Programozáselmélet
Összefoglaló / gyakorlat jegyzet

Készült Borsi Zsolt előadásai
és Gregorics Tibor gyakorlatai alapján

Sárközi Gergő, 2021-22-1. félév
Nincsen lektorálva!

Tartalomjegyzék

1. ZH 1 összefoglaló

1.1. Jelölések, matematikai alapok

- Jelölni "ha $l : \mathbb{L}$ igaz, akkor abc" így: $l \to abc$
- Tömb: $x : \mathbb{Z}^n$ ahol $n \in \mathbb{N}$ (nem kell külön bevezetni); első elem: $x[1]$
- Intervallum: $i : [1..n]$ vagy $\forall j \in [1..n]$
- Reláció: $R \subseteq A \times B$ (üres halmaz is az)
- Függvény: $R \in A \to B$ ($|R(a)| \leq 1$); vagy $R : A \to B$ ($D_R = A$)
- Kompozíció: $Q \circ P = \{(a, c) \in A \times C \mid \exists b \in B : (a, b) \in P \wedge (b, c) \in Q\}$
- Sorozatok: $H^{**} = H^* \cup H^\infty$ (véges/végtelen hossz); pl.: <1,2,2,...>
  - $*$ véges sokat, akár nullát jelöl. Példa: $< 0, 1, (2, 3)^*, (4, 5)^\infty >$
- $\lceil R \rceil = \{a \in A | R(a) = igaz\}$ és $(Q \implies R) \Leftrightarrow (\lceil Q \rceil \subseteq \lceil R \rceil)$
  - $R \in A \to \mathbb{L}$ akkor $\lceil R \rceil$ csak megegyezés miatt adhatja meg D_R-t

1.2. Programok, programfüggvények, stb. relációi

- Ezek mind levezethetők definíciókból.
- $S_1 \subseteq S_2 \implies D_{p(S_2)} \subseteq D_{p(S_1)}$
- $S_1 \subseteq S_2 \implies \forall a \in D_{p(S_2)} : p(S_1)(a) \subseteq p(S_2)(a)$
- $S' \subseteq S$ és S megoldja F-et, akkor S' is
- $F \subseteq F'$ és S megoldja F'-t, attól még S nem oldja meg F-t
- F, S és $p(S)$ determinisztikussága nem nagyon következtethető ki egymásból.

1.3. lf tulajdonságai

- ha $Q \implies R$ akkor $lf(S, Q) \implies lf(S, R)$
- $lf(S, Q) \wedge lf(S, R) = lf(S, Q \wedge R)$ ($\vee$-val is igaz)
- Számolás: R-be behelyettesítjük S-t: (és S nem abortál)
  $lf((x := 3), (x < 10)) = (igaz \wedge (x < 10)^{\leftarrow x:=3}) = (3 < 10) = igaz$

1.4. Alapfogalmak

- Állapot: változókkal címkézett értékek; állapottér: lehetséges állapotok
  - Alap-állapottér: segédváltozók nélküli (azaz A, nem pedig $\overline{A}$)
- Feladat: $F \subseteq A \times A$ ahol A egy állapottér
- Program: $S \subseteq A \times (\overline{A} \cup \{fail\})^{**}$ ahol:
  - $D_S = A$ és a végrehajtási sorozat (vs) első tagja a bemenet
  - Az utolsó állapot lehet csak *fail*; valamint vagy *fail* vagy A-beli
  - Semmit nem változtató értékadás is bekerül a végrehajtási sorozatba.
- Programfüggvény: $p(S) \subseteq A \times A$ ahol S program A felett
  - $D_{p(S)} = \{a \in A | S(a) \subseteq \overline{A}^*\}$ (ahol S véges, nem *fail* vs-t ad)
  - $p(S)(a) = \{b \in A | \exists vs \in S(a) : b = vs_{|vs|}\}$ (utolsó vs tagok)
- Megoldás: S m.o. F-et ha $D_F \subseteq D_{p(S)}$ és $\forall a \in D_F : p(S)(a) \subseteq F(a)$
- Gyenge programfüggvény: $\tilde{p}(S) \subseteq A \times (A \cup \{fail\})$
  - $D_{\tilde{p}(S)} = \{a \in A | S(a) \cap (\overline{A} \cup \{fail\})^* \neq \varnothing\}$ (ahol min 1 véges vs)
  - $\widetilde{p}(S)(a) = \{b \in A \cup \{fail\} | \exists vs \in S(a) \cap (\overline{A} \cup \{fail\})^* : b = vs_{|vs|}\}$
- Parciális helyesség: S parc. helyes F-re ha $\forall a \in D_F : \widetilde{p}(S)(a) \subseteq F(a)$
- Leggyengébb előfeltétel: $lf(S,R) : A \to \mathbb{L}$ (ahonnan S után R igaz)
  - $\lceil lf(S,R) \rceil = \{a \in A | a \in D_{p(S)} \wedge p(S)(a) \subseteq \lceil R \rceil\}$

1.5. Paramétertér, specifikáció

- B az F paramétertere ha $F = F_2 \circ F_1$ (ahol $F_1 \subseteq A \times B$ és $F_2 \subseteq B \times A$)
  - minden feladatnak végtelen sok van, pl.: $B = A$, $F_1 = i\mathrm{d}$, $F_2 = F$
  - paraméter: paramétertér egy eleme (hasonló: állapottér vs állapot)
- Specifikáció: A (állapottér), B (paramtér), Q (előfeltétel), R (utófeltétel)
  - $\forall b \in B : \lceil Q_b \rceil = F_1^{-1}(b)$ (a-k, amikhez F_1 rendel b-t)
  - $\forall b \in B : \lceil R_b \rceil = F_2(b)$ (a-k, amiket F_2 rendel egy b-hez)
  - ha $\forall b \in B : Q_b \implies lf(S, R_b)$ akkor S megoldja F-t
- Szigorúbb feladat: gyengített előfeltétel és/vagy szigorúbb utófeltétel.

2. ZH 2 összefoglaló

2.1. Programkonstrukciók

2.1.1. Szekvencia: $(S_1; S_2)$

- $(S_1; S_2)(a) =$
  $\{vs \in \overline{A}^{\infty} \mid vs \in S_1(a)\}$ (S_1 végtelen vs-t ad)
  $\cup\{vs \in (\overline{A} \cup \{fail\})^* \mid vs \in S_1(a) \wedge vs_{|vs|} = fail\}$ (S_1 vs vége fail)
  $\cup\{vs \in (\overline{A} \cup \{fail\})^{**} \mid vs = x \oplus y \wedge x \in S_1(a) \wedge |x| < \infty \wedge$
  $x_{|x|} \neq fail \wedge y \in S_2(x_{|x|})\}$ (S_1 vs véges és nem fail: S_2-vel folytatjuk)
- $p((S_1; S_2)) = p(S_2) \odot p(S_1)$ (szigorú kompozíció)
  $Q \odot P = \{(a, c) \in A \times C \mid \exists b \in B : (a, b) \in P \wedge (b, c) \in Q \wedge P(a) \subseteq D_Q\}$
- $D_{p((S_1;S_2))} = \{a \in D_{p(S_1)} \mid p(S_1)(a) \subseteq D_{p(S_2)}\}$

2.1.2. Elágazás: $IF = IF(\pi_1 : S_1, ..., \pi_n : S_n)$

- $IF(a) = w_0(a) \cup \bigcup_{i=1..n} w_i(a)$
  - $w_0(a) =< a, fail >$ ha $\forall i : (a \in D_{\pi_i} \wedge \neg\pi_i(a))$ egyébként $\varnothing$
  - $w_i(a) = \begin{cases} S_i(a) \text{ ha } a \in D_{\pi_i} \wedge \pi_i(a) \\ \varnothing \quad \text{ ha } a \in D_{\pi_i} \wedge \neg\pi_i(a) \\ \{< a, fail >\} \text{ ha } a \notin D_{\pi_i} \end{cases}$
- $p(IF)(a) = \bigcup_{i=1..n} \{x \in p(S_i)(a) \mid a \in \lceil \pi_i \rceil \}$
- $D_{p(IF)} = \{a \in A \mid a \in \bigcap D_{\pi_i} \wedge a \in \bigcup \lceil \pi_i \rceil \wedge (a \in \lceil \pi_i \rceil \implies a \in D_{p(S_i)})\}$

2.1.3. Ciklus: $DO = DO(\pi : S)$

- $DO(a) = \begin{cases} (S; DO)(a) \text{ ha } a \in D_\pi \wedge \pi(a) \\ < a > \quad \text{ ha } a \in D_\pi \wedge \neg\pi(a) \\ < a, fail > \text{ ha } a \notin D_\pi \end{cases}$
- $p(DO)(a) =?$ (nem hangzott el EA-n)
- $D_{p(DO)} = \lceil \neg\pi \rceil \cup \{x \in \lceil \pi \rceil \cap D_{p(S)} \mid p(S)(x) \subseteq D_{p(DO)}\}$ (nem volt EA-n)

2.1.4. Atomi program: $[S](a) = S(a)$

- Atomikus: nem megszakítható
  - Értékadások, logikai függvények kiértékelése alapból atomikus
- S nem tartalmazhat sem ciklust, sem várakoztató utasítást (Miért?)
- $p([S]) = p(S)$ és $D_{p([S])} = D_{p(S)}$

2.1.5. Várakoztató utasítás: *await* β *then* S *ta*; stukiban: $\overset{\cap}{\underset{S}{\beta}}$

- S nem tartalmazhat sem ciklust, sem várakoztató utasítást (Miért?)
- $(await\ \beta\ then\ S\ ta)(a) = \begin{cases} [S](a) \text{ ha } a \in D_\beta \wedge \beta(a) \ \ (\beta \text{ és } S \text{ együtt atomikus}) \\ (skip, await\ \beta\ then\ S\ ta)(a) \text{ ha } a \in D_\beta \wedge \neg\beta(a) \\ < a, fail > \text{ ha } a \notin D_\beta \end{cases}$
- $p(AWAIT) = p(S)$ (nem hangzott el EA-n)
- $D_{p(AWAIT)} = \lceil \neg\beta \rceil \cup (\lceil \beta \rceil \cap D_{p(S)})$ (nem hangzott el EA-n)
- Holtpontban van, ha nem egy párhuzamos blokk része és $Q \implies \neg\beta$

2.1.6. Párhuzamos blokk: *parbegin* $S_1||...||S_n$ *parend*; stukiban: $S_1||S_2$

- $(parbegin\ S_1||...||S_n\ parend)(a) = \bigcup_{i=1}^{n}(u_i; parbegin\ ...||S_{i-1}||T_i||S_{i+1}||...\ parend)(a)$
  - ahol $S_i = (u_i, T_i)$ (T_i nincs, ha S_i atomikus)
- $p(PAR) =?$ (nem hangzott el EA-n)
- $D_{p(PAR)} =?$ (nem hangzott el EA-n)
- Holtpontra jutott, ha minden (és legalább 1) be nem fejeződött komponense (S_i) tartalmaz hamis őrfeltételű várakoztató utasítást (blokkolt).
- Interferencia: értékadások "megtámadhatják" a komponensek helyességét.
  - Formálisan: kritikus utasítások támadhatnak, amik vagy értékadások, vagy atomikus programok, bennük értékadással.

2.2. Levezetési szabályok

2.2.1. Specifikáció Tétele

- Legyen F feladat specifikációja A, B, Q, R
- Ha $Q \implies lf(S, R)$ akkor S program megoldja F-et

2.2.2. SZekvencia Levezetési SZabálya

- Legyen $S = (S_1; S_2)$; elő- és utófeltételek: $Q \to (S_1) \to Q' \to (S_2) \to R$
- Ha $Q \implies lf(S_1, Q')$ és $Q' \implies lf(S_2, R)$ akkor $Q \implies lf(S, R)$

2.2.3. Elágazás Levezetési SZabálya

- Legyen $IF = IF(\pi_1 : S_1, ..., \pi_n : S_n)$
- Ha az alsók teljesülnek, akkor $Q \implies lf(S, R)$
  - $Q \implies AND_{i=1}^{n} \pi_i \vee \neg\pi_i$ (mindegyik mindenhol értelmes)
  - $Q \implies OR_{i=1}^{n} \pi_i$ (mindenhol legalább egy teljesül)
  - $\forall i \in [1..n] : Q \wedge \pi_i \implies lf(S_i, R)$ (bármelyik ágon R igaz lesz)

2.2.4. Ciklus Levezetési SZabálya

- Legyen $DO = DO(\pi : S)$, P invariáns és t terminálló függvény
- Ha az alsók teljesülnek, akkor $Q \implies lf(DO, R)$
  - $Q \implies P$ (invariáns alapból teljesül)
  - $P \wedge \neg\pi \implies R$ (jó helyen áll meg)
  - $P \implies \pi \vee \neg\pi$ (ciklusfeltétel nem abortál)
  - $P \wedge \pi \implies t > 0$ (cikluson belül van még lépés)
  - $P \wedge \pi \wedge t = t_0 \implies lf(S, P \wedge t < t_0)$ (invariáns marad, lépés - -)

2.2.5. Atomi program Levezetési SZabálya

- Ha $Q \implies lf(S, R)$ akkor $Q \implies lf([S], R)$

2.2.6. Várakoztató utasítás Levezetési SZabálya

- Ha az alsók teljesülnek, akkor $Q \implies lf(await\ \beta\ then\ S\ ta, R)$
  - $Q \implies \beta \vee \neg\beta$ (őrfeltétel nem abortál)
  - $Q \wedge \beta \implies lf(S, R)$
- Holtpontban van, ha nem egy párhuzamos blokk része és $Q \implies \neg\beta$

2.2.7. Párhuzamos blokk Levezetési SZabálya

- Ha az alsók teljesülnek, akkor $Q \implies lf(parbegin\ S_1||...||S_n\ parend, R)$
  - $Q \implies Q_1 \wedge ... \wedge Q_n$ (belépési feltétel)
  - $R_1 \wedge ... \wedge R_n \implies R$ (jó helyen állunk meg)
  - $\forall i \in [1..n] : Q_i \implies lf(S_i, R_i)$
  - $Q_i \implies lf(S_i, R_i)$ teljes helyességi interferencia formulák interferenciamentesek
  - párhuzamos blokk holtpontmentes
- Interferenciamentesség: u nem interferál $Q \implies lf(S, R)$-rel, ha:
  - $pre_u \wedge Q \implies lf(u, Q)$
  - $pre_u \wedge R \implies lf(u, R)$
  - Ha S-ben van ciklus: $pre_u \wedge t = t_0 \implies lf(u, t \leq t_0)$
  - Ezt be kell látnia az összes u értékadásra és az összes, u-tól különböző komponensben található $Q \implies lf(S, R)$ helyességi formulára.
  - Egyszerűsítési lehetőség: u nem változtat a $Q \implies lf(S, R)$ által használt változókon. ("$1 \rightsquigarrow 2$: nincs krit. ut.")
  - Hamis bármit implikál: $\implies$ előtt ellentmondás az egyből jó
- Holtpontmentesség: nem lehet holtpontban (mert pl. nincs várakoztatás)
  - Be kell látni, hogy hamis: minden komponens vagy kész van, vagy blokkolt és legalább egy komponens blokkolt.
  - Példa: $parbegin\ (await\ \beta\ then\ S_1\ ta)\ ||\ S_2\ parend$
    - Be kell látni, hogy hamis: $pre(AWAIT) \wedge \neg\beta \wedge post(S_2)$
  - Példa: $parbegin\ await\ \beta_1\ then\ S_1\ ta\ ||\ await\ \beta_2\ then\ S_2\ ta\ parend$
    - Be kell látni, hogy ezek mindegyike hamis:
    - $pre(A_1) \wedge \neg\beta_1 \wedge post(A_2)$
    - $pre(A_2) \wedge \neg\beta_2 \wedge post(A_1)$
    - $pre(A_1) \wedge \neg\beta_1 \wedge pre(A_2) \wedge \neg\beta_2$

2.3. Helyességbizonyítás

- Menete: specifikáció tételének felhasználása, majd mindent, ami nem értékadás, az ő levezetési szabályával levezetünk.
- $Q \implies$ ... esetén vegyem úgy, hogy Q igaz és lássam be ...-ot
- Helyezzek ki pipákat, ha belátok valamit, pl. hogy $1 + 1 < 3$
- Értékkiválasztás: $lf(x :\in h, x\ even) = (h \neq \varnothing \wedge \forall e \in h : e\ even)$
- $(\forall k \in [1..n] : x[k] = ...)$-on $(x[i] = ...)$ végrehajtása: $\forall k$ 3 részre bontása: $(\forall k \in [1..i-1] : ...)$ és $(x[i] = ...)$ és $(\forall k \in [i+1..n] : ...)$
- $\vee$ (vagy)-ból következés: esetszétválasztás, akár negált is felhasználható, pl.: $(a \vee b \implies ...)$-nél: $(a \implies)$ vagy $(\neg a \wedge b \implies)$
- Értékadások és számolások, amik abortálhatnak:
  - $a := b$ ahol $a : A$ és $b : B$ abortál, ha $B \not\subseteq A$
  - $a \mod b$ abortál, ha $b = 0$
  - $x[i]$ ahol $x : \mathbb{Z}^n$ abortál, ha $i \notin [1..n]$