PTI. BSc. Záróvizsga

12. Algoritmusok tervezése és elemzése

Algoritmus Jól meghatározott lépések egymásutánja, amelyek már elég pontosan, egyértelműen megfogalmazottak ahhoz, hogy gépiesen végrehajthatók legyenek. Ennél pontosabb meghatározására általában nem szoktak vállalkozni, mert több különböző értelemben használják.

Számítási modellek Számos számítási modellt definiáltak törekedve azok minél nagyobb hatékonyságára, azaz arra, hogy a függvények minél nagyobb körének kiszámítására legyenek képesek. Definiálták a Turing-gép, Markov-algoritmus, Post-féle kanonikus rendszer stb. fogalmát, amelyekről kiderült, hogy azonos hatékonyságúak.

Turing-gép fogalma Legyen k egy pozitív egész. Egy k-szalagos Turing-gép egy hetessel jellemezhető:

$$M = (Q, T, ü, I, q_0, F, \delta),$$

ahol

Q : egy véges halmaz, az M gép belső állapotainak a halmaza.
T : egy véges halmaz, a szalagjelek halmaza.
$ü$: $ü \in T$ az üresjel
I : $I \subseteq T \setminus \{ü\}$ a bemenő jelek halmaza
q_0 : $q_0 \in Q$ a kezdő állapot
F : $F \subseteq Q$ az elfogadó állapotok halmaza
δ : $(Q \setminus F) \times T^k \to Q \times (T \times \{B, H, J\})^k$ a mozgásfüggvény. Ez egy parciális (nem mindenütt értelmezett) függvény.

A Turing-gép működése: kezdetben a gép a q_0 állapotban van, az első szalag tartalmazza – cellánként egy-egy betű – az input szót, tőle balra és jobbra a cellákban csupa üresjelek vannak, a többi szalag teljesen üres. Az első szalag író-olvasó feje az input szó első betűjén – ha a szó üres volt, akkor az egyik üres jelen –, a többi szalagé tetszőleges helyen áll. A Turing-gép a mozgásfüggvénye alapján teszi meg a lépéseit: az aktuális állapot és a leolvasott k darab jel alapján a δ által meghatározott új állapotba kerül, a megadott k szalagjelet rendre kiírja a szalagokra, és a megadott k iránynak megfelelően az író-olvasó fejek egyet lépnek vagy helyben maradnak. Ha a mozgásfüggvény az aktuális argumentumokra nem definiált, akkor a Turing-gép megáll.

Def. Az M Turing-gép által elfogadott L_M nyelv azokból az $s \in I^*$ szavakból áll, amelyekkel mint bemenetekkel elindítva az M-t az megáll, mégpedig F-beli állapotban.

Def. Legyen M egy kitüntetett output szalaggal rendelkező Turing-gép. Az M által kiszámított $f_M : I^* \to I^*$ parciális függvényt így értelmezzük: $f_M(s) = w$, ha M az $s \in I^*$ inputtal megáll, és megállás után az output szalag tartalmának – csupa üresjelekkel közrefogott legszűkebb szó – I^*-beli leghosszabb kezdőszelete w.

Szimuláció Jelölje rendre $T_M(n)$ és $S_M(n)$ az M Turing-gép maximális számolási idejét (lépésszámát), illetve maximális tárigényét (érintett cellák számát) az n hosszú bemenetekre.

Azt mondjuk, hogy az M Turing-gépet az M' szimulálja, ha ugyanazt a nyelvet ismeri fel, vagy ugyanazt a függvényt számítja ki. Tetszőleges k szalagos M Turing-gép szimulálható olyan egyszalagos M' Turing-géppel, amelyre $T_{M'}(n) \leq 2T_M^2(n)$ és $S_{M'}(n) \leq S_M(n) + n$.

Kiszámíthatóság

Def. Az $L \subseteq I^*$ nyelvet rekurzíve felsorolhatónak nevezzük, ha van olyan M Turing-gép, amelyre $L = L_M$. Az L nyelvet rekurzívnak nevezzük, ha van olyan M Turing-gép, amelyre $L = L_M$, és M minden inputra megáll.

Van olyan nyelv, amely nem rekurzíve felsorolható. Van olyan rekurzíve felsorolható nyelv, amely nem rekurzív.

Def. Az $f : I^* \to I^*$ parciális függvény parciálisan rekurzív, ha van olyan M Turing-gép, amelyre $f = f_M$. Az f függvény rekurzív, ha az előbbieken túl még totális is.

Church–Turing-tézis Ami algoritmussal, azaz véges eljárással kiszámítható/eldönthető, az Turing-értelemben is kiszámítható/eldönthető. Nevezetesen: Egy parciális függvény kiszámítható pontosan akkor, ha az parciálisan rekurzív; egy totális függvény kiszámítható pontosan akkor, ha az rekurzív; egy nyelvhez tartozás problémája algoritmussal eldönthető pontosan akkor, ha a nyelv rekurzív.

Bonyolultság fogalmak Legyenek f és g a pozitív egészeken értelmezett valós értékű függvények. Ekkor $f = O(g)$ (nagy ordó g) jelöli azt a tényt, hogy vannak olyan c és n_0 állandók, hogy $|f(n)| \leq c|g(n)|$, ha $n > n_0$.

Def. Az M Turing-gép $t(n)$ időkorlátos, ha $T_M(n) \leq t(n)$, illetve $s(n)$ tárkorlátos, ha $S_M(n) \leq s(n)$.

Bonyolultsági osztályok Def. Jelölje $TIME(t(n))$ az $O(t(n))$ időkorlátos Turing-gépekkel felismerhető nyelvek osztályát, illetve $SPACE(s(n))$ az $O(s(n))$ tárkorlátos Turing-gépekkel felismerhető nyelvek osztályát. Továbbá $FTIME(t(n))$ az $O(t(n))$ időkorlátos Turing-gépek által kiszámítható parciális függvények osztályát, illetve $FSPACE(s(n))$ az $O(s(n))$ tárkorlátos Turing-gépek által kiszámítható parciális függvények osztályát.

Ezek alapján például $TIME(n)$ a lineáris , $\cup_{k\geq 1} TIME(n^k)$ (ezt P-vel szokás jelölni) a polinomiális, $\cup_{k\geq 1} TIME(2^{n^k})$ az exponenciális időben felismerhető nyelvek osztálya, vagy $SPACE(\log n)$ a logaritmikus tárban felismerhető nyelvek osztálya, illetve $\cup_{k\geq 1} FTIME(n^k)$ a polinomiális időben kiszámítható parciális függvények osztálya stb.

Tár-idő tétel. Ha $L \in SPACE(s(n))$, akkor van olyan L-től függő c konstans, mellyel $L \in TIME(c^{s(n)})$ teljesül. Ha $f \in FSPACE(s(n))$, akkor van olyan f-től függő c konstans, mellyel $f \in FTIME(c^{s(n)})$ teljesül.

A gyakorlatban olyan nyelvek (eldöntési feladatok) fordulnak elő, amelyek exponenciális időben felismerhetőek.

NP-teljesség A nemdeterminisztikus Turing-gép abban tér el az egyszalagos Turing-géptől, hogy a δ átmenetfüggvény nem determinisztikus, a képtér $Q \times T \times \{B, H, J\}$ helyett ennek hatványhalmaza (összes részhalmazok halmaza), azaz $2^{Q\times T\times\{B,H,J\}}$. Amúgy a δ totális is – nincs szükség megengedni a definiálatlanságot. Egy nemdeterminisztikus

Turing-gép elfogad egy szót, ha ilyen szalagtartalommal indítva el tud jutni – van olyan számítási út – valamelyik elfogadási állapotba.

Def. Egy M nemdeterminisztikus Turing-gép $t(n)$ időkorlátos, illetve $s(n)$ tárkorlátos, ha az n hosszú inputok esetén minden számítási út mentén legfeljebb $t(n)$ lépést megtéve, illetve legfeljebb $s(n)$ cellát érintve megáll.

Def. Jelölje $NTIME(t(n))$ az $O(t(n))$ időkorlátos nemdeterminisztikus Turing-gépekkel felismerhető nyelvek osztályát, illetve NP a polinomiális időben nemdeterminisztikus Turing-gépekkel felismerhető nyelvek osztályát, azaz $NP = \cup_{k\geq 1} NTIME(n^k)$.

Napjaink fontos kérdése: vajon $P = NP$? $P \subseteq NP$, így a kérdés átfogalmazható: vajon $P \supseteq NP$?

Def. Azt mondjuk, hogy az $f : I^* \to I^*$ leképezés az $L_1 \subseteq I^*$ nyelv Karp-redukciója az $L_2 \subseteq I^*$ nyelvre, ha az f polinomiális időben számítható, és tetszőleges $x \in I^*$ szó esetén $x \in L_1$ pontosan akkor teljesül, ha $f(x) \in L_2$ is teljesül.

Def. Az $L \subseteq I^*$ nyelv NP-teljes, ha $L \in NP$ és tetszőleges $L' \in NP$ esetén létezik L'-nek L-re való Karp-redukciója.

Vannak NP-teljes nyelvek. Ha valamelyikük polinomiális időben determinisztikus Turing-géppel felismerhető lenne, akkor minden NP-beli nyelv is ilyen lenne.

Szekvenciális programformák Szekvenciális programok alatt a klasszikus - Boehm, Jacopini, Mills, Wirth féle - strukturális programokat értünk, azaz olyan programokat, amelyek az értékadó utasításokból mint atomi utasításokból kiindulva a három vezérlési szerkezet - szekvencia, szelekció, iteráció - segítségével építhetők fel.

Absztrakt szintaxisuk egy elsőrendű matematikai logikai nyelv felett a következő:

$$C ::= \textsf{skip} \mid X := E \mid C_1 \,;\, C_2 \mid \textsf{if } B \textsf{ then } C_1 \textsf{ else } C_2 \textsf{ fi} \mid \textsf{while } B \textsf{ do } C \textsf{ od},$$

ahol a C-k utasításokat, a X változót, E termet, B kvantormentes formulát jelöl. Az utasítások neve rendre *üres utasítás*, *(determinisztikus) értékadás*, *kompozíció*, *feltételes utasítás*, *ciklus utasítás*. Szemantikát pl. az operációs, a denotációs vagy az axiomatikus szemantika szerint rendelhetünk hozzájuk.

Helyességfogalmak Az axiomatikus szemantika az utasításokhoz (programokhoz) kétféle helyesség fogalmat rendel: parciális és totális helyességet. Mindkét fogalom az ún. feladat – rendezett formula páros, az input és output feltételek párosa – és a program viszonyát fejezi ki.

Def. Azt mondjuk, hogy a C utasítás *parciálisan helyes* a (P, Q) feladatra nézve, ha valahányszor a változók kezdeti tartalma kielégíti P-t, és a C utasítás végrehajtása eredményesen befejeződik, a változók végrehajtás utáni tartalma kielégíti Q-t. Ezt a tényt a $\{P\}\, C\, \{Q\}$ ún. Hoare-formula fejezi ki.

Def. Azt mondjuk, hogy a C utasítás *totálisan helyes* a (P, Q) feladatra nézve, ha valahányszor a változók kezdeti tartalma kielégíti P-t, akkor a C utasítás végrehajtása eredményesen befejeződik, és a változók végrehajtás utáni tartalma kielégíti Q-t. Ezt a tényt a $[P]\, C\, [Q]$ ún. Hoare-formula fejezi ki.

A szekvenciális programok helyességének bizonyítási módszerei

Mindkét típusú helyesség bizonyítására egy-egy levezetési rendszer (kalkulus) szolgál. A két levezetési rendszer nagyban hasonlít egymásra, csupán a ciklus utasításra vonatkozó szabályban térnek el egymástól. Az atomi utasításokhoz egy-egy axióma séma, az összetett utasításokhoz egy-egy levezetési szabály tartozik. Ez egészül ki az úgynevezett következmény szabállyal, amely a számlálójában nemcsak Hoare-formulákat, hanem közönséges formulákat, bizonyos implikációkat is tartalmaz. A bizonyítás ezeken az implikációkon keresztül kapcsolódik az adatok elméletéhez. A parciális helyesség levezetési rendszere a következő:

Axióma sémák

a skip axióma sémája	$\{P\}\,\textbf{skip}\,\{P\}$
az értékadás axióma sémája	$\{Q^X_E\}\,X := E\,\{Q\}$

Levezetési szabályok

kompozíciós szabály	$\dfrac{\{P\}\,C_1\,\{R\},\ \{R\}\,C_2\,\{Q\}}{\{P\}\,C_1; C_2\,\{Q\}}$
feltételes szabály	$\dfrac{\{P \wedge B\}\,C_1\,\{Q\},\ \{P \wedge \neg B\}\,C_2\,\{Q\}}{\{P\}\,\textbf{if}\ B\ \textbf{then}\ C_1\ \textbf{else}\ C_2\ \textbf{fi}\,\{Q\}}$
ciklus (while) szabály	$\dfrac{\{P \wedge B\}\,C\,\{P\}}{\{P\}\,\textbf{while}\ B\ \textbf{do}\ C\ \textbf{od}\,\{P \wedge \neg B\}}$
következmény szabály	$\dfrac{P \supset P',\ \{P'\}\,C\,\{Q'\},\ Q' \supset Q}{\{P\}\,C\,\{Q\}}$

A parciális helyesség Hoare-féle kalkulusa helyes: Ha a $\{P\}\,C\,\{Q\}$ Hoare-formula levezethető, és a levezetésben szereplő implikációk univerzális lezártjai – az adott interpretációban – igazak, akkor a formulához rendelt logikai érték igaz.

A parciális helyesség Hoare-féle elmélete szemantikailag teljes: Ha a $\{P\}\,C\,\{Q\}$ Hoare-formulához rendelt logikai érték igaz, akkor léteznek a formula levezetéséhez szükséges feltételek mint állapot halmazok, viszont nem biztos, hogy ezek formalizálhatóak.

A totális ciklus szabály:

$$\frac{[P \wedge B \wedge U = N]\ C\ [P \wedge U < N],\quad P \wedge B \supset U > 0}{[P]\,\textbf{while}\ B\ \textbf{do}\ C\ \textbf{od}\,[P \wedge \neg B]}$$

ahol U egész értékű függvény, N egész típusú változó, amely nem paramétere P-nek, B-nek, U-nak és C-nek. P neve ciklusinvariáns, U neve ciklusszámláló.

A totális helyesség Hoare-féle kalkulusa is helyes, illetve szemantikailag teljes.

A helyesség kifejezhető ún. leggyengébb előfeltételekkel ($wp(C,Q)$), leggyengébb megengedő előfeltételekkel ($wlp(C,Q)$), illetve legerősebb megengedő utófeltétellel ($slp(P,C)$). A kapcsolatot a következő ekvivalenciák fogalmazzák meg:

$$[P]\,C\,[Q] \Leftrightarrow P \supset wp(C,Q)$$

$$\{P\}\,C\,\{Q\} \Leftrightarrow P \supset wlp(C,Q) \qquad \{P\}\,C\,\{Q\} \Leftrightarrow slp(P,C) \supset Q$$

Nemdeterminisztikus programok

A helyes eredmény nem minden esetben egyértelmű. Sok olyan feladat fogalmazható meg, amelyben több eltérő eredmény helyesnek fogadható el: Pl. vektor maximális komponensének az indexe vagy adott pontossággal való közelítés stb. A klasszikus programok által előállított eredmény minden esetben egyértelmű volt, a több elfogadható eredményű feladatokhoz készült programok esetében is – ezekben az esetekben a programozó döntötte el, hogy mi lesz az egyértelmű eredmény. Az ilyen feladatoknak jobban megfelelnek azok a programmodellek, amelyekben a végrehajtás lépései, és így a végrehajtás eredménye sem feltétlenül egyértelmű. A lehetséges lépésalternatívák közül nem a programozó választ, hanem egy ütemező. Nem egyértelmű működés többféle módon jelenhet meg: az értékadásban szereplő kifejezés értéke nemdeterminisztikus, a vezérlés, ütemezés nemdeterminisztikus. Ennek megfelelően a nemdeterminisztikus utasítások a következők: nemdeterminisztikus értékadás, nemdeterminisztikus szelekció és iteráció.

A párhuzamos utasítások – ezekben az ütemezés a nemdeterminisztikus – is ide sorolhatóak.

Szintaxis:

$$C ::= X := ? \mid \textsf{if}\ B_1 \to C_1 [] B_2 \to C_2\ \textsf{fi} \mid \textsf{do}\ B_1 * C_1 [] B_2 * C_2\ \textsf{od}$$

Az utasítások neve rendre *nem determinisztikus értékadás*, *alternatív konstrukció*, *repetitív konstrukció*. Az utasításokban szereplő feltételeket *őrfeltételeknek* nevezzük. Az utóbbi két utasítás definíciója és elnevezése Dijkstra-tól származik. (Ő 2 helyett tetszőleges n komponenst definiál.) A nemdeterminisztikus értékadás végrehajtása során a változó egy tetszőleges értéket fog felvenni. Az alternatív konstrukció végrehajtása a feltételek kiértékelésével kezdődik, majd az igazak egyikének megfelelő utasítás végrehajtódik. Ha nincs igaz értékű feltétel, akkor a teljes utasítás 'abortál'. A repetitív konstrukció végrehajtása során alternatív konstrukciós lépéseket hajtunk végre mindaddig, amíg minden őrfeltétel hamissá nem válik.

Párhuzamos programok

Owiczki és Gries definiálták azokat az utasítás konstruktorokat, amelyekkel kiegészítve a szekvenciális utasítások konstruktorait párhuzamos utasításokhoz juthatunk.

Szintaxis:

$$C ::= \textsf{parbegin}\ C_1 \parallel C_2\ \textsf{parend} \mid \textsf{await}\ B\ \textsf{then}\ C\ \textsf{fi}$$

Ezek a párhuzamosító és szinkronizációs utasítások.

A párhuzamosító utasításban a két utasítás végrehajtása szimultán, azaz időben átlapolt, de eredményét tekintve mindig megegyezik egy olyan szimulált párhuzamos végrehajtás

eredményével, amelynek során nemdeterminisztikus módon váltogatva hol az egyik, hol a másik ágban teszünk meg egy-egy atomi (oszthatatlan) lépést (üres utasítást, értékadó utasítást, szinkronizációs utasítást hajtunk végre, vagy feltétel-kiértékelés és annak megfelelő vezérlésátadás történik). A szinkronizációs utasítás utasítás komponense csak értékadást és szekvenciát tartalmazhat. Végrehajtása oszthatatlan.

Helyességfogalmak Def. Azt mondjuk, hogy a C (nemdeterminisztikus vagy párhuzamos) utasítás *parciálisan helyes* a (P, Q) feladatra nézve, ha valahányszor a változók kezdeti tartalma kielégíti P-t, és a C utasítás végrehajtása eredményesen befejeződik, a változók végrehajtás utáni tartalma kielégíti Q-t.

Def. Azt mondjuk, hogy a (nemdeterminisztikus vagy párhuzamos) C utasítás *totálisan helyes* a (P, Q) feladatra nézve, ha valahányszor a változók kezdeti tartalma kielégíti P-t, akkor a C utasítás végrehajtása minden esetben eredményesen befejeződik, és a változók végrehajtás utáni tartalma minden esetben kielégíti Q-t.

A programhelyesség bizonyítási módszerei A nemdeterminisztikus értékadás axióma sémája:

$$\{P\}\, X :=? \,\{\exists X P\}$$

(Utófeltétel: Volt olyan X, amelyre P teljesült.)

A párhuzamos utasítások parciális helyességének bizonyításához ki kell egészíteni három szabállyal a Hoare-kalkulust.

parbegin szabály
$$\frac{\{P_1\}\, C_1\, \{Q_1\},\ \{P_2\}\, C_2\, \{Q_2\},\ \text{és ezek interferenciamentesek}}{\{P_1 \wedge P_2\}\ \textsf{parbegin}\ C_1 \parallel C_2\ \textsf{parend}\ \{Q_1 \wedge Q_2\}}$$

await-szabály
$$\frac{\{P \wedge B\}\, C\, \{Q\}}{\{P\}\ \textsf{await}\ B\ \textsf{then}\ C\ \textsf{fi}\ \{Q\}}$$

segédváltozós szabály
$$\frac{\{P\}\, C\, \{Q\}}{\{P\}\, C'\, \{Q\}}$$

Az interferenciamentesség fogalma biztosítaná a Hoare-formulák bizonyításának a sértetlenségét: a két ág bizonyításában szereplő előfeltételeinek és az ágak utófeltételeinek invariánsaknak kell lenni a másik ág változótartalmakat módosító oszthatatlan utasításaival szemben. Egy $\{P\}\, C\, \{Q\}$ Hoare-formulában változók egy halmaza segédváltozók halmazának tekinthető, ha ezek a változók nem paraméterei P-nek, Q-nak, és csak értékadásokban fordulnak elő, méghozzá olyanokban, ahol a bal oldal is a halmazból való. A segédváltozós szabályban C' abban különbözik C-től, hogy már nem szerepelnek benne a segédváltozós értékadások.

A kiegészített kalkulus helyes, de csak szemantikailag teljes.