2. A programozás alapfogalmai

Ahhoz, hogy a programozásról beszélhessünk, definiálnunk kell, hogy mit értünk a programozás egyes fogalmain. Ha belegondolunk, nem is olyan könnyű megfogalmazni, mi is az a program, vagy hogy mikor old meg egy program egy feladatot.

Amikor programot írunk, általában egy a külvilágban adott feladatot akarunk számítógéppel megoldani. Ehhez kiválasztjuk a külvilág azon elemeit, amelyek kapcsolódnak a feladathoz, és megpróbáljuk elkészíteni ezeknek az elemeknek egy számítógépes modelljét. Ám a számítógépek változnak!

Ezért a továbbiakban bevezetünk egy absztrakt modellt, amelyben leírjuk azokat fogalmakat, amelyek a programozás során előkerülnek, és e modell segítségével a gyakorlatban használható eszközöket adunk.

2.1. Az állapottér fogalma

Az elsőként bevezetendő absztrakt fogalom tulajdonképpen a számítógép memóriájának ad egy a továbbiakban kényelmesen használható megfelelőt.

1. DEFINÍCIÓ: ÁLLAPOTTÉR

Legyenek $A_1, A_2, \ldots, A_n$ tetszőleges véges vagy megszámlálható nem üres halmazok. Ekkor az $A = A_1 \times A_2 \times \cdots \times A_n$ halmazt *állapottérnek*, az A_i halmazokat pedig *típusértékhalmazoknak* nevezzük.

Amikor egy modellt készítünk, el kell döntenünk, hogy a valóság mely részét kívánjuk modellezni, és melyek azok a jellemzők – és milyen értékeket vehetnek fel – amiket a modellünkben figyelembe akarunk venni.

Az állapottér fenti definíciójában az egyes komponenseket tekintsük úgy, mint egyes jellemzők lehetséges értékeinek halmazát. A típusértékhalmaz elnevezés arra utal, hogy ezek a halmazok bizonyos közös tulajdonsággal rendelkező elemekből állnak. A későbbiekben majd kitérünk arra is, hogy ez a közös tulajdonság mit is jelent. Mivel a jellemzők értékhalmaza lehet azonos, az állapottér komponensei között egy halmaz többször is szerepelhet.

2.2. A feladat

Az állapottér fogalmának segítségével könnyen megfogalmazhatjuk, hogy mit értünk programozási feladaton. Azt kell megfogalmaznunk, hogy a memória egy adott állapotából (azaz az állapottér egy pontjából) milyen memóriaállapotba (azaz az állapottér mely pontjába) akarunk eljutni.

2. DEFINÍCIÓ: FELADAT

Feladatnak nevezzük az $F \subseteq A \times A$ relációt.

A feladat fenti definíciója, ha belegondolunk, természetes módon adódik, hiszen a feladatot legegyszerűbben úgy formalizálhatjuk, ha a feladatot egy leképezésnek tekintjük az állapottéren, és az állapottér minden pontjára megmondjuk, hogy hova kell belőle eljutni.

Az, hogy egy feladatnak mi lesz az állapottere, természetesen magától a feladattól függ, ám még a feladat ismeretében sem egyértelmű. Például egy pont síkbeli koordinátáit megadhatjuk a derészögű koordináta-rendszerben, de megadhatjuk polárkoordinátákkal is.

Mégis, az, hogy mit választunk állapottérnek, nagyon fontos dolog, hiszen meghatározza, hogy a továbbiakban mit, és hogyan tudunk leírni. Ha túl kevés jellemzőt vizsgálunk – azaz az állapottér túl kevés komponensből áll – akkor lehetnek olyan fogalmak, amiket nem tudunk benne leírni, ha túl sok a komponens, akkor túl bonyolult lesz a modell.

2.3. A program

Ha a gyakorlatban azt mondjuk, egy program fut, akkor amögött azt értjük, hogy a számítógép memóriájának tartalma folyamatosan változik.

A programfutás tehát egy időben dinamikus folyamat. Vezessünk be egy könnyebben kezelhető statikus modellt! Hogyan kaphatunk egy ilyen statikus modellt?

Tekintsük például az alábbi – a programozástól igazán messze eső – problémát: Adott egy kémiai kísérlet, amely túl gyorsan játszódik le ahhoz, hogy az ember pontosan regisztrálni tudja az egymásutáni eseményeket. Ez a programfutáshoz hasonlóan egy időben dinamikusan lejátszódó folyamat. Hogyan követhető nyomon mégis a kísérlet? Például úgy, hogy a kísérletet filmre vesszük, és a továbbiakban a képkockák által rögzített statikus állapotokat vizsgáljuk. Így az időben változó folyamatot egy statikus állapotsorozattal írjuk le.

A fenti példa szemléletesen mutatja, hogyan adhatunk statikus modellt egy dinamikus folyamat leírására.

A program definíciójában a program időbeni futásának jellemzésére az előbbi példával analóg módon vezetünk be egy statikus modellt: a futást állapottérbeli sorozatokkal írjuk le. Ahogy a program futása során a memóriatartalom változik, úgy jutunk az állapottér újabb és újabb pontjaiba, így ezeket a pontokat egy sorozatba fűzve valójában "filmre vesszük" a programfutást.

3. DEFINÍCIÓ: PROGRAM

Programnak nevezzük az $S \subseteq A \times A^{**}$ relációt, ha

1. $\mathcal{D}_S = A$,
2. $\forall a \in A : \forall \alpha \in S(a) : \alpha_1 = a$,
3. $\forall \alpha \in \mathcal{R}_S : \alpha = red(\alpha)$.

Az absztrakt program értékkészlete olyan sorozatokat tartalmaz, amelyek a konkrét program egy végrehajtását jellemzik.

A fenti megszorítások értelemszerűek: az első azt kívánja meg, hogy a program futását jellemző sorozat abból a pontból induljon el, amihez hozzárendeltük.

A programnak tetszőleges állapotból el kell indulnia, hiszen egy program és annak sincs értelme, hogy a program egy állapotban egymás után véges sokszor lehessen (a program áll?).

2.4. A programfüggvény

Ahhoz, hogy egy program és egy feladat viszonyát megvizsgáljuk, elegendő, ha a programról tudjuk, hogy az állapottér egy adott pontjából kiindulva, az állapottér mely pontjába jut, mert a megoldás szempontjából a közbülső állapotok lényegtelenek. Természetesen vannak olyan – a programok minőségére vonatkozó – további kritériumok, amelyek szempontjából egyáltalán nem mindegy, hogy a program hogyan oldja meg a feladatot (ilyen lehet például a hatékonyság, program idő- és tárigénye), de mi a továbbiakban ezekkel egyelőre nem foglalkozunk.

Ezért vezetjük be a programfüggvény fogalmát, amely tehát csak a program futásának eredményét jellemzi.

4. DEFINÍCIÓ: PROGRAMFÜGGVÉNY

A $p(S) \subseteq A \times A$ reláció az $S \subseteq A \times A^{**}$ program *programfüggvénye*, ha

1. $\mathcal{D}_{p(S)} = \{a \in A \mid S(a) \subseteq A^*\}$,
2. $p(S)(a) = \{b \in A \mid \exists \alpha \in S(a) : \tau(\alpha) = b\}$.

Az első követelmény azt fogalmazza meg, hogy csak azokban a pontokban van értelme azt vizsgálni, hogy hova jut egy program, ahonnét kiindulva a program nem "száll el". A második pont értelemszerűen azt írja le, hogy ahova a program eljut, az a sorozat utolsó eleme.

A programfüggvény elnevezés megtévesztő lehet, hiszen egy program programfüggvénye nem feltétlenül függvény, sőt az sem biztos, hogy determinisztikus reláció (parciális függvény). Történeti megfontolások alapján azonban mégis megtartottuk ezt az elnevezést.

2.5. Megoldás

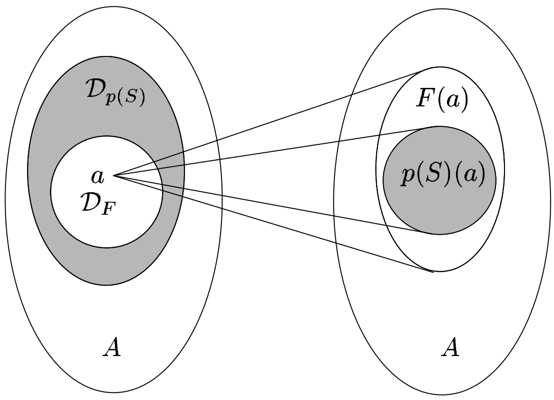
Vegyük észre, hogy a programfüggvény ugyanolyan típusú reláció mint a feladat volt. Így tehát a programfüggvény fogalmának bevezetésével lehetőségünk nyílik arra, hogy kapcsolatot teremtsünk egy adott feladat és egy adott program között. Természetesen ennek a kapcsolatnak azt kell leírnia, hogy mikor mondjuk egy programról azt, hogy megold egy adott feladatot.

5. DEFINÍCIÓ: MEGOLDÁS

Azt mondjuk, hogy az S program *megoldja* az F feladatot, ha

1. $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$,
2. $\forall a \in \mathcal{D}_F : p(S)(a) \subseteq F(a)$.

Ezzel a definícióval végül is azt kívánjuk meg, hogy az állapottér olyan pontjaihoz, ahol a feladat értelmezve van, a program csak véges sorozatokat rendeljen (terminál-jon) és a sorozatok végpontjait a feladat hozzárendelje a kezdőponthoz.



2.1. ábra. Megoldás

2.6. Példák

1. példa: Legyen $A_1 = \{1,2\}$, $A_2 = \{1,2\}$, $A_3 = \{1,2,3,4,5\}$, $A = A_1 \times A_2 \times A_3$. $F = \{((a,b,c),(d,e,f)) \mid f = a+b\}$. $F(1,1,1) = ?$ Hány olyan pontja van az állapottérnek, amelyekhez a feladat ugyanazt rendeli, mint az $(1,1,1)$-hez?

Megoldás:
$F(1,1,1) = \{(1,1,2), (1,2,2), (2,1,2), (2,2,2)\}$.
Mivel a feladat hozzárendelése nem függ az állapottér harmadik komponensétől, a feladat ugyanezeket a pontokat rendeli az összes $(1,1,*)$ alakú ponthoz. Más pontokhoz viszont nem rendelheti ugyanezeket a pontokat, mert akkor az összeg nem lehetne 2! Tehát öt olyan pontja van az állapottérnek amelyhez a feladat ugyanazt rendeli, mint az $(1,1,1)$-hez.

2. példa: Legyen $A = \{1,2,3,4,5\}, S \subseteq A \times A^{**}$.

$$
\begin{array}{llllll}
S = \{ & (1, \langle 1251 \rangle), & (1, \langle 14352 \rangle), & (1, \langle 132 \ldots \rangle), & (2, \langle 21 \rangle), & \\
 & (2, \langle 24 \rangle), & (3, \langle 333333 \ldots \rangle), & (4, \langle 41514 \rangle), & (4, \langle 431251 \rangle), & \\
 & (4, \langle 41542 \rangle), & (5, \langle 524 \rangle), & (5, \langle 534 \rangle), & (5, \langle 5234 \rangle) & \}
\end{array}
$$

$F = \{(2,1)\ (2,4)\ (4,1)\ (4,2)\ (4,5)\}$.

a) Adjuk meg $p(S)$-t!

b) Megoldja-e S a feladatot?

Megoldás:

a) Mivel a program az 1-hez és a 3-hoz végtelen sorozatot is rendel, a programfüggvény értelmezési tartománya:

$$\mathcal{D}_{p(S)} = \{2,4,5\}.$$

Ekkor a programfüggvény:

$$p(S) = \{(2,1), (2,4), (4,4), (4,1), (4,2), (5,4)\}.$$

b) A megoldás definíciója két pontjának teljesülését kell belátnunk.

$$
\begin{array}{ll}
i. & \mathcal{D}_F = \{2,4\} \subseteq \{2,4,5\} = \mathcal{D}_{p(S)}. \\
ii. & p(S)(2) = \{1,4\} \subseteq \{1,4\} = F(2), \\
 & p(S)(4) = \{4,1,2\} \not\subseteq \{1,2,5\} = F(4),
\end{array}
$$

tehát az S program nem megoldása az F feladatnak.

3. példa: Fejezzük ki a programok uniójának programfüggvényét a programok programfüggvényeivel!
Megoldás: Legyenek $S_1, S_2 \subseteq A \times A^{**}$ programok. Ekkor a programfüggvény értelmezési tartományának definíciójából kiindulva:

$$
\begin{aligned}
\mathcal{D}_{p(S_1 \cup S_2)} &= \{a \in A \mid p(S_1 \cup S_2)(a) \subseteq A^*\} = \\
&= \{a \in A \mid p(S_1)(a) \subseteq A^* \land p(S_2)(a) \subseteq A^*\} = \\
&= \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)}.
\end{aligned}
$$

Legyen $a \in \mathcal{D}_{p(S_1)} \cap \mathcal{D}_{p(S_2)}$. Ekkor

$$
\begin{aligned}
p(S_1 \cup S_2)(a) &= \{\tau(\alpha) \mid \alpha \in (S_1 \cup S_2)(a)\} = \\
&= \{\tau(\alpha) \mid \alpha \in S_1(a) \lor \alpha \in S_2(a)\} = \\
&= p(S_1)(a) \cup p(S_2)(a).
\end{aligned}
$$

4. példa: Legyen F_1 és F_2 egy-egy feladat ugyanazon az állapottéren! Igaz-e, hogy ha minden program, ami megoldása F_1-nek, az megoldása F_2-nek is, és minden program, ami megoldása F_2-nek, az megoldása F_1-nek is, akkor F_1 és F_2 megegyeznek?
Megoldás: A leggyakoribb hiba, amit ennek a feladatnak a megoldásakor el szoktak követni, az az, hogy összekeverik az állítás feltételrendszerét magával a bizonyítandó állítással, és azt próbálják bebizonyítani, hogy valamelyik feladatnak minden program megoldása. Természetesen általában ez nem igaz, de nem is ez a feladat! Abból kell tehát kiindulnunk, hogy pontosan ugyanazok a programok oldják meg mindkét feladatot, és meg kell vizsgálnunk, hogy következik-e ebből az, hogy a két feladat megegyezik.

Induljunk ki abból, hogy minden program, ami megoldása F_1-nek, az megoldása F_2-nek, és válasszunk egy olyan programot, amelynek programfüggvénye megegyezik az F_1 relációval. Ekkor a választott program triviálisan megoldja az F_1 feladatot, tehát meg kell oldania F_2-t is, azaz:

$$\begin{array}{ll} i. & \mathcal{D}_{F_2} \subseteq \mathcal{D}_{F_1}, \\ ii. & \forall a \in \mathcal{D}_{F_2} : F_1(a) \subseteq F_2(a) \end{array}$$

Most felhasználva, hogy minden program, ami megoldása F_2-nek, az megoldása F_1-nek is, és egy olyan program választásával, amelynek programfüggvénye megegyezik F_2-vel, az előzőekkel analóg módon adódnak a fordított irányú állítások:

$$\begin{array}{ll} iii. & \mathcal{D}_{F_1} \subseteq \mathcal{D}_{F_2}, \\ iv. & \forall a \in \mathcal{D}_{F_1} : F_2(a) \subseteq F_1(a). \end{array}$$

Az $i.$ és $iii.$ állításokból következik, hogy a két feladat értelmezési tartománya megegyezik, míg az $ii.$ és $iv.$ állítások garantálják, hogy ezen közös értelmezési tartomány egyes pontjaihoz mindkét feladat ugyanazokat a pontokat rendeli, azaz $F_1 = F_2$.

5. példa: $F_1 \subseteq F_2$. Az S program megoldja F_2-t. Igaz-e, hogy S megoldja F_1-et is?
Megoldás: Próbáljuk meg bebizonyítani az állítást. Ehhez a megoldás definíciója két pontját kell belátnunk.

$i.$ $\mathcal{D}_{F_1} \subseteq \mathcal{D}_{p(S)}$,

$ii.$ $\forall a \in \mathcal{D}_{F_1} : p(S)(a) \subseteq F_1(a)$.

Az $i.$ pont teljesülése könnyen látható, ugyanis S megoldása F_2-nek, tehát

$$\mathcal{D}_{F_1} \subseteq \mathcal{D}_{F_2} \subseteq \mathcal{D}_{p(S)}.$$

Az $ii.$ pont bizonyításánál azonban gond van, hiszen az alábbi két állítás áll rendelkezésünkre:

$$\begin{array}{ll} \forall a \in \mathcal{D}_{F_1} : & p(S)(a) \subseteq F_2(a), \\ \forall a \in \mathcal{D}_{F_1} : & F_1(a) \subseteq F_2(a). \end{array}$$

és ezekből a kívánt állítás nem bizonyítható. Elakadtunk a bizonyításban, lehet, hogy nem igaz az állítás? Készítsünk ellenpéldát felhasználva azt, hogy hol akadtunk el a bizonyításban!

Legyen $A = \{1,2\}$, $F_1 = \{(1,1)\}$, $F_2 = \{(1,1),(1,2)\}$ és $p(S)$ egyezzen meg az F_2 feladattal. Ekkor S triviálisan megoldja F_2-t, de nem megoldása F_1-nek, ui.

$$1 \in \mathcal{D}_{F_1} \wedge p(S)(1) = F_2(1) = \{1,2\} \not\subseteq \{1\} = F_1(1).$$

Tehát az állítás nem igaz.

2.7. Feladatok

1. Legyen $A = \{\Omega, \Phi, \Psi, \Theta, \Gamma\}$, $S \subseteq A \times A^{**}$.

$$S = \{ \begin{array}{lll} (\Omega, \langle \Omega\Phi\Gamma\Omega \rangle), & (\Omega, \langle \Omega\Theta\Psi\Gamma \rangle), & (\Omega, \langle \Omega\Psi\Phi \dots \rangle), \\ \Phi, \langle \Phi\Omega \rangle), & \Psi, \langle \Psi\Theta \rangle), & \Psi, \langle \Psi\Psi\Psi\Psi\Psi\Psi \dots \rangle), \\ \Theta, \langle \Theta\Omega\Gamma\Omega\Theta \rangle), & \Theta, \langle \Theta\Psi\Omega\Phi\Gamma\Omega \rangle), & \Theta, \langle \Theta\Omega\Gamma\Theta\Phi \rangle), \\ \Gamma, \langle \Gamma\Phi\Psi \rangle), & \Gamma, \langle \Gamma\Psi \rangle), & \Gamma, \langle \Gamma\Phi\Psi\Omega \rangle) \end{array} \}$$

$F = \{(\Phi, \Omega)\ (\Phi, \Psi)\ (\Theta, \Omega)\ (\Theta, \Phi)\ (\Theta, \Theta)\}$.

   a) Adjuk meg $p(S)$-t!
   b) Megoldja-e S a feladatot?

2. Legyen S program, F olyan feladat, hogy S megoldása F-nek. Igaz-e, hogy
   a) ha F nem determinisztikus, akkor S sem az?
   b) ha F determinisztikus, akkor S is az?
   c) ha F nem determinisztikus, akkor $p(S)$ sem az?
   d) ha $p(S)$ determinisztikus, akkor F is az?
   e) ha F determinisztikus, akkor $p(S)$ is az?
   f) ha S nem determinisztikus, akkor $p(S)$ sem az?

3. Igaz-e, hogy $p(S)$ értelmezési tartománya éppen A^* őseképe S-re nézve?

4. Mondhatjuk-e, hogy az S program megoldja az F feladatot, ha igaz a következő állítás:

$$q \in \mathcal{D}_F \Rightarrow S(q) \subseteq A^* \wedge p(S)(q) \subseteq F(q).$$

5. Legyenek S_1 és S_2 programok, F pedig egy feladat egy tetszőleges közös állapottéren. Teggyük fel továbbá, hogy $S_1 \subseteq S_2$ és S_2 megoldja az F feladatot. Igaz-e, hogy S_1 megoldja F-et?

6. Legyen $A = \mathbb{N} \times \mathbb{N}$, $F_1, F_2 \subseteq A \times A$.
   $F_1 = \{((u,v),(x,y)) \mid \ \ y|u\}$,
   $F_2 = \{((u,v),(x,y)) \mid x = u \wedge y|u\}$.
   Ugyanaz-e a két feladat? (Van-e valamilyen összefüggés közöttük?)

7. $F \subseteq A \times A$. S_1, S_2 programok A-n. Az S_1 és az S_2 is megoldja az F feladatot. Igaz-e, hogy az $S = (S_1 \cup S_2)$ program is megoldja az F feladatot?

8. Tekintsük a következő szövegesen megadott feladatot: Adott egy sakktábla, és két rajta lévő bástya helyzete. Helyezzünk el a táblán egy harmadik bástyát úgy, hogy az mindkettőnek az ütésében álljon! Készítsük el a modellt: írjuk fel az állapotteret és az F relációt!

9. Tudjuk, hogy S megoldja F-et (az A állapottéren). Igaz-e, hogy

$$(a \in A \wedge (S(a) \not\subseteq A^* \vee p(S)(a) \not\subseteq F(a))) \Rightarrow a \notin \mathcal{D}_F?$$

10. Legyen $F \subseteq A \times A$ egy feladat és $S \subseteq A \times A^{**}$ egy program. Jelöljük FP-vel azt a relációt, amely F és $p(S)$ metszeteként áll elő. Igaz-e, hogy

    a) ha $\mathcal{D}_{FP} = \mathcal{D}_F$, akkor S megoldja F-et?

    b) ha S megoldja F-et, akkor $\mathcal{D}_{FP} = \mathcal{D}_F$?