
$\Longrightarrow, \Longleftrightarrow, \Longleftarrow$ - "ha, akkor", "akkor és csak akkor", ill. "akkor, ha"
$R^{(-1)}$ - inverz reláció
R^{tdl} - reláció tranzitív diszjunktív lezártja
$[R]$ az $R =\uparrow$ állítás rövidítése, ahol $R \subseteq A \times \mathcal{L}$
$v_i : A \longmapsto A_i$ - változó
$\mathcal{N}$ - pozitív egészek halmaza
$\mathcal{N}_0$ - nemnegatív egészek halmaza

ω - a természetes számok halmazának rendszáma
$\eta X : G(X)$ - G legnagyobb fixpontja X szerint
$\mu Y : F(Y)$ - F legkisebb fixpontja Y szerint
$P \rhd Q$ - P stabil feltéve, hogy nem Q
$P \mapsto Q$ - P biztosítja Q-t
$P \hookrightarrow Q$ - P-ből elkerülhetetlen Q
FP $\Rightarrow R$ - R teljesül fixpontban
$Q \in$ INIT - Q igaz kezdetben
inv$P$ - P invariáns
$Q \hookrightarrow$ FP, $Q \in$ TERM - Q-ból indítva a program biztosan fixpontba jut
$VR(P)$ - azok a változók, amelyektől a P (logikai) reláció (függvény) függ
F - feladat
B - paramétertér
s - utasítás
I - a változók és az állapottérkomponensek indexeinek halmaza
J - a program utasításainak indexhalmaza
$p(s)$ - az s utasítás hatásrelációja
$SKIP$ - üres utasítás
$VL(s)$ - az s utasítás baloldalán álló változók
$VR(s)$ - az s utasítás jobboldalán álló változók
$V(s)::=VR(s) \cup VL(s)$
S - absztrakt program
$E(S)(a)$ - S program a-ból elérhető állapotainak halmaza
$VL(S)$ - az S programban baloldalon álló változók
$VR(S)$ - az S programban jobboldalon álló változók
$V(S)::=VR(S) \cup VL(S)$
$fixpont_S$ - az S absztrakt program fixpontjait jellemző állítás
P', F', S' - altéren definiált logikai fgv., feladat, program kiterjesztése
$s_1 \| s_2$ - feltételes értékadások szuperpozíciója
$S_1 \cup S_2$ - programok uniója
$F_1 \sqcup F_2$ - feladatok egyesítése
$S_1; S_2$ - programok szekvenciája
BT - elágazó idejű temporális logika
LT - lineáris idejű temporális logika

rész I

Párhuzamos programok modelljei

rész II

Feladat, program, megoldás

4. Fejezet

A relációs modell alapfogalmai

> Programozási modell*nek* *nevezzük azt a matematikai modellt, amely megadja feladatok és programok szemantikai jelentését, konstrukciós műveleteket definiál feladatok és programok felett, valamint megadja, hogy egy program mikor old meg egy feladatot. Relációs modellről beszélünk, ha a szemantikai tartományok elemei relációk.*

Gondolkodásunk során a programozási feladat [Fót 83] fogalmából indulunk ki. Programozási feladatot mindig egy állapottéren [Dij 76] fogalmazunk meg. A feladat megfogalmazásához tehát meg kell alkotnunk a feladat matematikai modelljét[1], absztrakcióra van szükség. A feladat megfogalmazásához vezető utat most nem vizsgáljuk.[2]

Az állapottér véges sok legfeljebb megszámlálhatóan végtelen típusértékhalmaz direkt szorzata [Fót 83].

4.1. Definíció. *(Állapottér)*
$I \subset \mathcal{N}. \forall i_j \in I : A_{i_j}$ megszámlálható halmaz. Az $A ::= \overset{\times}{_{i_j \in I}} A_{i_j}$ halmazt *állapottér*nek, az A_{i_j} halmazokat *típusértékhalmaz*oknak nevezzük.

4.2. Definíció. *(Állapot)*
Az állapottér elemeit, az $a = (a_{i_1}, \ldots, a_{i_n}) \in A$ pontokat *állapot*oknak nevezzük.

---

[1] A modell szót általános értelemben használjuk. Ha a matematikai logikában, a modellelméletben használt modellfogalomra gondolunk, akkor erre külön hivatkozunk. Egy feladat megfogalmazása nem köthető pl. egy rögzített temporális logikai struktúrához, mert ez esetben a feladat már nem fogalmazható meg függetlenül az időstruktúrát definiáló programtól. A feladatot nem azonosítjuk az őt leíró formulák halmazával, mint szintaktikus egységekkel sem, ugyanis egy interpretálatlan formulahalmaz minden interpretációban más és más szemantikai jelentést hordoz. A feladatot mindig egy rögzített állapottér felett, azon értelmezett relációk segítségével adjuk meg.

[2] Nem vizsgáljuk azt, hogy egy feladat formális alakja valóban azt a feladatot írja-e le, amit valamilyen természetes nyelven megfogalmaztak. A választott programozási modell keretein túlmutat ennek a kérdésnek a vizsgálata.

A feladat matematikai megfogalmazásához szükségünk lesz a reláció és a sorozat fogalmára. Megismételjük a reláció, bináris reláció és a reláció értelmezési tartománya definícióját (3. fejezet).

4.3. Definíció. *(Reláció)*
$I \subset \mathcal{N}$. Az $R \subseteq \overset{\times}{_{i_j \in I}} A_{i_j}$ halmazt *reláció*nak[3] nevezzük.

Az $R \subseteq A \times B$-t *bináris reláció*nak nevezzük.

4.4. Definíció. *(Reláció értéke)*
$R \subseteq A \times B$. $R(a) ::= \{\, b \,|\, (a,b) \in R \,\}$ halmaz az R reláció *értéke*[4] *az a pontban.*

4.5. Definíció. *(Reláció értelmezési tartománya)*
$R \subseteq A \times B$. Az R *reláció értelmezési tartománya*:
$\mathcal{D}_R ::= \{\, a \in A \,|\, \exists b \in B : (a,b) \in R\}$. [5]

Jelölések:
véges sorozat: $\alpha = (\alpha_1, \ldots, \alpha_n)$
végtelen sorozat: $\alpha = (\alpha_1, \ldots\ldots)$
A^* : A elemeiből képzett véges sorozatok halmaza
A^∞ : A elemeiből képzett végtelen sorozatok halmaza
$A^{**} ::= A^* \cup A^\infty$

4.6. Definíció. *(Hatványhalmaz)*
Az A halmaz részhalmazainak halmazát az A *hatványhalmaz*ának nevezzük és $\mathcal{P}(A)$-val jelöljük.

A feladat matematikai megfelelője feltételek együttese. Minden egyes feltétel az állapottér hatványhalmaza felett értelmezett reláció. Az állapottér részhalmazait logikai relációkkal jellemezzük. Megadjuk a logikai reláció, logikai függvény és igazsághalmazuk definícióját.

4.7. Definíció. *(Parciális függvény)*
Az $R \subseteq A \times B$ reláció *determinisztikus reláció*, (vagy *parciális függvény*), ha $\forall a \in A : |\, R(a) \,| \leq 1$.
Parciális függvények esetén az $R : A \longrightarrow B$ jelölést alkalmazzuk.

4.8. Definíció. *(Függvény)*
Az $R \subseteq A \times B$ reláció *függvény*, ha $\forall a \in A : |\, R(a) \,| = 1$.
Függvények esetén az $R : A \longmapsto B$ jelölést használjuk.

---

[3] Példa: $A ::= \{1,2,5\}$, $B ::= \{2,3\}$. $R \subseteq A \times B$. $R ::= \{(1,3),(1,2),(5,2)\}$.

[4] A példa R relációjának képe az 1 ponban: $R(1) = \{2,3\}$.

[5] A példa R relációjának értelmezési tartománya: $\mathcal{D}_R = \{1,5\}$.

4.9. Definíció. *(Logikai függvény, logikai reláció)*
$f \subseteq A \times \mathcal{L}$ reláció *logikai reláció* , ahol $\mathcal{L}::=\{igaz, hamis\}$ a logikai értékek halmaza. *Logikai függvény*-ről beszélünk, ha a logikai reláció függvény.

Jelölés:
$\uparrow: A \longmapsto \mathcal{L}$ - az azonosan igaz,
$\downarrow: A \longmapsto \mathcal{L}$ - az azonosan hamis logikai függvény.

4.10. Definíció. *(Logikai függvény igazsághalmaza)*
$\lceil f \rceil ::= \{a \in A | f(a) = \{igaz\}\}$ az *f logikai függvény (állítás) igazsághalmaza*[6].

4.1. Megjegyzés. *(Logikai műveletek)*
A logikai relációk felett értelmezzük a $\wedge, \vee, \rightarrow, \Rightarrow, \neg$ műveleteket, oly módon, hogy azok megfeleljenek a relációk igazsághalmazaira vonatkozó halmazműveleteknek.
Azaz: $P \Rightarrow Q ::= \lceil P \rceil \subseteq \lceil Q \rceil$, $\lceil P \wedge Q \rceil ::= \lceil P \rceil \cap \lceil Q \rceil$, $\lceil P \vee Q \rceil ::= \lceil P \rceil \cup \lceil Q \rceil$, $\lceil \neg Q \rceil ::= A \setminus \lceil Q \rceil$, és $\lceil P \rightarrow Q \rceil ::= \lceil \neg P \vee Q \rceil$.
A $\Longrightarrow, \Longleftrightarrow, \Longleftarrow$ jeleket bizonyítások szövegének rövidítésére használjuk, a "ha, akkor", "akkor és csak akkor", ill. "akkor, ha" állítások leírásának rövidítésére.

4.11. Definíció. *(Reláció inverz képe)*
$R^{(-1)}(H) ::= \{ a \in A \,|\, \exists h \in H : (a, h) \in R \}$ a $H \subseteq B$ halmaz $R \subseteq A \times B$ relációra vonatkozó *inverz kép*e.

4.12. Definíció. *(Reláció ősképe)*
$R^{-1}(H) ::= \{ a \in A \,|\, R(a) \subseteq H \wedge R(a) \neq \emptyset \}$ a $H \subseteq B$ halmaz $R \subseteq A \times B$ relációra vonatkozó *ősképe* [WRMP 95].

4.13. Definíció. *(Relációk kompozíciója)*
$R_2 \circ R_1 ::= \{ (a, c) \,|\, \exists b \in B : (a, b) \in R_1 \wedge (b, c) \in R_2 \}$ az $R_1 \subseteq A \times B$ és az $R_2 \subseteq B \times C$ relációk *kompozíció*ja.

4.14. Definíció. *(Relációk szigorú kompozíciója)*
$R_2 \odot R_1 ::= \{ (a, c) \,|\, R_1(a) \subseteq \mathcal{D}_{R_2} \wedge \exists b \in B : (a, b) \in R_1 \wedge (b, c) \in R_2 \}$ az $R_1 \subseteq A \times B$ és az $R_2 \subseteq B \times C$ relációk *szigorú kompozíció*ja [Hor 90].

4.15. Definíció. *(Reláció igazsághalmaza)*
$\lceil R \rceil ::= R^{-1}(\{igaz\})$ az $R \subseteq A \times \mathcal{L}$ *reláció igazsághalmaza.*

4.16. Definíció. *(Reláció tranzitív diszjunktív lezártja)*
$R^{tdl} \subseteq \mathcal{P}(A) \times \mathcal{P}(A)$ reláció az $R \subseteq \mathcal{P}(A) \times \mathcal{P}(A)$ reláció tranzitív diszjunktív lezártja, ha R^{tdl} a legkisebb olyan reláció, amelyre: $R \subseteq R^{tdl}$, ha $(a, b) \in R^{tdl}$ és $(b, c) \in R^{tdl}$, akkor $(a, c) \in R^{tdl}$ és bármely W megszámlálható halmazra: $(\forall m : m \in W :: (a(m), b) \in R^{tdl}) \Longrightarrow ((\bigcup_{m \in W} a(m)), b) \in R^{tdl}$.

[6] Példa: $A ::= \{1, 2, 5\}$. $R : A \mapsto \mathcal{L}$. $R ::= \{(1, igaz), (2, hamis), (5, igaz)\}$. $\lceil R \rceil = \{1, 5\}$.

4.17. Definíció. *($\lceil R \rceil$)*
Legyen $R \subseteq A \times \mathcal{L}$. $\lceil R \rceil$ annak az állításnak a rövid megfogalmazása, hogy az R reláció igazsághalmaza megegyezik A-val [Dij Sch 89].

4.2. Megjegyzés. $\lceil P \rceil \subseteq \lceil Q \rceil \iff P \Rightarrow Q \iff [P \to Q]$.

4.18. Definíció. *(Változó)*
A $v_{i_j} : A \longmapsto A_{i_j}$ projekciókat *változó*knak nevezzük.[7] A változók megadásakor a projekció értelmezési tartományát, az állapotteret, általában elhagyjuk: $v_{i_j} : A_{i_j}$.

4.19. Definíció. *(Vetítés altérre)*
Legyen A_1 az A direkt szorzat altere. $pr_{A_1} : A \longmapsto A_1$ függvény az A-beli pontokhoz az A_1-beli vetületüket rendeli.
A pr_{A_1} függvényt általánosítjuk A részhalmazaira, A-beli elemek sorozataira, A felett értelmezett bináris relációkra oly módon, hogy a részhalmazok és sorozatok elemeit ill. a relációk elemeinek komponenseit pontonként vetítjük az altérre [WRMP 95].

4.20. Definíció. *(Reláció független egy változótól)*
Legyen $A ::= \overset{\times}{_{i\in[1,n]}} A_i$ és $R \subseteq A \times B$. Azt mondjuk, hogy az R reláció *független* az A_i komponenstől és a $v_i : A \longmapsto A_i$ változótól, ha
$\forall a, b \in \mathcal{D}_R : (\forall k \in ([1, i-1] \cup [i+1, n]) : a_k = b_k) \Rightarrow R(a) = R(b)$

Jelöljük $VR(R)$-rel azon változók halmazát, amelyektől az R reláció függ.

4.21. Definíció. *(Reláció nem változtatja meg)*
Legyen $A ::= \overset{\times}{_{i\in[1,n]}} A_i$ és $R \subseteq A \times A$. Azt mondjuk, hogy az R reláció *nem változtatja meg* az A_i komponens és a $v_i : A \longmapsto A_i$ változó értékét, ha $v_i \circ R = v_i$.

Jelöljük $VL(R)$-rel azon változók halmazát, amelyeket az R reláció megváltoztat. $V(R) ::= VL(R) \cup VR(R)$.

[7] A fenti definíció szerint a változó tehát nem szintaktikus fogalom. A továbbiakban amíg a választott programozási modell keretein belül maradunk nem vizsgáljuk formális nyelvek és szemantikai tartományok lehetséges kapcsolatrendszereit. Ha szükséges, a későbbiek során könnyen definiálható formális nyelv és szemantikus leképezés. A szemantikai tartomány elemei az általunk megfogalmazott modellben definiált egyes matematikai objektumok lehetnek.

5. Fejezet

A feladat fogalmának általánosítása

A feladat definíciójának megfogalmazásakor általánosítjuk azt a specifikációs módszert, amely relációk segítségével megfogalmazott elő- és utófeltételeket használ. A most bevezetett feladatfogalom magában foglalja azt az esetet is, amikor egy vagy több nem feltétlenül terminálló folyamat, egy zárt rendszer *együttes viselkedésére teszünk előírásokat. Megjegyezzük, hogy a feladat függetlenül megfogalmazható bármely lehetséges megoldásától, összehasonlítható más feladatokkal, illetve összevethető tetszőleges vele közös állapottéren futó programmal abból a szempontból, hogy az megoldja-e. A feladat megoldása nem feltétlenül csak párhuzamos program lehet.*

A feladat matematikai megfelelője *specifikációs reláció*k együttese. A specifikációs relációkat az állapottér hatványhalmaza felett értelmezzük, a relációk elemeit *specifikációs feltétel*eknek nevezzük.

5.1. Specifikációs feltételek

A specifikációs feltételek a programra, mint az állapottér feletti mozgásra fogalmaznak meg kikötéseket. Ezeket a kikötéseket csoportosíthatjuk típusuk szerint. Egy feladat leírásához hét féle feltételt használunk. Az azonos feltételtípushoz tartozó feltételeket egy relációban gyűjtjük össze, így hét specifikációs relációt vezetünk be.

Legyen $P, Q, R, U : A \longmapsto \mathcal{L}$ logikai függvény.
$\rhd, \mapsto, \hookrightarrow \subseteq \mathcal{P}(A) \times \mathcal{P}(A)$ relációk, és $\mathrm{FP}, \mathrm{INIT}, \mathrm{inv}, \mathrm{TERM} \subseteq \mathcal{P}(A)$ halmazok[1].

A relációk megadásakor infix jelölést alkalmazunk, ezért bevezetjük az alábbi *jelöléseket*. Zárójelben megadjuk azt is, hogy hogyan olvassuk azt, ha egy halmaz vagy egy halmazpár eleme az adott relációnak. Az állapottér részhalmazait logikai relációkkal jellemezzük.

[1] unáris relációk

Jelölések:

$P \rhd Q ::= (\lceil P \rceil, \lceil Q \rceil) \in \rhd$ (P stabil feltéve, hogy nem Q),
$P \mapsto Q ::= (\lceil P \rceil, \lceil Q \rceil) \in \mapsto$ (P biztosítja Q-t),
$P \hookrightarrow Q ::= (\lceil P \rceil, \lceil Q \rceil) \in \hookrightarrow$ (P-ből elkerülhetetlen Q),
$Q \hookrightarrow \mathrm{FP} ::= \lceil Q \rceil \in \mathrm{TERM}$ (Q-ból a program biztosan fixpontba jut),
$\mathrm{FP} \Rightarrow R ::= \lceil R \rceil \in \mathrm{FP}$ (R teljesül fixpontban),
$\mathrm{inv}P ::= \lceil P \rceil \in \mathrm{inv}$ (P invariáns).
$Q \in \mathrm{INIT} ::= \lceil Q \rceil \in \mathrm{INIT}$ (Q igaz kezdetben),

5.1. Példa. *(Specifikációs reláció)*
Legyen az állapottér $A ::= \mathcal{N}$ egyelemű direkt szorzat. Az egyetlen komponenshez tartozó változót jelöljük i-vel. Legyen $\rhd ::= \{(\lceil i = k \rceil, \lceil i = k+1 \rceil) \mid k \in \mathcal{N}\}$. Ekkor pl. az $i = 5 \rhd i = 6$ feltétel azt a kikötést fogalmazza meg, hogy a program az $a = (5)$ állapotból csak az $a = (6)$ állapotba juthat. A teljes reláció azt a feltételegyüttest adja meg, amely megköveteli, hogy a program futása során az i változó értéke csak egyesével növekedhet. Megkönnyíti a relációk kezelését, ha egy-egy relációnak csak néhány eleme van. Ezért célszerű a reláció k paraméter szerinti felbontása egyelemű relációkra, erre a célra vezetjük majd be a paramétertér fogalmát. □

A programra, mint az állapottér feletti mozgásra vonatkozó specifikációs feltételeket négy csoportra osztjuk aszerint, hogy milyen típusú kikötéseket fogalmazunk meg segítségükkel[2]. *A relációcsoportok és az egyes specifikációs relációk elnevezései azt tükrözik, hogy az adott reláció segítségével milyen jellegű feltételeket kívánunk megfogalmazni.* A specifikációs relációk pontos szemantikáját az adja meg, hogy egy program mikor felel meg egy adott relációhoz tartozó feltételnek. Ennek megfogalmazásához szükséges az absztrakt program (6.15. def.) és a megoldás (7.1. def.) definíciójának ismerete[3]. Az alábbiakban röviden és informálisan már most megadjuk az egyes feltételek jelentését.

- A $P \rhd Q$ és az $\mathrm{inv}P$ alakú feltételeket *biztonságossági feltétel*eknek nevezzük. Ha a program állapotára teljesül a $P \wedge \neg Q$ feltétel, akkor $P \rhd Q$ tiltja, hogy a program Q érintése nélkül közvetlenül egy $\neg P \wedge \neg Q$-beli állapotba jusson. $\mathrm{inv}P$ pedig kiköti, hogy a P feltétel igazsághalmazából a program minden elemi lépése a P igazsághalmazába vigyen, valamint, hogy P "kezdetben"[4] is teljesüljön.

[2] A kikötések teljesülését - az invariánsok kivételével - általában nem a teljes állapottér felett vizsgáljuk majd meg, hanem csak az állapottér egy olyan részhalmaza felett, amely tartalmazza az összes elérhető állapotot.

[3] A feladat szemantikáját a specifikációs relációk segítségével meg tudjuk adni oly módon, hogy bármely feladat függetlenül leírható bármely azt megoldó vagy meg nem oldó programtól, azaz a feladatok a programoktól független szemantikával rendelkeznek a modellben.

[4] A kezdeti értékadás végrehajtása után.

- A $P \mapsto Q$, illetve $P \hookrightarrow Q$ *haladási feltétel*ek előírják, hogy ha a program egy P-beli állapotba jut, akkor abból előbb - utóbb Q-ba jusson. $P \mapsto Q$ további megszorítást tesz a haladási irányra. $Q \hookrightarrow$ FP kikötésnek megfelelő program előbb-utóbb *biztosan fixpontba jut* Q-beli állapotából.

- A FP $\Rightarrow R$ *fixpont feltétel*ekkel szükséges feltételeket fogalmazunk meg arra, hogy a program fixpontba jusson.

- Elégségesnek tekintjük, ha $Q \in$ INIT *kezdeti feltétel*ekkel meghatározott állapotokból indítva helyesen működik a program.

5.1. Megjegyzés. *Stabilitási feltétel*nek nevezzük P-t , ha $P \rhd \downarrow$. P *konstans feltétel*, ha $\neg P$ is és P is stabilitási feltétel.

A továbbiakban a $\rhd, \mapsto, \hookrightarrow,$ FP, INIT, inv, TERM relációkat *specifikációs reláció*knak, elemeiket *specifikációs feltétel*eknek, ezen belül az $\rhd$, $\hookrightarrow$, $\mapsto$, inv, TERM relációk elemeit *átmenetfeltétel*eknek, az INIT, FP relációk elemeit pedig *peremfeltétel*eknek nevezzük. Az INIT reláció a *környezeti előírás*ok csoportjába tartozik.

5.2. A programozási feladat definíciója

5.1. Definíció. *(Programozási feladat)*
Legyen A egy állapottér, B pedig egy tetszőleges, megszámlálható halmaz. Rendeljünk hozzá a $b \in B$ pontokhoz rendezett reláció heteseket. Minden egyes rendezett hetes kettő, peremfeltételeket megadó, illetve öt, átmenetfeltételeket leíró relációt tartalmaz.
Az $F \subseteq B \times (\overset{\times}{_{i\in[1..3]}} \mathcal{P}(\mathcal{P}(A) \times \mathcal{P}(A)) \overset{\times}{_{i\in[1..4]}} \mathcal{P}(\mathcal{P}(A)))$ relációt az A állapottér felett definiált *feladat*nak, B-t pedig a feladat *paraméterter*ének nevezzük.
A $\overset{\times}{_{i\in[1..3]}} \mathcal{P}(\mathcal{P}(A) \times \mathcal{P}(A))$ és $\overset{\times}{_{i\in[1..4]}} \mathcal{P}(\mathcal{P}(A))$ direktszorzat $b \in B$-hez rendelt $h \in F(b)$ elemének komponenseit rendre $\rhd_h, \mapsto_h, \hookrightarrow_h$, TERM_h,FP_h, inv_h, INIT_h-val jelöljük. Ha $F(b)$ egyelemű, akkor h helyett b-t írunk vagy a h indexet teljesen elhagyjuk, ha ez nem okoz félreértést.

5.2. Példa. *(Programozási feladat)*
Példaként megadjuk az elemenkénti feldolgozás feladatának specifikációját:
A specifikációs relációk közül négy üres, három pedig egyelemű. Kikötjük, hogy bármely fixpontban az y rendezett halmaz-m-es értéke éppen $f(x')$ legyen ((5.3.) feltétel), ahol x' az x változó kezdeti értéke ((5.1.) feltétel). Megköveteljük, hogy a program biztosan elérje valamelyik fixpontját ((5.2.) feltétel). A feladat megfogalmazásában az f függvény argumentuma, mint a specifikációs feltételek paramétere jelenik meg.

$$A = X \times Y, \quad x : X, y : Y, B = X, \quad x' : X.$$

$$(x = x') \in INIT_{x'} \tag{5.1.}$$
$$\uparrow\!\hookrightarrow \mathrm{FP}_{x'} \tag{5.2.}$$
$$\mathrm{FP}_{x'} \Rightarrow y = f(x'), \tag{5.3.}$$

ahol f elemenként feldolgozható.

Az X, Y típus specifikációja, az elemenként feldolgozható függvény fogalma és a megoldó program megtalálható a 11. fejezetben. □

A paramétertér helyes megválasztásával elérhetjük, hogy a $\rhd_h$, $\hookrightarrow_h$, FP_h, stb. relációk végesek legyenek, vagy éppen csak egyetlen egy halmaz ill. halmazpár legyen az elemük. Ha a B paramétertér végtelen, akkor így összességében végtelen sok relációt adunk meg. Ezek a relációk azonban általában csak a b paraméter értékében különböznek egymástól, így a megoldás definícióját elegendő lesz egyetlen $b \in B$ paraméteres esetre megvizsgálni.

A paramétertér bevezetésével könnyen megfogalmazhatunk olyan feladatokat, amelynek megoldása több alternatív viselkedésminta szerint is lehetséges[5].

5.2. Megjegyzés. A paramétertér általában maga is az állapottérhez hasonló direktszorzat. Sok esetben van az állapottérnek és a paramétertérnek nem üres, közös altere. A paramétertér projekcióit is változóknak nevezzük, ezeket a változókat megkülönböztetésül $'$ jellel egészítjük ki, pl.: v'.[6]

5.3. Megjegyzés. A feladat fenti definíciója a [Fót 83, Fót Hor 91]-ben ismertett specifikációs módszer általánosítása. Legyen $\forall b \in B : |F(b)| = 1$ és $Q_b \in \mathrm{TERM}_b$, $\{Q_b\} = \mathrm{INIT}_b$ és $\{R_b\} = \mathrm{FP}_b$.

5.4. Megjegyzés. A specifikációs feltételek szintaktikus alakjától a feladat, mint reláció független. Lényegében ugyanazt a feladatot (5.5. def.) azonban több ekvivalens specifikációs feltételhalmazzal is megfogalmazhatjuk.

Az 7.1. definícióban megadjuk, hogy az F feladatnak mikor *megoldás*a egy program, azaz mikor elégíti ki a feladatban előírt feltételeket[7].

---

[5] Ha a feladat determinisztikus, akkor megfogalmazhatnánk a feladatot a paramétertér bevezetése nélkül is, mint a $\overset{\times}{_{i\in[1..3]}}\mathcal{P}(\mathcal{P}(A)\times\mathcal{P}(A))_{i\in[1..4]}\overset{\times}{\mathcal{P}}(\mathcal{P}(A))$ direktszorzat elemét. Ebben az esetben azonban a feladat egyes komponenseinek számossága kezelhetetlenül nagy lehet, pl. egy-egy átmenetfeltételnek általában végtelen sok halmazpár eleme lenne.

[6] Elsőrendű temporális logikai nyelvekben szokás az állapottér változóit lokális változóknak, a paramétertér változóit globális vagy rigid változóknak nevezni.

[7] Azt mondjuk, hogy az S program megoldja az F feladatot, ha $\forall b \in B : \exists h \in F(b)$, hogy az S program *megfelel* a h-ban adott $\mathrm{inv}_h P$, $P \rhd_h U$, $P \mapsto_h U$, $P \hookrightarrow_h U$, $\mathrm{FP}_h \Rightarrow R$, $Q \in \mathrm{TERM}_h$ alakú specifikációs feltételek mindegyikének a $Q \in \mathrm{INIT}_h$ kezdeti feltételek mellett.

5.3. Feladat kiterjesztése

Legyen A_1 az A altere, F az A_1 állapottér és B paramétertér felett definiált feladat. Az F_1 A térre vett kiterjesztése az az F' feladat, amely a kiegészítő altér változóira nem tesz kikötéseket és az A_1 altérre vett vetülete megegyezik F_1-gyel.

Ha P szerepel az F feladat egy specifikációs feltételében, akkor a kiterjesztett feladat megfelelő specifikációs feltételében egy olyan P' logikai függvény szerepel, amelynek A_1-re vett vetülete P és nem függ a kiegészítő altér változóitól.

5.2. Definíció. *(Logikai függvény kiterjesztése)*
Jelöljük P'-vel a P altéren definiált *logikai függvény* teljes térre való *kiterjesztését*. P' igazsághalmaza az a legbővebb halmaz, amelynek vetülete P igazsághalmaza.

5.3. Definíció. *(Feladat kiterjesztése)*
Legyen A_1 az A altere, F az A_1 állapottér és B paramétertér, F' az A tér és a B paramétertér felett definiált feladat. F'-t az F kiterjesztésének nevezzük, ha $\forall b \in B : pr_{A_1}(F'(b)) = F(b)$ és F' specifikációs feltételeiben előforduló logikai függvények az F specifikációs feltételeiben adott logikai függvények kiterjesztései[8].

5.4. A feladat finomítása

Célunk, hogy a modell eszközei segítségével a feladat specifikációját helyettesíteni tudjuk olyan feladatok specifikációival, amely feladatok megoldása esetén a rendelkezésre álló matematikai eszközökkel belátható az eredeti feladat megoldásának helyessége [Var 81, Fót Hor 91, Cha Mis 89, Bac Ser 90, Mor 87]. Arra törekszünk, hogy a megoldás előállításával párhuzamosan a megoldás helyességének bizonyítását is előállítsuk.

5.5. Megjegyzés. A lépésenkénti finomítás szokásos megfogalmazásától eltérően nem a megoldó programot finomítjuk [Bac Ser 90][9]. A feladat finomításának elve különbözik Morris [Mor 87], ill. Lamport [Lam 91] felfogásától is, mert ezekben a modellekben magát a programot is specifikációs eszköznek tekintik és ennek megfelelően finomítják a specifikációt.

A specifikáció finomításának leggyakoribb módja az állapottér bővítése, a régi és új komponensekre további, általában a korábbiaknál szigorúbb feltételek megfogalmazása.

[8] A 5.3. def. a szekvenciális modell [Fót 83, Fót 88] kiterjesztési definíciójának általánosítása.

[9] Egy program finomítása egy másiknak, ha minden olyan specifikációnak megfelel, amelyiknek az eredeti program is megfelelt [Bac Ser 90].

Azt, hogy egy feladat mikor finomítása egy másiknak egy rögzített állapottér felett, a program (6.15. def.) és a megoldás (7.1. def.) definíciójának felhasználásával indirekt úton adjuk meg. Ez a definíciós módszer alkalmas arra, hogy a feladatok lépésenkénti finomítása során az egyes lépéseink helyességét formálisan is igazoljuk[10]. A feladatok felett értelmezett finomítás relációt tehát a feladatok és programok között értelmezett megoldás reláció indukálja.

5.4. Definíció. *(Feladat finomítása)*
Azt mondjuk, hogy az F_1 feladat *finomítás*a az F_2 feladatnak, ha minden olyan S program, ami megoldása az F_1 feladatnak az megoldása az F_2 feladatnak is.

5.3. Példa. *(Feladat finomítása)*
Az alábbi specifikáció finomítása a (5.1.)-(5.3.) feltételekkel megadottnak.

$$(x = x') \in INIT_{x'} \tag{5.4.}$$
$$\uparrow\hookrightarrow \mathrm{FP}_{x'} \tag{5.5.}$$
$$\mathrm{FP}_{x'} \Rightarrow \forall i \in [1..n] : (x_i = \emptyset) \tag{5.6.}$$
$$inv_{x'}(\forall j \in [1,m] : (y_j \cup f_j(x_1,...,x_n) = f_j(x'_1,...,x'_n))) \tag{5.7.}$$
$$inv_{x'}(\forall j \in [1,m] : (y_j \cap f_j(x_1,...,x_n) = \emptyset)) \tag{5.8.}$$
$$inv_{x'}(\forall i,j \in [1,n] : (x'_i \setminus x_i) \cap x_j = \emptyset), \tag{5.9.}$$

ahol f elemenként feldolgozható.

Annak bizonyítása, hogy a fenti specifikáció valóban finomítása a (5.1.)-(5.3.) feltételekkel megadottnak megtalálható a 11. fejezetben. □

5.5. Definíció. *(Ekvivalens feladat)*
Azt mondjuk, hogy az F_1 feladat *ekvivalens* az F_2 feladattal, ha az F_1 finomítása az F_2-nek és az F_2 finomítása az F_1-nek.

5.6. Megjegyzés. *(Absztrakt feladat)*
Nevezzük a most bevezetett ekvivalenciareláció által létrejött ekvivalenciaosztályokat *absztrakt feladat*nak. A most bevezetett ekvivalenciareláció indukál egy homomorfizmust a feladatokról az absztrakt feladatokra[11].

Négyféle módon finomítjuk a feladat matematikai modelljét:

- az állapottér komponenseit finomítjuk és mint altereket tekintjük őket,
- az állapottér alterein fogalmazunk meg feladatokat,

[10] A programok felett értelmezett finomítás reláció is a megoldás fogalmához kötött [Bac Ser 90, Mor 87], és a finomítást támogató kalkulus alapja.

[11] Két absztrakt feladat pontosan akkor különbözik egymástól, ha az egyikhez található olyan megoldás, amelyik a másiknak nem megoldása.

- az állapotteret további komponensekkel bővítjük, a hozzájuk tartozó változókra kikötéseket teszünk,
- *állapottér transzformáció*t [Fót 86], vagy más néven koordinátatranszformációt [Dij Sch 89] alkalmazunk.

5.7. Megjegyzés. Ha el akarjuk dönteni, hogy egy feladat finomítása-e egy másiknak abban az esetben, amikor a két feladat állapottere különbözik, akkor a két feladat állapotterét feleltessük meg egy kiválasztott állapottér egy-egy alterének és adjunk meg egy-egy olyan függvényt, amelyik a kiválasztott altér hatványhalmazára a feladat állapotterének hatványhalmazát leképezi. Ezek a leképezések definiálják a feladatok megfelelőit az új állapottér alterein. A feladatokat ezek után kiterjeszthetjük a közös állapottérre. Legtöbbször a választott állapottér a két feladat egyikének állapottere, a másik feladat állapottere a közös tér egy altere, a leképezés pedig az identitás.

5.8. Megjegyzés. Feladatok specifikációjának finomításakor támaszkodunk a *nyitott specifikáció* technikájára. Az állapottér egy alterén definiált részfeladat környezeti feltételeiként olyan biztonságossági-, haladási- és fixpontfeltételeket adunk meg, amelyek az altér felett specifikált komponens és környezete által együttesen alkotott *zárt rendszer*től [Jär 92] elvárt viselkedésre vonatkoznak (9. fejezet) [Col 94, Cha Mis 89]. Az alterekre vonatkozó feltételeket megkülönböztetésül felső indexszel jelöljük, pl.: $\rhd_h^E$, $\mapsto_h^E$, $\hookrightarrow_h^E$, TERM_h^E, FP_h^E, inv_h^E. A részfolyamat egyes tulajdonságainak vizsgálatakor felhasználjuk a teljes rendszer (a külső környezet) ismert vagy feltételezett tulajdonságait [Cha Mis 89, Col 94].

A lépésenkénti finomítás során újabb és újabb részletekkel egészítjük ki a specifikációt, majd az utolsó lépésben előállítjuk a megoldó programot. A program előállítása általában egyszerű, a specifikáció finomítása nehezebb feladat. Minden egyes lépés után igazolnunk kell, hogy az új és részletesebb specifikációt megoldó program megoldja az eredeti feladatot is. A specifikáció finomítása során építjük be a megoldásba mindazt a tudást, amelyet a feladat elemzése során, vagy korábban szereztünk. A finomítás iránya kisebb vagy nagyobb mértékben befolyásolja, hogy milyen architektúrán implementálható hatékonyan a kapott megoldás és melyiken nem. Ezért akárcsak a szekvenciális programok levezetése során, a párhuzamos programok fejlesztésekor is előfordulhat, hogy visszatérünk egy korábban megfogalmazott specifikációhoz és más irányban folytatjuk a specifikáció finomítását.

6. Fejezet

Párhuzamos absztrakt program

Az absztrakt párhuzamos program a UNITY-ből ismert programfogalom relációs alapú megfogalmazása. Ahhoz, hogy eldönthessük, hogy egy program megold-e egy feladatot (7. fejezet), össze kell vetnünk a feladatot definiáló relációt a program viselkedését leíró relációval. A programhoz annak viselkedési relációját hozzárendelő leképezést tekinkthetjük úgy is, mint egy olyan szemantikai leképezést, amelynek absztrakciós szintje megegyezik az absztrakt feladat szemantikájának absztrakciós szintjével.

6.1. Az absztrakt program struktúrája

Az absztrakt program struktúrája nem eredményezheti, hogy olyan szinkronizációs kényszerek épüljenek be a megoldásba, amelyek feleslegesek, valódi párhuzamos architektúrán szükségtelenül lassítják a program futását. Ha a programot szekvenciális folyamatok halmazának tekintenénk, ahogy ezt pl. CSP-ben [Hoa 78] vagy Adában [ALRM 83] megszoktuk, akkor ezzel eleve végrehajtási sorrendet definiálnánk utasítások nagy részhalmazai felett. Ezért a párhuzamos programot utasítások halmaza segítségével definiáljuk.

6.1. Példa. *(Absztrakt program megadása)*
S::=$(SKIP, \{(x := x + 1, \text{ha } x \leq y \parallel y := y + x), z := x + y\})$.□

A nemdeterminisztikus végrehajtási sorrend korlátozására szinkronizációs feltételeket használunk (pl. termelő-fogyasztó esetén üres pufferből nem lehet fogyasztani). Az absztrakt program tehát szimultán feltételes értékadások (6.9. def.) [Fót 83, Hor 93] véges halmazával adható meg [Cha Mis 89], ahol az egyes értékadások jobboldalán függvénykompozíciók is szerepelhetnek (pl. kapcsos zárójellel megadott függvény).

6.1.1. A feltételes értékadás fogalma

6.1. Definíció. *(Utasítás)*
Az $s \subseteq A \times A^{**}$ relációt *utasítás*nak nevezzük, ha

- $\mathcal{D}_s = A$,
- $\forall a \in A \ : \ \forall \alpha \in s(a) \ : \ \alpha_1 = a$,
- $(\alpha \in \mathcal{R}_s \wedge \alpha \in A^{\infty}) \Rightarrow (\forall i \in N \ (\alpha_i = \alpha_{i+1} \rightarrow (\forall k(k > 0) : \alpha_i = \alpha_{i+k})))$,
- $(\alpha \in \mathcal{R}_s \wedge \alpha \in A^{*}) \Rightarrow (\forall i \ (1 \leq i < |\alpha|) : \alpha_i \neq \alpha_{i+1})$.

6.2. Definíció. *(Hatásreláció)*
A $p(s) \subseteq A \times A$ reláció az $s \subseteq A \times A^{**}$ utasítás *hatásrelációja*, ha

- $\mathcal{D}_{p(s)} = \{a \in A | s(a) \subseteq A^{*}\}$,
- $p(s)(a) = \{b \in A | \exists \alpha \in s(a) \ : \ \tau(\alpha) = b\}$,

ahol $\tau \ : \ A^{*} \rightarrow A$ függvény az $\alpha = (\alpha_1, \ldots, \alpha_n) \in A^{*}$ véges sorozathoz annak végpontját rendeli. $\tau(\alpha)::=\alpha_n$.

6.3. Definíció. .

- Azt mondjuk, hogy az $v_i : A_i$ változó *konstans* függvény az s utasításban, ha $\forall a \in A : \forall \alpha \in s(a) : (\forall \alpha_k \in \alpha : \alpha_{ki} = v_i(a))$.
- Azt mondjuk, hogy az s utasítás végrehajtása *biztosan nem változtatja meg* az $v_i : A_i$ változót, ha $\forall a \in \mathcal{D}_{p(s)} : p(s)(a)_i = v_i(a)$.

Elemi utasítás az értékadás és az üres utasítás.

6.4. Definíció. *(Üres utasítás, SKIP)*
*Üres*nek nevezzük, és SKIP-pel jelöljük azt az utasítást, amire $\forall a \in A : \ SKIP(a) = \{(a)\}$.

6.5. Definíció. *(Általános értékadás)*
Legyen $A = A_1 \times \ldots \times A_n$, $F = (F_1, \ldots, F_n)$, ahol $F_i \subseteq A \times A_i$. Az s utasítás *általános értékadás* [Fót 83] , ha
$s = \{(a, red(a,b)) | a, b \in A \ \wedge \ a \in \bigcap_{i \in [1,n]} \mathcal{D}_{F_i} \ \wedge \ b \in F(a)\} \cup$
$\{(a, (aaa \ldots)) | a \in A \ \wedge \ a \notin \bigcap_{i \in [1,n]} \mathcal{D}_{F_i}\}$, ahol
az $\alpha \in A^{**}$ *redukáltjá*nak nevezzük, és $red\,(\alpha)$-val jelöljük azt a sorozatot, amit úgy kapunk, hogy az α sorozat minden azonos elemekből álló véges részsorozatát a részsorozat egyetlen elemével helyettesítjük.

6.6. Definíció. *(Változó az értékadás baloldalán)*
Azt mondjuk, hogy a $v_i : A \longmapsto A_i$ változó az *értékadás baloldalán áll*, az értékadás *értéket ad a v_i változónak*, ha az $F_i \subseteq A \times A_i$ reláció nem egyenlő a v_i projekcióval [Fót 86], azaz az értékadás hatásrelációja megváltoztatja a v_i változót (4.21. def.). Az s értékadás baloldalán álló változók halmazát $VL(s)$-sel jelöljük. Az értékadás azoknak a változóknak ad értéket, amelyek a baloldalán állnak[1].

6.1. Megjegyzés. *(Szimultán értékadás)*
Az értékadás egyszerre több változó értékét is megváltoztathatja, ezért ún. *szimultán* értékadásról van szó.

6.7. Definíció. *(Egyszerű értékadás)*
Ha legfeljebb egy változó áll az értékadás baloldalán, akkor *egyszerű* értékadásról beszélünk.

6.8. Definíció. *(Változó az értékadás jobboldalán)*
Azt mondjuk, hogy a $v_i : A \longmapsto A_i$ változó az értékadás *jobboldalán áll*, ha az értékadás hatásrelációja nem független (4.20. def.) az A_i állapottérkomponenstől. Az s értékadás jobboldalán álló változók halmazát $VR(s)$-sel jelöljük.

6.9. Definíció. *(Feltételes értékadás)*
Legyen $A = A_1 \times \ldots \times A_n$, $F = (F_1, \ldots, F_n)$, ahol $F_i \subseteq A \times A_i$. Legyen $\lceil \pi_i \rceil ::= \mathcal{D}_{F_i}$. $F_i|_\uparrow$ az F_i *kiterjesztése a $\uparrow$ feltételre nézve*[2]: $F_i|_\uparrow(a) = F_i(a)$, ha $a \in \lceil \pi_i \rceil$ és $F_i|_\uparrow(a) = a_i$, különben. Az $F||_\uparrow = (F_1|_\uparrow, \ldots, F_n|_\uparrow)$ relációval megadott általános s értékadást *feltételes értékadás*nak nevezzük, ha $\forall a \in A : |p(s)(a)| < \omega$.

6.2. Megjegyzés. A v_i változóra vonatkozó egyszerű, determinisztikus, feltételes értékadást a $(v_i := F_i(v_1, .., v_n)$, ha $\pi_i(v_1, ..., v_n))$ alakban adjuk meg. Röviden: $((v_i := F_i$, ha $\pi_i)$-vel jelöljük. Szimultán, nemdeterminisztikus, feltételes értékadás megadható a $(v_i :\in F_i(v_1, .., v_n)$, ha $\pi_i \parallel (v_k :\in F_k(v_1, .., v_n)$, ha $\pi_k)$ alakban. Ha lehetséges, akkor sok változó esetén a $\mathop{\Big\|}_{i \in [1,n]}(...)$ rövidítést alkalmazzuk.

Ha a feltételes értékadásban szereplő valamelyik egyszerű értékadáshoz rendelt egyetlen feltétel egy $a \in A$ állapothoz *hamis* értéket rendel, akkor ez a 6.9. definíció szerint annak a rövid megfogalmazása, hogy az értékadás az a pontból indítva nem változtatja meg a baloldalon álló változó értékét. Ezáltal a feltételes értékadások mindenütt értelmezve vannak az állapottér felett.

---

[1] A 6.6. definíció azokat a változókat nevezi az értékadás baloldalán állónak, amelyek az értékadás végrehajtása során megváltozhatnak, azaz van olyan $a \in A$ állapot, amelyre $a_i = v_i(a) \neq v_i \circ F(a) = F_i(a)$. A definíció tehát független az értékadás szintaktikus alakjától.

[2] Általánosabban egy R *reláció π feltételre vonatkozó kiterjesztését* az alábbi módon definiálhatjuk:
Legyen B altere A-nak. $pr_B : A \longmapsto B$. A pr_B függvény az A-beli pontokhoz B-beli vetületüket rendeli hozzá [Fót 83]. $R \subseteq A \times B$. $R|_\pi ::= (R \cap (\lceil \pi \rceil \times B)) \cup \{(a, pr_B(a)) | a \in \lceil \pi \rceil \setminus \mathcal{D}_R\}$.

6.2. Állapotátmenetfák

Az absztrakt programot egy olyan bináris relációként definiáljuk, amelyik egy kezdeti feltételes értékadás hatásrelációja, illetve véges sok feltételes értékadás hatásrelációjának diszjunkt uniója által generált fák ekvivalenciaosztályait rendeli az állapottér egyes pontjaihoz (6.15. def.).

6.10. Definíció. *(Címkézett állapotátmenetfa)*
A címkézett állapotátmenetfa egy (r, N, V, L, S) rendezett ötös, ahol r a fa gyökere, N a csúcsok halmaza, $V \subseteq N \times N$ az élek halmaza, $L : N \longmapsto A$ a gráf csúcsaihoz állapotokat rendelő címkefüggvény, $S : V \longmapsto J$ az élekhez természetes számokat rendelő címkefüggvény, $\forall x \in N : (x, r) \notin V$ és pontosan egy út vezet r-ből minden $x \in N$ csúcsba.

6.11. Definíció. *(Izomorf állapotátmenetfák)*
Izomorfnak mondjuk a $G_1 = (r_1, N_1, V_1, L_1, S_1)$ és a $G_2 = (r_2, N_2, V_2, L_2, S_2)$ fát, ha van olyan $f : N_1 \longmapsto N_2$ bijekció, amelyre $\forall x \in N_1 : f(V_1(x)) = V_2(f(x)) \wedge L_1(x) = L_2(f(x))$ és $\forall x, y \in N_1 : (x, y) \in V_1 \equiv (f(x), f(y)) \in V_2 \wedge (x, y) \in V_1 : S_1(x, y) = S_2(f(x), f(y))$.

6.1. Tétel. *(Az izomorfia reláció ekvivalenciareláció)*
Az 6.11. definícióban megfogalmazott izomorfia reláció ekvivalenciareláció az A felett generált fák halmazán.

Biz.: A reláció reflexív, mert minden fához létezik adott tulajdonságú leképezés, az identitás. A reláció szimmetrikus, mert a bijekció inverze rendelkezik az adott tulajdonságokkal, ha a bijekció rendelkezett vele. Végül a reláció tranzitív, mert két adott tuljadonságú bijekció kompozíciója is rendelkezik a megkövetelt tulajdonságokkal. □

6.12. Definíció. *(Állapotátmenetfák ekvivalenciaosztályai)*
A^{***} jelölje az A felett generált fák ekvivalenciaosztályainak halmazát.

6.13. Definíció. *(Generált állapotátmenetfa)*
Legyenek az $R_0, R \subseteq A \times A$ relációk mindenütt értelmezve az A állapottér felett, azaz $\mathcal{D}_R = \mathcal{D}_{R_0} = A$. $J \subset \mathcal{N}_0$. Tetszőlges $a \in A$ pontra az (R_0, R) relációpár által az a ponthoz generált fának nevezzük a $GR(a) = (r, N_a, V_a, L_a, S_a)$ irányított fát, ha

- $L_a(r) = a$,
- $\forall x \in N_a \setminus \{r\} : L_a(V_a(x)) = R(L_a(x))$,
- $L_a(V_a(r)) = R_0(L_a(r)) = R_0(a)$.

Reprezentáljuk az R reláció által az a ponthoz generált gráfok ekvivalenciaosztályait annak egy elemével.

Legyen $S = (s_0, \{s_1, ...s_m\})$ egy s_0 feltételes értékadás és véges sok feltételes értékadás nem üres halmazának rendezett párja. J={1,..,m}, $UP(S)$ a $p(s_j)$ relációk diszjunkt uniója.

6.14. Definíció. *(Helyesen címkézett állapotátmenetfa)*
Az $(p(s_0), UP(S))$ relációpár által generált fa címkézése helyes, ha minden r-ből induló él címkéje 0 és minden j-vel címkézett élre amely a-val címkézett csúcsból b-vel címkézett csúcsba mutat teljesül, hogy $(a, b) \in p(s_j)$.

6.2. Lemma. *(Helyes címkézés és ekvivalencia)*
Ha egy állapotátmenetfa címkezése helyes, akkor a vele ekvivalens állapotátmenetfák címkézése is helyes.
Biz.: a 6.11. def. következménye. □

6.15. Definíció. *(Absztrakt program)*
Absztrakt programnak nevezzük az $UPG(S) \subseteq A \times A^{***}$ relációt, ha az állapottér pontjaihoz a $(p(s_0), UP(S))$ relációpár által generált azon állapotátmenetfák ekvivalenciaosztályait rendeli, amelyek címkézése helyes. Az $S = (s_0, \{s_1, ...s_m\})$ által definiált $UPG(S)$ programot röviden S-sel jelöljük, és azt mondjuk, hogy $s_j \in S$, ha $j \in J$.

6.3. Megjegyzés. *(Műveleti szemantika)*
A 6.15. definíció megadja a párhuzamos absztrakt program szemantikai jelentését. Ez a szemantika műveleti jellegű, a programnak megfelelő gráf valójában a műveleti szemantika címkézett állapotátmenet gráfjával azonosítható (1. fejezet). Megjegyezzük, hogy ez a szemantika olyan programok között is különbséget tesz, amelyek egyaránt megoldásai (7.1. def.) ugyanannak a feladatnak, de más-más végrehajtási utakat definiálnak.

6.16. Definíció. *(Végrehajtási út)*
A $b \in UPG(S)(a)$ ekvivalenciosztály reprezentánsának bármelyik útját *végrehajtási út*nak nevezzük.

6.17. Definíció. *(Elérhető állapotok halmaza)*
Az $UPG(S)(a)$ halmaz elemeinek végrehajtási útjain elhelyezkedő csúcsok címkéinek halmazát jelöljük $E(S)(a)$-val. $E(S)(a)$ az *a állapotból elérhető állapot*ok halmaza[3].

6.18. Definíció. *(Absztrakt program változói)*
Jelöljük $VL(S)$-sel az S program utasításainak baloldalán álló változók (6.6. def.) halmazát, azaz $VL(S) ::= \bigcup_{s \in S} VL(s)$. Jelöljük a jobboldalon álló változók (6.8. def.) halmazát $VR(S)$-sel. $V(S) ::= VL(S) \cup VR(S)$.

[3] Ezen állapotok az absztrakt program állapotátmenetfájában érhetőek el, de valamely ütemezési kikötés mellett a konkrét program által már nem feltétlenül elérhetőek.

Feltételezzük, ha több processzor hajtja végre a konkrét programot, akkor ez hatékonysági szempontoktól eltekintve hatásában megegyezik azzal, mintha egyetlen processzor válogatott volna valamilyen nemdeterminisztikus sorrendben az utasításhalmaz elemei közül. Megengedjük ugyan, hogy két vagy több processzor időben átfedve hajtsa végre ugyanazon utasítás különböző elemi lépéseit vagy különböző utasításokat, de az így kapott eredménynek meg kell egyeznie valamelyik eredménnyel azok közül, amelyet valamelyik végrehajtási út mentén egyetlen processzor állított volna elő. *Feltételezzük* tehát, hogy az absztakt programot oly módon implementáljuk, hogy elemi építőköveire, a szimultán feltételes értékadásokra párhuzamos végrehajtás esetén teljesül a *sorbarendezhetőség* [Lam Lyn 90] követelménye. Egy-egy sorrendet egy-egy végrehajtási út ír le.

6.4. Megjegyzés. Az 6.15. definíció alapján megállapíthatjuk, hogy a modellben szimultán feltételes értékadások *valós aszinkron párhuzamos* végrehajtását nem tudjuk kifejezni[4]. A modell programfogalma *összefésüléses* szemantikán (1. fejezet) alapszik. A szinkron párhuzamos végrehajtás leírására a szimultán értékadás alkalmas.

Az utasítások halmazát sok esetben halmazműveletek segítségével állítjuk majd elő a megoldás logikai struktúrájának megfelelő modulokból. Egy-egy modul leírhatja például egy-egy objektum viselkedését [Cha Mis 89, Sin 91]. A modulok uniójaként vagy szuperpozíciójaként kapott program [Cha Mis 89] utasításait hatékonysági szempontok figyelembevételével képezhetjük le logikai vagy fizikai processzorokra. A modul tehát programtervezési, a folyamat pedig implementációs fogalom.

6.2.1. Utasítások kiterjesztése, szuperpozíciója

Definiáljuk az A állapottér egy altere felett definiált s utasítás s' kiterjesztettjét oly módon, hogy abban a kiegészítő altér változói ne álljanak egyetlen utasítás baloldalán és jobboldalán sem és a kiterjesztett utasítás A_1-re vett vetülete éppen s legyen [Fót 83, Fót 88].

6.19. Definíció. *(Utasítás kiterjesztése)*
Legyen B altere az A állapottérnek, B' a B altér kiegészítő altere az A-ra.
Az $s \subseteq B \times B^{**}$ *utasítás kiterjesztése* A-ra:
$s' = \{(a,\alpha) \in A \times A^{**} \mid (pr_B(a), pr_B(\alpha)) \in s \wedge \forall i \in \mathcal{D}_\alpha : \ pr_{B'}(\alpha_i) = pr_{B'}(a)\}$.

6.3. Lemma. Legyen az A_1 tér az A állapottér altere. Legyen az A_1 altér felett definiált s utasítás kiterjesztése az s' utasítás. Ekkor: $p(s) = pr_{A_1}(p(s'))$.

[4] Valós párhuzamosság esetén összetett feladatok megoldását általában nem lehet modulokból előállítani (10. fejezet)

Biz.: Az utasítás kiterjesztésének definíciója szerint $pr_{A_1}(s') = s$, így $p(pr_{A_1}(s')) = p(s)$. Felhasználva, hogy egy utasítás hatásrelációjának (6.2. def.) kiszámítása és a vetítés kommutatív, a lemma állításához jutunk. □

Gyakran alkalmazzuk egyes utasítások definíciójánál azt a módszert, hogy egy meglévő és ismert hatásrelációjú feltételes értékadást módosítunk.

6.20. Definíció. *(Értékadás kiegészítése feltétellel)*
$((s_j), \text{ha } \pi) ::= \|_{i\in[1,n]}(v_i :\in F_{j_i}(v_1,..,v_n), \text{ha } \pi_{j_i} \wedge \pi)$.

6.4. Lemma. $p((s_j), \text{ha } \pi) = (p(s) \cap (\lceil \pi \rceil \times A)) \cup (id_A \cap (\lceil \neg\pi \rceil \times A))$.
Biz.: A 6.20. def. közvetlen következménye.□

6.21. Definíció. *(Feltételes értékadások szuperpozíciója)*
Legyen s_1 és s_2 azonos állapottéren adott két feltételes értékadás és legyen $VL(s_2) \cap V(s_1) = \emptyset$. Ekkor $s_1 \| s_2 ::=$
$\|_{v_i \notin VL(s_2)}(v_i :\in F_{1i}(v_1,..,v_n), \text{ha } \pi_{1i}) \; \|_{v_i \in VL(s_2)}(v_i :\in F_{2i}(v_1,..,v_n), \text{ha } \pi_{2i})$.

6.5. Megjegyzés. Tegyük fel, hogy $u : A \mapsto A_{i'}$ nem szerepel az s_j értékadás baloldalán (6.6. def.).
Ekkor az előző definíció értelmében speciális esetként definiálhatjuk az s_j feltételes értékadás és a $(u := F_{i'}, \text{ha } \pi_{j_{i'}})$ egyszerű értékadás szuperpozícióját: $s_j \| (u := F_{i'}, \text{ha } \pi_{j_{i'}}) ::=$
$\|_{i\in([1,n]\setminus\{i'\})}(v_i :\in F_{j_i}(v_1,..,v_n), \text{ha } \pi_{j_i} \wedge \pi) \| (u := F_{i'}, \text{ha } \pi_{j_{i'}})$.

6.5. Lemma. *(Szuperpozíció hatásrelációja)*
Legyen s_1 és s_2 azonos állapottéren adott két feltételes értékadás és legyen $VL(s_2) \cap V(s_1) = \emptyset$. Ekkor $p(s_1 \| s_2) = p(s_1) \circ p(s_2)$.

Biz.: Jelöljük $id \subseteq A \times A$-val azt a relációt, amelyik minden ponthoz önmagát rendeli ($id_i \subseteq A_i \times A_i$). Legyen $\forall a \in A : p(s)_i(a) ::= (p(s)(a))_i$.
$\forall v_i \notin VL(s_2) : p(s_2)_i = id_i$, így $\forall v_i \notin VL(s_2) : (p(s_1) \circ p(s_2)(a))_i = (p(s_1) \circ id)_i = p(s_1)_i$. $\forall v_i \in VL(s_2) : v_i \notin V(S)$, tehát $p(s_1)_i = id_i$, így $\forall v_i \in VL(s_2) : (p(s_1) \circ p(s_2)(a))_i = (id \circ p(s_2))_i = p(s_2)_i$. □

6.2.2. Program kiterjesztése

Definiáljuk az S_1 program A-ra való kiterjesztettjét utasításonként.

6.22. Definíció. *(Program kiterjesztése)*
Legyen az A_1 és A_2 tér az A állapottér két egymást kiegészítő altere. Legyen S program az A_1 altér, S' az A tér felett definiálva. Az S' programot az S program A-ra való kiterjesztésének nevezzük, ha minden utasítása kölcsönösen egyértelműen megfeleltethető az S program egy utasítása kiterjesztésének.

6.6. Megjegyzés. A 6.3. lemma és az absztrakt program definíciója alapján könnyen belátható, hogy
$\forall a \in A : pr_{A_2}(E(S')(a)) = \{pr_{A_2}(a)\}$ és $pr_{A_1}(UPG(S')) = UPG(S)$.

6.3. Pártatlan ütemezés fogalma

A megoldás definíciója kimondja majd, hogy a feladatban megfogalmazott feltételeknek csak azokra a végrehajtási utakra kell teljesülni, amelyekre teljesül a feltétlenül pártatlan ütemezés axiómája.

6.23. Definíció. *(Feltétlenül pártatlan ütemezés)*
Egy végrehajtási útról azt mondjuk, hogy teljesül rá a *feltétlenül pártatlan ütemezés* axiómája, ha az út mentén a feltételes értékadások halmazából minden utasítás végtelen sokszor kerül kiválasztásra, azaz a címkefüggvény az út mentén a J indexhalmaz minden elemét végtelen sokszor rendeli az élekhez[5].

Ha a konkrét, adott architektúrára leképezett programra teljesül, hogy feltétlenül pártatlan ütemezés mellett kerül végrehajtásra, akkor a többi utat valóban nem kell figyelembe venni a megoldás helyessége szempontjából. Ebben az esetben a program működése nemdeterminisztikus módon kiválasztott transzformációk iterációjával írható le, ahol a nemdeterminisztikusság véges de nem korlátos hasonló értelemben, ahogy relációk nem korlátos lezártjáról beszéltünk [Fót 83, Hor 90]. Több utasítás közül ugyanaz az utasítás véges, de nem korlátos sokszor nem kerül kiválasztásra közvetlenül egymás után.

Utasítások pártatlan kiválasztására sokféle feltételt lehet megfogalmazni [Fra 86, And 91], ezek közül az egyik leggyengébb a pártatlan ütemezés axiómája. Így feltétlenül pártatlan ütemezés mellett jól működő programok az általában szigorúbb ütemezési feltételek esetén is megoldják a feladatot.

Az ún. *őrfeltétel*ek [Dij 75, Hoa 78, And 91] hasonlítanak a feltételes értékadásokban szereplő π_i feltételekhez. A feltételes értékadások hatásrelációi azonban akkor is definiáltak, ha az adott állapotra a feltétel nem teljesül. Ezért a generált fában mindig megjelenik az értékadás hatásrelációjának megfelelő él. A hamis őrfeltételű műveletek azonban nem generálnak éleket.

- *Feltétlenül pártatlan*nak nevezünk egy ütemezést, ha minden őrfeltételhez nem kötött és végrehajtásra váró elemi művelet előbb utóbb végrehajtásra kerül (6.23. def.).

[5] Megkövetelhetünk azonban kevesebbet is, pl.: hogy a feladatban megfogalmazott feltételeknek csak azokra a végrehajtási utakra kell teljesülni, amelyekre teljesül az utófeltételekre vonatkozóan pártatlan ütemezés axiómája.

- *Gyengén pártatlan* egy ütemezés, ha feltétlenül pártatlan és minden olyan művelet, amelynek őrfeltétele igazzá válik és igaz is marad, előbb-utóbb végrehajtásra kerül.

- *Szigorúan pártatlan* egy ütemezés, ha feltétlenül pártatlan és minden olyan elemi művelet, amely végrehajtásra vár és őrfeltétele végtelen sokszor igaz, előbb-utóbb végrehajtásra kerül.

6.24. Definíció. *(Utófeltételekre pártatlan ütemezés)*
Nem teljesül egy végrehajtási útra az *utófeltételekre vonatkozóan pártatlan*[6] *ütemezés* axiómája, ha
- egy adott pontjától kezdődően minden pontjában kiválasztható olyan utasítás, amely az adott pontnak megfelelő állapotból egy adott logikai függvény igazsághalmazába visz és
- sohasem kerül ilyen utasítás kiválasztásra.

A feltételes értékadások halmazaként definiált absztrakt program nem tartalmaz őrfeltételeket. A továbbiakban feltételezzük, hogy az implementált program végrehajtása feltétlenül pártatlan.

6.7. Megjegyzés. Belátható, hogy feltétlenül pártatlan ütemezéssel nem pártatlan ütemezés is modellezhető [Cha Mis 89].

6.4. Az absztrakt program tulajdonságai

Az absztrakt programok tulajdonságait az állapottér hatványhalmaza felett értelmezett relációkkal írjuk le.

6.4.1. A leggyengébb előfeltétel és általánosítása

Az absztrakt programok jellemzésekor támaszkodunk a leggyengébb előfeltétel [Dij 76, Fót 83], a legszigorúbb utófeltétel [Lam 90], ill. a monoton leképezések fixpontjának (3. fejezet) fogalmára.

6.25. Definíció. *(Leggyengébb előfeltétel, legszigorúbb utófeltétel)*
Legyen s egy utasítás, Q, R pedig logikai függvények az A állapottér felett. A $wp(s,R) : A \longmapsto \mathcal{L}$ logikai függvény az R utófeltétel s utasításra vonatkozó *leggyengébb előfeltétele*, ahol
$\lceil wp(s,R) \rceil ::= \{a \in \mathcal{D}_{p(s)} \mid p(s)(a) \subseteq \lceil R \rceil\}$.
Az $sp(s,Q) : A \longmapsto \mathcal{L}$ logikai függvény a Q előfeltétel *legszigorúbb utófeltétele* az s-re nézve, ahol $\lceil sp(s,Q) \rceil ::= p(s)(\lceil Q \rceil)$.

[6] gyengén pártatlan

6.6. Lemma. *(Leggyengébb előfeltétel alaptulajdonságai)*

(1) $wp(s, \downarrow) = \downarrow$ (csoda kizárásának elve),
(2) Ha $\mathcal{D}_{p(s)} = \lceil\uparrow\rceil$, akkor $wp(s, \uparrow) = \uparrow$,
(3) $\lceil wp(s, R)\rceil = \lceil R \circ p(s)\rceil$ (utófeltételbe helyettesítés módszere),
(4) Ha $P \Rightarrow Q$, akkor $wp(s, P) \Rightarrow wp(s, Q)$ (monotonitás),
(5) $wp(s, Q) \vee wp(s, R) \Rightarrow wp(s, Q \vee R)$ (gyenge additivitás),
(6) $wp(s, Q) \wedge wp(s, R) = wp(s, Q \wedge R)$ (multiplikativitás).

Biz.: Az állítások közvetlenül a 6.25. definícióból következnek. ((1), (3), (4), (5), (6) bizonyítása megtalálható [Fót 83, Fót Hor 91]-ben.) □

6.8. Megjegyzés. A lemma (2)-es állítása az absztrakt programban előforduló utasításokra, a feltételes értékadásokra (6.9. def.) mindig teljesül.

6.7. Lemma. *(Kiterjesztés és leggyengébb előfeltétel)*
Legyen R az A_1 altéren definiált logikai függvény, R' pedig az R logikai függvény kiterjesztése A-ra. s' jelölje az s utasítás A-ra vonatkozó kiterjesztését. Legyen a' egy tetszőleges olyan pont, amelyre $a = pr_{A_1}(a')$, Ekkor: $a \in wp(s, R) \Longleftrightarrow a' \in (wp(s', R'))$ és $a \in sp(s, R) \Longleftrightarrow a' \in (sp(s', R'))$

Biz.: $a' \in wp(s', R') \Longleftrightarrow a' \in \mathcal{D}_{p'(s)}$ és $p(s')(a) \subseteq R' \Longleftrightarrow$ (a 6.3. lemma alkalmazásával) $\Longleftrightarrow a \in \mathcal{D}_{p(s)}$ és $p(s)(a) \subseteq R \Longleftrightarrow a \in wp(s, R)$. A legszigorúbb utófeltételre vonatkozó állítás ugyanígy bizonyítható. □

6.1. Következmény. $wp(s, R)' = (wp(s', R'))$ és $sp(s, Q)' = (sp(s', Q'))$.

6.8. Lemma. *(Kiegészítés és leggyengébb előfeltétel)*
Ha $P \Rightarrow wp(s, Q)$, akkor $P \vee \neg\pi \Rightarrow wp((s, \text{ ha } \pi), Q \vee \neg\pi)$.

Biz.: Ha $a \in P \wedge \pi$, akkor $p(s, \text{ ha } \pi)(a) = p(s)(a) \subseteq \lceil Q\rceil$ (6.4. lemma). Ha $a \in \neg\pi$, akkor $p(s, \text{ ha } \pi)(a) = id_A(a) \subseteq \neg\pi$. □

6.9. Lemma. *(Szuperpozíció és leggyengébb előfeltétel)*
Legyen Q, R egy-egy logikai függvény és $VR(Q) \cap VL(s_1) = \emptyset$, $VR(R) \cap VL(s_1) = \emptyset$. Ekkor $wp(s_1, Q) = Q$, ill. ha $R \Rightarrow wp(s, Q)$, akkor $R \Rightarrow wp(s \| s_1, Q)$.

Biz.: A 6.6. lemmát többször alkalmazva bizonyítunk. A feltétel szerint $Q \circ p(s_1) = Q$, azaz $wp(s_1, Q) = Q \circ p(s_1) = Q$. A feltétel szerint $p(s_1)(\lceil R\rceil) = \lceil R\rceil$ és $\lceil R\rceil \subseteq \lceil Q \circ p(s)\rceil$, így $p(s_1)(\lceil R\rceil) \subseteq \lceil Q \circ p(s)\rceil$, azaz $p(s) \circ p(s_1)(\lceil R\rceil) \subseteq \lceil Q\rceil$. A 6.5. lemma szerint $p(s \| s_1, Q) = p(s) \circ p(s_1)$, így éppen a lemma állítását kaptuk. □

A továbbiakban jelöljön S egy absztrakt programot (6.15. def.), az állapottér legyen $A ::= \overset{\times}{_{i\in[1..n]}} A_i$. $S = (s_0, \{s_1, ...s_m\})$, ahol s_0 és $\forall s_j \in S$ egy (szimultán, nemdeterminisztikus) feltételes értékadás.
$s_j : \overset{\|}{_{i\in[1,n]}}((v_i :\in F_{j_i}(v_1, .., v_n), \text{ ha } \pi_{j_i})$.

Általánosítjuk a leggyengébb előfeltétel fogalmát:

6.26. Definíció. *(Leggyengébb előfeltétel általánosítása)*
$wp(S, R) ::= \forall s \in S : wp(s, R)$.
$wpa(S, R) ::= \exists s \in S : wp(s, R)$ ($wpa(S, R)$ az ún. "angyali" leggyengébb előfeltétel [Mor 90]).[7]

6.10. Lemma. *(Általánosított leggyengébb előfeltétel alaptulajdonságai)*

(1) $wp(S, \downarrow) = \downarrow$,
(2) $wp(S, \uparrow) = \uparrow$,
(3) Ha $P \Rightarrow Q$, akkor $wp(S, P) \Rightarrow wp(S, Q)$,
(4) $wp(S, Q) \vee wp(S, R) \Rightarrow wp(S, Q \vee R)$,
(5) $wp(S, Q) \wedge wp(S, R) = wp(S, Q \wedge R)$.

Biz.: Az állítások az a 6.26. definícióból, a 6.6. lemmából és a logikai és függvénykompozíció asszociativitásából és kommutativitásából következnek. □

6.4.2. Invariánsok és elérhető állapotok

Jelöljük $\text{inv}_S(\lceil Q \rceil)$-val azon P logikai függvények igazsághalmazainak halmazát, amelyek az S programra nézve invariánsok[8], ha a program $\lceil Q \rceil$-beli állapotból indul. A $\lceil P \rceil \in \text{inv}_S(\lceil Q \rceil)$-t röviden $P \in \text{inv}_S(Q)$-val jelöljük. Jelölje $\text{INV}_S(Q)$ azon P logikai függvények konjunkcióját, amelyekre $P \in \text{inv}_S(Q)$[9].

6.27. Definíció. *(Invariáns tulajdonság)*
$inv_S : \mathcal{P}(A) \longmapsto \mathcal{P}(\mathcal{P}(A))$. $\text{inv}_S(\lceil Q \rceil) \subseteq \mathcal{P}(A)$.
$\text{inv}_S(\lceil Q \rceil) ::= \{\lceil P \rceil \mid sp(s_0, Q) \Rightarrow P \text{ és } P \Rightarrow wp(S, P)$.

6.9. Megjegyzés. *($\text{inv}_S(Q)$ nem üres)*
A definíció szerint $\forall S, Q :\uparrow \in \text{inv}_S(Q)$.

[7] A *wpa* leképezés definíciója hasonlít a J.R. Rao által egyes utasításokra definiált *wpp* operátor definíciójához, de attól eltérően absztrakt programra vonatkozik. A *wpp* operátort a UNITY valószínűségi alapon nemdeterminisztikus kiegészítése során használják [Rao 95].

[8] Szigorú invariáns [Pra 94].

[9] $\text{INV}_S(Q)$ a legszigorúbb invariáns [Pra 94].

6.11. Lemma. *(Invariások konjunkciója)*
$inv_S(Q)$ zárt a $\wedge$ műveletre nézve [Pra 94].

Biz.: Legyen $P, K \in inv_S(Q)$. 6.27. def. felhasználásával: $sp(s_0, Q) \Rightarrow P$ és $sp(s_0, Q) \Rightarrow K \Longrightarrow sp(s_0, Q) \Rightarrow P \wedge K$. $P \Rightarrow wp(S, P)$ és $K \Rightarrow wp(S, K) \Longrightarrow P \wedge K \Rightarrow wp(S, P) \wedge wp(S, K)$. A 6.10. lemma szerint: $wp(S, P) \wedge wp(S, K) = wp(S, P \wedge K)$, így $P \wedge K \Rightarrow wp(S, P \wedge K)$. □

6.2. Következmény. *(Legszigorúbb invariáns)*
Az $inv_S(Q)$ halmaznak egyértelműen létezik legkisebb (3.13. def.) eleme és az éppen $INV_S(Q)$.

6.28. Definíció. *(Legszigorúbb invariáns)*
Az $inv_S(Q)$ halmaz legkisebb elemét, $INV_S(Q)$-t *a legszigorúbb invariáns*nak nevezzük.

6.12. Tétel. *(Invariáns konjunkciója kezdetben igaz állítással)*
Ha $sp(s_0, Q) \Rightarrow J$, $I \in inv_S(Q)$ és $I \wedge J \Rightarrow wp(S, J)$, akkor $I \wedge J \in \mathrm{inv}_S(Q)$ ($I \wedge J$ invariáns).

Biz.: Ha $I \in \mathrm{inv}_S(Q)$, akkor $sp(s_0, Q) \Rightarrow I$. Így $sp(s_0, Q) \Rightarrow I \wedge J$ az első feltétel szerint. Ha $I \in inv_S(Q)$, akkor $I \Rightarrow wp(S, I)$, a harmadik feltétel és 6.10. lemma felhasználásával: $I \wedge J \Rightarrow wp(S, I) \wedge wp(S, J) = wp(S, I \wedge J)$. □

6.10. Megjegyzés. *(Invariáns tulajdonság felbontása)*
A 6.11. tétel nem megfordítható. Ha $I \wedge J$ invariáns, akkor nem feltétlenül igaz, hogy akár I, akár J invariáns lenne. Annak bizonyítása, hogy egy $P = \bigwedge_{i \in [1..n]} P_i$ állítás invariáns tulajdonság, a 6.12. tétel segítségével azonban sok esetben részekre bontható pl. úgy, hogy

- belátjuk, hogy P_1 invariáns,
- megmutatjuk, hogy $\forall i : P_i$ kezdetben igaz,
- igazoljuk, hogy $\forall i : P^i \Rightarrow wp(S, P_i)$, ahol $P^i ::= \bigwedge_{j \in [1..i-i]} P_j$.

6.29. Definíció. *(Mindig igaz)*
$true_S : \mathcal{P}(A) \longmapsto \mathcal{P}(\mathcal{P}(A))$. $true_S(\lceil Q \rceil) \subseteq \mathcal{P}(A)$.
$true_S(\lceil Q \rceil) ::= \{\lceil P \rceil \mid \mathrm{INV}_S(Q) \Rightarrow P\}$.

6.11. Megjegyzés. *($true_S(Q)$ nem üres)*
A definíció szerint $\forall S, Q : \uparrow \in true_S(Q)$.

Azokat a logikai függvényeket, amelyek igazsághalmaza eleme a $true_S(Q)$ halmaznak, a Q-ból elérhető állapotok felett a program futása során *mindig igaz* állításoknak nevezzük[10].

6.13. Lemma. *(Az invariáns mindig igaz)*
$inv_S(Q) \subseteq true_S(Q)$.
Biz.: A 6.28. következmény szerint, ha $P \in \mathrm{inv}_S(Q) \Longrightarrow \mathrm{INV}_S(Q) \Rightarrow P$. □

6.14. Lemma. *(Mindig igaz állítások konjunkciója mindig igaz)*
Ha $J \in true_S(Q)$ és $I \in inv_S(Q)$, akkor $I \wedge J \in true_S(Q)$.

Biz.: Ha $J, I \in true_S(Q)$, akkor $sp(s_0, Q) \Rightarrow J$ és $sp(s_0, Q) \Rightarrow I$. Így $sp(s_0, Q) \Rightarrow I \wedge J$. Ha $I, J \in true_S(Q)$, akkor $\mathrm{INV}_S(Q) \Rightarrow J$ és $\mathrm{INV}_S(Q) \Rightarrow I$. Így $\mathrm{INV}_S(Q) \Rightarrow I \wedge J$. □

6.3. Következmény. Mindig igaz és invariáns konjunkciója mindig igaz.

6.4. Következmény. *(A legszigorúbb mindig igaz)*
A $true_S(Q)$ halmaznak egyértelműen létezik legkisebb eleme és az éppen $INV_S(Q)$, *a legszigorúbb invariáns.*

$INV_S(Q)$ tehát az a legszűkebb igazsághalmazú logikai függvény, amelyiknek igazsághalmazát a program soha nem hagyja el a $\lceil Q \rceil$-ból indulva. Így kimondhatjuk az alábbi tételt:

6.15. Tétel. *($\mathrm{INV}_S(Q)$ és a Q-ból elérhető állapotok)*
$\mathrm{INV}_S(Q)$ igazsághalmaza éppen a $\lceil Q \rceil$-ból *elérhető állapot*ok (6.17. def.) halmaza [Pra 94].

Nem minden esetben lesz egy mindig igaz állítás és egy invariáns konjunkciója invariáns, hiszen pl. a $\uparrow$ is invariáns és konjunkciója egy mindig igaz, de nem invariáns állítással nem eredményezhet invariáns állítást.

6.16. Lemma. *(Mindig igaz és invariáns konjunkciója)*
Ha $J \in true_S(Q)$, $I \in inv_S(Q)$ és $I \wedge J \Rightarrow wp(S, J)$, akkor $I \wedge J \in \mathrm{inv}_S(Q)$.
Biz.: Ha $J \in true_S(Q)$, akkor $sp(s_0, Q) \Rightarrow J$. Így az állítás következik a 6.12. tételből. □

[10] A mindig igaz állításokat gyenge invariánsoknak is nevezik [San 91, Pra 94]

6.4.3. Biztonságossági tulajdonságok

Jelöljük $\rhd_S$-sel azon P, Q logikai függvények igazsághalmazai rendezett párjainak halmazát, amelyekre az S program végrehajtása során igaz, hogy P *stabil feltéve, hogy* nem Q. Jelölés: $P \rhd_S Q ::= (\lceil P \rceil, \lceil Q \rceil) \in \rhd_S$.

6.30. Definíció. *(Stabil feltéve, hogy – tulajdonság)*
$\rhd_S \subseteq \mathcal{P}(A) \times \mathcal{P}(A)$.
$\rhd_S ::= \{(\lceil P \rceil, \lceil Q \rceil) \mid (P \wedge \neg Q \Rightarrow wp(S, (P \vee Q))\}$[11]

6.12. Megjegyzés. *(Stabil tulajdonság)*

Azt mondjuk, hogy S rendelkezik a P stabil tulajdonsággal, ha $P \rhd_S \downarrow$.

6.17. Lemma. *($\rhd_S$ és a stabil tulajdonságok)*
Ha $P \rhd_S Q$ és $K \rhd_S \downarrow$, akkor $P \wedge K \rhd_S Q \wedge K$.
Biz.: A 6.30. def. alapján $P \wedge \neg Q \Rightarrow wp(S, P \vee Q)$ és $K \wedge \uparrow \Rightarrow wp(S, K)$. Ebből a 6.10. lemma alkalmazásával: $P \wedge K \wedge \neg Q \Rightarrow wp(S, P \vee Q) \wedge wp(S, K) = wp(S, (P \vee Q) \wedge K) = wp(S, (P \wedge K) \vee (Q \wedge K))$. A 6.30. def. alkalmazásával a kívánt állításhoz jutunk. □

6.18. Lemma. *(Az invariánsok stabil tulajdonságok)*
Ha $\exists Q : K \in inv_S(Q)$, akkor $K \rhd_S \downarrow$.
Biz.: A 6.27. és 6.30. definíciók közvetlen következménye. □

6.19. Tétel. *($\rhd_S$ és az invariánsok szigoríthatósága)*
Ha $(P \wedge J) \rhd_S (Q \wedge J)$ és $J \in inv_S(Q)$ és $K \in inv_S(Q)$, akkor $J \wedge K \in inv_S(Q)$ és $(P \wedge J \wedge K) \rhd_S (Q \wedge J \wedge K)$[12].
Biz.: Az állítás első része következik a 6.11. lemmából. Az állítás második részét pedig a 6.18. és 6.17. lemma alkalmazásával kapjuk. □

6.20. Tétel. *($\rhd_S$ és a legszigorúbb invariáns)*
Ha $(P \wedge J) \rhd_S (R \wedge J)$ és $J \in inv_S(Q)$, akkor
$P \wedge \mathrm{INV}_S(Q) \rhd_S R \wedge \mathrm{INV}_S(Q)$.
Biz.: $\mathrm{INV}_S(Q) \in \mathrm{inv}_S(Q)$ miatt alkalmazható a 6.19. tétel. $\mathrm{INV}_S(Q)$ def. alapján viszont $INV_S(Q) \wedge J = INV_S(Q)$. □

[11] A $\rhd_S$ definíciója megfelel a [Cha Mis 89]-ben adott *unless* fogalmának.
[12] A tétel Prasetya tételének relációs átfogalmazása [Pra 94].

6.4.4. Haladási tulajdonságok

Jelöljük $\mapsto_S$-sel azon P, Q logikai függvények igazsághalmazai rendezett párjainak halmazát, amelyekre az S program végrehajtása során igaz, hogy P stabil feltéve, hogy nem Q és van egy olyan $s_j \in S$ feltételes értékadás, amely garantálja, hogy a $\lceil P \rceil$-ből $\lceil Q \rceil$-ba jutunk. Jelölés: $P \mapsto_S Q ::= (\lceil P \rceil, \lceil Q \rceil) \in \mapsto_S$.

6.31. Definíció. *(Biztosítja tulajdonság)*
$\mapsto_S \subseteq \mathcal{P}(A) \times \mathcal{P}(A)$.
$\mapsto_S ::= \{(\lceil P \rceil, \lceil Q \rceil) \mid (P, Q) \in \rhd_S \wedge \exists j \in J : (P \wedge \neg Q \Rightarrow wp(s_j, Q))\}$[13]

6.21. Lemma. *($\mapsto_S$ és a stabil tulajdonság)*
Ha $P \mapsto_S Q$ és $K \rhd_S \downarrow$, akkor $P \wedge K \mapsto_S Q \wedge K$.

Biz.: A 6.31. def. és a 6.17. lemma alapján elegendő azt bizonyítani, hogy $\exists s \in S : (P \wedge K \wedge \neg(Q \wedge K) \Rightarrow wp(s, Q \wedge K)$. A feltételekből tudjuk, hogy $\exists s \in S : P \wedge \neg Q \Rightarrow wp(s, Q)$. Válasszunk egy ilyen s utasítást. K stabil, ezért erre az $s \in S$-re is: $K \Rightarrow wp(s, K)$. Az s-re vonatkozó két állításból logikai és művelet és egyszerűsítés után a $P \wedge K \wedge \neg Q \Rightarrow wp(s, Q \wedge K) \wedge wp(s, K)$ eredményre jutunk. A leggyengébb előfeltétel ismert tulajdonsága alapján (6.6. lemma) $wp(s, Q \wedge K) \wedge wp(s, K) = wp(S, Q \wedge K \wedge K) = wp(s, Q \wedge K)$. □

6.22. Tétel. *($\mapsto_S$ és az invariánsok szigoríthatósága)*
Ha $(P \wedge J) \mapsto_S (Q \wedge J)$ és $J \in inv_S(Q)$ és $K \in inv_S(Q)$, akkor $J \wedge K \in inv_S(Q)$ és $(P \wedge J \wedge K) \mapsto_S (Q \wedge J \wedge K)$[14].
Biz.: Az állítás első része következik a 6.11. lemmából. Az állítás második részét pedig a 6.18. és 6.21. lemma alkalmazásával kapjuk. □

6.23. Tétel. *($\mapsto_S$ és a legszigorúbb invariáns)*
Ha $(P \wedge J) \mapsto_S (R \wedge J)$ és $J \in inv_S(Q)$, akkor
$P \wedge \mathrm{INV}_S(Q) \mapsto_S R \wedge \mathrm{INV}_S(Q)$.
Biz.: 6.20. tételhez hasonlóan. □

6.32. Definíció. *(Elkerülhetetlen tulajdonság)*
Legyen $\hookrightarrow_S \subseteq \mathcal{P}(A) \times \mathcal{P}(A)$ a $\mapsto_S$ reláció tranzitív diszjunktív lezártja (4.16. def.), vagyis az a legkisebb reláció[15], amelyre teljesül, hogy

(1) $\mapsto_S \subseteq \hookrightarrow_S$.

[13] A $\mapsto_S$ definíciója megfelel a [Cha Mis 89]-ben adott *ensures* fogalmának, ha az ütemezés megfelel a feltétlenül pártatlan ütemezés axiómájának.

[14] A tétel Prasetya tételének relációs átfogalmazása [Pra 94].

[15] A $\hookrightarrow_S$ definíciója megfelel a [Cha Mis 89]-ben adott *leads-to* fogalmának, ha az ütemezés megfelel a feltétlenül pártatlan ütemezés axiómájának.

(2) Tranzitivitás: ha $(\lceil P\rceil, \lceil Q\rceil) \in \hookrightarrow_S$ és $(\lceil Q\rceil, \lceil R\rceil) \in \hookrightarrow_S$, akkor $(\lceil P\rceil, \lceil R\rceil) \in \hookrightarrow_S$.

(3) Diszjunkció: ha bármely W megszámlálható halmazra: $\forall m : (m \in W :: (\lceil P(m)\rceil, \lceil Q\rceil) \in \hookrightarrow_S$, akkor $((\lceil \exists m : m \in W :: P(m))\rceil, \lceil Q\rceil) \in \hookrightarrow_S$.

Jelölés: $P \hookrightarrow_S Q ::= (\lceil P\rceil, \lceil Q\rceil) \in \hookrightarrow_S$.

A 6.32. def. alapján $P \hookrightarrow_S Q$ pontosan akkor, ha a 6.32. def. (1),(2),(3) szabályainak véges sokszori alkalmazásával $P \hookrightarrow_S Q$ levezethető[16].

6.13. Megjegyzés. *(Egyértelműen létezik legkisebb adott tulajdonságú reláció)* A $\mathcal{P}(A)\times\mathcal{P}(A)$ rendelkezik a megadott tulajdonságokkal. Ha X és Y rendelkezik a megadott tulajdonságokkal, akkor $X \cap Y$ is rendelkezik velük. Így $\hookrightarrow_S$ egyértelműen definiált.

6.14. Megjegyzés.
A UNITY különböző relációs kiterjesztéseiben [Pac 92, Jut Kna Rao 89] nem a feladat specifikációs feltételeinek, hanem a program által definiált inv_S, $\mapsto_S$, $\hookrightarrow_S$ relációknak *megfelelő* relációkat definiálnak. Pachl [Pac 92] az $\mapsto_S$ reláció értelmezési tartományát az elérhető állapotok halmazára (6.17. def.) korlátozza.

6.24. Lemma. *($\Rightarrow$ és $\hookrightarrow_S$)*
Ha $P \subseteq Q$, akkor $(P, Q) \in \hookrightarrow_S$ tetszőleges S programra.
Biz.: $(P, P) \in \mapsto_S$, így $(P, Q) \in \mapsto_S$ a 6.31. definíció szerint. □

6.25. Lemma. *($\hookrightarrow_S$ és a stabil tulajdonság)*
Ha $P \hookrightarrow_S Q$ és $K \triangleright_S \downarrow$, akkor $P \wedge K \hookrightarrow_S Q \wedge K$.

Biz.: Struktúrális indukcióval az induktív 6.32. def. alapján.
Alapeset: $P \hookrightarrow_S Q$-t 1 lépésben vezettük le $P \mapsto_S Q$-ból. Ekkor a 6.21. lemma szerint $P \wedge K \mapsto_S Q \wedge K$. 6.32. def. (1) pontja szerint ekkor $P \wedge K \hookrightarrow_S Q \wedge K$.
Indukciós feltevés: Ha $X \hookrightarrow_S Y$-t kevesebb, mint n lépésben vezettük le, akkor $X \wedge K \hookrightarrow_S Y \wedge K$.
Tegyük fel, hogy a $P \hookrightarrow_S Q$-t n lépésben vezettük le. Az indukciós feltevés felhasználásával megmutatjuk, hogy $P \wedge K \hookrightarrow_S Q \wedge K$.
a) eset: az utolsó lépésben a 6.32. def. (2) pontját, a tranzitivitást alkalmaztuk, azaz: $P \hookrightarrow_S Q_1$ (kevesebb, mint n lépésben) és $Q_1 \hookrightarrow_S Q$ (kevesebb, mint n lépésben). Az indukciós feltétel szerint: $P \wedge K \hookrightarrow_S Q_1 \wedge K$ és $Q_1 \wedge K \hookrightarrow_S Q \wedge K$. A 6.32. def. (tranzitivitás) alapján: $P \wedge K \hookrightarrow_S Q \wedge K$.
b) eset: az utolsó lépésben a 6.32. def. (3) pontját, a diszjunktivitást alkalmaztuk, azaz: $P = \exists m : m \in W :: P(m)$ és $\forall m : m \in W :: (P(m) \hookrightarrow_S Q)$ (kevesebb, mint n lépésben). Az indukciós feltétel szerint: $\forall m : (m \in W :: (P(m) \wedge K \hookrightarrow_S Q \wedge K)$, amiből a 6.32. def. (diszjunktivitás alapján) $P \wedge K \hookrightarrow_S Q \wedge K$. □

[16] A (3)-as szabály egyetlen lépésben is végtelen sok elemre alkalmazható.

6.15. Megjegyzés. A tétel általánosítható a következő alakban[17]: Ha $P \hookrightarrow_S Q$ és $R \rhd B$, akkor $P \wedge R \hookrightarrow_S (Q \wedge R) \vee B$.
Bizonyítás strukturális indukcióval.

6.26. Tétel. *($\hookrightarrow_S$ és az invariánsok szigoríthatósága)*
Ha $(P \wedge J) \hookrightarrow_S (Q \wedge J)$ és $J \in inv_S(Q)$ és $K \in inv_S(Q)$, akkor $J \wedge K \in inv_S(Q)$ és $(P \wedge J \wedge K) \hookrightarrow_S (Q \wedge J \wedge K)$[18].

Biz.: Az állítás első része következik a 6.11. lemmából. Az állítás második részét pedig a 6.18. és 6.25. lemma alkalmazásával kapjuk. □

6.27. Tétel. *($\hookrightarrow_S$ és a legszigorúbb invariáns)*
Ha $(P \wedge J) \hookrightarrow_S (R \wedge J)$ és $J \in inv_S(Q)$, akkor
$P \wedge \mathrm{INV}_S(Q) \hookrightarrow_S R \wedge \mathrm{INV}_S(Q)$.
Biz.: 6.20. tételhez hasonlóan. □

6.28. Tétel. *($\hookrightarrow_S$ egyelemű részhalmazokra)*
$(X, Y) \in \hookrightarrow_S \Longleftrightarrow \forall x \in X : (\{x\}, Y) \in \hookrightarrow_S$.
Biz.: Ha $\forall x \in X : (\{x\}, Y) \in \hookrightarrow_S$, akkor $(X, Y) \in \hookrightarrow_S$ 6.32. def. alapján (diszjunktív lezárás). Ha $(X, Y) \in \hookrightarrow_S$, akkor $\forall x \in X : \{x\} \subseteq X$. A 6.24. lemma alapján $(\{x\}, X) \in \hookrightarrow_S$. Ha $(X, Y) \in \hookrightarrow_S$, akkor a tranzitív lezárás miatt $(\{x\}, Y) \in \hookrightarrow_S$.□

6.29. Lemma. *($\hookrightarrow_S$ – jobboldal gyengítése)*
Ha $P \hookrightarrow_S Q$ és $Q \Rightarrow R$, akkor $P \hookrightarrow_S R$.
Biz.: 6.24. lemma és a 6.32. def. (tranzitivitás) következménye. □

6.33. Definíció. *(Elkerülhetetlen feltétlenül pártatlan ütemezés mellett)*
$(P, Q) \in \rightsquigarrow_S$, akkor és csak akkor, ha $\forall a \in P$ az S által az a-hoz rendelt fákban bármelyik, a feltétlenül pártatlan ütemezésnek megfelelő végrehajtási úton[19] véges (esetleg nem korlátos) távolságban van olyan pont, amelynek címkéje eleme Q halmaznak.

6.30. Tétel. *($\rightsquigarrow_S$ egyelemű részhalmazokra)*
$(X, Y) \in \rightsquigarrow_S \Longleftrightarrow \forall x \in X : (\{x\}, Y) \in \rightsquigarrow_S$.
Biz.: A 6.33. def. közvetlen következménye. □

6.31. Tétel. *($\hookrightarrow_S$ helyessége és teljessége)*
$\hookrightarrow_S = \rightsquigarrow_S$[20].

[17] A PSP tétel [Cha Mis 89] relációs alakja.
[18] A tétel Prasetya bizonyítás nélkül publikált tételének relációs átfogalmazása [Pra 94].
[19] v.ö. 3.10. def.
[20] A tétel Pachl tételének általánosítása [Pac 92]

Biz.:[21]
a) $\hookrightarrow_S \subseteq \rightsquigarrow_S$. Biz.: $\mapsto_S \subset \rightsquigarrow_S$ és $\rightsquigarrow_S$ tranzitív és diszjunktív.
b) $\rightsquigarrow_S \subseteq \hookrightarrow_S$. A 6.28., 6.30. tételek alapján elegendő bizonyítani, hogy
$(\{x\}, P) \in \rightsquigarrow_S \Longrightarrow (\{x\}, P) \in \hookrightarrow_S$.
$A(P) ::= \{y | (\{y\}, P) \in \hookrightarrow_S\}$. $E(P) ::= \{y | (\{y\}, P) \notin \hookrightarrow_S\}$.
Legyen $w \in E(P)$. Jelöljük $E(w, P)$-vel azon pontok halmazát, amelyekre igaz, hogy w-ből olyan úton érhetőek el, amelynek minden pontja eleme $E(P)$-nek. $\forall w \in E(P) : \forall s \in S : \exists z \in E(w, P) : p(s)(z) \not\subseteq A(P)$, ellenkező esetben $\exists w \in E(P) : \exists s \in S : \forall z \in E(w, P) : p(s)(z) \subseteq A(P)$, azaz $(E(w, P), A(P)) \in \mapsto_S$ és $w \in E(w, P)$ miatt $w \in A(P)$ következne, ami ellentmondás. Tehát $\forall w \in E(P)$-re megkonstruálható egy olyan út, amelyen $\forall s \in S$ címkéje szerepel a feltétlenül pártatlan ütemezés aximómája szerint és az út $E(P)$ belsejében halad. Tegyük fel indirekt, hogy $x \in E(P)$. Ekkor a fentiek alapján $(\{x\}, P) \notin \rightsquigarrow_S$. Azaz $(\{x\}, P) \in \rightsquigarrow_S \Longrightarrow x \notin E(P)$, azaz $x \in A(P)$. □

6.5. Következmény. Ha egy program rendelkezik a $P \rightsquigarrow_S Q$ tulajdonsággal, akkor és csak akkor rendelkezik a $P \hookrightarrow_S Q$ tulajdonsággal is, amely a tranzitivitás és a diszjunkció szabályának véges számú alkalmazásával levezethető a program $U \mapsto_S V$ alakú tulajdonságaiból[22].

6.4.5. Fixpont tulajdonságok

A *konkrét program* az absztrakt program végrehajtásának prefixe. Az absztrakt program nem terminál abban az értelemben, hogy több elemi művelet nem kerül végrehajtásra. Terminálnak tekinthetünk azonban egy izolált programot akkor, ha elérte egy fixpontját, azaz a további műveletek hatására állapotváltozás már nem következhet be.

Az S programról azt mondjuk, hogy *fixpontba jutott az A altér felett*, ha az A altérhez tartozó változókra vonatkozó egyszerű értékadások mindegyikére teljesül, hogy az értékadás hatásrelációja független az altérhez nem tartozó állapottérkomponensel és

- az értékadás determinisztikus és a jobboldalán álló függvénykompozíciók (kifejezések) értéke azonos a baloldalon álló változók értékével vagy
- az értékadás jobboldalán szereplő kapcsoszárójel függvény feltétele hamis, vagy

[21] [Pac 92]-ben adott bizonyítás általánosítható a teljes állapottérre (lásd 6.14 lábjegyzet) és nemdeterminisztikus feltételes értékadások esetére.

[22] A tranzitív diszjunktív lezárási tulajdonságok felhasználásával *Cook-féle relatív teljes* [Rao 95] levezetési szabályrendszert alkothatunk a program haladási tulajdonságaira nézve.

- az értékadás nemdeterminisztikus és az altér azon részhalmazának, amely felett az értékadás determinisztikus, és a kapcsoszárójel függvény feltétele igazsághalmazának metszete felett az értékadás jobboldalán álló függvénykompozíciók (kifejezések) értéke azonos a baloldalon álló változók értékével.

Legyen $S = (s_0, \{s_1, ...s_m\})$, ahol $\forall s_j \in S$ egy (szimultán, nemdeterminisztikus) feltételes értékadás, $s_j : \|_{i\in[1,n]}((v_i :\in F_{j_i}(v_1, .., v_n), \text{ ha } \pi_{j_i})$.

Jelöljük $\pi_{j_{id}}$-vel azt a logikai függvényt, amelynek igazsághalmazára leszűkítve az F_{j_i} reláció determinisztikus, azaz: $\pi_{j_{id}}(a) \Leftrightarrow (|F_{j_i}(a)| = 1)$.

6.34. Definíció. *(Fixpontok halmaza)*
$fixpont_S ::= (\bigwedge_{j\in J, i\in[1..n]}(\neg\pi_{j_i} \vee (\pi_{j_{id}} \wedge v_i = F_{j_i}(v_1, \ldots, v_n))))$

Ha az értékadások mindegyike determinisztikus, akkor a program fixpontjait jellemző logikai függvényt a szimultán értékadások egyenlőséggé való átalakításával, feltételeik implikációs előtaggá való kiemelésével és konjunkciójával kapjuk meg: $fixpont_S = (\bigwedge_{j\in J, i\in[1..n]}(\pi_{j_i} \to v_i = F_{j_i}(a))$.

6.2. Példa. *(Program fixpontjainak halmaza)*
$S = (SKIP, \{k := k + 1, \text{ ha } k < N\})$.
$fixpont_S = (k < N \to k = k + 1) \equiv k \geq N$ [Cha Mis 89]. □

6.35. Definíció. *(Fixpont tulajdonság)*
Jelöljük FP_S-sel azon R logikai függvények igazsághalmazainak halmazát, amelyekre $fixpont_S \Rightarrow R$.

6.16. Megjegyzés. A megoldás definíciója szerint (7.1.) egy program teljesíti a FP $\Rightarrow R$ specifikációs feltételt, ha az R fixpontfeltétel tartalmazza a $fixpont_S$ állítás igazsághalmazának és (legalább) az elérhető állapotok halmazának metszetét.

6.32. Lemma. *(Fixpont tulajdonság gyengítése)*
Ha $R \Rightarrow Q$ és $R \in \mathrm{FP}_S$, akkor $Q \in \mathrm{FP}_S$.
Biz.: Az 6.35. def. közvetlen következménye. □

6.4.6. Terminálási tulajdonságok

6.36. Definíció. *(Biztosan fixpontba jut – tulajdonság)*
Jelöljük TERM_S-sel azon Q logikai függvények igazsághalmazainak halmazát, amelyekre $Q \hookrightarrow_S fixpont_S$.

A program biztosan fixpontba jut, ha egy alkalmasan megválasztott *variáns függvény*[23] értéke bármely állapot elérése után a jövőben elkerülhetetlenül csökken (8.4. tétel).

[23] pl. az állapottér változóiból függvénykompozícióval alkotott nemnegatív egészértékű függvény

6.5. Az absztrakt program viselkedési relációja

Egy program *szemantikai jelentését* azonosíthatjuk az általa az állapottér hatványhalmaza felett definiált - invariánsok, biztonságossági, haladási, fixpont és terminálási tulajdonságoknak megfelelő unáris és bináris - relációk együttesével. Egy ilyen szemantika leíró jellegű[24] (1. fejezet) [Jut Kna Rao 89] és absztrakciós szintje lényegében megegyezik a bevezetett megoldásfogalom absztrakciós szintjével.

6.37. Definíció. *(Viselkedési reláció)*
Legyen S program az A állapottér felett. Az A állapottér felett adott $(\rhd_S, \mapsto_S, \hookrightarrow_S, \mathrm{FP}_S, \mathrm{inv}_S, \mathrm{TERM}_S)$ rendezett relációhatost az S *absztrakt párhuzamos program viselkedési relációjá*nak hívjuk és $p(S)$-sel jelöljük.

6.6. Haladási tulajdonságok fixpontos definíciói

Ebben a bekezdésben példákat mutatunk arra, hogyan lehet logikai függvények és funkcionálok segítségével zárt alakban kifejezni a $\mapsto_S, \hookrightarrow_S$ relációkat. A továbbiakban a félreértések elkerülése érdekében $\mapsto_S$ megfelőjét $\mapsto\!\!\!\!\circ\ _S$-sel, a $\hookrightarrow_S$ megfelelőjét $\hookrightarrow\!\!\!\!\circ\ _S$-sel jelöljük.

6.6.1. Haladási tulajdonságok fixpontos alakja utófeltételre pártatlan ütemezés esetén

Az egyszerűbb megfogalmazás kedvéért feltesszük, hogy az ütemezés megfelel az utófeltételre pártatlan ütemezés axiómájának (6.24. def.). A bekezdésben bevezetett fogalmakra és tételekre a későbbiekben általában csak egyes megjegyzésekben és lábjegyzetekben utalunk, így a további fejezetek megértéséhez ez a bekezdés nem szükséges.

6.38. Definíció. Definiáljuk a $G(S,P)(Y,X) : ((A \longmapsto \mathcal{L}) \times (A \longmapsto \mathcal{L})) \longmapsto (A \longmapsto \mathcal{L})$ kétváltozós függvényt a következő függvénykompozíció segítségével: $G(S,P)(Y,X) ::= P \vee ((wpa(S,Y) \wedge wp(S, X \vee Y))$

6.17. Megjegyzés. A $G(S,P)(Y,X)$ jelölés helyett a $G(P,Y,X)$ jelölést alkalmazzuk, ha ez nem okoz félreértést.
A $G(S,P,Y)(X) : (A \longmapsto \mathcal{L}) \longmapsto (A \longmapsto \mathcal{L})$ függvény a $G(S,P)(Y,X)$-ből származtatható az első argumentum, az Y rögzítésével.

6.39. Definíció. *(Biztosítja tulajdonság)*
$(Q,P) \in \mapsto\!\!\!\!\circ\ _S$, ha $(Q \Rightarrow (P \vee ((wpa(S,P) \wedge wp(S, Q \vee P)))$,
azaz $(Q \Rightarrow G(P,P,Q))$.

[24] Leíró szemantikáról csak akkor beszélhetünk, ha a szemantikus leképezés kompozícionális. A *kompozcionalitás* azonban csak részben teljesül a modellben (9. fejezet), amely általában elegendő ahhoz, hogy a megoldást részfeladatok megoldásából programkonstrukciók segítségével előállítsuk, de nem felel meg a kompozicionalitás szigorú matematikai követelményének.

6.18. Megjegyzés. A 6.39. definícióban erősen kihasználjuk az *utófeltételre* pártatlan ütemezés meglétét.

6.3. Példa.

$$s_0: \quad x, i := -1, 0$$

$$S: \{ \quad s_1: x, i := \begin{cases} 0, i+1, & \text{ha } x < 0 \\ -x, i+1, & \text{ha } x \geq 0 \end{cases}, \quad s_2: x, i := \begin{cases} 0, i+1, & \text{ha } x > 0 \\ -x, i+1, & \text{ha } x \leq 0 \end{cases} \quad \}$$

Könnyen belátható, hogy $(x \neq 0, x = 0) \in \mapsto_S$ utófeltételre pártatlan ütemezés és a 6.39. definíció mellett. Létezik azonban olyan feltétlenül pártatlan ütemezés: $s_2, s_1, s_2, s_1, ...$, amelyre $x = 0$ elkerülhető. □

6.33. Lemma. *(G monoton)*
G monoton P, Y, X-ben.

Biz: $wp(S, X)$ és $wpa(S, Y)$ monoton X, Y-ban. □

6.6. Következmény. *(Legnagyobb fixpont létezik)*
$\forall P, Y : \eta X : G(P, Y, X)$ létezik.

6.40. Definíció. $F(P, Y) ::= \eta X : G(P, Y, X)$

6.34. Lemma. *(F monoton)*
$F(P, Y)$ monoton P, Y -ban.

Biz: Tegyük fel, hogy $Y \Rightarrow R$ és $P \Rightarrow Q$. $F(P, Y) = \{$ fixpont (3.1./f) $\} = G(P, Y, F(P, Y)) \Rightarrow \{ G$ monoton $\} \Rightarrow G(Q, R, F(P, Y))$
$\{$ a fixpont indukció (3.1./e) szerint $\}$
$F(P, Y) \Rightarrow \eta X : G(Q, R, X) = F(Q, R)$ □

6.7. Következmény. *(Legkisebb fixpont létezik)*
$\forall P : \mu Y : F(P, Y)$ létezik.

6.41. Definíció. *(Elkerülehetetlen tulajdonság)*
(elkerülhetetlenül P) $\quad \rightsquigarrow P ::= \mu Y : F(P, Y)$

6.35. Lemma. *($\rightsquigarrow P$ monoton)*
$\rightsquigarrow P$ monoton P-ben.

Biz: Tegyük fel, hogy $P \Rightarrow Q$. $F(P, \rightsquigarrow Q) \Rightarrow \{F$ monoton $\}$
$F(Q, \rightsquigarrow Q) = \{$fixpont(3.1./e)$\} = \rightsquigarrow Q$.
A fixpont indució (3.1./b) szerint: $\mu Y : F(P, Y) \Rightarrow \ (\rightsquigarrow Q)$ { 6.41. def. }
$(\rightsquigarrow P) \Rightarrow (\rightsquigarrow Q)$ □

6.36. Lemma. $P \Rightarrow (\rightsquigarrow P)$.

Biz: $P \vee (wpa(S, \rightsquigarrow P) \wedge wp(S, \rightsquigarrow P)) = \{$ fixpont (3.1./c,f) $\} = \rightsquigarrow P$.
A baloldal gyengítésével: $P \Rightarrow (\rightsquigarrow P)$ □

6.8. Következmény. $(\rightsquigarrow P) \Rightarrow \rightsquigarrow (\rightsquigarrow P)$.

6.42. Definíció. $(Q, P) \in \hookrightarrow_S$, ha $(Q \Rightarrow (\rightsquigarrow P))$ [25]

6.6.2. Haladási tulajdonságokra vonatkozó alaptételek

6.43. Definíció. *(Elkerülhetetlen utófeltételre pártatlan ütemezés mellett)*
$a \in \lceil \sim P \rceil$, akkor és csak akkor, ha az S által az a-hoz rendelt fákban bármelyik, utófeltételre pártatlan ütemezésnek megfelelő végrehajtási úton véges (esetleg nem korlátos) távolságban van olyan pont, amelynek címkéje eleme a $\lceil P \rceil$ halmaznak.

6.37. Tétel. $\lceil \rightsquigarrow P \rceil \subseteq \lceil \sim P \rceil$.

A tétel bizonyításához definiáljuk az FF leképezést:

6.44. Definíció.

$$FF^0(P) = \downarrow$$

$$FF^i(P) = \begin{cases} F(P, FF^{i-1}(P)), & \text{ha } i \text{ nem határrendszám} \\ \bigvee_{j \in [0,i]} FF^j(P), & \text{ha i határrendszám.} \end{cases}$$

6.38. Lemma.

a) Ha $(i \leq j)$, akkor $FF^i(P) \Rightarrow FF^j(P)$,

b) $(\mu Y : F(P, Y)) = FF^i(P)$ valamely i-re,
azaz $(\rightsquigarrow P) = FF^i(P)$ valamely i-re,

c) $FF^i(P) \Rightarrow wpa(S, FF^{i-1}(P)) \vee P$, ha i nem határrendszám,

d) $FF^i(P) \Rightarrow wp(S, FF^i(P)) \vee P$, ha i nem határrendszám.

[25] A 6.42. definíció Park eredményeiből merít [Par 79]. Park felismerte, hogy pártatlansági feltételek fennállása esetén iteratív programok leggyengébb előfeltétele nem fejezhető ki önmagában csak a legkisebb vagy csak a legnagyobb fixpontoperátor segítségével. A két fixpontoperátor együttes alkalmazásával a leképezés olyan fixpontjait is definiálhatjuk, amelyek nem egyeznek meg sem a legkisebb, sem a legnagyobb fixponttal. Példaként Park megadja két végtelen sorozatból fair összefésüléssel előállítható sorozatok halmazát [Par 79]. Rekurzív eljárások leggyengébb előfeltételét határozza meg predikátum-transzformerek és fixpontoperátorok segítségével Morris *utófeltételre* pártatlan ütemezés esetén. A 6.37. tétel bizonyításakor [Mor 90]-ban leírt technikát is alkalmazzuk. A megközelítés azonban Parkéhoz hasonló: halmazelméleti eszközöket használunk és iteratív programstruktúrákat vizsgálunk.

A lemma bizonyítása: F monoton, így a) és b) speciális esete [Mor 90] 3. lemmájának.

c) és d) bizonyítása: $FF^i(P) = \{\text{ def }\} = F(P, FF^{i-1}(P)) = \{\text{def}\} = \eta X : G(P, FF^{i-1}(P), X) = \{\text{def}\} = \eta X : (P \vee ((wpa(S, FF^{i-1}(P)) \wedge wp(S, X \vee FF^{i-1}(P)))) =$ {fixpont (3.1./f} $= (P \vee ((wpa(S, FF^{i-1}(P))) \wedge wp(S, FF^i(P) \vee FF^{i-1}(P)))) = \{$ a) és disztributivitás miatt $\}$
$= (P \vee ((wpa(S, FF^{i-1}(P))) \wedge (P \vee wp(S, FF^i(P))))$. A jobboldal gyengítésével $FF^i(P) \Rightarrow P \vee wpa(S, FF^{i-1}(P)))$ és $F^i(P) \Rightarrow \vee wp(S, FF^i(P)))$ □

A 6.37. tétel bizonyítása:

A bizonyítás hasonló a [Mor 90]-ben leírthoz. Válasszunk egy olyan t végrehajtási utat, amely $\lceil(\rightsquigarrow P)\rceil$-ból indul. Megmutatjuk, ha P sohasem teljesül t mentén, akkor a t-ben szereplő feltételes értékadások kiválasztása nem felel meg a utófeltételre pártatlan ütemezés szabályainak. A t út t_j pontjához rendeljük hozzá azt az $i > 0$ rendszámot, amelyre $FF^i(P)(t_j)$ és $\neg FF^{i-1}(P)(t_j)$. A 6.38./d. lemma és a $\rightsquigarrow P$ definíciója szerint ilyen i rendszám létezik és nem lehet határrendszám. Mivel a t út pontjaihoz hozzárendelt rendszámok sorozata monoton csökkenő (6.38./d. lemma), ezért van olyan k és h rendszám, amelyekre $\forall j > h : FF^k(P)(t_j) \wedge \neg FF^{k-1}(P)(t_j)$. A (6.38./c. lemma szerint a t út nem felel meg a utófeltételre pártatlan ütemezésnek, ugyanis t_j pontjától kezdve mindig kiválasztható egy olyan utasítás, ami $\lceil FF^{k-1}(P)\rceil$-be visz és ez egyetlen pontban sem kerül kiválasztásra [Mor 90]. □

6.9. Következmény. $(\rightsquigarrow (\rightsquigarrow P)) \Rightarrow (\rightsquigarrow P)$.
Biz.: $\rightsquigarrow\rightsquigarrow P = FF^i(\rightsquigarrow P) = FF^i(FF^j(P)) = \rightsquigarrow P$. □

6.39. Lemma. Ha $Q \mapsto_S P$, akkor $Q \hookrightarrow_S P$[26].

Biz.: $(Q \mapsto_S P)\{$6.39. def. $\} = Q \Rightarrow G(P, P, Q)$. A fixpont indució szerint (3.1./e): $Q \Rightarrow F(P, P)$. Mivel $\{P \Rightarrow (\rightsquigarrow P)$ a 6.36. lemma szerint és F monoton, ezért $F(P, P) \Rightarrow F(P, \rightsquigarrow P) = \{$fixpont$\} = \rightsquigarrow P$.
Tehát: $(Q \Rightarrow \rightsquigarrow P)$ {def} $(Q \hookrightarrow_S P)$. □

6.40. Lemma. *($\hookrightarrow_S$ tranzitív)*
Ha $P \hookrightarrow_S Q$ és $Q \hookrightarrow_S R$, akkor $P \hookrightarrow_S R$[27].

Biz.: $Q \hookrightarrow_S R$ {6.42. def. $\} Q \Rightarrow (\rightsquigarrow R)$ {$\rightsquigarrow$ monoton$\}$ $(\rightsquigarrow Q) \Rightarrow (\rightsquigarrow (\rightsquigarrow R))$ {6.9. köv. $\}$ $(\rightsquigarrow Q) \Rightarrow (\rightsquigarrow R)$. Másrészt a feltétel és 6.42. definició szerint: $P \Rightarrow (\rightsquigarrow Q)$. A $\Rightarrow$ tranzitivitása miatt: $P \Rightarrow (\rightsquigarrow R)$ □

6.41. Lemma. *($\hookrightarrow_S$ diszjunktív)*
Legyen I egy tetszőleges megszámlálható halmaz.
Ha $\forall i \in I : (P_i \hookrightarrow_S Q)$, akkor $(\exists i : P_i) \hookrightarrow_S Q$[28].

[26] v.ö.: 6.33. def.
[27] v.ö.: 6.33. def.
[28] v.ö.: 6.33. def.

Biz.: $\forall i \in I : (P_i \hookrightarrow\!\!\!\!\!\!\ominus\ {}_S Q)\{\text{def.}\}\ \forall i \in I : P_i \Rightarrow (\rightsquigarrow Q)$.
Ezért $P_{i_1} \vee \ldots \vee P_i \Rightarrow (\rightsquigarrow Q)$, azaz $(\exists i : P_i) \Rightarrow (\rightsquigarrow Q)$.
A 6.42. definíciója szerint: $(\exists i \in P_i) \hookrightarrow\!\!\!\!\!\!\ominus\ {}_S Q$. □

6.42. Tétel. $\lceil \sim P \rceil \subseteq \lceil \rightsquigarrow P \rceil$.

Biz.: $A(P)::=\lceil \rightsquigarrow P \rceil$, $E(P)::=\lceil \neg\rightsquigarrow P \rceil$. Legyen $w \in E(P)$. Jelöljük $E(w,P)$-vel azon pontok halmazát, amelyekre igaz, hogy w-ből olyan úton érhetőek el, amelynek minden pontja eleme $E(P)$-nek. $\forall w \in E(P) : \exists z \in E(w,P) : \forall s \in S : p(s)(z) \not\subseteq A(P)$, ellenkező esetben $\exists w \in E(P) : \forall z \in E(w,P) : \exists s \in S : p(s)(z) \subseteq A(P)$, azaz $(E(w,P), A(P)) \in \mapsto\!\!\!\!\!\!\ominus\ {}_S$ és $w \in E(w,P)$ miatt $w \in A(P)$ következne 6.39.,6.40., lemmák miatt, ami ellentmondás. Tehát $\forall w \in E(P)$-re megkonstruálható egy olyan út, amelyre az utófeltételre pártatlan ütemezés aximómája teljesül, elkerülheti $A(P)$-t, azaz az út $E(P)$ belsejében halad. Tegyük fel indirekt, hogy $x \in E(P)$. Ekkor a fentiek alapján $x \notin \lceil \sim P \rceil$. Azaz $x \in \lceil \sim P \rceil \Longrightarrow x \notin E(P)$, azaz $x \in A(P)$. □

6.43. Tétel. *($\rightsquigarrow P$ helyessége és teljessége)*
Legyen $a \in A$ tetszőleges. Legyen az S ütemezése utófeltételre pártatlan és legyen az S program az a állapotban. $a \in \lceil \rightsquigarrow P \rceil$ akkor és csak akkor, ha S elkerülhetetlenül olyan állapotba jut, ahol P teljesül[29].

Biz.: A 6.37. és a 6.42. tétel és a 6.43. def. következménye. □

[29] azaz az S által generált fákban bármelyik a-val címkézett útból induló, utófeltételre pártatlan ütemezésnek megfelelő végrehajtási úton véges (esetleg nem korlátos) távolságban van olyan pont, amelynek címkéje eleme P igazsághalmazának.

7. Fejezet

A megoldás fogalma

Megadjuk, hogy egy absztrakt program mikor old meg egy feladatot. A megoldás fogalmát a leggyengébb előfeltétel *fogalmára építjük fel. A megoldás definíciója így egy olyan verifikációs kalkulus alapja, amely helyes program esetén a specifikációs feltételek és az absztrakt program utasításainak számával lineárisan arányos számú lépésben véget ér. Kimondunk néhány tételt, amelyek a verifikációt egyszerűsítik, illetve igazolják, hogy a bevezetett megoldásfogalom megfelel elvárásainknak.*

7.1. A megoldás definíciója

7.1. Definíció. *(Megoldás)*
Azt mondjuk, hogy az S program megoldja az F feladatot (5.1. def.), ha $\forall b \in B$: $\exists h \in F(b)$, hogy az S program megfelel a h-ban adott $\mathrm{inv}_h P$, $P \rhd_h U$, $P \mapsto_h U$, $P \hookrightarrow_h U$, $\mathrm{FP}_h \Rightarrow R$, $Q \in \mathrm{TERM}_h$ alakú specifikációs feltételek mindegyikének a $Q \in \mathrm{INIT}_h$ kezdeti feltételek mellett.

Az alábbiakban sorra megadjuk, hogy egy S program mikor *felel meg* az egyes specifikációs feltételeknek a $Q \in \mathrm{INIT}_h$ kezdeti feltételek mellett.

7.1. Megjegyzés. *(Specifikációs feltételek és elérhető állapotok)*
Rögzített program esetén indokolt a specifikációs feltételek vizsgálatát az elérhető állapotok halmazára korlátozni [Lam Lyn 90, San 91, Pra 94]. Ha a program megfelel egy specifikációs feltételnek az elérhető állapotok felett, akkor a specifikációs feltétel nem sérül a program futása során.
A gyakorlatban azonban általában az elérhető állapotok halmazánál tágabb halmazt választunk (v.ö.: 7.3. megjegyzés), szélső esetben akár az összes állapotot, a teljes állapotteret is figyelembe vehetjük.

7.2. Megjegyzés. A megoldás definíciójának megadásakor az absztrakt program viselkedési relációjának 6.5. bekezdésben adott definíciójára (6.37. def.) támaszkodunk

az absztrakt program (6.15. def.) definíciója helyett. A viselkedési reláció és a feladat hasonló szerkezetű, így könnyen összehasonlíthatóak.

7.2. Átmenetfeltételek

7.2.1. Biztonságossági feltételek

7.2. Definíció. *(Megfelel* $(\mathrm{inv}_h P)$*-nek)*
Az S program pontosan akkor felel meg az $(\mathrm{inv}_h P)$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ invariáns tulajdonág, amely mellett megfelel a feltételnek.

7.3. Definíció. Az S program pontosan akkor felel meg az $(\mathrm{inv}_h P)$ specifikációs feltételnek a $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ mellett, ha
$P \wedge K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$.

7.3. Megjegyzés. A program bármely invariáns tulajdonságának igazsághalmaza tartalmazza az elérhető állapotok halmazát (6.17. def., 6.15. tétel), így a a továbbiakban a 7.1. megjegyzésnek megfelelően megengedjük, hogy a program az egyes specifikációs feltételeknek csak egy-egy kiválasztott *invariáns tulajdonság* igazsághalmaza felett feleljen meg[1]. Az alábbiakban megmutatjuk, hogy programok egyes specifikációs feltételekre vonatkozó helyességének bizonyítása során az invariáns tulajdonságokat segédtételként felhasználhatjuk.

7.4. Megjegyzés. *(Specifikációs feltételek és mindig igaz állítások)*
A specifikációs feltételek vizsgálatát megszoríthatnánk egyes *mindig igaz* állítások igazsághalmazára is, amelyek igazsághalmaza az invariáns tulajdonságokhoz hasonlóan tartalmazza az elérhető állapotok halmazát. A mindig igaz állítások azonban nem használhatóak fel segédtételként az utasítások *leggyengébb előfeltételének kiszámítására épülő bizonyítások* során. (Ha J mindig igaz, de nem invariáns, akkor $J \not\Rightarrow wp(S, P)$.) A mindig igaz állításokra az sem igaz, hogy szigoríthatóak az átmenetfeltételekre nézve (6.19. 6.22. 6.26. tétel) [Pra 94].

7.4. Definíció. *(Megfelel* $P \rhd_h Q$*-nak)*

S megfelel a $P \rhd_h Q$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ invariáns tulajdonság ami mellett megfelel a feltételnek.

7.5. Definíció. Az S program pontosan akkor felel meg az $P \rhd_h Q$ specifikációs feltételnek a $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ mellett, ha $P \wedge K \rhd_S Q \wedge K$.

[1] Ez a döntés megfelel a helyettesítési axiómának [Cha Mis 89, UN 88-93, Pra 94]

7.2.2. Haladási feltételek

7.6. Definíció. *(Megfelel $P \mapsto_h Q$-nak)*
S megfelel a $P \mapsto_h Q$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ invariáns tulajdonság ami mellett megfelel a feltételnek.

7.7. Definíció. Az S program pontosan akkor felel meg az $P \mapsto_h Q$ specifikációs feltételnek a $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ mellett, ha $P \wedge K \mapsto_S Q \wedge K$.

7.5. Megjegyzés. *(Haladási feltételek és az ütemezés)*
A definíció a 6.31. definícióra épül, amelyben erősen kihasználjuk a feltétlenül pártatlan ütemezés meglétét [Cha Mis 89].

7.8. Definíció. *(Megfelel $P \hookrightarrow_h Q$-nak)*
S pontosan akkor felel meg a $P \hookrightarrow_h Q$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ invariáns tulajdonság ami mellett megfelel a feltételnek.

7.9. Definíció. S pontosan akkor felel meg a $P \hookrightarrow_h Q$ specifikációs feltételnek a $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ mellett, ha $P \wedge K \hookrightarrow_S Q \wedge K$.

7.10. Definíció. *(Megfelel $P \hookrightarrow \mathrm{FP}_h$-nak)*
S pontosan akkor felel meg a $P \hookrightarrow \mathrm{FP}_h$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ invariáns tulajdonság ami mellett megfelel a feltételnek.

7.11. Definíció. S pontosan akkor felel meg a $P \hookrightarrow \mathrm{FP}_h$ specifikációs feltételnek a $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ mellett, ha $(sp(s_0, P) \wedge K) \in \mathrm{TERM}_S$.

7.3. Peremfeltételek

7.3.1. Fixpont feltételek

7.12. Definíció. *(Megfelel $\mathrm{FP}_h \Rightarrow R$-nek)*
S pontosan akkor felel meg a $\mathrm{FP}_h \Rightarrow R$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ invariáns tulajdonság ami mellett megfelel a feltételnek.

7.13. Definíció. S pontosan akkor felel meg a $\mathrm{FP}_h \Rightarrow R$ specifikációs feltételnek $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ mellett, ha $fixpont_S \wedge K \Rightarrow R$.

7.4. Megoldás K invariáns tulajdonság mellett

7.14. Definíció. *(Megoldás K invariáns mellett)*
Azt mondjuk, hogy az S program megoldja az F feladatot (5.1. def.) a K invariáns tulajdonság mellett, ha $\forall b \in B : \exists h \in F(b)$, hogy $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ és az S program K mellett megfelel a h-ban adott $\mathrm{inv}_h P$, $P \rhd_h U$, $P \mapsto_h U$, $P \hookrightarrow_h U$, $\mathrm{FP}_h \Rightarrow R$, $Q \in \mathrm{TERM}_h$ alakú specifikációs feltételek mindegyikének a $Q \in \mathrm{INIT}_h$ kezdeti feltételek mellett.

7.5. A megoldás definíciójának vizsgálata

7.1. Lemma. *(Megfelel $(\mathrm{inv}_h P)$-nek)*
S megfelel $(\mathrm{inv}_h P)$ specifikációs feltételnek, ha van olyan $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$, hogy
$sp(s_0, (\bigwedge_{Q \in \mathrm{INIT}_h} Q)) \Rightarrow P \wedge K$ és $P \wedge K \Rightarrow wp(S, P \wedge K)$.

Biz.: 6.27. és 7.2. definíciók közvetlen következménye. □

7.1. Következmény. S megfelel $(\mathrm{inv}_h P)$ specifikációs feltételnek, ha
$(\exists Q \in \mathrm{INIT}_h, \exists K) : sp(s_0, Q) \Rightarrow P \wedge K$ és $K \Rightarrow wp(S, K)$ és $P \wedge K \Rightarrow wp(S, P \wedge K)$.

7.2. Tétel. *(Megfelel inv_h-nak INV_S mellett)*
Az S program pontosan akkor felel meg a $P \in \mathrm{inv}_h$ specifikációs feltételnek, ha megfelel a legszigorúbb invariáns mellett, azaz, ha P mindig igaz ($P \in true_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$).

Biz.: Ha megfelel a legszigorúbb invariáns mellett, akkor van olyan invariáns, amely mellett megfelel, így a 7.2. definíció szerint megfelel a feltételnek. Ekkor $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. $\mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ a legszigorúbb invariáns, így: $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) = \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$, azaz P mindig igaz.
Ha megfelel a feltételnek, akkor van olyan J invariáns amely mellett megfelel, így a 6.11. tétel szerint a legszigorúbb invariáns szerint is megfelel a feltételnek. □

7.3. Lemma. *(Megfelel $P \rhd_h Q$-nak)*

S megfelel a $P \rhd_h Q$ specifikációs feltételnek *a* $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ *invariáns tulajdonság mellett*, ha $(P \wedge \neg Q \wedge K \Rightarrow wp(S, (P \vee Q) \wedge K))$. Biz: A 6.30.,7.4. definíciók közvetlen következménye. □

7.4. Tétel. *(Megfelel $P \rhd_h Q$-nak INV_S mellett)*
Az S program pontosan akkor felel meg a $P \rhd_h Q$ specifikációs feltételnek, ha megfelel a legszigorúbb invariáns mellett, azaz:
$P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \rhd_S Q \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$.

Biz.: Ha megfelel a legszigorúbb invariáns mellett, akkor van olyan invariáns, amely mellett megfelel, így a 7.4. definíció szerint megfelel a feltételnek. Ha megfelel a feltételnek, akkor van olyan J invariáns amely mellett megfelel, így a 6.20. tétel szerint a legszigorúbb invariáns szerint is megfelel a feltételnek. □

7.6. Megjegyzés.
Azt mondjuk, hogy S megfelel a P stabil$_h$ feltételnek, ha megfelel a $P \rhd_h \downarrow$ feltételnek.

7.5. Lemma. *(Megfelel $(P \mapsto_h Q)$-nak)*
Az S program megfelel $(P \mapsto_h Q)$ specifikációs feltételnek a $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ *invariáns tulajdonság mellett*, ha S megfelel a $P \rhd_h Q$ feltételnek K mellett, és $\exists j \in J : (P \wedge \neg Q \wedge K \Rightarrow wp(s_j, Q \wedge K))$.

Biz.: 6.31. 7.6. definíciók és a 7.3. lemma közvetlen következménye. □

7.6. Tétel. *(Megfelel $P \mapsto_h Q$-nak INV_S mellett)*
Az S program pontosan akkor felel meg a $P \mapsto_h Q$ specifikációs feltételnek, ha megfelel a legszigorúbb invariáns mellett, azaz:
$P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \mapsto_S Q \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$.

Biz.: 7.4. tétel bizonyításához hasonlóan a 6.23. tétel felhasználásával. □

7.7. Tétel. *(Megfelel $P \hookrightarrow_h Q$-nak INV_S mellett)*
Az S program pontosan akkor felel meg a $P \hookrightarrow_h Q$ specifikációs feltételnek, ha megfelel a legszigorúbb invariáns mellett, azaz:
$P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$.

Biz.: 7.4. tétel bizonyításához hasonlóan a 6.27. tétel felhasználásával. □

7.8. Lemma. *(Megfelel $P \hookrightarrow_h Q$-nak INV_S mellett)*
Az S program pontosan akkor felel meg a $P \hookrightarrow_h Q$ specifikációs feltételnek, ha $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q$.

Biz.: Ha $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q$, akkor a 6.25. tétel miatt (a legszigorúbb invariáns stabil): $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. Az állítást az előző tétel alkalmazásával kapjuk. Ha S megfelel $P \hookrightarrow_h Q$-nak, akkor az előző tétel szerint $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. A 6.29. lemma szerint a $\hookrightarrow_S$ jobboldala gyengíthető, így $P \wedge \mathrm{INV}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q$. □

7.2. Következmény. Az S program pontosan akkor felel meg a $P \hookrightarrow_h Q$ specifikációs feltételnek, ha $P \wedge \mathrm{INV}_S(\overset{\wedge}{_{Q\in \mathrm{INIT}_h}} Q) \rightsquigarrow_S Q$.
Biz.: A 6.31. tétel szerint $\hookrightarrow_S = \rightsquigarrow_S$. □

7.9. Lemma. *(Megfelel $P \hookrightarrow \mathrm{FP}_h$-nak INV_S mellett)*
Az S program pontosan akkor felel meg a $P \hookrightarrow \mathrm{FP}_h$ specifikációs feltételnek, ha $sp(s_0, P) \wedge INV_S(\overset{\wedge}{_{Q\in \mathrm{INIT}_h}} Q) \hookrightarrow_S fixpont_S$.

Biz.: A 6.36.,7.2.,7.10. def. és 7.8. lemma közvetlen következménye. □

7.10. Lemma. *(Megfelel $\mathrm{FP}_h \Rightarrow R$-nek)*
S megfelel a $(\mathrm{FP}_h \Rightarrow R)$ specifikációs feltételnek az A állapottér felett, *a* $K \in \mathrm{inv}_S(\overset{\wedge}{_{Q\in \mathrm{INIT}_h}} Q)$ *invariáns mellett*, ha $fixpont_S \wedge K \Rightarrow R$, ahol a $fixpont_S$ logikai függvény az S program A feletti fixpontjainak halmazát adja meg (6.34. def.).

Biz.: A 6.35. és a 7.12. def. közvetlen következménye. □

7.11. Lemma. *(Megfelel $\mathrm{FP}_h \Rightarrow R$-nek INV_S mellett)*
S pontosan akkor felel meg a $(\mathrm{FP}_h \Rightarrow R)$ kikötésnek, ha $fixpont_S \wedge INV_S(\overset{\wedge}{_{Q\in \mathrm{INIT}_h}} Q) \Rightarrow R$.

Biz.: Az előző lemma következménye. □

7.7. Megjegyzés. *(Megoldás K mellett)*
Ha S megoldja a feladatot egy K invariáns tulajdonság mellett, akkor S megoldja a feladatot. Ha S megoldja a feladatot, akkor megoldja a legszigorúbb invariáns mellett is.

7.12. Tétel. *(Program és feladat kiterjesztése)*
Legyen az F feladat és az S program az A állapottér A_1 altere felett definiálva. Ha S megoldja F-et, akkor S kiterjesztése A-ra megoldja az F feladat A-ra való kiterjesztését[2].

Biz.: A 7.1. def. szerint az S program viselkedési relációja a megoldás definíciója szerint tartalmaz minden olyan programtulajdonságot, amely ahhoz szükséges, hogy S megfeleljen az F specifikációs feltételeinek. A programtulajdonságok mindegyikének definíciója a leggyengébb előfeltételre épül, ezért 6.22. def. és a 6.7. lemma szerint a programtulajdonságok kiterjesztései tulajdonságai a kiterjesztett programnak is. Így a kiterjesztett feladat (5.3. def.) specifikációs feltételeinek kielégítéséhez szükséges valamennyi programtulajdonság szerepel a kiterjesztett program viselkedési relációjában. □

[2] A tétel a [Fót 88] cikk egyik kiterjesztési tételének általánosítása. A többi kiterjesztési tétel megfelelője is megfogalmazható.

8. Fejezet

Levezetési szabályok

Ebben a fejezetben olyan tételeket bizonyítunk be, amelyeket gyakran alkalmazunk feladatok finomítása során, vagy amelyek megkönnyítik annak bizonyítását, hogy egy program megfelel egy adott specifikációs feltételnek. Ezeket a tételeket levezetési szabály*oknak nevezzük. A leggyakrabban invariánsok, elkerülhetetlenséget kifejező, ill. fixpontfeltételek finomítására van szükség.*

8.1. Biztonságossági feltételek finomítása

8.1. Lemma. *(Invariáns feltétel felbontása)*
Ha egy S absztrakt program megfelel az $(\mathrm{inv}_h P_1)$, $(\mathrm{inv}_h P_2)$ specifikációs feltételeknek, akkor megfelel a $(\mathrm{inv}_h P_1 \wedge P_2)$ specifikációs feltételnek is.
Biz.: a 6.11. tétel következménye.□

8.2. Haladási feltételek finomítása

8.2. Lemma. *($\hookrightarrow_h$ finomítása véges sok lépésben)*
S megfelel a $P \hookrightarrow_h Q$ specifikációs feltételnek, ha az alábbi szabályok véges számú alkalmazásával levezethető:

(1) Ha S megfelel $(P \mapsto_h Q)$-nak, akkor S megfelel $(P \hookrightarrow_h Q)$-nak is.

(2) Tranzitivitás: ha S megfelel $(P \hookrightarrow_h Q)$-nak és S megfelel $(Q \hookrightarrow_h R)$-nak, akkor S megfelel $(P \hookrightarrow_h R)$-nak is.

(3) Diszjunkció: bármely W megszámlálható halmazra: ha
$\forall m : m \in W :: S$ megfelel $(P(m) \hookrightarrow_h Q)$-nak, akkor
S megfelel $((\exists m : m \in W :: p(m)) \hookrightarrow_h Q)$-nak is.

Biz.: 6.32. def. és a 7.6.,7.7. tételek felhasználásával: (1) A feltétel szerint $P \wedge INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \mapsto_S Q \wedge INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. Ekkor 6.32. def. szerint $P \wedge INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S Q \wedge INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. (2)-es és (3)-as esetben hasonlóan. □

8.3. Lemma. *($P \hookrightarrow \mathrm{FP}_h$ feltétel bizonyítása)*
Az S program pontosan akkor felel meg a $P \hookrightarrow \mathrm{FP}_h$ specifikációnak, ha megfelel a $sp(s_0, P) \hookrightarrow_h \ fixpont_S$ feltételnek, tehát S *biztosan fixpontba jut.*

Biz.: A 7.9. lemma szerint $sp(s_0, P) \wedge INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \hookrightarrow_S \ fixpont_S$, így a 7.8. lemma felhasználásával: S megfelel a $sp(s_0, \uparrow) \hookrightarrow_h \ fixpont_S$ feltételnek. □

8.1. Definíció. *(Variáns függvény)*
Variáns függvénynek nevezzük a $t : A \longmapsto \mathcal{Z}$, az állapotokhoz egészeket rendelő függvényeket. Legyen $m \in \mathcal{Z}$ tetszőleges egész szám. A $t = m, t > m \ : A \longmapsto \mathcal{L}$ logikai függvényeket definiáljuk az igazsághalmazukkal:
$\lceil t = m \rceil ::= \{a \in A \mid t(a) = m\}$, $\lceil t > m \rceil ::= \{a \in A \mid t(a) > m\}$.

A következő tételben a UNITY indukciós elvét [Cha Mis 89] fogalmazzuk meg a ciklus levezetési szabályához hasonló alakban [Fót 83, WRMP 95].

8.4. Tétel. *(Variánsfüggvény alkalmazása)*
Legyen $P, Q : A \longmapsto \mathcal{L}$ logikai függvény és $t : A \longmapsto \mathcal{Z}$ egy olyan variáns függvény, amelyre teljesül, hogy $(P \wedge \neg Q) \Rightarrow t > 0$. Ha $\forall m \in \mathcal{N} :: (P \wedge \neg Q \wedge t = m) \hookrightarrow_S ((P \wedge t < m) \vee Q)$, akkor $P \hookrightarrow_S Q$.

Biz.: Teljes indukcióval belátjuk, hogy $\forall m \in \mathcal{N} : (P \wedge \neg Q \wedge t = m \hookrightarrow_S Q)$. Ebből a 6.32. def. alapján, a diszjunktivitás felhasználásával
$P \wedge \neg Q \wedge (\bigvee_{m \in \mathcal{N}} (t = m)) \hookrightarrow_S Q$, azaz $P \wedge \neg Q \wedge t > 0 \hookrightarrow_S Q$ adódik.
A feltétel szerint $P \wedge \neg Q \Rightarrow t > 0$, így $(P \wedge \neg Q \wedge t > 0) \equiv P \wedge \neg Q$.
Tehát $P \wedge \neg Q \hookrightarrow_S Q$. A 6.24. lemma szerint $P \wedge Q \hookrightarrow_S Q$. A 6.32. def., a diszjunktivitás alkalmazásával: $P \hookrightarrow_S Q$.
Teljes indukció:
Alapeset: $m = 1$. A tétel feltétele szerint: $(P \wedge \neg Q \wedge t = 1) \hookrightarrow_S ((P \wedge t < 1) \vee Q)$. Tudjuk, hogy $P \wedge \neg Q \Rightarrow t > 0$, így $(P \wedge t < 1) \vee Q \equiv (P \wedge \neg Q \wedge t < 1) \vee (P \wedge Q \wedge t < 1) \vee Q \equiv \downarrow \vee (P \wedge Q \wedge t < 1) \vee Q \equiv Q$.
Indukciós feltevés: $\forall k : k > 0 \wedge k < m :: (P \wedge \neg Q \wedge t = k \hookrightarrow_S Q)$. A 6.32. def. alapján, a diszjunktivitás felhasználásával: $(P \wedge \neg Q \wedge t > 0 \wedge t < m) \hookrightarrow_S Q$. A 6.24. lemma alapján: $Q \hookrightarrow_S Q$. A diszjunkció ismételt alkalmazásával: $(P \wedge \neg Q \wedge t > 0 \wedge t < m) \vee Q \hookrightarrow_S Q$. A tétel feltétele szerint: $(P \wedge \neg Q \wedge t = m) \hookrightarrow_S ((P \wedge \neg Q \wedge t < m) \vee (P \wedge Q \wedge t < m) \vee Q)$. $P \wedge \neg Q \Rightarrow t > 0$, így $((P \wedge \neg Q \wedge t < m) \vee (P \wedge Q \wedge t < m) \vee Q) \equiv (P \wedge \neg Q \wedge t > 0 \wedge t < m) \vee Q$. Tehát $(P \wedge \neg Q \wedge t = m) \hookrightarrow_S (P \wedge \neg Q \wedge t > 0 \wedge t < m) \vee Q$. A 6.32. def. szerint, a tranzitivitás alkalmazásával: $(P \wedge \neg Q \wedge t = m) \hookrightarrow_S Q$. □

8.5. Tétel. *($\hookrightarrow_h$ finomítása variánsfüggvény alkalmazásával)*
Legyen $P, Q : A \longmapsto \mathcal{L}$ logikai függvény és $t : A \longmapsto \mathcal{Z}$ egy olyan variáns függvény, amelyre teljesül, hogy $P \wedge \neg Q \Rightarrow t > 0$. Ha $\forall m \in \mathcal{N} :: S$ megfelel a $(P \wedge \neg Q \wedge t = m) \hookrightarrow_h ((P \wedge t < m) \vee Q)$ specifikációs feltételnek, akkor S megfelel a $P \hookrightarrow_h Q$ specifikációs feltételnek is.

Biz.: Az előző tétel (8.4.) bizonyítása alapján a bizonyítás megkonstruálható. A bizonyítás során a 6.32. def. helyett a 8.2. lemmára kell hivatkozni. A 6.24. lemma helyett pedig a lemma azon következményére van szükség, amely szerint tetszőleges S program megfelel a $Q \wedge P \hookrightarrow_h Q$ feltételnek. □

8.6. Lemma. Legyen $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$, válasszunk egy olyan t variáns függvényt, amelyre: $K \wedge \neg\ fixpont_S \Rightarrow t > 0$. Ha minden $t' \in \mathcal{N}$-re $(K \wedge \neg\ fixpont_S \wedge t = t') \hookrightarrow_S (K \wedge t < t') \vee fixpont_S$, akkor S megfelel tetszőleges Q logikai függvény esetén a $Q \in \mathrm{TERM}_h$ specifikációs feltételnek (pl. a $\uparrow \in \mathrm{TERM}_h$ specifikációs feltételnek is).

Biz.: A feltétel és a 8.4. tétel szerint: $K \hookrightarrow_S\ fixpont_S$. A 6.28. tétel alapján bármely P logikai függvénnyel szűkíthetjük $\hookrightarrow_S$ baloldalát: $P \wedge K \hookrightarrow_S\ fixpont_S$. A 7.10. def. alapján $P ::= sp(s_0, \uparrow)$, ill. $P ::= sp(s_0, Q)$ választás mellett S megfelel a $\uparrow \in \mathrm{TERM}_h$, $Q \in \mathrm{TERM}_h$ specifikációs feltételeknek. □

Legyen $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. Ha választunk egy olyan t variáns függvényt, amelyre $K \wedge \neg\ fixpont_S \Rightarrow t > 0$, akkor $INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \Rightarrow K$ miatt: $INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \wedge \neg\ fixpont_S \Rightarrow t > 0$. Ha minden $t' \in \mathcal{N}$-re $(INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \wedge \neg\ fixpont_S \wedge t = t') \hookrightarrow_S (INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q) \wedge t < t') \vee fixpont_S$, akkor S megfelel a tetszőleges Q logikai függvény esetén a $Q \in \mathrm{TERM}_h$ specifikációs feltételnek a tétel szerint. A 6.15. tétel szerint $INV_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ éppen a program által elérhető állapotok halmaza. Ezért megfogalmazhatjuk az alábbi tételt:

8.7. Tétel. *(Biztosan fixpontba jut)*
Legyen $K \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. Válasszunk egy olyan t variáns függvényt, amelyre $K \wedge \neg\ fixpont_S \Rightarrow t > 0$.
Ha S megfelel a $\neg\ fixpont_S \wedge t = t' \hookrightarrow_h (t < t') \vee\ fixpont_S$ specifikációs feltételnek minden $t' \in \mathcal{N}$-re, akkor S megfelel a $\uparrow \in \mathrm{TERM}_h$ specifikációs feltételnek is, azaz biztosan fixontba jut.

8.1. Megjegyzés. A 8.7. tétel nem használható feladatok finomítására, mert csak akkor alkalmazható, ha a $fixpont_S$ állítás ismert, azaz a program adott. A 8.7. tétel helyességbizonyítási eszköz.

8.8. Tétel. *(A fixpontfeltétel finomítása)*
Ha S megfelel a $\mathrm{inv}_h P$, $\mathrm{FP}_h \Rightarrow R$ specifikációs feltételeknek és $P \wedge R \Rightarrow Q$, akkor megfelel a $\mathrm{FP}_h \Rightarrow Q$ specifikációs feltételnek is.

Biz.: A 6.32. és a 7.10. lemma alapján elegendő megmutatni, hogy S megfelel a $\mathrm{FP}_h \Rightarrow P \wedge R$ feltételnek. A feltétel és a 7.10. lemma szerint van olyan I invariáns, hogy $I \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ és $fixpont_S \wedge I \Rightarrow R$. 6.11. lemma szerint ekkor $I \wedge P \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$. Így található olyan $I' ::= I \wedge P$ invariáns, hogy $I' \in \mathrm{inv}_S(\bigwedge_{Q \in \mathrm{INIT}_h} Q)$ és $\mathrm{FP}_s \wedge I' \Rightarrow P \wedge R$. A 7.10. lemma alapján ezzel beláttuk, hogy S megfelel a $\mathrm{FP}_h \Rightarrow P \wedge R$ feltételnek. □

9. Fejezet

Programkonstrukciók

Ebben a fejezetben programok és feladatok konstrukciós módszereit ismertetjük, azaz megadjuk, hogy meglévő feladatokat hogyan bonthatunk részfeladatokra, illetve programokból milyen szabályok alapján hozhatunk létre összetett programot. Bevezetjük a kompozícionális modell fogalmát, majd megvizsgáljuk, hogy a programkonstrukciós szabályokkal kiegészített modell mennyiben felel meg a kompozícionalitás követelményének.

Feladatok és programok konstrukcióit, mint egy vagy több relációhoz egy relációt hozzárendelő leképezéseket definiáljuk.

*Megengedett konstrukciós művelet*nek nevezünk a továbbiakban minden olyan relációs műveletet, amely
- relációk uniója, metszete, különbsége,
- relációk adott pontban felvett értékének uniója, metszete, különbsége,
- relációk vetítése, kiterjesztései, rendezett reláció n-esek komponenseire bontása és komponensekből rendezett reláció n-esek előállítása,
- relációk direkt szorzata,
- relációk kompozíciója, szigorú kompozíciója és lezártjai (lezárt, korlátos lezárt, tranzitív, diszjunktív lezárt),

véges sokszori egymás utáni alkalmazásaként[1] megfogalmazható.

A modellben *feladatkonstrukció*nak nevezzük azokat a megengedett konstrukciós műveleteket, amelyek egy vagy több feladatból egy feladatot állítanak elő. *Programkonstrukció*nak nevezzük azokat a megengedett konstrukciós műveleteket, amelyek egy vagy több programból egy új, összetett programot állítanak elő.

9.1. Definíció. Kompozícionálisnak nevezünk egy programozási modellt, ha a modell minden $\mathcal{F}$ feladatkonstrukciójához létezik olyan $\mathcal{S}$ programkonstrukció, hogy S_1 megoldása F_1 és S_2 megoldása F_2 esetén $S_1 \mathcal{S} S_2$ megoldja $F_1 \mathcal{F} F_2$-t.

[1] A definíció nem formális. A modell keretein túlmutat annak vizsgálata, hogy a fent felsorolt elemi műveletek valamilyen értelembe teljes rendszert alkotnak-e.

9.2. Definíció. *Részlegesen kompozícionális* egy programozási modell, ha egyes kiválasztott $\mathcal{F}$ feladatkonstrukciók esetén megadható olyan $\mathcal{S}$ programkonstrukció, amelyre S_1 megoldása F_1 és S_2 megoldása F_2 esetén $S_1 \mathcal{S} S_2$ megoldja $F_1 \mathcal{F} F_2$-t.

A továbbiakban megmutatjuk, hogy a bevezetett modell részlegesen kompozícionális.

Jelöljük az S_1 program viselkedési relációját
$p(S_1)::=(\rhd_{S_1}, \mapsto_{S_1}, \hookrightarrow_{S_1}, \mathrm{FP}_{S_1}, \mathrm{TERM}_{S_1}, \mathrm{inv}_{S_1})$-vel, az S_2 program viselkedési relációját
$p(S_2)::=(\rhd_{S_2}, \mapsto_{S_2}, \hookrightarrow_{S_2}, \mathrm{FP}_{S_2}, \mathrm{TERM}_{S_2}, \mathrm{inv}_{S_2})$-vel.

9.1. Unió

9.3. Definíció. *(Unió)*
Legyen az A_1, A_2 tér az A állapottér altere. Jelölje B az A_1, A_2 terek legnagyobb közös alterét. Legyen S_1 az A_1, S_2 az A_2 altér felett definiált program kiterjesztése (6.22. def.) A-ra. $S_1 = (s_{1,0}, \{s_{1,1}, \ldots, s_{1,k}\})$, $S_2 = (s_{2,0}, \{s_{2,1}, \ldots, s_{2,m}\})$.
Az $S = (s_0, \{s_1, \ldots, s_k, s_{k+1}, s_{k+2}, \ldots, s_{k+m}\})$ programot, amely az A állapottéren adott, az S_1 és S_2 program *unió*jának nevezzük és $S_1 \cup S_2$-vel jelöljük, ha
minden B altérhez tartozó v_i változóra teljesül, hogy az $s_{1,0}$ és $s_{2,0}$ értékadások azonos értéket rendelnek hozzá, azaz: $F_{1,0_i} = F_{2,0_i}$ és
$s_0 = s_{1,0} \| s_{2,0}$,
$\forall i \in [1..k] : s_i = s_{1,i}$,
$\forall i \in [k+1..k+m] : s_i = s_{2,i-k}$.

9.1. Tétel. *(Unió viselkedési relációja)*
Legyen $S::=(S_1 \cup S_2)$. Ekkor[2]:

(1) $\rhd_S = \rhd_{S_1} \cap \rhd_{S_2}$

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

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

(4) $\forall Q$-ra, amelyre $sp(s_{1,0} \| s_{2,0}, Q) \Rightarrow sp(s_{1,0}, Q) \wedge sp(s_{2,0}, Q)$:
$\mathrm{inv}_{S_1}(Q) \cap \mathrm{inv}_{S_2}(Q) \subseteq \mathrm{inv}_S(Q)$[3]

(5) $fixpont_S = fixpont_{S_1} \wedge fixpont_{S_2}$

(6) $\mathrm{FP}_{S_1} \cap \mathrm{FP}_{S_2} \subseteq \mathrm{FP}_S$

[2] A tétel (1)-es,(2)-es és (5)-ös pontja a UNITY unió tételének relációs alakja [Cha Mis 89].

[3] Az állítás általánosítható:
$\forall Q, P$: ha $sp(s_{1,0} \| s_{2,0}, Q) \Rightarrow P$ and $P \in (\rhd_{S_1}^{(-1)}(\downarrow) \cap \rhd_{S_1}^{(-1)}(\downarrow))$, akkor $P \in \mathrm{inv}_S(Q)$.

(7) $((\rhd_{S_1} \cap \rhd_{S_2} \cap (\mapsto_{S_1} \cup \mapsto_{S_2}))^{tdl})^{(-1)}(fixpont_{S_1} \wedge fixpont_{S_2}) = \mathrm{TERM}_S$.

Biz.:
(1) A 6.26. és a 9.3. def. alapján: $wp(S_1 \cup S_2, P \vee Q) = \bigwedge_{s \in S_1 \cup S_2} wp(s, P \vee Q) =$
$= \bigwedge_{s \in S_1} wp(s, P \vee Q) \ \wedge \ \bigwedge_{s \in S_2} wp(s, P \vee Q) = wp(S_1, P \vee Q) \wedge wp(S_2, P \vee Q)$.

Tegyük fel, hogy $P \rhd_S Q$. Ekkor a 6.30. def. és a $wp(S_1 \cup S_2, P \vee Q) = wp(S_1, P \vee Q) \wedge wp(S_2, P \vee Q)$ egyenlőség felhasználásával:
$P \wedge \neg Q \Rightarrow wp(S_1, P \vee Q) \wedge wp(S_2, P \vee Q)$. A jobboldal gyengítésével: $P \wedge \neg Q \Rightarrow wp(S_1, P \vee Q)$ ill. $P \wedge \neg Q \Rightarrow wp(S_2, P \vee Q)$, azaz $P \rhd_{S_1} Q$ és $P \rhd_{S_2} Q$.
Ha $P \rhd_{S_1} Q$ és $P \rhd_{S_2} Q$, akkor $P \wedge \neg Q \Rightarrow wp(S_1, P \vee Q)$ és $P \wedge \neg Q \Rightarrow wp(S_2, P \vee Q)$, így $P \wedge \neg Q \Rightarrow wp(S_1, P \vee Q) \wedge wp(S_2, P \vee Q)$. A bizonyított egyenlőség és a 6.30. def. alapján: $P \rhd_S Q$.

(2) Tegyük fel, hogy $P \mapsto_S Q$.
A 6.31. def. szerint $P \rhd_{S_1 \cup S_2} Q \ \wedge \ \exists s \in S_1 \cup S_2 : P \wedge \neg Q \Rightarrow wp(s, Q)$
$\Longleftrightarrow$ { (1) és a 9.3. def. felhasználásával }
$P \rhd_{S_1} Q$, $P \rhd_{S_2} Q$ és
$(\exists s \in S_1 : P \wedge \neg Q \Rightarrow wp(s, Q) \ \vee \ \exists s \in S_2 : P \wedge \neg Q \Rightarrow wp(s, Q))$, azaz
$\Longleftrightarrow$ { 6.31. def. }
$P \rhd_{S_1} Q$ és $P \rhd_{S_2} Q$ és ($P \mapsto_{S_1} Q$ vagy $P \mapsto_{S_2} Q$).

(3) A 6.32. def. és (2) következménye.

(4) Tegyük fel, hogy $P \in inv_{S_1}(Q)$ és $P \in inv_{S_2}(Q)$. A 6.27. def. szerint ekkor $sp(s_{1,0}, Q) \Rightarrow P$ és $sp(s_{2,0}, Q) \Rightarrow P$. Ekkor $sp(s_{1,0}, Q) \wedge sp(s_{2,0}, Q) \Rightarrow P$ is teljesül, így a feltétel miatt $sp(s_{1,0} \| s_{2,0}, Q) \Rightarrow P$.
A 6.27. def. alapján $P \Rightarrow wp(S_1, P)$ és $P \Rightarrow wp(S_2, P)$. A 6.26. és 9.3. def. szerint $P \Rightarrow wp(S_1 \cup S_2, P)$.

(5) 6.34. def. és 9.3. def. közvetlen következménye.

(6) (5) következménye. Ha $fixpont_{S_1} \Rightarrow R$ és $fixpont_{S_2} \Rightarrow R$, akkor $fixpont_{S_1} \wedge fixpont_{S_2} \Rightarrow R$.

(7) A 6.36. def. és a (3),(5) állítások következménye. □

9.1. Megjegyzés. *($\exists S_1, S_2, Q : \mathrm{inv}_{S_1 \cup S_2}(Q) \neq \mathrm{inv}_{S_1}(Q) \cap \mathrm{inv}_{S_2}(Q)$)*
A 9.1. tétel (3) pontjának bizonyításához felhasználtuk, hogy $sp(s_{1,0}, Q) \Rightarrow P$ és $sp(s_{2,0}, Q) \Rightarrow P$ esetén $sp(s_{1,0}, Q) \wedge sp(s_{2,0}, Q) \Rightarrow P$ is teljesül. Ez a segédállítás nem megfordítható. A 9.1. példa segítségével megmutatjuk, hogy az összetett program invariánsa nem invariánsa az összetevőknek.

9.1. Példa. S_1::=$(x := 1, \{SKIP\})$. S_1::=$(y := 1, \{SKIP\})$. Q::= $\uparrow$.
P::=$(x = 1 \wedge y = 1)$. Könnyen ellenőrizhető, hogy $sp(s_{1,0}, Q) = (x = 1)$, $sp(s_{2,0}, Q) = (y = 1)$. $sp(s_{1,0}, Q) \wedge sp(s_{2,0}, Q) \Rightarrow P$, de $sp(s_{1,0}, Q) \not\Rightarrow P$ ill. $sp(s_{2,0}, Q) \not\Rightarrow P$. □

9.2. Megjegyzés. *($\exists S_1, S_2 : \mathrm{FP}_{S_1} \cap \mathrm{FP}_{S_2} \neq \mathrm{FP}_S$)*
Az előző megjegyzésben leírt gondolatmenethez hasonlóan belátható, hogy $fixpont_{S_1} \wedge fixpont_{S_2} \Rightarrow R \not\Longrightarrow fixpont_{S_1} \Rightarrow R$ és $fixpont_{S_2} \Rightarrow R$.

9.3. Megjegyzés. *($\hookrightarrow_{S_1 \cup S_2}$ és $\mathrm{TERM}_{S_1 \cup S_2}$)*
A $fixpont_{S_1 \cup S_2}$ logikai függvény, ill. a $\mapsto_{S_1 \cup S_2}$ reláció meghatározható az összetevő programok viselkedési relációjából, így 6.32. és a 6.36. def. alapján a $\hookrightarrow_{S_1 \cup S_2}$ ill. a $\mathrm{TERM}_{S_1 \cup S_2}$ reláció is kifejezhető közvetett módon az összetevő komponensek viselkedési relációjából a tétel (3) és (7) pontja szerint. Az unió viselkedési relációjának egyes komponensei, pl. $\hookrightarrow_{S_1 \cup S_2}$, ill. $\mathrm{TERM}_{S_1 \cup S_2}$ általában azonban nem határozhatóak meg az összetevő programok viselkedési relációjának *megfelelő* komponenseiből. Az alábbi példa (9.2.) szerint általában nem igaz, hogy $P \hookrightarrow_{S_1} Q$ és $P \hookrightarrow_{S_2} Q$ esetén $P \hookrightarrow_{S_1 \cup S_2} Q$ is teljesül.

9.2. Példa. *($\hookrightarrow_{S_1 \cup S_2}$)*
S_1::=$(SKIP, \{s_1 : x := x + 1\})$. S_2::=$(SKIP, \{s_2 : x := x - 1\})$.
P::=$(0 < x < 5)$. Q::=$(x \leq 0 \vee x \geq 5)$. A 6.32. def. alapján könnyen igazolható, hogy $P \hookrightarrow_{S_1} Q$ és $P \hookrightarrow_{S_2} Q$. A 6.31. tétel alapján, ha van olyan feltétlenül pártatlan ütemezésnek megfelelő végrehajtási útja $S_1 \cup S_2$-nek, amely mentén P-ből elkerülhető Q, akkor $P \hookrightarrow_{S_1 \cup S_2} Q$ nem teljesül. Válasszuk kezdőállapotnak az $x = 3$ állapotot. Az s_1, s_2, s_1, s_2.... ütemezés feltétlenül pártatlan és sohasem jut a program Q-beli állapotba. □

9.4. Definíció. *(Feladatok egyesítése)*
Legyen F_1 és F_2 közös állapot és paramétertéren[4] adott feladat.
$\forall b \in B$:
$(F_1 \sqcup F_2)(b)$::=$\{((\rhd_{h_1} \cap \rhd_{h_2}), (\rhd_{h_1} \cap \rhd_{h_2} \cap (\mapsto_{h_1} \cup \mapsto_{h_2})),$
$(\rhd_{h_1} \cap \rhd_{h_2} \cap (\mapsto_{h_1} \cup \mapsto_{h_2}))^{tdl}, ((\rhd_{h_1} \cap \rhd_{h_2} \cap (\mapsto_{h_1} \cup \mapsto_{h_2}))^{tdl})^{(-1)}(fixpont_S),$
$(\mathrm{FP}_{h_1} \cap \mathrm{FP}_{h_2}), (\mathrm{inv}_{h_1} \cap \mathrm{inv}_{h_2}), (\mathrm{INIT}_{h_1} \cup \mathrm{INIT}h_2)) \mid h_1 \in F_1(b), h_2 \in F_2(b)\}$.

9.2. Tétel. *(Unió levezetési szabálya)*
Legyen F_1 és F_2 az A állapottér és a B paramétertér felett megadott feladat. S_1 és S_2 az A állapottérre kiterjesztett programok, amelyek uniója értelmezett (9.3. def.). Ha S_1 megoldja F_1-et K mellett és S_2 megoldja F_2-t K mellett és $\forall b \in B : \forall h_1 \in F_1(b) : \forall h_2 \in F_2(b) : sp(s_{1,0} \| s_{2,0}, (\bigwedge_{Q \in \mathrm{INIT}_{h_1}} Q) \wedge (\bigwedge_{Q \in \mathrm{INIT}_{h_2}} Q)) \Rightarrow sp(s_{1,0}, (\bigwedge_{Q \in \mathrm{INIT}_{h_1}} Q)) \wedge sp(s_{2,0}, (\bigwedge_{Q \in \mathrm{INIT}_{h_2}} Q))$, akkor $S_1 \cup S_2$ megoldja $F_1 \sqcup F_2$-t.

[4]Ha az állapottér nem közös, akkor válasszunk egy olyan teret, amelynek mindkét állapottér altere. A feladatokat terjesszük ki a közös térre (5.3. def.).

Biz.: 7.14., 9.4. def. és 9.1. tétel következménye. □

Az unió levezetési szabálya (9.2. tétel) azt mondja ki, ha a modell szemantikája összefésüléses és az összetevők rendelkeznek egy közös *globális invariáns*sal [And 91], akkor az ezen invariáns tulajdonság mellett megoldott feladatok megfelelő kompozícióját az összetett program is megoldja. A 10.1. példán megmutatjuk, hogy az összefésüléses szemantika milyen lényeges az unió viselkedési relációjának meghatározhatóságában [Cha 90].

Bizonyos esetekben programok uniójának viselkedési relációja könnyen kifejezhető az összetevők viselkedési relációja alapján. Ez akkor lehetséges, ha az összetevők kölcsönhatása (*interferenciája*) az unió levezetési szabályában tett megszorításokon túl további tulajdonságokkal is jellemezhető.

A kölcsönhatás jellemzésének alkalmas módja lehet a *nyitott specifikáció*, amely az eredő program egyszerűbb (általában biztonságossági) tulajdonságaira tett kikötések segítségével tesz indirekt kikötéseket az egyik vagy mindkét összetevő tulajdonságaira. A nyitott specifikáció módszerét részletesen bemutatja Chandy és Misra [Cha Mis 89].

A szekvencia programkonstrukció esetében (9.7. def.) az unió olyan speciális esetét fogalmazzuk majd meg, ahol a unióhoz tartozó értékadásokat olyan diszjunkt csoportokba sorolhatjuk, hogy az állapottér egy alkalmasan megválasztott részhalmaza felett legfeljebb egyetlen csoporthoz tartozó értékadások hatásrelációi különböznek az identitástól. Az alábbi két tétel erre a speciális esetre vonatkozik.

9.3. Tétel. *(Unió és az állapottér részhalmazai)*
Legyen $S = S_1 \cup S_2$, π logikai függvény az A állapottéren oly módon, hogy $\forall s \in S_2 : p(s) \cap (\lceil \pi \rceil \times A) = id_A \cap (\lceil \pi \rceil \times A)$ és $\pi \rhd_{S_1} \downarrow$.
Ekkor

(1) ha $P \rhd_{S_1} Q$, akkor $P \wedge \pi \rhd_S Q \wedge \pi$,

(2) ha $P \mapsto_{S_1} Q$, akkor $P \wedge \pi \mapsto_S Q \wedge \pi$,

(3) ha $P \hookrightarrow_{S_1} Q$, akkor $P \wedge \pi \hookrightarrow_S Q \wedge \pi$.

Biz.:
A feltétel alapján: $\forall s \in S_2 : \forall Z : A \longmapsto \mathcal{L} :: p(s)(\lceil Z \wedge \pi \rceil) = \lceil Z \wedge \pi \rceil$, így: $Z \wedge \pi \Rightarrow wp(s, (Z \wedge \pi))$.
(1) A feltételek és 6.17. lemma szerint: $\forall s \in S_1 : P \wedge \pi \wedge \neg(Q \wedge \pi)) \Rightarrow wp(s, (P \wedge \pi) \vee (Q \wedge \pi))$ és $\forall s \in S_2 : P \wedge \pi \wedge \neg(Q \wedge \pi) \Rightarrow wp(s, P \wedge \pi \wedge \neg(Q \wedge \pi)) \Rightarrow wp(s, (P \wedge \pi) \vee (Q \wedge \pi))$.
(2) Az előző állítás szerint: $P \wedge \pi \rhd_S Q \wedge \pi$. A feltétel és 6.21. lemma szerint: $\exists s \in S_1 : P \wedge \pi \wedge \neg(Q \wedge \pi) \Rightarrow wp(s, Q \wedge \pi)$. Ha $s \in S_1$, akkor $s \in S$, így az állítást igazoltuk.
(3) Struktúrális indukcióval az induktív 6.32. def. alapján.
Alapeset: $P \hookrightarrow_{S_1} Q$-t 1 lépésben vezettük le $P \mapsto_{S_1} Q$-ból. Az előző állítás szerint

ekkor $P \wedge \pi \mapsto_S Q \wedge \pi$, az 6.32. def. szerint: $P \wedge \pi \hookrightarrow_S Q \wedge \pi$.
Indukciós lépés: a) eset: az utolsó lépésben a 6.32. def. (2) pontját, a tranzitivitást alkalmaztuk, azaz: $P \hookrightarrow_{S_1} Q_1$ és $Q_1 \hookrightarrow_{S_1} Q$. Az indukciós feltétel szerint: $P \wedge \pi \hookrightarrow_S Q_1 \wedge \pi$ és $Q_1 \wedge \pi \hookrightarrow_S Q \wedge \pi$. A 6.32. def. (tranzitivitás) alapján: $P \wedge \pi \hookrightarrow_S Q \wedge \pi$.
b) eset: az utolsó lépésben a 6.32. def. (3) pontját, a diszjunktivitást alkalmaztuk, azaz: $P = \exists m : m \in W :: P(m)$ és $\forall m : m \in W :: (P(m) \hookrightarrow_{S_1} Q)$. Az indukciós feltétel szerint: $\forall m : (m \in W :: (P(m) \wedge \pi \hookrightarrow_S Q \wedge \pi)$, amiből a 6.32. def. (diszjunktivitás alapján) $P \wedge \pi \hookrightarrow_S Q \wedge \pi$. □

A tétel egy kicsit módosított feltételekkel is kimondható. A (3)-as állítás megfogalmazásánál alkalmazzuk a nyitott specifikáció módszerét.

9.4. Tétel. *(Unió és az állpottér részhalmazai (2.))*
Legyen $S = S_1 \cup S_2$, π logikai függvény az A állapottéren oly módon, hogy $\forall s \in S_2 : p(s) \cap (\lceil \pi \rceil \times A) = id_A \cap (\lceil \pi \rceil \times A)$, $\pi \rhd_{S_1} Q$. Ekkor

(1) ha $P \rhd_{S_1} Q$, akkor $P \wedge \pi \rhd_S Q$,

(2) ha $P \mapsto_{S_1} Q$, akkor $P \wedge \pi \mapsto_S Q$,

(3) ha $\neg\pi \wedge \neg Q \rhd_{S_1} \downarrow$ és $P \hookrightarrow_{S_1} Q$, akkor $P \wedge \pi \hookrightarrow_S Q$.

Biz.:
A feltétel alapján: $\forall s \in S_2 : \forall Z : A \longmapsto \mathcal{L} :: p(s)(\lceil Z \wedge \pi \rceil) = \lceil Z \wedge \pi \rceil$, így: $Z \wedge \pi \Rightarrow wp(s, (Z \wedge \pi))$.
(1) A feltételek szerint: $\forall s \in S_1 : P \wedge \neg Q \Rightarrow wp(s, P \vee Q)$ és $\pi \wedge \neg Q \Rightarrow wp(s, \pi \vee Q)$. A 6.6. lemma alapján: $P \wedge \pi \wedge \neg Q \Rightarrow wp(s, (P \vee Q) \wedge (\pi \vee Q)) = wp(s, (P \wedge \pi) \vee Q)$ és $\forall s \in S_2 : P \wedge \pi \wedge \neg Q \Rightarrow wp(s, P \wedge \pi \wedge \neg Q) \Rightarrow wp(s, (P \wedge \pi) \vee Q)$.
(2) Az előző állítás szerint: $P \wedge \pi \rhd_S Q$. A feltétel szerint: $\exists s \in S_1 : P \wedge \neg Q \Rightarrow wp(s, Q)$. $P \wedge \pi \wedge \neg Q \Rightarrow P \wedge \neg Q$. Ha $s \in S_1$, akkor $s \in S$, így az állítást igazoltuk.
(3) Struktúrális indukcióval az induktív 6.32. def. alapján.
Alapeset: $P \hookrightarrow_{S_1} Q$-t 1 lépésben vezettük le $P \mapsto_{S_1} Q$-ból. Az előző állítás szerint ekkor $P \wedge \pi \mapsto_S Q$, a 6.32. def. szerint: $P \wedge \pi \hookrightarrow_S Q$.
Indukciós lépés: a) eset: az utolsó lépésben a 6.32. def. (2) pontját, a tranzitivitást alkalmaztuk, azaz: $P \hookrightarrow_{S_1} Q_1$ és $Q_1 \hookrightarrow_{S_1} Q$. A PSP tételt (9.8. lemma) a $\neg\pi \wedge \neg Q \rhd_{S_1} \downarrow$ feltételre és a $Q_1 \hookrightarrow_{S_1} Q$ indukciós feltételre alkalmazva: $Q_1 \wedge \neg\pi \wedge \neg Q \hookrightarrow_{S_1} Q \wedge \neg\pi \wedge \neg Q$. A jobboldal hamis, így a 9.9. lemma alkalmazásával azt kapjuk, hogy a baloldal is hamis, azaz: $Q_1 \Rightarrow (\pi \vee Q)$. Az indukciós feltétel szerint: $P \wedge \pi \hookrightarrow_S Q_1$. Beláttuk, hogy $Q_1 \Rightarrow Q_1 \wedge (\pi \vee Q)$, így a 6.24. lemma szerint $Q_1 \hookrightarrow_S (Q_1 \wedge \pi) \vee (Q_1 \wedge Q)$, a 6.29. lemma felhasználásával $Q_1 \hookrightarrow_S (Q_1 \wedge \pi) \vee Q$. Az indukciós feltétel szerint: $Q_1 \wedge \pi \hookrightarrow_S Q$. 6.24. lemma alapján: $Q \hookrightarrow_S Q$ A 6.32. def. (diszjunktivitás) alapján: $(Q_1 \wedge \pi) \vee Q \hookrightarrow_S Q$. A 6.32. def. (tranzitivitás) kétszeri alkalmazásával: $P \wedge \pi \hookrightarrow_S Q$.

b) eset: az utolsó lépésben a 6.32. def. (3) pontját, a diszjunktivitást alkalmaztuk, azaz: $P = \exists m : m \in W :: P(m)$ és $\forall m : m \in W :: (P(m) \hookrightarrow_{S_1} Q)$. Az indukciós feltétel szerint: $\forall m : (m \in W :: (P(m) \wedge \pi \hookrightarrow_S Q)$, amiből a 6.32. def. (diszjunktivitás alapján) $P \wedge \pi \hookrightarrow_S Q$. □

Az összetett program olyan programtulajdonságait is megfogalmazhatjuk, amelyek érvényessége csak olyan változóktól függ, amelyek csak az egyik összetevőben állnak a baloldalon. Az ilyen tulajdonságokat *lokálisaknak* nevezzük [Cha Mis 89].

Kimondjuk az általános lokalitás tételt ([UN 88-93]/17-90), a bizonyítás Singh, Misra és Knapp bizonyításai alapján a relációs modell eszközeivel is megkonstruálható. A tétel 4. állítása a nyitott specifikáció technikáját alkalmazza.

9.5. Tétel. *(Lokalitás tétel - általános alak)*
Legyen S_1 és S_2 közös állapottéren értelmezett program. $X ::= V(S_1) \cap V(S_2)$. Ha $VR(P) \subseteq V(S_1)$[5], akkor

(1) $P \rhd_{S_1} Q \Longrightarrow P \wedge (X = M) \rhd_{S_1 \cup S_2} Q \vee (X \neq M)$

(2) $P \mapsto_{S_1} Q \Longrightarrow P \wedge (X = M) \mapsto_{S_1 \cup S_2} Q \vee (X \neq M)$

(3) $P \hookrightarrow_{S_1} Q \Longrightarrow P \wedge (X = M) \hookrightarrow_{S_1 \cup S_2} Q \vee (X \neq M)$

(4) $P \hookrightarrow_{S_1} Q$ és $P \wedge (X = M) \rhd_{S_1 \cup S_2} (P \wedge (X < M)) \vee Q \implies P \hookrightarrow_{S_1 \cup S_2} Q.$

Biz.: Ha $VR(P) \subseteq V(S_1)$, akkor $VR(P) \cap V(S_2) \subseteq X$.
Lemma:[6] $P \wedge (X = M) \rhd_{S_2} (X \neq M)$.
Lemma bizonyítása[7]: Legyen $s \in S_2$. Ha $p(s)(\lceil P \wedge X = M \rceil) \subseteq \lceil X = M \rceil$, akkor $\forall a \in \lceil P \wedge X = M \rceil : \forall v \in VR(P) : v \circ p(s)(a) = v(a)$, így:
$p(s)(\lceil P(VR(P)) \wedge X = M \rceil) \subseteq \lceil P(VR(P)) \rceil$.
Ha $p(s)(\lceil P \wedge X = M \rceil) \not\subseteq \lceil X = M \rceil$, akkor
$p(s)(\lceil P(VR(P)) \wedge X = M \rceil) \cap \lceil X = M \rceil \subseteq \lceil P(VR(P)) \rceil$ és
$p(s)(\lceil P(VR(P)) \wedge X = M \rceil) \cap \lceil X \neq M \rceil \subseteq \lceil X \neq M \rceil$, azaz
$p(s)(\lceil P(VR(P)) \wedge X = M \rceil) \subseteq \lceil X \neq M \vee P \rceil$. A 6.10. lemma szerint $P \wedge X = M \Rightarrow wp(S_2, P \vee X \neq M)$, amelyből ekvivalens átalakítással: $P \wedge X = M \wedge X = M \Rightarrow wp(S_2, (P \wedge X = M) \vee X \neq M)$, azaz a 6.30. def. szerint: $P \wedge X = M \rhd_{S_2} X \neq M$. □
(1), (2), (3), (4) biz. megtalálható [UN 88-93]-ban. A bizonyítás a most bizonyított lemmára és a $\rhd_S$, $\mapsto_S$, $\hookrightarrow_S$ relációk korábban bizonyított tulajdonságaira épül. □

---

[5] $VR(P) \cap V(S_2) \subseteq X$

[6] Misra lokalitási axiómája

[7] $P(VR(P))$-vel azt a függvénykompozíciót jelöljük, amely a $VR(P)$ halmazhoz tartozó változókból, mint az állapottér projekciós függvényeiből, a $VR(P)$-ben nem szereplő változók helyett az identitásfüggvényből, ill. a P logikai függvényből állítható elő. Ha a $VR(P)$-hez tartozó függvények értéke változatlan, akkor $P(VR(P))$ függvénykompozíció értéke sem változik meg.

9.2. Szuperpozíció

9.5. Definíció. *(Szuperpozíció)*
Legyen az $S = (s_0, \{s_1, \ldots, s_m\})$ program az A állapottér A_1 altere és az s feltételes értékadás az A állapottér felett definiálva oly módon, hogy $VL(s)$ az A_1 altér egyetlen változóját sem tartalmazza. Jelölje $s_j \| s$ az s_j és az s utasítás szuperpozícióját (6.21. def.). Legyen az $S' = (s'_0, \{s'_1, \ldots, s'_m\})$ az S kiterjesztése A-ra (6.22. def.). Az
a) $(s'_0, \{s'_1, \ldots, s'_m, s\})$ ill. az
b) $S = (s'_0, \{s'_1, \ldots (s'_j \| s), \ldots s'_m\})$, ahol $i \in [1, m]$
alakú programokat az S program és az s utasítás szuperpozíciójának nevezzük.

9.6. Tétel. *(Szuperpozíció viselkedési relációja)*
Legyen az A állapottéren adott S'' program az A_1 alterén adott S program és az $s ::= \|_{i\in[1,n]}(v_i :\in F_i(v_1, .., v_n)$, ha $\pi_i)$. utasítás egy szuperpozíciója. Jelöljük az A_1 altéren adott P, Q logikai függvények A-ra való kiterjesztését P', Q'-vel. Ekkor[8]:

(1) $P \rhd_S Q \Longrightarrow P' \rhd_{S''} Q'$,

(2) $P \mapsto_S Q \Longrightarrow P' \mapsto_{S''} Q'$,

(3) $P \hookrightarrow_S Q \Longrightarrow P' \hookrightarrow_{S''} Q'$,

(4) $\forall Q : P \in \mathrm{inv}_S(Q) \Longrightarrow P' \in \mathrm{inv}_{S''}(Q')$,

(5) $fixpont_{S''} = fixpont'_S \wedge fixpont_s$,

(6) $R \in \mathrm{FP}_S \Longrightarrow R' \in \mathrm{FP}_{S''}$,

ahol $fixpont'_S$ a $fixpont_S$ logikai függvény kiterjesztése és
$fixpont_s ::= (\bigwedge_{i\in[1..n]}(\neg\pi \vee (\pi_{id} \wedge v_i = F_i(v_1, \ldots, v_n))))$.

Biz.:
(1), (2), (4) a 6.1. következmény és 6.9. lemma következménye.
(3) a (2)-es pontból 6.32. def. alapján ($P \hookrightarrow_S Q$ előállítása szerinti struktúrális indukcióval).
(5) 5.2. és 6.34. definíciókból közvetlenül adódik.
(6) az (5) állítás és 6.35. def. következménye. □

9.4. Megjegyzés. Ha a szuperpozíció a) típusú, akkor a tétel (1),(2),(3) állítása a lokalitás tétel következménye.

[8] A tétel (1)-es, (2)-es, (3)-as, (4)-es pontja a UNITY szuperpozíció tételének relációs alakja [Cha Mis 89].

9.6. Definíció. *(Feladat gyenge kiterjesztése)*
F'' az F feladat kiterjesztésének gyengítése, ha az F kiterjesztéséből, F'-ből, a $Q \in \mathrm{TERM}_h$ típusú specifikációs feltételek elhagyásával kapjuk.

9.7. Tétel. *(Szuperpozíció levezetési szabálya)*
Legyen F az A állapottér A_1 altere és a B paramétertér felett megadott feladat. Ha az S program megoldja az F feladatot, akkor az S program és az s utasítás bármely szuperpozíciója megoldja az F feladat gyenge kiterjesztését.

Biz.: A 7.1. definícióból és a 9.6. tételből következik. □

9.3. Szekvencia

9.7. Definíció. *(Szekvencia)*
Legyen $S_1 = (s_{1,0}, \{s_{1,1}, \ldots, s_{1,k}\})$ az A állapottér $A_1 = \overset{\times}{_{i\in I_1}} A_{1i}$ altere felett, $S_2 = (s_{2,0}, \{s_{2,1}, \ldots, s_{2,m}\})$ pedig az $A_2 = \overset{\times}{_{i\in I_2}} A_{2i}$ altér felett definiált program. Legyen u egy logikai változó, amelyekhez tartozó állapottérkomponensek nem tartoznak sem az A_1 sem az A_2 altérhez.
Jelöljük S^1-gyel az $\overset{\times}{_{i\in I_1}} A_{1i} \times \mathcal{L}$ altéren definiált $(s_0, \{s_1, \ldots, s_k\})$ programot, ahol
$s_0 = (s'_{1,0} \| u := hamis)$ ($\rightarrow$ 6.21. def.),
$\forall i \in [1..k] s_i = ((s'_{1,i}),$ ha $\neg u)$ ($\rightarrow$ 6.20. def.).
Jelöljük S^2-vel az $\mathcal{L} \overset{\times}{_{i\in I_2}} A_{2i}$ állapottéren definiált $(SKIP, \{s_{k+2}, \ldots, s_{k+m+1}\})$ programot, ahol $\forall i \in [k+2..k+m+1] : s_i = ((s'_{2,i-(k+2)+1}),$ ha $u)$.
Legyen $s_{k+1} = ((s'_{2,0} \| u := igaz),$ ha $\neg u \wedge fixpont_{S_1})$,
Az $S = (S^{1\prime} \cup S^{2\prime} \cup (SKIP, \{s_{k+1}\})')$ programot az S_1, S_2 programok *szekvenciájá*nak nevezzzük, és $S_1; S_2$-vel jelöljük.

A szekvenciát tehát feltételekkel kiegészített értékadások és unió segítségével definiáltuk.

Kimondunk néhány segédtételt:

9.8. Lemma. *(PSP)*
[9] Ha $P \hookrightarrow_S Q$ és $R \rhd_S B$, akkor $P \wedge R \hookrightarrow_S (Q \wedge R) \vee B$.
Biz.: struktúrális indukcióval.

9.9. Lemma. *(Csoda kizárása és $\hookrightarrow_S$)*
[10] Ha $P \hookrightarrow_S \downarrow$, akkor $P \equiv \downarrow$.
Biz.: struktúrális indukcióval.

[9] Chandy-Misra
[10] Chandy-Misra

9.10. Lemma. [11] Ha $P \subseteq Q$, akkor $inv_S(Q) \subseteq inv_S(P)$. Hasonlóan: $true_S(Q) \subseteq true_S(P)$.

A feltételek gyengíthetőek. Elegendő, ha $sp(s_0, P) \subseteq sp(s_0, Q)$, így bármely I, amelyet a Q kezdeti feltétel kezdetben igazzá tesz, P-ből indulva is teljesül. Ha $I \in inv_S$, akkor $I \rhd_S \downarrow$. A második állítás az első állításból és $INV_S(R) = \bigcap\{I | I \in inv_S(R)\}$-ból következik.

9.11. Lemma. [12] Tetszőleges S programra, ha $(P \wedge fixpont_S) \hookrightarrow_S Q$, akkor $(P \wedge fixpont_S) \subseteq Q$.

Az előző lemma és a PSP tétel (9.8. lemma) felhasználásával: $(fixpont_S \wedge \neg Q)$, azaz
$(P \wedge fixpont_S \wedge \neg Q) \hookrightarrow_S \downarrow$, így $(P \wedge fixpont_S \wedge \neg Q) = \downarrow$ a csoda kizárásának elve (9.9. lemma) miatt.

9.12. Tétel. *(Szekvencia viselkedési relációjáról)*
Legyen $S = S_1; S_2$. Ekkor[13]:

(1) ha $P \rhd_{S_1} fixpont_{S_1}$, akkor $P' \wedge \neg u \rhd_S fixpont'_{S_1} \wedge \neg u$,

(2) ha $P \mapsto_{S_1} fixpont_{S_1}$, akkor $P' \wedge \neg u \mapsto_S fixpont'_{S_1} \wedge \neg u$,

(3) ha $P \hookrightarrow_{S_1} fixpont_{S_1}$, akkor $P' \wedge \neg u \hookrightarrow_S fixpont'_{S_1} \wedge \neg u$,

(4) ha $P \rhd_{S_2} Q$, akkor $P' \wedge u \rhd_S Q' \wedge u$,

(5) ha $P \mapsto_{S_2} Q$, akkor $P' \wedge u \mapsto_S Q' \wedge u$,

(6) ha $P \hookrightarrow_{S_2} Q$, akkor $P' \wedge u \hookrightarrow_S Q' \wedge u$,

(7) $u \wedge fixpont'_{S_2} = fixpont_S$,

(8) Ha $R \in \mathrm{FP}_{S_2}$ akkor $R' \in \mathrm{FP}_S$.

(9) Ha $P \in inv_{S_1}(Q)$, akkor és csak akkor $P' \wedge u \in inv_S(Q')$; valamint, ha $P \in inv_{S_2}(Q)$, akkor és csak akkor $P' \wedge \neg u \in inv_S(Q')$.

(10) Ha $P \in inv_{S_1}(Q)$ és $P \in inv_{S_2}(P \wedge fixpont_{S_1})$, akkor és csak akkor $P' \in inv_S(Q')$.

(11) Ha $P \hookrightarrow_{S_1} Q$, akkor $(P' \wedge \neg u) \hookrightarrow_S (Q' \wedge \neg u)$.

[11] Kozsik T.
[12] Kozsik T.
[13] (9)-(12): Kozsik T.

(12) Ha $P \hookrightarrow_{S_1} Q$ és $P \hookrightarrow_{S_2} Q$, akkor $P' \hookrightarrow_S Q'$.

Biz.:
(1), (2), (3): A 5.2., 6.19., 6.30., 6.31. def. alapján könnyen belátható, hogy $P \triangleright_{S_1} Q \Longrightarrow P' \wedge \neg u \triangleright_{S^1} Q' \wedge \neg u$ és $P \mapsto_{S_1} Q \Longrightarrow P' \wedge \neg u \mapsto_{S^1} Q' \wedge \neg u$. Az utóbbiból $\hookrightarrow_{S_1}$ előállítása szerinti struktúrális indukcióval: $P \hookrightarrow_{S_1} Q \Longrightarrow P' \wedge \neg u \hookrightarrow_{S^1} Q' \wedge \neg u$. A szekvencia definíciója (9.7. def.) és 6.30. def. alapján ellenőrizhető, hogy: $(u \vee fixpont'_{S_1}) \wedge \neg(fixpont'_{S_1} \wedge \neg u) \triangleright_{S^1} \downarrow$, $\neg u \wedge \neg fixpont'_{S_1} \triangleright_{S^1} fixpont'_{S_1} \wedge \neg u$ és $p(s_{1,i}$, ha $\neg u) = p(s_{1,i}$, ha $\neg u \wedge \neg fixpont_{S_1})$, így alkalmazható a 9.4. tétel a $\pi ::= \neg u \wedge \neg fixpont'_{S_1}$, $Q ::= fixpont'_{S_1} \wedge \neg u$ megfeleltetéssel.
(4),(5),(6): A szekvencia (9.7. def.) és 6.30. def. alapján: $u \triangleright_{S^2} \downarrow$. Alkalmazható a 9.3. tétel a $\pi ::= u$ megfeleltetéssel.
(7) Ha $\neg u \wedge \mathrm{FP}_{S_1}$, akkor s_{k+1} miatt nem lehet fixpont. Ha $\neg u \wedge \neg \mathrm{FP}_{S_1}$, akkor S^1, ha $u \wedge \neg fixpont_{S_2}$, akkor S^2 változtat állapotot. Ezért $fixpont_S \Rightarrow u \wedge fixpont_{S_2}$. A másik irány a 6.34. definíció következménye.
(8) A (7)-es állítás következménye. □
(9),(10) A szekvencia programjára vonatkozó leggyengébb előfeltételek kiszámításával bizonyítható.
(11)
$P \hookrightarrow_{S_1} Q$ struktúrája szerinti indukcióval. Alapesetben a 9.11. lemmát alkalmazzuk.
(12) Az első feltételből és (11)-ből: $(P' \wedge \neg u) \hookrightarrow_S (Q' \wedge \neg u)$, így $(P' \wedge \neg u) \hookrightarrow_S Q'$. Hasonlóan (6) felhasználásával: $(P' \wedge u) \hookrightarrow_S Q'$, így $P' \hookrightarrow_S Q'$.□

9.13. Tétel. *(Szekvencia levezetési szabálya)*
Legyen F_1 és F_2 az A állapottér A_1 ill. A_2 altere és a B paramétertér felett értelmezett determinisztikus feladat, $S1; S2$ az A_1 altéren definiált S_1 és az A_2 altéren adott S_2 szekvenciája. Tetszőleges $b \in B$-re jelöljük $F_1(b)$ komponenseit ^{F_1}, $h \in F_2(b)$ komponenseit ^{F_2} indexszel.
Ha S_1 megfelel a $P \in \mathrm{TERM}_b^{F_1}$ és a $R \in \mathrm{FP}_b^{F_1}$ feltételnek a $P \in \mathrm{INIT}_b^{F_1}$ kezdeti feltétel mellett, S_2 megfelel $Q \in \mathrm{TERM}_b^{F_2}$ és $Z \in \mathrm{FP}_b^{F_2}$ feltételeknek a $Q \in \mathrm{INIT}_b^{F_2}$ feltétel mellett, és $R' \Rightarrow Q'$, akkor $S_1; S_2$ megfelel $P' \in \mathrm{TERM}_b$ és $Z' \in \mathrm{FP}_b$ feltételeknek a $P' \in \mathrm{INIT}_b$ kezdeti feltétel mellett[14].

Biz.: A feltételekből a 7.9. tétel szerint $sp(s_{1,0}, P) \wedge \mathrm{INV}_{S_1}(P) \hookrightarrow_{S_1} fixpont_{S_1}$. $sp(s_{1,0}, P) \Rightarrow \mathrm{INV}_{S_1}(P)$ (6.27. def.). a 9.12. tétel (3)-as pontja és a 6.27. tétel felhasználásával: $sp(s_{1,0}, P)' \wedge \neg u \wedge \neg fixpont'_{S_1} \hookrightarrow_S fixpont'_{S_1} \neg u$. $\mathrm{INV}_{S_1}(P)' \in \mathrm{inv}_{S^1}(P')$, így: $sp(s_{1,0}, P)' \wedge \neg u \wedge \neg fixpont'_{S_1} \hookrightarrow_S fixpont'_{S_1} \wedge \mathrm{INV}_{S_1}(P)' \wedge \neg u$.

[14] A tétel állítása Misra programszekvenciára vonatkozó - *nem bizonyított* - állításával rokon. Misra eredeti állítása a következőképpen fogalmazható meg a relációs modellben:
ha $s_{2,0} = SKIP$, $P \hookrightarrow_{S_1} Q \vee R$, $R \Rightarrow fixpont_{S_1}$ és S_2' megfelel a $P' \vee R' \hookrightarrow_h Q'$ specifikációs feltételnek a $fixpont'_{S_1} \in \mathrm{INIT}_h$ kezdeti feltétel mellett, akkor $S_1; S_2$ megfelel a $P' \hookrightarrow_h Q'$ specifikációs feltételnek ([UN 88-93]/16-90). Misra a szekvencia fogalmát nem definiálja formálisan, így a tételt sem bizonyítja. Műveleti szemantikai megfontolásokra és a helyettesítési axiómára hivatkozva indokolja az állítás helyességét és példákon keresztül mutatja meg, hogy használata helyes következtetések levonásához vezet.

$sp(s_1, P') = sp(s_{1,0}, P)' \wedge \neg u$ felhasználásával $sp(s_1, P') \wedge \neg fixpont'_{S_1} \hookrightarrow_S fixpont'_{S_1} \wedge \mathrm{INV}_{S_1}(P)' \wedge \neg u$.

7.11. tétel és 5.2. def. szerint $fixpont'_{S_1} \wedge \mathrm{INV}_{S_1}(P)' \Rightarrow R'$, azaz $sp(s_1, P') \wedge \neg fixpont'_{S_1} \hookrightarrow_S fixpont'_{S_1} \wedge \neg u \wedge R'$ (6.29. lemma).

Mivel $sp(s_1, P') \wedge fixpont'_{S_1} \Rightarrow INV_{S_1}(P)' \neg u \wedge fixpont'_{S_1} \Rightarrow R' \wedge \neg u \wedge fixpont'_{S_1}$, ezért a (6.24. lemma) és a 6.32. def. (diszjunktivitás) alkalmazásával: $sp(s_1, P') \hookrightarrow_S fixpont'_{S_1} \wedge \neg u \wedge R'$.

$\neg u \wedge fixpont'_{S_1} \wedge R' \mapsto_S sp(s_{k+1}, R') \wedge u$.

6.32. def. (tranzitivitás) alkalmazásával: $sp(s_1, P') \hookrightarrow_S sp(s_{k+1}, R') \wedge u$.

A fentihez hasonló gondolatmenet alapján $sp(s_{2,0}, Q) \hookrightarrow_{S_2} fixpont_{S_2} \wedge INV_{S_2}(Q)$, amelyből a 9.12. tétel (6)-os pontja felhasználásával: $sp(s_{2,0}, Q)' \wedge u \hookrightarrow_S fixpont'_{S_2} \wedge INV_{S_2}(Q)' \wedge u$. $sp(s_{k+1}, R') \Rightarrow sp(s_{2,0}, Q)' \wedge u$, így $sp(s_{k+1}, R') \wedge u \hookrightarrow_S fixpont{S_2}' \wedge INV_{S_2}(Q) \wedge u$. A 6.32. def. (tranzitivitás) alkalmazásával: $sp(s_1, P') \hookrightarrow_S fixpont{S_2}' \wedge u$. 9.12. tétel (7)-es és (8)-as állításaival a tétel állítását kapjuk. □

rész III

Programszintézis

11. Fejezet

Programozási tételek

Ebben a fejezetben általánosan megfogalmazott programozási feladatokat oldunk meg. A kapott megoldásokat programozási tételeknek nevezzük, mert széles körben alkalmazhatóak konkrét feladatok megoldása során. Ilyen alapfeladat például:

- asszociatív művelet eredményének párhuzamos kiszámítása (11.1. pont),
- elemenként feldolgozható (11.2. pont).

Megvizsgáljuk, hogy a kapott megoldások milyen architektúrákon implementálhatók hatékonyan. Olyan megoldásokat dolgozunk ki, amelyek osztott és aszinkron osztott memóriás rendszerekre is könnyen leképezhetőek.

11.1. Asszociatív művelet eredményének kiszámítása

Legyen H tetszőleges halmaz. $\circ : H \times H \longmapsto H$ tetszőleges kétoperandusú asszociatív alapművelet H-n.

$f : H^* \longmapsto H$ függvény. f a $\circ$ művelet egyszeri vagy ismételt alkalmazásának felel meg. $\circ$ asszocivitása miatt tetszőleges legalább 3 hosszú $x \in H^*$ sorozatra: $f(\ll x_1, \ldots, x_{|x|} \gg) = f(\ll f(\ll x_1, \ldots, x_{|x|-1} \gg), x_{|x|} \gg) = f(\ll x_1, f(\ll x_2, \ldots, x_{|x|} \gg) \gg)$. A továbbiakban a $(h_1 \circ h_2)$ helyett mindig $f(\ll h_1, h_2 \gg) - t$ írunk. f-et kiterjeszthetjük az egyetlen elemből álló sorozatokra is, legyen $f(\ll h \gg) = h$.

Adott $a \in H^*$, H-beli elemek véges, nem üres sorozata. Tegyük fel, hogy a sorozat egyes elemei közvetlenül elérhetőek. Az a sorozat elemeit a szokásostól eltérően fordított sorrendben indexeljük, utolsó elemét a_1-gyel jelöljük. Ha a sorozat hossza n, akkor a sorozat első eleme a_n. $a = \ll a_n, ..., a_1 \gg$, $(n \geq 1)$. Számítsuk ki a $\mathcal{G}_a : [1..n] \longmapsto H$ függvény értékét minden $i \in [1..n]$-re, ahol $n \geq 1$ és $\mathcal{G}_a(i) = f(\ll a_i, ..., a_1 \gg)$.

11.1.1. A feladat specifikációja

Reprezentáljuk az a sorozatot és a $\mathcal{G}_a$ függvényt egy-egy vektorral, amelyeket a-val, illetve g-vel jelölünk. A vektorok elemei H-beli értékek. Kikötjük, hogy fixpontban a g vektor i. eleme éppen $\mathcal{G}_a(i)$ legyen (11.3.), illetve a program biztosan elérje egy fixpontját (11.2.).

$$\begin{array}{lllll} A = & G \times & G, & \text{ahol} & G = vektor([1..n], H), \quad n \geq 1 \\ & g & a & & \\ B = & G & & & \\ & a' & & & \end{array}$$

$$(a = a') \in INIT_{a'} \tag{11.1.}$$

$$\uparrow\!\!\hookrightarrow \mathrm{FP}_{a'} \tag{11.2.}$$

$$\mathrm{FP}_{a'} \Rightarrow (a = a' \wedge \forall i \in [1..n] : g(i) = f(\ll a_i, ..., a_1 \gg)) \tag{11.3.}$$

Megállapíthatjuk, hogy a $\mathcal{G}_a$ függvény i helyen felvett értékének meghatározását megkönnyíti, ha ismerjük f értékét bármely $[u..v] \subseteq [i..1]$ intervallum elemeivel indexelt $f(\ll a_u, ..., a_v \gg)$ részsorozatra.[1] Megfigyelhetjük azt is, hogy egy részsorozatra kapott eredményt bármely a részsorozatot tartalmazó részsorozatra vonatkozó eredmény meghatározásánál hasznosíthatjuk.

Ezen gondolatmenet alapján bővítsük a feladat állapoterét és finomítsuk a specifikációt. Vezessük be a h függvényt oly módon, hogy $h(a, i, k)$ jelentse f értékét a azon részsorozatára, amelynek első eleme a_i és hossza vagy éppen 2^k vagy a_1 az utolsó eleme, ha $i < 2^k$. A gs kétdimenziós vektort azért vezetjük be, hogy segítségével a h függvényt változóval helyettesítsük [Fót 83]. A gs, k, t változók és h kapcsolatát invariáns állítással írjuk le (11.6.)-(11.8.). Ugyanezt jelenítjük meg szemléletes alakban a 11.1. ábrán. Az ábrán szereplő vonalak a gs mátrix elemei között fennálló kapcsolatot írják le a 11.2. lemma alapján, azaz $gs(i, k) = h(a, i, k)$, ha $k \leq k(i)$.

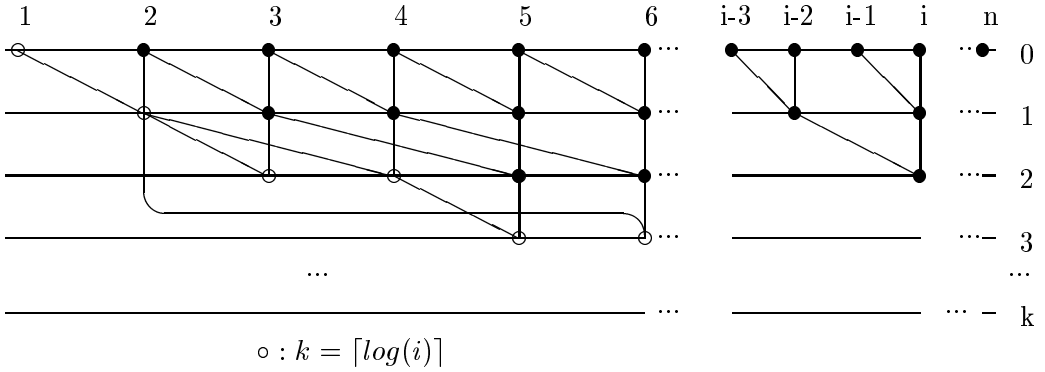
$$\begin{array}{lllll} A' = & G\times & G\times & GS\times & K\times \quad T \\ & g & a & gs & k \quad t \end{array}$$

$$\begin{array}{ll} G & = vektor([1..n], H), \\ GS & = vektor([1..n, 0..(\lceil \log(n) \rceil)], H) \\ K & = vektor([1..n], \mathcal{N}_0), \\ T & = vektor([1..n], \mathcal{N}_0), \quad n \geq 1 \end{array}$$

Megadjuk a $h : G \times [1..n] \times \mathcal{N}_0 \longrightarrow H$ parciális függvény pontos definícióját:

$$h(a, i, k) = \begin{cases} f(\ll a_i, ..., a_1 \gg), & \text{ha } i - 2^k + 1 \leq 1 \\ f(\ll a_i, ..., a_{(i-2^k+1)} \gg), & \text{ha } i - 2^k + 1 \geq 1. \end{cases}$$

[1] A lépésenkénti finomítás során alkalmazzuk a [Qui 87] 3-2. pontjában és [Cha Mis 89] 5. fejezetében ill. 6.9. pontjában bemutatott megoldási módszerek egyes elemeit.



11.1. ábra: Asszociatív művelet kiszámításának részeredményei,
a gs mátrix elemei között fennálló kapcsolatok

Válasszuk a $v : A \longmapsto \mathcal{N}_0$ variánsfüggvényt a következőképpen:

$$v = 4 * n * n - \sum_{i=1}^{n} \left(k(i) + \chi(k(i) = \lceil \log(i) \rceil \wedge g(i) = gs(i, k(i))) \right),$$

ahol $\chi : \mathcal{L} \longmapsto \{0, 1\}$. $\chi(igaz) = 1$, $\chi(hamis) = 0$. A variánsfüggvény lényegében azt adja meg, hogy a gs mátrix egyes oszlopaiban összesen hány olyan elem van, amely nem azonos a h függvény megfelelő helyen felvett értékével, illetve a g vektor értéke hány helyen különbözik a $\mathcal{G}_a$ függvény értékétől.

11.1. Lemma. *(Asszociatív művelet - a feladat finomítása)*
Az alábbi specifikáció finomítása az eredetinek:

$$(a = a') \in INIT_{a'} \tag{11.4.}$$

$$\uparrow\!\hookrightarrow \mathrm{FP}_{a'} \tag{11.5.}$$

$$\mathrm{FP}_{a'} \Rightarrow \forall i \in [1..n] : (k(i) = \lceil \log(i) \rceil) \ \wedge (g(i) = gs(i, \lceil \log(i) \rceil)) \tag{11.6.}$$

$$\mathrm{inv}_{a'}(\forall i \in [1..n] : k(i) \leq \lceil \log(i) \rceil \wedge \forall k : k \leq k(i) : gs(i, k) = h(a, i, k)) \tag{11.7.}$$

$$\mathrm{inv}_{a'}(\forall i \in [1..n] : t(i) = 2^{k(i)}) \tag{11.8.}$$

$$\mathrm{inv}_{a'}(a = a') \tag{11.9.}$$

Bizonyítás:
Fixpontban (11.6.) szerint $k(i) = \lceil \log(i) \rceil$ és $g(i) = gs(i, \lceil \log(i) \rceil))$, tehát a 8.8. lemma és (11.7.) alapján $g(i) = gs(i, \lceil \log(i) \rceil) = h(a, i, \lceil \log(i) \rceil)$. $2^{\lceil \log(i) \rceil} \geq i$, így h definícióját felhasználva $h(a, i, \lceil \log(i) \rceil) = f(\ll a_i, ..., a_1 \gg)$, így a (11.9.) invariáns tulajdonság és 8.8. lemma alkalmazásával igazoltuk, hogy a (11.6.),(11.7.),(11.9.) feltételek együttesen finomítják a (11.3.) feltételt. □

11.1. Megjegyzés. A variáns függvényre vonatkozó 8.7. tétel segítségével bizonyíthatjuk majd azt, hogy a program megfelel a (11.2.)=(11.5.) feltételnek. Ebben az értelemben a variáns függvény megválasztása is egy finomítási lépésként fogható fel.

11.2. Megjegyzés. A (11.8.) feltétel csak az új állapottérkomponensekre tesz kikötést, így ezt a feltételt nem kellett felhasználnunk a bizonyítás során.

11.2. Lemma.
Ha $(i - 2^k \geq 1)$, akkor $f(\ll h(a,i,k), h(a,i-2^k,k) \gg) = h(a,i,k+1)$.

Bizonyítás:
Tudjuk, hogy $i - 2^k \geq 1$, tehát $h(a,i,k) = f(\ll a_i, \ldots, a_{(i-2^k+1)} \gg)$. Ha $(i - 2^k) - 2^k + 1 \geq 1$, akkor $h(a, i-2^k, k) = f(\ll a_{(i-2^k)}, ..., a_{(i-2^k-2^k+1)} \gg)$. Ekkor f asszociativitása miatt $f(\ll h(a,i,k), h(i{-}2^k,k) \gg) = f(\ll a_i, ..., a_{(i-2^k+1)}, a_{(i-2^k)}, ..., a_{(i-2^k-2^k+1)}$ $h(a,i,k{+}1)$. Ha $(i{-}2^k){-}2^k{+}1 < 1$, akkor $h(a, i{-}2^k, k) = f(\ll a_{(i-2^k)}, ..., a_1 \gg)$. f asszocivitása miatt $f(\ll h(a,i,k), h(a,i{-}2^k,k) \gg) = f(\ll a_i, ..., a_{(i-2^k+1)}, a_{(i-2^k)}, ..., a_1 \gg =$ $h(a,i,k+1)$. □

11.1.2. A megoldás

11.3. Tétel. *(Asszociatív művelet kiszámításának tétele I.)*
A 11.1. program megfelel a (11.4.)-(11.9.) specifikációnak, azaz megoldja az asszociatív művelet eredménye kiszámításának feladatát.

$$
\begin{aligned}
s_0 :\quad & \mathop{\square}_{i=[1..n]} gs(i,0), t(i), k(i) := f(\ll a_i \gg), 1, 0 \\
S : \{\quad & \mathop{\square}_{i=[1..n]} gs(i,k(i)+1), t(i), k(i) := \\
& \quad \begin{cases} f(\ll gs(i,k(i)), & gs((i-t(i)),k(i)) \gg), 2*t(i), k(i)+1, \\ & \text{ha } (i - 2*t(i) + 1 \geq 1) \wedge \\ & \quad \wedge (k(i-t(i)) \geq k(i)) \\ f(\ll gs(i,k(i)), & gs(i-t(i), k(i-t(i))) \gg), \\ & \quad 2*t(i), k(i)+1, \\ & \text{ha } (i - t(i) \geq 1) \wedge (i - 2*t(i) + 1 < 1) \\ & \quad \wedge (k(i-t(i)) = \lceil \log(i-t(i)) \rceil \end{cases} \\
& \mathop{\square}_{i=[1..n]} g(i) := gs(i,k(i)) \text{ha } (k(i) = \lceil \log(i) \rceil) \\
\}\quad &
\end{aligned}
$$

11.1. absztrakt program: Asszociatív művelet I. változat

11.3. Megjegyzés. $\square_{i=[1,.n]}$ n utasítás rövidítése. Az egyes értékadásokat példányosítással kapjuk oly módon, hogy az általános alakban az i változót konkrét értékkel helyettesítjük.

Bizonyítás:
(11.9.): A programban a elemeire vonatkozó értékadás nincs. Így a (11.4.) kezdeti feltétel egyben invariáns tulajdonság is.
(11.8.): $sp(s_0, \uparrow)$-ben: $t(i) = 1$ és $k(i) = 0$, tehát a feltétel kezdetben teljesül. Az értékadások mindegyike együtt változtatja $k(i)$ és $t(i)$ értékét, ezért a (11.8.) feltétel invariáns tulajdonság.
(11.7.): $sp(s_0, \uparrow) \Rightarrow gs(i, k(i)) = h(a, i, k(i))$, mert $h(a, i, 0) = f(\ll a(i) \gg)$. $sp(s_0, \uparrow) \Rightarrow (k(i) \leq \lceil \log(i) \rceil)$, mert $k(i)$ kezdetben 0.
Értékadás leggyengébb előfeltételének meghatározása után elég azt megmutatni, hogy

- $(i - 2 * t(i) + 1 \geq 1) \wedge (k(i - t(i)) \geq k(i))$-ből és $\forall k : k \leq k(i) : gs(i, k) = h(a, i, k)$-ból következik az egyenlőség $k(i) + 1$-re is, azaz: $f(\ll gs(i, k(i)), gs(i - t(i), k(i)) \gg) = h(a, i, k(i) + 1)$ és $k(i) + 1 \leq \lceil \log(i) \rceil$
- $(i - t(i) \geq 1) \wedge (i - 2 * t(i) + 1 < 1) \wedge (k(i - t(i)) = \lceil \log(i) \rceil$-ből és $\forall k : k \leq k(i) : gs(i, k) = h(a, i, k)$-ból következik az egyenlőség $k(i) + 1$-re is, azaz: $f(\ll gs(i, k(i)), gs(i - t(i), (\lceil \log(i - t(i)) \rceil)) \gg) = h(a, i, k(i) + 1)$ és $k(i) + 1 \leq \lceil \log(i) \rceil$.

$(i - 2 * t(i) + 1 \geq 1) \wedge (t(i) \geq 1) \Rightarrow (i - t(i) \geq 1) \Rightarrow k(i) \leq \log(i - 1) < \log(i) \leq \lceil \log(i) \rceil)$.
Az első esetben: $k(i) \leq k(i)$ miatt $gs(i, k(i)) = h(a, i, k(i))$, $(k(i - t(i)) \geq k(i))$ miatt $gs(i - t(i), k(i)) = h(a, i - t(i), k(i))$. A második esetben: $k(i) \leq k(i)$ miatt $gs(i, k(i)) = h(a, i, k(i))$, $k(i - t(i)) = \lceil \log(i - t(i)) \rceil$ miatt $gs(i - t(i), (\lceil \log(i) \rceil)) = h(a, i - t(i), (\lceil \log(i) \rceil))$. Mindkét esetben a 11.2. lemma alkalmazásával kapjuk a bizonyítandó állítást.
A 6.10. megj. szerint (11.7.),(11.8.) és (11.9.) feltételeinek konjunkciója invariáns tulajdonság.
(11.5.): (11.7.),(11.8.),(11.9.) feltételeinek konjunkciójából következik, hogy $\forall i \in [1..n] : k(i) \leq n$, így: $v > 0$. A 8.7. lemma szerint elegendő belátni, hogy a program minden utasítására igaz, hogy vagy pontosan 1-gyel csökkenti a variáns függvényt, vagy nem okoz állapotváltozást. Ha a program nincs fixpontban, akkor van olyan $i \in [1..n]$ és megfelelő értékadás, amely $k(i)$ értékét növeli, vagy van olyan i, hogy $k(i) = \lceil \log(i) \rceil$ és $g(i)$ még nem vette fel a $gs(i, (\lceil \log(i) \rceil))$ értéket.
(11.6.): a fixpont definíciója (6.34.,7.12. def.) alapján

$$\forall i \in [1..n] : \\ (k(i) = \lceil \log(i) \rceil) \rightarrow g(i) = gs(i, k(i)) \ \wedge \qquad (11.10.)$$

$$((i - 2 * t(i) + 1 < 1) \vee (k(i - t(i)) < k(i)) \ \wedge \tag{11.11.}$$
$$(i - t(i) < 1) \vee (i - 2 * t(i) + 1 \geq 1) \vee$$
$$\vee (k(i - t(i)) \neq \lceil \log(i - (t(i)) \rceil)) \tag{11.12.}$$

Ebből i szerinti indukcióval $\forall i \in [1..n] : (k(i) = \lceil \log(i) \rceil)$. $i = 1$-re: (11.7.)-ből és $sp(s_0, \uparrow)$-ból következik $(k(1) = \lceil \log(1) \rceil)$. Tegyük fel, hogy $\forall j < i : (k(j) = \lceil \log(j) \rceil)$. $t(i) \geq 1$ ezért $(k(i - t(i)) \neq \lceil \log(i - (t(i)) \rceil)$ ellentmond az indukciós feltételnek. Így (11.12.) az $(i - t(i) < 1) \vee (i - 2 * t(i) + 1 \geq 1)$ feltétellel helyettesíthető.
Ha $(i - 2 * t(i) + 1 \geq 1)$, akkor $k(i - t(i)) < k(i)$, különben (11.11.) nem teljesül. Az indukciós feltétel szerint $t(i) \geq 1$ miatt $k(i - t(i)) = \lceil \log(i - t(i)) \rceil$, tehát: $\lceil \log(i - t(i)) \rceil < k(i)$. Ez azonban ellentmond a $(i - 2 * t(i) + 1 \geq 1)) \Rightarrow (i - t(i) - t(i) + 1 \geq 1)) \Rightarrow \lceil \log(i - (t(i)) \rceil \geq k(i)$ kezdeti feltételnek. Tehát: $(i - 2 * t(i) + 1 < 1))$.
$(i - 2 * t(i) + 1 < 1)) \Rightarrow (i - t(i) < 1)$, különben (11.12.) nem teljesül. $(i - t(i) < 1) \Rightarrow k(i) \geq \lceil \log(i) \rceil$. A (11.7.) feltétel (invariánstulajdonság része) miatt $k(i) = \lceil \log(i) \rceil$. Ekkor (11.10.) alapján: $g(i) = gs(i, k(i)) = gs(i, \lceil \log(i) \rceil)$ is.

11.1.3. Programtranszformáció

Tegyük fel, hogy $\Theta(n)$ processzor párhzamosan hajtja végre a kapott programot. A vektorok i. komponenseire vonatkozó értékadásokat az i. logikai processzorra képezhetjük le. A variáns függvény definiciójából és a fenti bizonyításból közvetlenül adódik, hogy a program legkésőbb $O\lceil \log(n) \rceil$ állapotváltozás után fixpontba jut. Az egyes logikai processzorok egymáshoz képest aszinkron és szinkron módon is működhetnek.

A program jelenlegi alakjában azonban még nem felel meg sem a finom atomicitás szabályának [And 91](2.4), sem az osztott változós sémának [Cha Mis 89], ezért további komponensekkel bővítjük az állapotteret. Célszerű a log függvényt is kitranszformálni.

Jelöljük gst(i)-vel $gs(i - t(i), k(i))$, $kt(i)$-vel $k(i - t(i))$,
$gstk(i)$-vel $gs(i - t(i), kt(i))$ értékét, ha az szükséges és ismert az i. processzor számára és $kt(i)$ értéke elegendően nagy ahhoz, hogy a gs mátrix i. oszlopának következő $(k(i)+1.)$ elemét meghatározhassuk (11.13.). Vezessük be a $ktf(i), gstf(i), gstkf(i)$ logikai vátozókat a segédvektorok kezelésének megkönnyítésére. A segédvektorok i. komponense lokális az i. processzorra nézve. A transzformált program esetén teljesül majd, hogy minden egyes értékadásban pontosan legfeljebb egy olyan változóra (vektorkomponensre) hivatkozunk, amely nem lokális az i. processzorra nézve.

A specifikáció finomítása

A (11.4.)-(11.9.) specifikációt bővítjük az alábbi invariánsokkal:

$$\text{inv}_{a'} \qquad \forall i \in [1..n]: \quad (kt(i) \leq k(i - t(i)) \ \wedge$$
$$ktf(i) \rightarrow (kt(i) \geq k(i) \vee kt(i) = l(i - t(i)))) \qquad (11.13.)$$
$$\text{inv}_{a'} \qquad \forall i \in [1..n]: (gstf(i) \rightarrow ktf(i) \wedge (i - 2 * t(i) + 1 \geq 1)$$
$$\wedge gst(i) = gs(i - t(i), k(i))) \qquad (11.14.)$$
$$\text{inv}_{a'} \quad \forall i \in [1..n]: (gstkf(i) \rightarrow ktf(i) \wedge (i - t(i) \geq 1) \wedge (i - 2 * t(i) + 1 < 1)$$
$$\wedge gstk(i) = gs(i - t(i), kt(i)) = gs(i - t(i), k(i - t(i)))) \qquad (11.15.)$$
$$\text{inv}_{a'} \qquad \forall i \in [1..n]: \lceil \log(i) \rceil = l(i) \qquad (11.16.)$$

A transzformált program

11.4. Tétel. *(Asszociatív művelet kiszámításának tétele II.)*
A 11.2. program megfelel a (11.4.)-(11.9.),(11.13.)-(11.16.) specifikációnak.

Bizonyítás: az (11.13.)-(11.16.) invariánsok teljesülését a leggyengébb előfeltételek és $sp(s_0, \uparrow)$ kiszámolásával könnyen igazolhatjuk. A (11.5.),(11.7.)-(11.8.) kikötések a transzformált programra is teljesülnek, mert a kikötésekben szereplő változókra vonatkozó értékadások a (11.13.)-(11.16.) invariáns állítások miatt ekvivalensek az eredetiekkel. A (11.6.) fixpontfeltétel teljesüléséhez azt kell megmutatni, hogy ha a transzformált program fixpontba jut, akkor az eredeti is fixpontban van és a (11.10.)-(11.12.) feltételek teljesülnek. □

11.1.4. Hatékonyság és általánosság

A fenti megoldás egyszerűen implementálható szinkron, aszinkron arhitektúrán is, és osztott rendszerben is [Cha Mis 89]. Szinkron arhitektúra esetén egyszerűsíthető a megoldás, kevesebb új változó bevezetésével is megoldható a feladat. Osztott rendszer esetén csak akkor hatékony ez a megoldás, ha elegendően sok, legalább $\Omega(\lceil \log(n) \rceil)$ csatorna áll rendelkezésre processzoronként és a kommunikációs költség alacsony. Ilyen architektúra pl. a *hiperkocka* [Qui 87]. Adatcsatornás megoldásra mutat hatékony megoldást [Loy Vor 90].

A tétel segítségével nagyon sok klasszikusnak számító feladatot oldhatunk meg egyszerű *visszavezetés*sel[2] [Fót 86]. Pl.: párhuzamos összeadás, emelkedő számsorozatok összehasonlítása [Cha Mis 89], stb.

[2] Az asszociatív függvény kiszámításának tételét eddig már kb. 200 egyetemi hallgató alkalmazta a gyakorlatban is sikeresen konrét feladatok megoldása során. PowerXplorer típusú, 16 processzoros, párhuzamos számítógépen, PVM-ben implementált aszinkron, párhuzamos, konkrét program futási ideje a felhasznált processzorok számának emelése mellett egyre rövidebb, ha a $\circ$ művelet elvégzéséhez szükséges processzoridő elegendően nagy a kommunikációs költségekhez képest.

$$s_0: \quad \overset{\square}{_{i=[1..n]}} gs(i,0), t(i), k(i), l(i), ktf(i), gstkf(i), gstf(i), kt(i) := f(\ll a_i \gg), 1, 0, \lceil \log(i) \rceil, hamis, hamis, hamis, 0$$

$$S: \{ \quad \overset{\square}{_{i=[1..n]}} kt(i) := k(i - t(i)), \text{ha } \neg ktf(i) \wedge (i - t(i)) \geq 1$$

$$\overset{\square}{_{i=[1..n]}} ktf(i) := igaz, \text{ha } \neg ktf(i) \wedge (i - t(i)) \geq 1 \wedge \wedge (kt(i) \geq k(i) \vee kt(i) = l(i - t(i)))$$

$$\overset{\square}{_{i=[1..n]}} gst(i), gstf(i) := gs(i - t(i), k(i)), igaz, \text{ ha } ktf(i) \wedge (i - 2 * t(i) + 1 \geq 1) \wedge (kt(i) \geq k(i)) \wedge \neg gstf(i)$$

$$\overset{\square}{_{i=[1..n]}} gstk(i), gstkf(i) := gs(i - t(i), kt(i)), igaz, \text{ ha } ktf(i) \wedge (i - t(i) \geq 1) \wedge (i - 2 * t(i) + 1 < 1) \wedge (kt(i) = l(i - t(i)) \wedge \neg gstkf(i)$$

$$\overset{\square}{_{i=[1..n]}} gs(i, k(i) + 1), t(i), k(i), ktf(i), gstf(i), gstkf(i), kt(i) :=$$

$$\begin{cases} f(\ll gs(i,k(i)), & gst(i) \gg), 2 * t(i), k(i) + 1, hamis, hamis, hamis, 0 \\ & \text{ha } gstf(i) \\ f(\ll gs(i,k(i)), & gstk(i) \gg), 2 * t(i), k(i) + 1, hamis, hamis, hamis, 0 \\ & \text{ha } gstkf(i) \end{cases}$$

$$\overset{\square}{_{i=[1..n]}} g(i) := gs(i, k(i)), \quad \text{ha } (k(i) = l(i))$$

$$\}$$

11.2. absztrakt program: Asszociatív művelet II. változat

11.2. Elemenként feldolgozható függvények

Legyen H egy tetszőleges halmaz. Az elemenkénti feldolgozás tárgyalása során[3] az alábbi jelöléseket használjuk:

$$\begin{aligned} X ::= \;& X_1 \times ... \times X_n, X_i \subseteq \mathcal{P}(H)\ (i \in [1,..,n]), \\ Y ::= \;& Y_1 \times ... \times Y_m, Y_j \subseteq \mathcal{P}(H)\ (j \in [1,..,m]), \\ & x, \overline{x}, \overline{\overline{x}} \in X. \end{aligned}$$

11.1. Definíció. *(Teljesen diszjunkt felbontás)*
$\overline{x}, \overline{\overline{x}}$ az $x \in X$ *teljesen diszjunkt felbontás*a, ha $\forall i \in [1,n] : x_i = \overline{x}_i \cup \overline{\overline{x}}_i$ és $\forall i,j \in [1,n] : \overline{x}_i \cap \overline{\overline{x}}_j = \emptyset$.

11.2. Definíció. *(Elemenként feldolgozható függvény)*
Legyen $f : X \longmapsto Y$ függvény. Ha minden $x \in X$ bármely $\overline{x}, \overline{\overline{x}}$ teljesen diszjunkt felbontására

$$f(\overline{x}) \cup f(\overline{\overline{x}}) = f(x), \tag{11.17.}$$
$$f(\overline{x}) \cap f(\overline{\overline{x}}) = \emptyset, \tag{11.18.}$$

akkor f *elemenként feldolgozható függvény* [Fót 83].

11.2.1. A feladat specifikációja

Kikötjük, hogy bármely fixpontban az y rendezett halmaz m-es értéke éppen $f(x')$ legyen (11.21.), ahol x' az x változó kezdeti értéke (11.19.). Megköveteljük, hogy a program biztosan elérje valamelyik fixpontját (11.20.).
$A = X \times Y \quad x : X, y : Y,\ B = X \quad x' : X.$

$$(x = x') \in INIT_{x'} \tag{11.19.}$$
$$\uparrow\!\hookrightarrow \mathrm{FP}_{x'} \tag{11.20.}$$
$$\mathrm{FP}_{x'} \Rightarrow y = f(x'), \tag{11.21.}$$

ahol f elemenként feldolgozható.

11.4. Megjegyzés. $\forall j \in [1..m] : y_j = f_j(x'_1, ..., x'_n)$.

[3] Az asszociatív művelet eredményének kiszámításakor minden egyes finomítási lépést, ill. az absztrakt program helyességét is részletes számításokkal bizonyítottuk. Az elemenkénti feldolgozás tárgyalása során a feladat részfeladatokra bontásának bemutatására és a különböző programkonstrukciók alkalmazására helyezzük a hangsúlyt. Az egyes lépések bizonyítása és az absztrakt program helyességének igazolása az előző tételben bemutatott módszerekkel könnyen elvégezhető.

Feltesszük, hogy x és $x' \backslash x$ invariáns[4] módon az x' teljesen diszjunkt felbontása és $x' \setminus x$-re már ismerjük az f függvény értékét. Felhasználva az elemenként feldolgozható függvények azon tulajdonságát, hogy értékük az argumentum teljesen diszjunkt felbontása esetén komponensenkénti diszjunkt unióval meghatározható a (11.19.)-(11.21.) specifikáció fixpont feltételét az $x_i = \emptyset$ feltétellel helyettesíthetjük.

11.5. Lemma. *(Elemenkénti feldolgozás - a feladat finomítása)*
Az alábbi specifikáció finomítása a (11.19.)-(11.21.) feltételekkel megadottnak.

$$(x = x') \in INIT_{x'} \tag{11.22.}$$
$$\uparrow\hookrightarrow \mathrm{FP}_{x'} \tag{11.23.}$$
$$\mathrm{FP}_{x'} \Rightarrow \forall i \in [1..n] : (x_i = \emptyset) \tag{11.24.}$$
$$inv_{x'}(\forall j \in [1,m] : (y_j \cup f_j(x_1, ..., x_n) = f_j(x'_1, ..., x'_n))) \tag{11.25.}$$
$$inv_{x'}(\forall j \in [1,m] : (y_j \cap f_j(x_1, ..., x_n) = \emptyset)) \tag{11.26.}$$
$$inv_{x'}(\forall i,j \in [1,n] : (x'_i \setminus x_i) \cap x_j = \emptyset), \tag{11.27.}$$

ahol f elemenként feldolgozható.

Biz.: A 8.8. lemma alapján. A bizonyításhoz szükséges matematikai meggondolások indoklása megtalálható [Fót 83]-ban.

11.2.2. A megoldás

Definiáljuk az sl függvényt a következőképpen: $sl : \mathcal{P}(\{1, \ldots, n\}) \times H \longmapsto Y$,

$$sl(\{i_1, \ldots, i_k\}, e)_i ::= \begin{cases} \{e\}, & \text{ha } i \in \{i_1, \ldots, i_k\} \\ \emptyset, & \text{ha } i \notin \{i_1, \ldots, i_k\}, \end{cases}$$

ahol $i_1, \ldots, i_n$ az $1, \ldots, n$ számok egy permutációja.

Jelöljük a p és $\{e\}$ halmazok unióját $p \tilde{\cup} \{e\}$-vel, ha $e \notin p$. Hasonlóan, jelölje $p \tilde{-} \{e\}$ a $p \setminus \{e\}$ halmazt, ha $e \in p$. A $e : mem\ p$ művelet egy nemdeterminisztikus feltételes értékadás, amely e-nek értékül adja p egy tetszőlegesen kiválasztott elemét, ha p nem üres.

11.6. Tétel. *(Elemenkénti feldolgozás)*
A 11.3. absztrakt program megoldja az elemenként feldolgozható függvény értéke kiszámításának feladatát, azaz megfelel a (11.22.)-(11.27.) specifikációnak.

Biz.: A bizonyításhoz szükséges matematikai meggondolások indoklása megtalálható [Fót 83]-ban. □

[4] A halmazkivonás komponensenként értendő.

$$s_0: \quad \mathop{\Box}_{j=[1..m]} y_j := \emptyset \quad \Box ch := hamis$$

$$S: \{ \quad \mathop{\Box}_{i=[1..n]} e : mem\ x_i \| ch := igaz, \quad \text{if } x_i \neq \emptyset \wedge \neg ch$$

$$ch, x_{i_1}, \ldots, x_{i_k}, y := \begin{cases} \ldots \\ hamis, & x_{i_1} \tilde{-} \{e\}, \ldots, x_{i_k} \tilde{-} \{e\}, \quad y \tilde{\cup} f(sl(\{i_1 \ldots i_k\}, e)) \\ & \text{ha } e \in x_{i_1} \wedge \ldots \wedge e \in x_{i_k} \wedge e \notin x_{i_{k+1}} \wedge \ldots \wedge e \notin x_{i_n} \\ & \quad \wedge ch \\ \ldots \end{cases}$$

$$\}$$

Az elágazások száma $2^n - 1$.

11.3. absztrakt program: Elemenkénti feldolgozás

11.3. Definíció. Megfigyelhetjük, hogy az elemenkénti feldolgozás 11.3. programja az $U ::= \bigcup_{i \in [1..n]} \{x_i\}$ halmaz számosságával arányos számú állapotváltozás után jut fixpontba. A továbbiakban, amikor x *méretéről* beszélünk, akkor U számosságára gondolunk. Jelöljük x méretét $|x|$-szel.

11.2.3. Teljesen diszjunkt felbontás párhuzamos előállítása

Reprezentáljuk a továbbiakban az x_i halmazokat olyan szigorúan növekvő monoton sorozatok formájában, amelyeknek elemei és részsorozatai közvetlenül hozzáférhetőek az indexek megadásával[5]. x_i első, legkisebb elemét jelölje $x_i(1)$. x_i hosszát $x_i.dom$-mal jelöljük. $x_i[i,j]$-vel jelöljük x_i azon részsorozatát, amely a $k : k \in (i,j]$ indexű elemeket tartalmazza.

Feltételezve, hogy $\Theta(n)$ processzor áll rendelkezésünkre, felbontjuk x-et n páronként teljesen diszjunkt részre. A felbontás kiegyensúlyozott, ha a legnagyobb és a legkisebb rész méretének különbsége legfeljebb 1. Kiegyensúlyozott felbontás esetén n processzor segítségével kb. n-szeresére gyorsíthatjuk f kiszámítását. Az egyes részekre kiszámított részeredményeket végül komponensenkénti diszjunkt unióval egyesíthetjük (11.2. def.).

Megadjuk a páronként diszjunkt felbontás feladatának formális specifikációját:

$A = X \times M \quad x : X, m : M,\ B = X \quad x' : X,$
$M = vektor([1..n, 0..n], \mathcal{N}_0)$

[5] Függvény típusú reprezentációt választunk [Fót 86].

$$(x = x') \in INIT_{x'} \tag{11.28.}$$

$$\uparrow\!\hookrightarrow \mathrm{FP}_{x'} \tag{11.29.}$$

$$\mathrm{FP}_{x'} \Rightarrow cdd(m, x) \wedge (x = x'), \tag{11.30.}$$

ahol $cdd(m, x)$ igaz, ha az m mátrix x páronként teljesen diszjunkt felbontását definiálja, azaz: $cdd(m, x) = \forall i, j \in [1, n] : \forall k, l \in [0, n-1] : k \neq l \rightarrow x_i[m(i, k), m(i, k+1)] \cap x_j[m(j, l), m(j, l+1)] = \emptyset$.

Ahhoz, hogy egy páronként teljesen diszjunkt felbontásról eldönthessük, hogy kiegyensúlyozott-e, meg kell határoznunk nem feltétlenül diszjunkt halmazok uniójának számosságát. Nem diszjunkt halmazok uniója azonban elemenként feldolgozható függvény. Ha az unió elemeinek számát az unió elemenkénti feldolgozással történő kiszámítása útján határozzuk meg, akkor önmagában annak eldöntése, hogy a felbontás kiegyensúlyozott-e ugyanannyi számítási lépést igényel, mint a teljes elemenkénti feldolgozás. Ez a módszer nyilvánvalóan nem vezet eredményre. Nyitott kérdés, hogy más úton lehetséges-e n (esetleg n^2) processzorral páronként teljesen diszjunkt és egyben kiegyensúlyozott felbontást hatékonyan előállítani.

A továbbiakban megmutatjuk hogyan lehet hatékonyan páronként teljesen diszjunkt felbontást előállítani anélkül, hogy a kiegyensúlyozottságot garantálnánk[6]. A felbontást a legnagyobb komponens egyenlő részekre osztásával definiáljuk, azaz ezen egyetlen komponens felosztását terjesztjük ki fokozatosan a többire oly módon, hogy a teljesen diszjunkt felbontás létrejöjjön.

Tegyük fel, hogy x_1 az x legnagyobb elemszámú komponense[7]. Bevezetjük a t vektort, amely tájékoztat arról, hogy a páronként teljesen diszjunkt felbontás mely összetevőit ismerjük. Ezen összetevők együttesét *x részlegesen meghatározott páronként teljesen diszjunkt felbontásának* nevezzük.

$t : vektor([1..n], \{0, .., n-1\})$. Jelöljük $pccd(m, t, x)$-vel, ha az m mátrix elemeivel és a t vektor értékeivel meghatározott, az $\{m(i, j) | j \leq t(i)\}$ *osztáspontokkal* megadott, részleges felbontás megfelel a páronként teljesen diszjunkt felbontás követelményeinek, azaz: $pcdd(m, t, x) = \forall i \in [1..n] : (m(i, 0) = 0 \wedge m(i, n) = x_i.dom) \wedge \forall i, j \in [1, n] : \forall k, l \in [0, n-1] : k \neq l \wedge k < t(i) \wedge l < t(j) \rightarrow x_i[m(i, k), m(i, k+1)] \cap x_j[m(j, l), m(j, l+1)] = \emptyset$.

A részlegesen meghatározott, páronként teljesen diszjunkt felbontás fogalmának bevezetésével fionomíthatjuk a (11.28.)-(11.30.) specifikációt.

11.7. Lemma. *(Páronként diszjunkt felbontás - a feladat finomítása)*
Az alábbi specifikáció finomítása a (11.28.)-(11.30.) specifikációnak:

$$(x = x') \in INIT_{x'} \tag{11.31.}$$

[6] A [Fót Hor Kozs 95] cikkben egy módszert mutattunk arra, hogy milyen módon oldható meg más úton az a feladat, hogy az egyes processzorok között a feladatmegosztást kiegyensúlyozzuk.

[7] A legnagyobb elemszámú komponens megtalálása visszavezethető egy maximumkeresésre, amely asszociatív művelet. A 11.3. tétel szerint ezt a feladatot $O(\log(n))$ lépésben megoldhatjuk. A későbbiekben látjuk majd, hogy $O(\log(n))$ lépés elhanyagolható a megoldáshoz szükséges összes állapotváltozáshoz képest.

$$\uparrow\hookrightarrow \mathrm{FP}_{x'} \tag{11.32.}$$

$$\mathrm{FP}_{x'} \Rightarrow \forall i \in [1.n] : t(i) = n - 1 \wedge (x = x'), \tag{11.33.}$$

$$inv_{x'}(pccd(m, t, x)) \tag{11.34.}$$

Biz.: A korábbiakhoz hasonlóan a 8.8. lemma alapján. □

Válasszuk a $v : A \longmapsto \mathcal{N}_0$ variáns függvényt a következőképpen:
$v ::= n * n - |\{m(i, j)|j \leq t(i)\}|$.

11.5. Megjegyzés. A (11.29.) feltételt nem finomítottuk. A variáns függvényre vonatkozó 8.7. tétel segítségével bizonyíthatjuk majd azt, hogy a program megfelel a (11.29.)=(11.32.) feltételnek. Ebben az értelemben a variáns függvény megválasztása is egy finomítási lépésként fogható fel.

A variáns függvény értéke akkor csökken, ha a $t(i)$ vektor elemeinek értéke nő. Ez azt jelenti, hogy a részlegesen meghatározott felbontást ki kell terjeszteni, az m mátrix további elemei értékének meghatározásával.

Az (11.34.) invariáns, $pccd(m, t, x)$ igaz marad, ha $m(i, t(i)+1)$-t azon x_i-beli elem indexének választjuk, amely elem kisebb vagy egyenlő $x_1(m(1, t(i)+1))$-nél, és amely elemre rákövetkező elem x_i-ben nagyobb, mint $x_1(m(1, t(i)+1))$. Jelöljük a $(j = m \vee (j \in (m, n] \wedge x_i(j) \leq h)) \wedge ((j + 1 \in [m, n] \wedge x_i(j + 1) > h) \vee (j = n))$ logikai függvényt $\Gamma(x_i, j, h)$-vel. A (11.34.) feltételt a

$$inv_{x'}(\forall i, j \in [1, n] : \Gamma(x_i, m(i, j), x_1(m(1, j)))) \tag{11.35.}$$

feltétellel finomítjuk.

Mivel x_i monoton, a (11.35.) feltétellel definiált feladat visszavezethető szekvenciális logaritmikus keresésre [Fót 83].

Logaritmikus keresés

Általánosítsuk a (11.35.) feltétellel definiált feladatot. Legyen H egy rendezett halmaz, $(m, n]$ az egész számok nem üres intervalluma és $f : (m, n] \longrightarrow H$ monoton növő függvény. Adott egy $h \in H$ érték. Keressük meg azt $j \in [m, n]$ egész számot, amelyre a $\gamma(j)$ tulajdonság teljesül, ahol $\gamma(j)::=(j = m \vee (j \in (m, n] \wedge f(j) \leq h)) \wedge ((j + 1 \in [m, n] \wedge f(j + 1) > h) \vee (j = n))$.

$$A = \mathcal{Z} \times \mathcal{Z} \times \mathcal{Z} \times H, \qquad m, n, j : \mathcal{Z}, \quad h : H.$$

$$B = \mathcal{Z} \times \mathcal{Z} \times H, \qquad m', n' : \mathcal{Z}, \quad h' : H.$$

$$Q ::= (m \leq n) \wedge (m = m' \wedge n = n' \wedge h = h' \wedge$$

$$\wedge \forall k, l \in (m, n] : k \leq l \Rightarrow f(k) \leq f(l))$$

$$Q \hookrightarrow \mathrm{FP}_{m',n',h'} \tag{11.36.}$$

$$Q \in \mathrm{INIT}_{m',n',h'}, \tag{11.37.}$$

$$\mathrm{FP}_{m',n',h'} \Rightarrow (h = h' \wedge j \in [m', n'] \wedge \gamma(j)). \tag{11.38.}$$

Mivel az $[m,n]$ intervallum nem üres, ezért biztosan létezik olyan $j \in [m,n]$: $\gamma(j)$. Finomítsuk a specifikációt egy invariáns és egy variáns függvény bevezetésével. Az állapotteret két új komponsnes bevetetésével bővítjük. $u, v : \mathcal{N}_0$.

Válasszuk a $v_1 : A \longmapsto \mathcal{N}_0$ variáns függvényt a következőképpen: $v_1 ::= v - u + 1$.

$$inv_{m',n',h'}(h = h' \wedge [u,v] \subseteq [m',n'] \wedge u \leq v \wedge j \in [u,v]) \qquad (11.39.)$$
$$inv_{m',n',h'}(\forall k \in [m',n'] \setminus [u,v] : \neg\gamma(k)). \qquad (11.40.)$$

11.8. Tétel. *(Logaritmikus keresés tétele)*
A 11.4. absztrakt program megfelel a (11.36.)-(11.40.) specifikációnak.

$$\begin{array}{rl} s_0: & u,v,j := m,n,\lceil (m+n)/2 \rceil \\ \\ S: \{ & u,j := j,\lceil (j+v)/2 \rceil, \text{ha } \neg\gamma(j) \wedge f(j) \leq h \ , \\ & v,j := j,[(u+j)/2], \text{ha } \neg\gamma(j) \wedge f(j) > h \\ \} & \end{array}$$

11.4. absztrakt program: Szekvenciális logaritmikus keresés

Bizonyítás: A bizonyítás a [Fót 83]-ban adott bizonyítás alapján elvégezhető. □

Egy megoldás a párhuzamos páronként diszjunkt felbontás feladatára

Alkalmazzuk a logaritmikus keresés programját (11.4. prg.) n sorozatra (szuperpozíció), egyenként $n-1$-szer (explicit szekvencializálás $mod(n)$ [Lam Sin 79]). Ezzel a módszerrel meghatározhatjuk az m mátrix értékét úgy, hogy az x egy páronként teljesen diszjunkt felbontását definiálja, azaz megkapjuk azt a programot, amely megfelel a (11.31.)-(11.34.) specifikációnak. A megoldás helyességét a szuperpozíció és a szekvencia levezetési szabályára hivatkozva igazolhatjuk. $n-1$ folyamat szekvenciáját egyszerű transzformációval ciklussá alakítjuk.

11.2.4. Diszjunkt halmazok uniója

Az f elemenként feldolgozható függvény értékét a páronként teljesen diszjunkt felbontással kapott szeletekre függetlenül, párhuzamosan meghatározhatjuk. A teljes eredményt a szeletekre kapott eredmény diszjunkt uniójaként állítjuk elő (11.2. def.).

$$
\begin{array}{ll}
s_0: & \mathop{\Box}_{i=[1..n]} m[i,0], m[i,n], t(i) := 0, x_i.dom, 0 \\
 & \\
 & \mathop{\Box}_{i=[1..n]} m[1,i] := i * \lceil (x_1.dom \text{ DIV } n) \rceil \\
 & \\
S: \{ & \mathop{\Box}_{i=[1..n]} u(i), v(i), m(i,t(i)+1) := 0, x_i.dom, \lceil x_i.dom/2 \rceil \\
 & \mathop{\Box}_{i=[1..n]} u(i), m(i,t(i)+1) := m(i,t(i)+1), \lceil (m(i,t(i)+1)+v(i))/2 \rceil, \\
 & \quad if \;\; x_i(m(i,t(i)+1)) \leq x_1(m(1,t(i)+1))) \wedge \\
 & \quad \neg\Gamma(x_i, m(i,t(i)+1), x_1(m(1,t(i)+1)))) \\
 & \\
 & \mathop{\Box}_{i=[1..n]} v(i), m(i,t(i)+1) := m(i,t(i)+1), [(u(i)+m(i,t(i)+1))/2], \\
 & \quad if \;\; x_i(m(i,t(i)+1)) > x_1(m(1,t(i)+1))) \wedge \\
 & \quad \neg\Gamma(x_i, m(i,t(i)+1), x_1(m(1,t(i)+1)))) \\
 & \\
 & \mathop{\Box}_{i=[1..n]} t(i), u(i), v(i), m(i,t(i)+1) := t(i)+1, 0, x_i.dom, \lceil x_i.dom/2 \rceil \\
 & \quad if \;\; \Gamma(x_i, m(i,t(i)+1), x_1(m(1,t(i)+1)))) \wedge t(i) < n-1 \\
\} &
\end{array}
$$

11.5. absztrakt program: Párhuzamos páronként teljesen diszjunkt felbontás

Tegyük fel, hogy f értékét ismerjük x mind az n páronként teljesen diszjunkt szeletére. Jelöljük a függvényértékeket rendre $p(1), \ldots, p(n)$-nel, ahol $p(i) = (p(i)_1, \ldots, p(i)_m) \in Y$. Tudjuk, hogy $\forall i \in [1..n] : \forall k, l \in [1..\underset{\sim}{p(i).dom}] : k \neq l \rightarrow p(i)_k \cap p(i)_l = \emptyset$. Tetszőleges $j \in [1..m]$-re: $y_j = f_j(x'_1, ..., x'_n) = \bigcup_{i \in [1..n]} p(i)_j$. Ahhoz, hogy megkapjuk az $y = f(x')$ értéket, ki kell számítanunk n diszjunkt halmaz unióját minden $j \in [1..m]$-re. Tegyük fel, hogy a halmazok sorozatok formájában adottak, és a sorozatok elemei indexeik alapján elérhetőek. Az halmazok unióját, mint a sorozatok konkatenációját állítjuk elő. $p(i)_j$ elemeit y_j azon részsorozatába kell bemásolnunk, amelyik kezdőindexe $\sum_{k=1}^{i-1} p(k)_j$, azaz meg kell határoznunk a $p(k)_j$ sorozatok hosszából kapott j szerint rendezett sorozat minden kezdőszeletének összegét. Az összeadás asszociatív művelet, így a feladat megoldható a 11.2. absztrakt program felhasználásával.

11.2.5. A párhuzamos elemenkénti feldolgozás tétele

11.9. Tétel. *(A párhuzamos elemenkénti feldolgozás tétele)*
A (11.19)-(11.21) specifikációs feltételek által definiált feladat megoldása a teljesen diszjunkt felbontás programjának (11.5. prg.), az elemenkénti feldolgozás programja

(11.3 prg.) n-szeres szuperpozíciójának és a diszjunkt halmazok uniójára adott megoldás m-szeres szuperpozíciójának szekvenciája.

11.2.6. Hatékonyság és általánosság

A fenti megoldás egyszerűen implementálható szinkron, aszinkron arhitektúrán is, és osztott rendszerben is [Cha Mis 89]. Osztott rendszer esetén csak akkor hatékony ez a megoldás, ha elegendően sok, $\Omega(\lceil\log(n)\rceil)$ (logikai) csatorna áll rendelkezésre processzoronként és a kommunikációs költség alacsony. Tegyük fel, hogy $\Theta(n)$ processzoron implementáljuk az absztrakt programot, $m = \Theta(n)$ és $|x|$ sokkal nagyobb, mint n. A párhuzamos teljesen diszjunkt felbontás eredményét $\Theta(n)$ processzor cseréli ki egymás között, hogy az egyes szeletekre megkezdődhessen az elemenkénti feldolgozás. Az elemenkénti feldolgozás eredményét ismét $\Theta(n)$ processzor cseréli ki egymás között[8]. A kommunikációs lépések száma tehát $\Theta(n)$ processzoronként. A teljesen diszjunkt felbontáshoz szükséges lépések száma $O(n{*}\log(|x|))$, a szelet elemenkénti feldolgozásához szükséges lépésszám: $\Omega(|x|/n)$ (kiegyensúlyozott felbontás esetén), a részeredmények konkatenációjához pedig $O(m * log(n))$ lépés szükséges a részletösszegek kiszámítása miatt. Kevés változó (n) és sok adat (x) és kiegyensúlyozott felbontás esetén a függvényérték meghatározásának jellemző költsége: $|x|/n$.

Elemenként feldolgoható függvény értékének kiszámítására vezethető vissza rendezett sorozatok összefésülése, halmazok uniója, az időszerűsítés [Fót Nyé 90], Conway problémája [Cha Mis 89] és még számos feladat.

[8] Egyes párhuzamos gépek processzorai számára a filerendszer párhuzamosan is elérhető. Ebben az esetben a kommunikációs igény kisebb.

Összefoglalás

Függelék

A. Függelék

Fontosabb tételek és lemmák

B. Függelék

Absztrakt programok

Tárgymutató

Irodalom

[ALRM 83] U.S. Department of Defense: *The Programming Language Ada, Reference Manual.* American National Standards Institute, Inc. ANSI/MIL-STD-1815A-1983, Lecture Notes in Computer Science, Vol. 155 (Springer, Berlin, 1983).

[And 91] Andrews, G.R.: *Concurrent Programming, Principles and Practice.* (Benjamin/Cummings, Redwood City, 1991).

[Bac Ser 90] Back, R.J.R.-Sere, K.: Stepwise Refinement of Parallel Algorithms. *Science of Computer Programming*, Vol. 13 (1989/90) 133-180.

[Bak War 91] de Bakker, J.W.-Warmerdam, J.H.A.: Four domains for concurrency. *Theoretical Computer Science* Vol. 90 (1991) 127-149.

[Ben 88] Benthem, J.: Time, Logic and Computation. In: *Linear Time, Branching Time and Partial Order in Logics an Models for Concurrency*, Lecture Notes in Computer Science, Vol. 354. (Springer, Berlin, 1989) 1-49.

[Best 83] Best, E.: Relational Semantics of Concurrent Programs. In: *Formal Description of Programming Concepts, II.* (1983) 431-452.

[Car 94] Carruth, A.: *Real-Time Unity.* Technical Report TR94-10, University of Texas at Austin, ftp://ftp.cs.utexas.edu. (March 29, 1994).

[Cha Mis 89] Chandy, K.M.-Misra, J.: *Parallel Program Design: A Foundation.* (Addison-Wesley, 1988, 1989).

[Cha 90] Chandy, K.M.: Reasoning about continuous systems. *Science of Computer Programming*, Vol. 14 (1990) 117-132.

[Col 94] Collette, P.: Composition of assumtion-commitment specifications in a UNITY style. *Science of Computer Programming*, Vol. 23 (1994) 107-125.

[Dij 75] Dijkstra, E.W.: Guarded Commands, Nondeterminacy and Formal Derivation of Programs. *Communications of the ACM*, Vol. 18, Num. 8 (1975) 453-457.

[Dij 76] Dijkstra, E.W.: *A Discipline of Programming.* (Prentice-Hall, 1976).

[Dij Sch 89] Dijkstra, E.W.-Scholten, C.S.: *Predicate Calculus and Program Semantics.* (Springer, New York, 1989).

[Eme Sri 88] Emerson, E.A.-Srinivasan, J.: Branching Time Temporal Logic. In: *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, Lecture Notes in Computer Science, Vol. 354 (Springer, Berlin, 1989) 123-172.

[Fáb 94] Fábián G.: *Párhuzamos algoritmusok specifikációja osztott objektumokat használó rendszerek esetén UNITY módszerrel. I. Osztott bináris fák.* Szakdolgozat, ELTE, Informatika Tanszékcsoport. (Témavezető: Horváth Z.) 1994.

[Flo Suz 81] Flon, L., Suzuki, N.: The Total Correctness of Parallel Programs. *SIAM Journal of Computing*, Vol. 10, No. 2 (May 1981) 227-246.

[Fót 83] Fóthi Á.: *Bevezetés a programozáshoz.* Egyetemi jegyzet (ELTE, TTK, Budapest, 1983).

[Fót 86] Fóthi Á.: Bevezetés a programozáshoz és Programozás c. előadásainak anyaga (1986-1987).

[Fót 88] Fóthi Á.: A Mathematical Approach to Programming. *Annales Uni. Sci. Budapest de R. Eötvös Nom. Sectio Computatorica*, Tom. IX. (1988) 105-114.

[Fót Hor 91] Fóthi Á.- Horváth Z.: The Weakest Precondition and the Theorem of the Specification. In: Koskimies,K.-Räihä,K., ed., *Proceedings of the Second Symposium on Programming Languages and Software Tools*, Pirkkala, Finland, August 21-23, 1991, Report A-1991-5, University of Tampere, Department of Computer Science (August, 1991) 39-47.

[Fót Hor 94] Fóthi Á.- Horváth Z.: A Parallel Elementwise Processing. In: Ferenczi Sz.-Kacsuk P., ed., *Proceedings of the 2nd Austrian-Hungarian Workshop on Transputer Applications*, September 29-October 1, 1994, Budapest, Hungary, KFKI-1995-2/M,N Report (1995) 273-282.

[Fót Hor Kozs 95] Fóthi Á.- Horváth Z.- Kozsik T.: Parallel Elementwise Processing – A Novel Version. In: Varga L., ed., *Proceedings of the Fourth Symposium on Programming Languages and Software Tools*, Visegrád, Hungary, June 9-10, 1995 (1995) 180-194. és *Annales Uni. Sci. Budapest de R. Eötvös Nom. Sectio Computatorica* (1996).

[Fót Nyé 90] Fóthi Á.-Nyékyné Gaizler J.: Some Problems of Updating Sequential Files. To appear.

[Fra 86] Franczez, N.: *Fairness.* (Springer, New York, 1986).

[Győr 94] Győrffy L.: *Párhuzamos algoritmusok specifikációja osztott objektumokat használó rendszerek esetén UNITY módszerrel. II. Hatványlisták.* Szakdolgozat, ELTE, Informatika Tanszékcsoport. (Témavezető: Horváth Z.) 1995.

[Hen 88] Hennessy, M.: *Algebraic Theory of Processes.* (The MIT Press, 1988).

[Hoa 78] Hoare, C.A.R.: Communicating Sequential Processes, *Communications of the ACM* Vol. 21, Num. 8 (1978) 666-677.

[Hoa 85] Hoare, C.A.R.: *Communicating Sequential Processes.* (Prentice-Hall Int., Englewood Cliffs, NJ, 1985).

[Hor 88] Horváth Z.: On-line folyamatirányító szakértői rendszerek fejlesztése. In: Fekete I., ed., *Szakértői rendszrek az ipari folyamatirányításban*, kutatási jelentés, ELTE, TTK, Általános Számítástudományi Tanszék (1988).

[Hor 90] Horváth Z.: Fundamental relation operations in the mathematical models of programming. *Annales Uni. Sci. Budapest de R. Eötvös Nom. Sectio Computatorica*, Tom. X. (1990) 277-298. {MR 92e68113 68Q55 68Q60}.

[Hor 93] Horváth Z.: The Weakest Precondition and the the Specification of Parallel Programs. In: *Proceedings of the Third Symposium on Programming Languages and Software Tools*, Käariku, Estonia, August 21-23, 1993 (1993) 24-33.

[Hor 93-96] Horváth Z.: Párhuzamos programozás alapjai. Jegyzet. Előkészületben. (ftp://augusta.inf.elte.hu/pub/parh) 1993-1996.

[Hor 95] Horváth Z.: Parallel asynchronous computation of the values of an associative function. *Acta Cybernetica*, Vol. 12, No. 1, Szeged (1995) 83-94.

[Hor 95a] Horváth Z.: The Formal Specification of a Problem Solved by a Parallel Program – a Relational Model. In: Varga L., ed., *Proceedings of the Fourth Symposium on Programming Languages and Software Tools*, Visegrád, Hungary, June 9-10, 1995 (1995) 165-179. és *Annales Uni. Sci. Budapest de R. Eötvös Nom. Sectio Computatorica* (1996).

[Hor Koz 94] Horváth Z.- Kozma L.: Parallel Programming Methodology. In: Bogdany J.-Vesztergombi G., ed., *Workshop on Parallel Processing. Technology and Applications.* Budapest, Hungary, 10-11 February, 1994, KFKI-94-09/M,N Report (1994) 57-65.

[Jär 92] Järvinen, H-M.: *The Design of a Specification Language for Reactive Systems.* Thesis for the degree of Doctor of Technology, Tampere University of Technology, Publications 95, Tampere, 1992.

[Jut Kna Rao 89] Jutla, C.S., Knapp, E., Rao, J. R.: A Predicate Transformer Approach to Semantics of Parallel Programs. In: *Proc. 8th Ann. ACM SIGACT/SIGOPS Symposyum on Principles of Distributed Computing*, Edmonton, Alberta, Canada, August 14-16, 1989 (1989) 249-263.

[Kna 90] Knapp, E.: A Predicate Transformer for Progress. *Information Processing Letters*, Vol. 33 (1989/90) 323-330.

[Kna 92] Knapp, E.: Derivation of concurrent programs: two examples. *Science of Computer Programing*, Vol. 19 (Oct. 1992) 1-23.

[Koz 94] Kozma L.: Synthesizing Methods of Parallel Systems. An Overview. In: *Proceedings of μP'94*, Technical University Budapest, Hungary (1994) 586-.

[Krö 87] Kröger, F.: *Temporal Logic of Programs.* (Spriger, 1987).

[Lam 77] Lamport, L.: Proving the Correctness of Multiprocess Programs, *IEEE Transactions on Software Engeneering*, Vol. SE-3, No., 2 (March 1977) 125-143.

[Lam 90] Lamport, L.: win and sin: Predicate Transformers for Concurrency. *ACM Transactions on Programming Languages and Systems*, Vol. 12, No. 3 (July 1990) 396-428.

[Lam 91] Lamport, L.: *The Temporal Logic of Actions.* Technical Report SRC Research Number TR79, Digital Equipment Corporation, Systems Research Center, Palo Alto, CA, ftp: gatekeeper.dec.com: pub/DEC/SRC/research-reports (December 1991).

[Lam Lyn 90] Lamport, L.-Lynch, N.: Distributed Computing Models and Methods. In: van Leeuwen, ed., *Handbook of Computer Science*, vol. B (Elsevier, Amsterdam, 1990) 1157-1199.

[Lam Sin 79] van Lamsweerde, A., Sintzoff, M.: Formal Derivation of Strongly Correct Concurrent Programs. *Acta Informatica*, Vol. 12, No. 1 (1979) 1-31.

[Lav 78] Laventhal, M.: *Synthesis of Synchronization Code for Data Abstractions.* Ph.D. Thesis (MIT, 1978).

[Loy Vor 90] Loyens, L.D.J.C.-van de Vorst, J.G.G.: Two Small Parallel Programming Exercises. *Science of Computer Programming*, Vol. 15 (1990) 159-169.

[Luk Sne 92] Lukkien, J., van de Snepscheut J.,L.,A.: Weakest Preconditions for Progress. *Formal Aspects of Computing*, Vol. 4 (1992) 195-236.

[Maz 89] Mazurkiewicz, A.: Basic Notions of Trace Theory. In: *Linear Time, Branching Time and Partial Order in Logics an Models for Concurrency*, Lecture Notes in Computer Science, Vol. 354. (Springer, Berlin, 1989) 285-362.

[Mil 89] Milner, R.: *Communication and Concurrency.* (Prentice Hall, 1989).

[Mis 94] Misra, J.: *A Logic for Concurrent Programming*, University of Texas, Austin, ftp: ftp.cs.utexas.edu., April 12, 1994.

[Mor 87] Morris, J.,M.: A Theoretical Basis for Stepwise Refinement and the Programming Calculus. *Science of Computer Programming*, Vol. 9 (1987) 287-306.

[Mor 90] Morris, J.,M.: Temporal Predicate Transformers and Fair Termination. *Acta Informatica*, Vol. 26, 287-313, 1990.

[Mak Ver 91] Mak, R.H., Verhoeff, T.: Classification of Models, Lecture Notes on Process Models, TU Eindhoven, manuscript, 1991.

[Mel 87] Melliar-Smyth, P.M.: Extending Interval Logic to Real Time Systems, Lecture Notes in Computer Science, Vol. 398 (1987) 224-242.

[Owi Gri 76] Owiczki, S.S., Gries, D.: An Axiomatic Proof Technique for Parallel Programs. *Acta Informatica*, Vol. 6, 319-340, 1976.

[Pac 92] Pachl, J.: A simple proof of a completeness result for leads-to in the UNITY logic. *Information Processing Letters*, Vol. 41 (1992) 35-38.

[Par 79] Park, D.: *On the semantics of fair parallelism* In LNCS 86, pp 504-526. Springer 1980.

[Pász 93] Pásztorné Varga K.: *Logikai alapozás alkalmazásokhoz. Matematikai logika - számítástudomány.* Egyetemi jegyzet, ELTE, TTK (Budapest, 1992).

[Pra 94] Prasetya, I.S.W.B.: Error in the UNITY Substitution Rule for Subscribed Operators. *Formal Aspects of Computing*, Vol. 6 (1994) 466-470.

[Pra 86] Pratt, V.: Modeling Concurrency with Partial Orders. *International Journal of Parallel Programing*, Vol. 15, No. 1 (1986) 33-71.

[Qui 87] Quinn, M.,J.: *Designing Efficient Algorithms for Parallel Computers.* (McGraw-Hill, Inc., 1987).

[Rácz 92] Rácz É.: *A Temporal Logic Specification of a Transaction Manager.* Ph.D. Thesis, ELTE (1992) /in Hungarian/

[Rao 95] Rao, J.,R.: *Extensions of the UNITY Methodology*, Lecture Notes in Computer Science, Vol. 908. (Springer, 1995).

[San 91] Sanders, B.A.: Eliminating the substitution axiom from the UNITY logic. *Formal Aspects of Computing*, Vol. 3 (1991) 189-205.

[Sin 91] Singh, A.,K.: Specification of concurrent objects using auxiliary variables. *Science of Computer Programming*, Vol. 16 (1991) 49-88.

[UN 88-93] Misra, J., et al.: *Notes on UNITY*, 1988-1993., The University of Texas, Austin, ftp://ftp.cs.utexas.edu.

[Var 81] Varga L.: *Programok analízise és szintézise.* (Akadémiai Kiadó, Budapest, 1981).

[WRMP 95] Workgroup on Relational Models of Programming – Fóthi Á. and Fekete I.-Gregorics T.-Horváth Z.- Koncz-Nagy M.-Kozics S.-Nyéky-Gaizler J.-Sike S.-Steingart F.-Tőke P.- Vargyas M.-Venczel T.: Some Concepts of a Relational Model of Programming. In: Varga L., ed., *Proceedings of the Fourth Symposium on Programming Languages and Software Tools*, Visegrád, Hungary, June 8-14, 1995 (1995) 434-446.

A dolgozat szerkesztése és szedése
LaTeX-ben készült.

Terjedelem: 10 (A/4) ív. Példányszám: 5.