Pavol Jozef Šafárik University in Košice
Faculty of Science

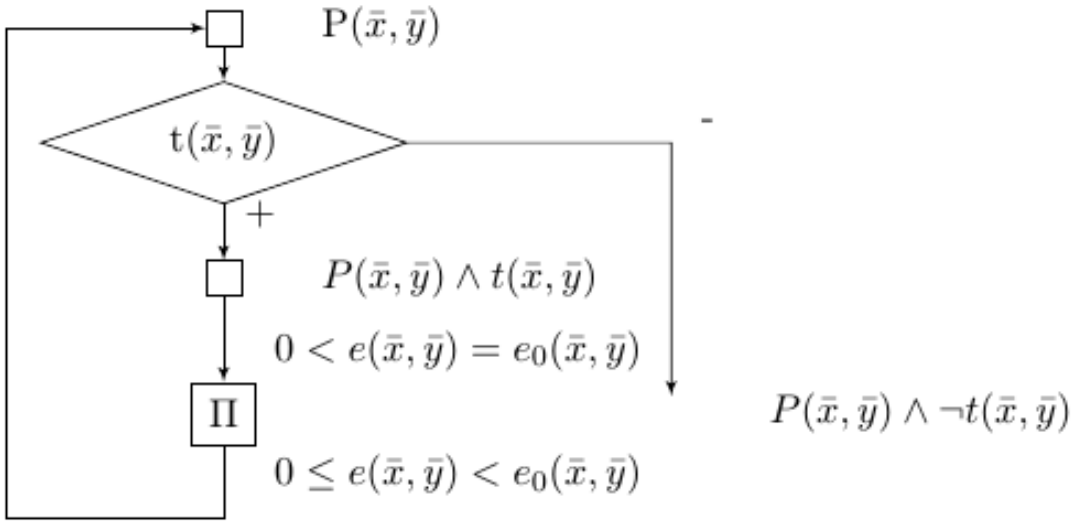
Floydova metóda, program končí

Gabriela Andrejková

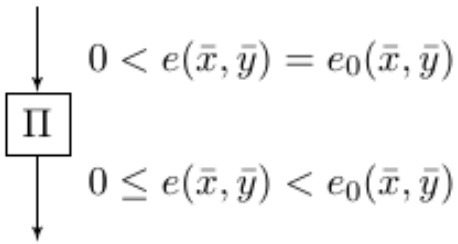
Program končí

Doposiaľ sme sa zaoberali len tým, či program dáva správne výsledky, ak skončí výpočet.

Iná rovnako dôležitá otázka je, či program skončí po konečnom počte krokov. Pretože výpočet podľa sekvencie príkazov bez cyklov zrejme končí, zaujíma nás, kedy končí výpočet podľa cyklu. Uvažujme, napríklad, cyklus



kde $e_0(\bar{x}, \bar{y})$ je celočíselný výraz. Predpokladajme, že platí $P(\bar{x}, \bar{y}) \wedge t(\bar{x}, \bar{y}) \rightarrow (e_0(\bar{x}, \bar{y}) > 0)$ a pre $e_0(\bar{x}, \bar{y}) \in N$ platí



To znamená, že $e(\bar{x}, \bar{y}) \geq 0$ je invariant cyklu, t.j. hodnota výrazu $e(\bar{x}, \bar{y})$ zostáva nezáporná počas celého výpočtu a navyše z druhého predpokladu vyplýva, že vykonaním Π sa hodnota $e(\bar{x}, \bar{y})$ znižuje.

Z toho vyplýva, že opakovaný proces je konečný, pretože $e(\bar{x}, \bar{y})$ musí spĺňovať vzťah $e(\bar{x}, \bar{y}) \geq 0$.

Sformulujme tento proces všeobecnejšie:

Ak je cyklus dobre skonštruovaný, hodnoty premenných v programe sa menia tak, aby raz bola splnená podmienka na ukončenie cyklu. Teda postupnosť hodnôt premenných v programe, vytváraná opakovaním príkazov cyklu, musí spĺňovať určité vlastnosti.

Ak pomocou nich a vstupných údajov je možné vykonaním príkazov cyklu skonštruovať postupnosť prvkov čiastočne usporiadanej množiny, v ktorej neexistuje nekonečná klesajúca postupnosť prvkov, potom taký cyklus určite skončí.

Čiastočne usporiadaná množina (skrátene ČUM) (W,$\sqsubset$) je dvojica tvorená neprázdnou množinou W a binárnou reláciou $\sqsubset$ nad W, ktorá pre všetky $a, b, c \in W$ splňuje:

a) tranzitívnosť: ak a $\sqsubset$ b a b $\sqsubset$ c, potom a $\sqsubset$ c;

b) asymetria: ak a $\sqsubset$ b, potom neplatí b $\sqsubset$ a;

c) antireflexívnosť: neplatí a $\sqsubset$ a.

Pri dokazovaní toho, že program končí, je možné aplikovať sémantické pravidlá, ktoré navrhol R. W. Floyd.

Ich použitie vyplynie z nasledujúceho:

Nech

P($\bar{x}$)	je vstupná podmienka programu Π;
$R_\alpha(\bar{x}, \bar{y})$	je predikát, ktorý zaručuje, že ak v bode A platí $R_\alpha(\bar{x}, \bar{y})$, tak výpočet pôjde po ceste α do bodu B.
$r_\alpha(\bar{x}, \bar{y})$	je funkcia, ktorú program počíta v príkazoch pozdĺž cesty α z bodu A do bodu B;
F($\bar{x}, \bar{y}$)	je funkcia, ktorej obor funkčných hodnôt je čiastočne usporiadaná množina, v ktorej neexistuje nekonečná klesajúca postupnosť - takú množinu budeme nazývať **fundovaná množina**. Indexom pri F vyjadríme funkčnú hodnotu v určitom deliacom bode.

Ak pre každú verifikačnú cestu α medzi dvoma deliacimi bodmi A a B ukážeme platnosť predikátu

$$\forall \bar{x} \forall \bar{y} [P(\bar{x}) \wedge R_\alpha(\bar{x}, \bar{y}) \rightarrow (F_B(\bar{x}, r_\alpha(\bar{x}, \bar{y})) \sqsubset F_A(\bar{x}, \bar{y})] \quad (1)$$

potom tento program končí, pretože platnosť predikátu zaručuje, že funkčná hodnota funkcie F v bode B je menšia než v bode A a zároveň platí, že množina funkčných hodnôt funkcie F je fundovaná množina.

Teda to, že program končí, je možné dokázať v nasledujúcich troch krokoch:

1. Zvoliť deliace body všetkých cyklov programu (sekvencia príkazov bez cyklov zrejme končí).
2. Pre každú verifikačnú cestu α medzi dvoma deliacimi bodmi vytvoriť vhodnú funkciu F s oborom funkčných hodnôt vo fundovanej množine.
3. pre každú verifikačnú cestu α dokázať platnosť predikátu, ktorý má tvar vyššie uvedený.

Z uvedeného vyplýva, že 2. krok, v ktorom sú vytvárané vhodné funkcie je najťažší, pretože v ňom nie je možné postupovať mechanicky, ale vyžaduje hlboké pochopenie algoritmu.

Poznámka:

Pri dokazovaní správnosti programov je potrebné uvedomiť si jednu dôležitú vlastnosť fundovaných množín, ktorú vyjadríme nasledovne:

Nech (W,$\sqsubset$) je fundovaná množina. Označme $W^n, n \geq 1$, množinu všetkých usporiadaných n-tíc prvkov z W (karteziánsky súčin) a $\sqsubset_n$ lexikografické čiastočné usporiadanie množiny W^n, ktoré je definované takto:
$< a_1, ..., a_n > \sqsubset < b_1, ..., b_n > <=> a_1 = b_1, ..., a_{i-1} = b_{i-1}$, a $a_i \sqsubset b_i$ pre nejaké $i, 1 \leq i \leq n$.

Potom $(W^n, \sqsubset_n)$ je fundovaná množina.

Príklad:

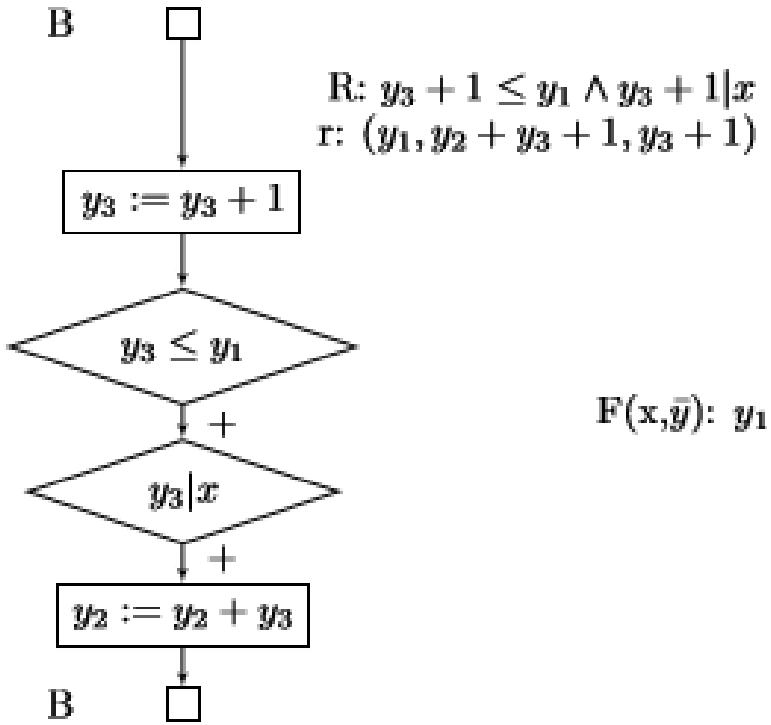
Dokážeme, že program z príkladu o dokonalých číslach končí.

1. Deliace body A, B, C zvolíme tak, ako boli zvolené pri dokazovaní čiastočnej správnosti. Deliace body pri dokazovaní čiastočnej správnosti a toho, že program končí, nemusia byť zvolené rovnako, ale v tomto prípade je to výhodné.
2. Vytvoríme vhodnú funkciu pre cesty $B++B$ a $B+-B$

$$F(x, \overline{y}): \quad y_1 - y_3, \quad y_1, y_3 \in N$$

Pretože $y_1 = x \, div \, 2$ v priebehu celého výpočtu a y_3 rastie, je funkcia F klesajúca. Pretože y_3 je zhora ohraničená, F je ohraničená zdola, a teda v obore funkčných hodnôt funkcií F neexistuje nekonečná klesajúca postupnosť.

a) Verifikačná cesta $B++B$:



b) Verifikačná cesta $B+-B$:

Funkcia F bude v tom istom tvare, a teda vyhovuje.

Predikáty majú tvar

$$\forall x \, \forall \bar{y} [x \in N \wedge y_3 + 1 \leq y_1 \wedge y_3 + 1 | x \rightarrow (y_1 - (y_3 + 1) \leq y_1 - y_3)]$$

a sú zrejme splnené. Analogicky je potrebné dokázať platnosť podmienky pre cestu $B + -B$.
Tým sme dokázali, že program končí.

Pri dokazovaní správnosti zložitejších úloh hľadanie vhodnej funkcie F je zložitejšie, pretože požiadavka nájsť funkciu, ktorá má požadované vlastnosti na celom definičnom obore, je príliš zaväzujúca.

Pre nás sú dôležité hodnoty, ktoré funkcia počas výpočtu programu nadobúda, preto dôkaz toho, že program končí, tomu prispôsobíme a budeme ho robiť v nasledujúcich krokoch (zostanú tri, avšak je potrebné urobiť v nich ďalšie činnosti):

1. činnosť

Zvolíme deliace body, pomocou ktorých sú preťaté všetky cykly programu. Ku každému deliacemu bodu X priradíme vhodnú induktívnu podmienku $\mathrm{Q}_X(\bar{x}, \bar{y})$ takú, že:

a) Pre každú verifikačnú cestu α z počiatočného bodu do bodu X platí

$$\forall \bar{x}[(P(\bar{x}) \wedge R_\alpha(\bar{x}) \rightarrow Q_X(\bar{x}, r_\alpha(\bar{x}, \bar{y}))]. \tag{2}$$

b) Pre každú verifikačnú cestu α z deliaceho bodu X do bodu Y platí

$$\forall \bar{x} \forall \bar{y}[Q_X(\bar{x}, \bar{y}) \wedge R_\alpha(\bar{x}, \bar{y}) \rightarrow Q_Y(\bar{x}, r_\alpha(\bar{x}, \bar{y}))]. \tag{3}$$

2. činnosť

Zvolíme vhodnú fundovanú množinu (W,$\sqsubset$) a ku každému bodu X priradíme parciálnu funkciu $\mathrm{F}_X(\bar{x},\bar{y}) : D\bar{x} \times D\bar{y} \longrightarrow W$, ktorá splňuje

$$\forall\bar{x}\forall\bar{y}[Q_X(\bar{x},\bar{y}) \longrightarrow F_X(\bar{x},\bar{y}) \in W] \tag{4}$$

Najčastejšie je zvolená (N,<) alebo (N+,<).

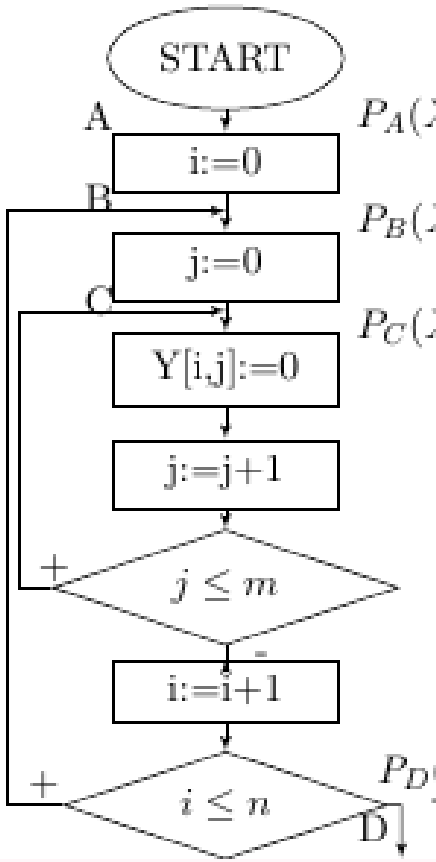
3. činnosť

Dokážeme platnosť predikátu, ktorý je podmienkou ukončenia, t.j. dokážeme, že pre každú verifikačnú cestu α z deliaceho bodu X do deliaceho bodu Y platí

$$\forall \bar{x} \forall \bar{y} [Q_X(\bar{x}, \bar{y}) \wedge R_\alpha(\bar{x}, \bar{y}) \longrightarrow F_Y(\bar{x}, r_\alpha(\bar{x}, \bar{y})) \sqsubset F_X(\bar{x}, \bar{y})]. \quad (5)$$

Úloha

Sú dané dve postupnosti (reťazce) znakov A=$a_1a_2...a_n$,
B=$b_1b_2...b_m$, $n, m \geq 1$, nad konečnou abecedou Σ.
Vytvorte program v jazyku vývojových diagramov, ktorý určí dĺžku najdlhšej spoločnej podpostupnosti týchto dvoch postupností.
Postupnosť C=$c_1c_2...c_p$ nazývame podpostupnosť postupnosti A, ak existuje postupnosť prirodzených čísel $i_1 < i_2 < ... < i_p$ taká, že platí $c_j = a_{i_j}$ pre $1 \leq j \leq p$.
Spoločná podpostupnosť postupností A a B je taká postupnosť, ktorá je podpostupnosťou postupnosti A a zároveň podpostupnosťou postupnosti B
Najdlhšia spoločná podpostupnosť je spoločná podpostupnosť maximálnej dĺžky.

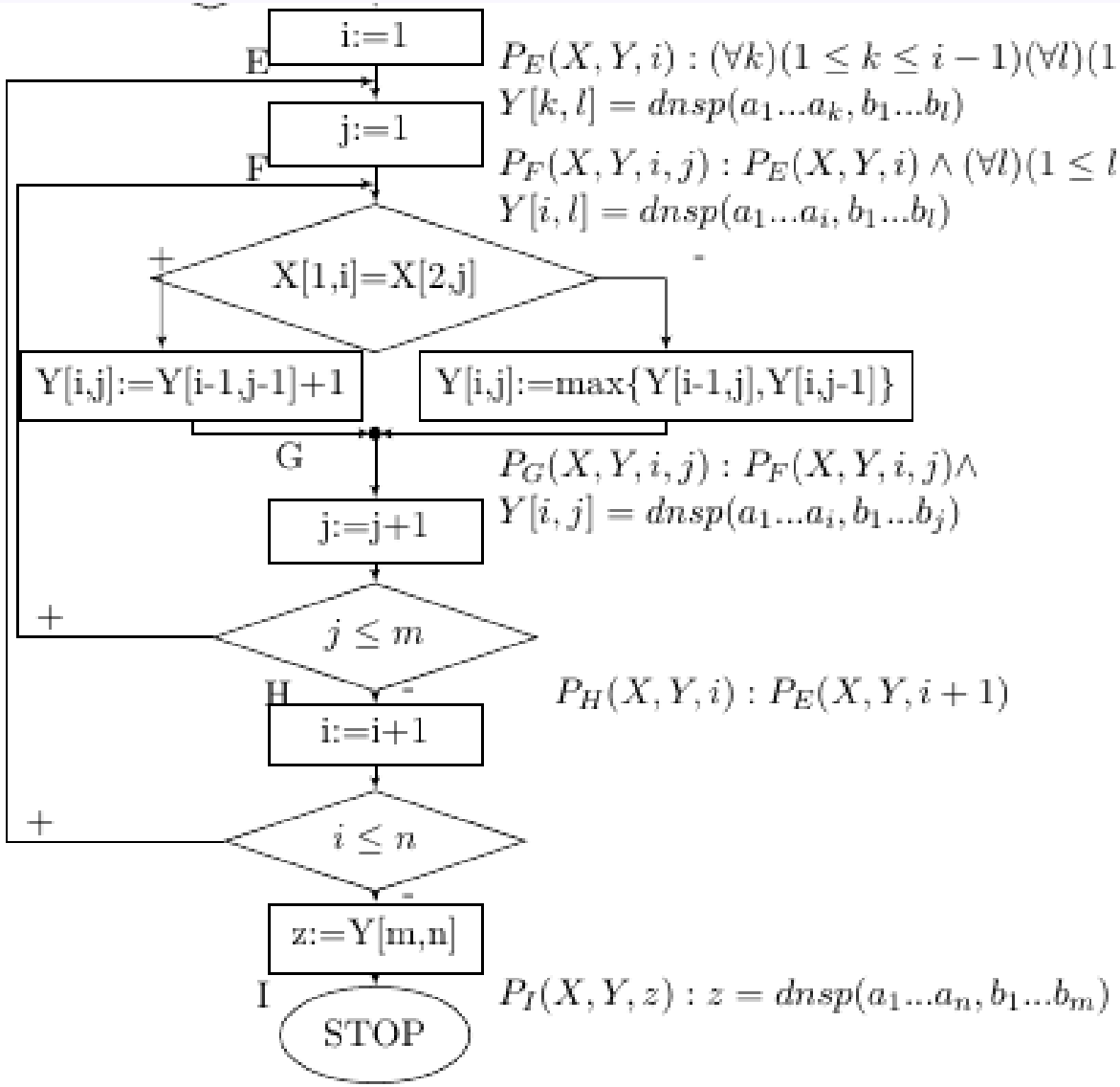


$P_A(X) : m, n \geq 1$

$P_B(X, Y, i) : (\forall k)(0 \leq k \leq i-1)(\forall l)(0 \leq l \leq m) Y[k, l] = 0$

$P_C(X, Y, i, j) : P_B(X, Y, i) \wedge (\forall l)(0 \leq l \leq j-1) Y[i, l] = 0$

$P_D(X, Y) : (\forall k)(0 \leq k \leq n)(\forall l)(0 \leq l \leq m) Y[k, l] = 0$

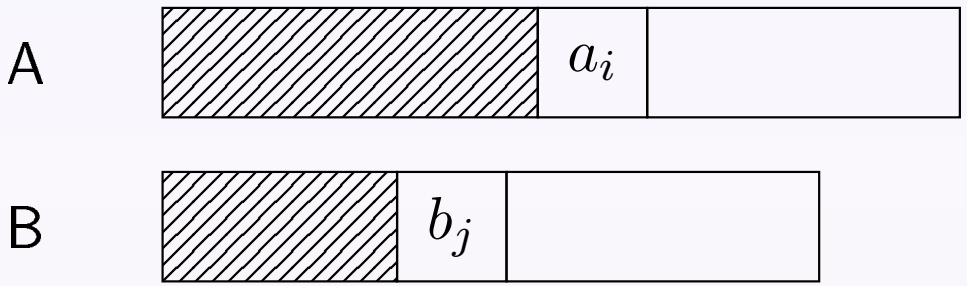


. 2.6: Program, ktorý určuje dĺžku najdlhšej spoločnej podpostupnosti dvoch reťazcov.

Induktívne podmienky vychádzajú z podstaty algoritmu, ktorý tu načrtneme.
Označme $Y[i,j]$ dĺžku najdlhšej spoločnej podpostupnosti postupností $a_1a_2...a_i$ a $b_1b_2...b_j$, skrátene

$$dnsp(a_1a_2...a_i,\, b_1b_2...b_j).$$

Predpokladajme, že v priebehu spracovania sme sa dostali k porovnaniu prvkov a_i a b_j.



V prípade, že $a_i = b_j$ je zrejmé, že $Y[i,j] = Y[i-1,j-1] + 1$.
Keďže platí $Y[i-1,j-1] \leq Y[i-1,j]$ a tiež
$Y[i-1,j-1] \leq Y[i,j-1]$,
v prípade, že $a_i \neq b_j$ je možné určiť $Y[i,j]$ pomocou $Y[i-1,j]$ a $Y[i,j-1]$ a síce $Y[i,j] = max\{Y[i-1,j], Y[i,j-1]\}$.

$$\text{if } a_i = b_j \text{ then } Y[i,j] := [i-1,j-1] + 1 \qquad (6)$$

else $Y[i,j] := max\{Y[i-1,j], Y[i,j-1]\}$, pre $1 \leq i \leq m$.

Vo funkciách r_α a predikátoch R_α, kde budeme pracovať s poľami a argumentom je pole, by mohlo dôjsť k nedorozumeniam, ak sú menené len niektoré hodnoty prvkov poľa, preto pomocou výrazu tvaru [X□$e\sharp$ X[indexy]] vyjadríme, ktorý prvok poľa je práve uvažovaný, kde X je premenná typu pole, indexy určujú prvok poľa X a e je výraz.

Vo funkciách r a predikátoch R budeme používať výrazy tvaru X□X[indexy]=h na vyjadrenie toho, že v poli X prvok X[indexy] nadobudol hodnotu h.

Verifikačná cesta AB:

$$R : true,\ r : (Y, 0, j)$$

Verifikačná podmienka

$$\forall X\,[n, m \geq 1] \rightarrow (\forall k)(0 \leq k \leq -1)(\forall l)(0 \leq l \leq m)Y[k, l] = 0]$$

je splnená triviálne, pretože k spĺňujúce uvedené nerovnosti neexistuje.

Verifikačná cesta BC:

$$R : true,\ r : (Y, i, 0)$$

Verifikačná podmienka

$$\forall X\, \forall Y [P_B(X, Y, i) \rightarrow P_C(X, Y, 0) \wedge (\forall l)(0 \leq l \leq 0 - 1) Y[i, l] = 0]$$

je opäť pravdivá, pretože neexistuje l, spĺňujúce uvedenú nerovnosť.

Verifikačná cesta $C + C$:

$$R : j + 1 \leq m, r : (Y \square Y[i, j] = 0, i, j + 1)$$

Verifikačná podmienka

$$\forall X \, \forall Y \, [P_C(X, Y, i) \wedge (\forall l)(0 \leq l \leq j - 1) Y[i, l] = 0 \wedge j + 1 \leq m$$

$$\rightarrow P_C(X, Y, i) \wedge (\forall l)(0 \leq l \leq j + 1 - 1) Y[i, l] = 0]$$

je splnená, čo vyplýva z vlastnosti funkcie r.

Verifikačná cesta $C - +B$:

$R : j + 1 > m \wedge i + 1 \leq n, r : (Y \square Y[i + 1, j + 1] = 0, i + 1, j + 1)$

Verifikačná podmienka
$\forall X \, \forall Y \, [P_C(X, Y, i) \wedge (\forall l)(0 \leq l \leq j - 1) Y[i, l] = 0 \wedge j + 1 > m \wedge i + 1 \leq n \rightarrow P_B(X, Y, i + 1)]$

je splnená, pretože $j > m - 1$, a teda $j \geq m$. V celom i-tom riadku platí $Y[i, l] = 0$, pre $0 \leq l \leq m$.

Verifikačná cesta $C--D$:

$$R : j+1 > m \wedge i+1 > n,\ r : (Y \Box Y[i+1, j+1] = 0, i+1, j+1)$$

Verifikačná podmienka

$$\forall X\, \forall Y\, [P_C(X, Y, i) \wedge (\forall l)(0 \leq l \leq j-1) Y[i, l] = 0 \wedge j+1 > m$$

$$\wedge i+1 > n \rightarrow P_D(X, Y)]$$

je splnená, pretože $i+1 > n$.

Verifikačná cesta DE:

$$R : true, r : (Y, 1, j)$$

Verifikačná podmienka

$$\forall X \, \forall Y \, [P_D(X, Y) \rightarrow P_E(X, Y, 1)$$

je splnená triviálne, pretože neexistuje k spĺňujúce $1 \leq k \leq 0$.

Verifikačná cesta EF:

$$R : true, r : (Y, 1, 1)$$

Verifikačná podmienka

$$\forall X \, \forall Y \, [P_E(X, Y, i) \rightarrow P_F(X, Y, 1, 1)]$$

je splnená, pretože neexistuje l spĺňujúce $1 \leq l \leq 0$.

Verifikačná cesta $F - G$:

$R : X[1,i] \neq X[2,j], r : (Y \square max\{Y[i-1,j], Y[i,j-1]\} \sharp Y[i,j], i, j)$

Verifikačná podmienka

$$\forall X \forall Y \, [P_F(X,Y,i,j) \wedge X[1,i] \neq X[2,j] \rightarrow P_F(X,Y,i,j) \wedge Y[i,j] =$$

$$dnsp(a_1 ... a_i, b_1 ... b_j)]$$

Platnosť tejto verifikačnej podmienky vyplýva zo vzťahu (1.1).

Verifikačná cesta F+G:

$$R : X[1,i] = X[2,j], r : (Y \square Y[i-1,j-1] + 1 \sharp Y[i,j], i, j)$$

Verifikačná podmienka

$$\forall X \forall Y \, [P_F(X,Y,i,j) \wedge X[1,i] = X[2,j] \rightarrow P_F(X,Y,i,j) \wedge Y[i,j] =$$

$$dnsp(a_1 ... a_i, b_1 ... b_j)].$$

je splnená, čo vyplýva z vyššie uvedeného vzťahu.

Verifikačná cesta $G + F$:

$$R : j + 1 \leq m, r : (Y, i, j + 1)$$

Verifikačná podmienka

$$\forall X \, \forall Y \, [P_G(X, Y, i, j) \wedge j + 1 \leq m \rightarrow P_F(X, Y, i, j + 1)$$

. Platnosť zrejmá.

Verifikačná cesta $G - H$:

$$R : j + 1 > m, r : (Y, i, j + 1)$$

Verifikačná podmienka

$$\forall X \, \forall Y \, [P_E(X, Y, i) \wedge (\forall l)(1 \leq l \leq j - 1)$$

$$Y[i, l] = dnsp(a_1 ... a_i, b_1 ... b_l)] \wedge Y[i, j] = dnsp(a_1 ... a_i, b_1 ... b_l)]$$

$$\wedge j + 1 > m \rightarrow P_E(X, Y, i + 1)].$$

Platnosť vyplýva z toho, že j+1>m.

Verifikačná cesta $H + E$:

$$R : i + 1 \leq n, r : (Y, i, 1, j)$$

Verifikačná podmienka

$$\forall X \left[P_E(X, Y, i + 1) \wedge i + 1 \leq n \rightarrow P_E(X, Y, i + 1) \right]$$

Platnosť zrejmá.

Verifikačná cesta $H - I$:

$$R : i > m, r : (Y, i + 1, j)$$

Verifikačná podmienka

$$\forall X \forall Y \left[P_H(X, Y, i) \wedge i > m \rightarrow P_I(X, Y, z) \right]$$

je splnená triviálne.

Tým sme ukázali, že program je čiastočne správny.

Je zrejmé, že v tomto príklade sme nemuseli zaviesť deliace body G a H, ale ich zavedením sme zjednodušili dôkaz. Tento príklad je zároveň aplikáciou Floydovej metódy pre programy, ktoré pracujú s poľami.

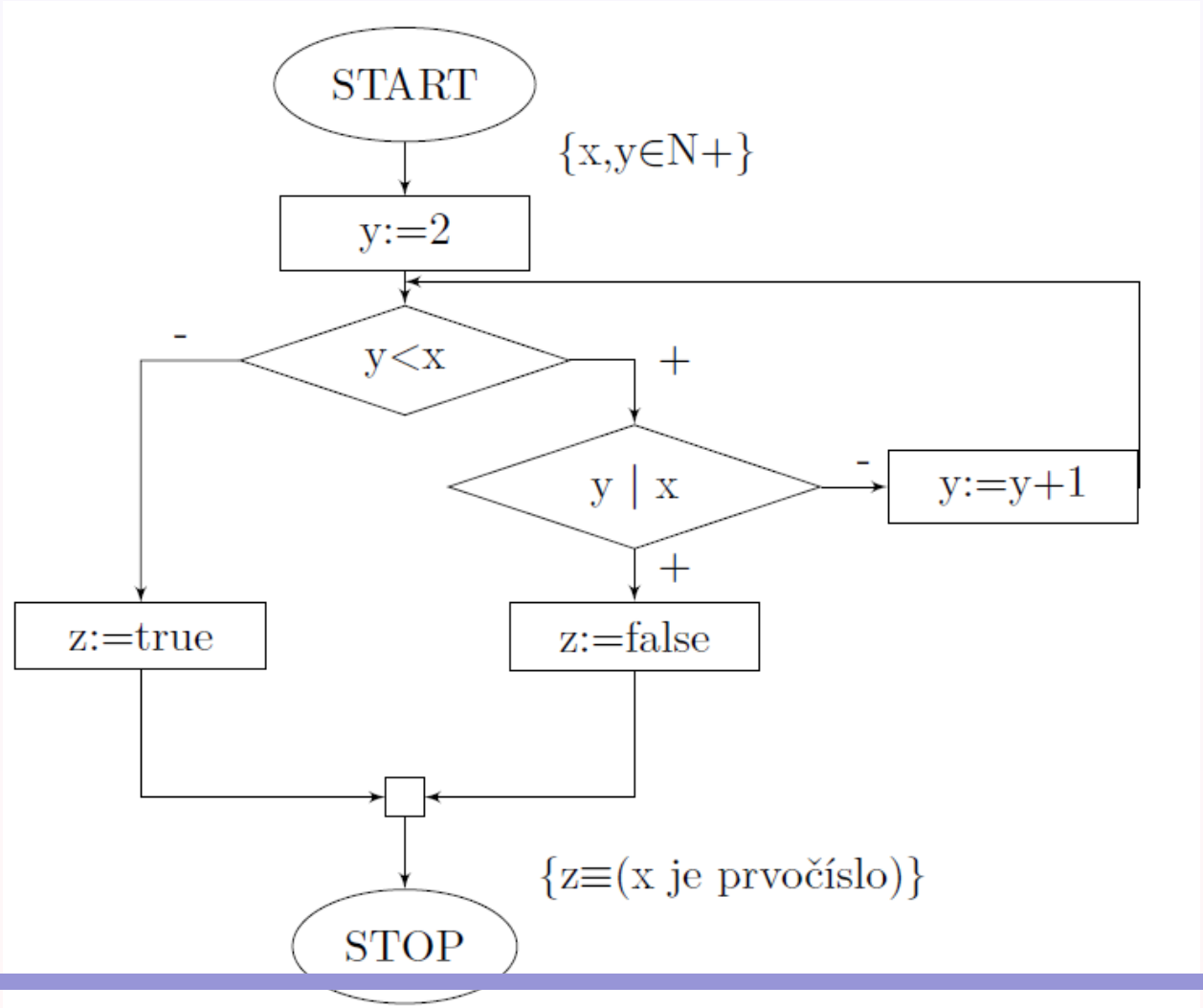
Príklad:

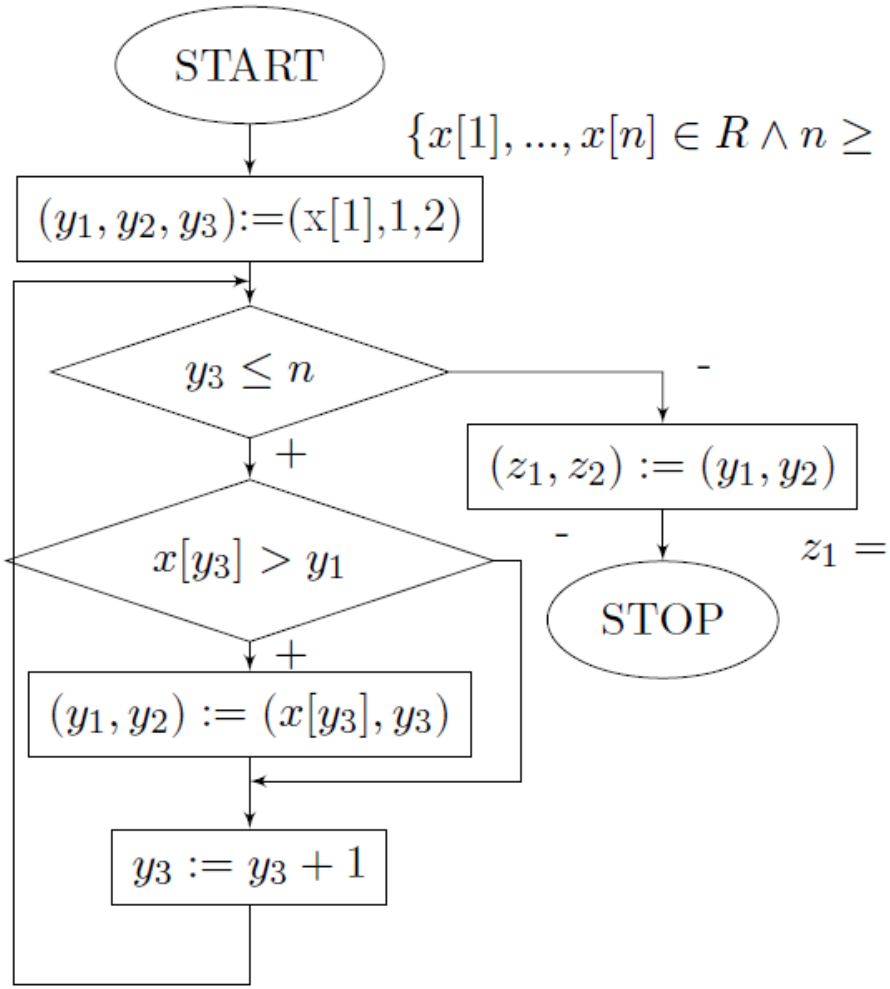
Pri určovaní podreťazcov v reťazcoch sa stretneme s funkciou definovanou takto:

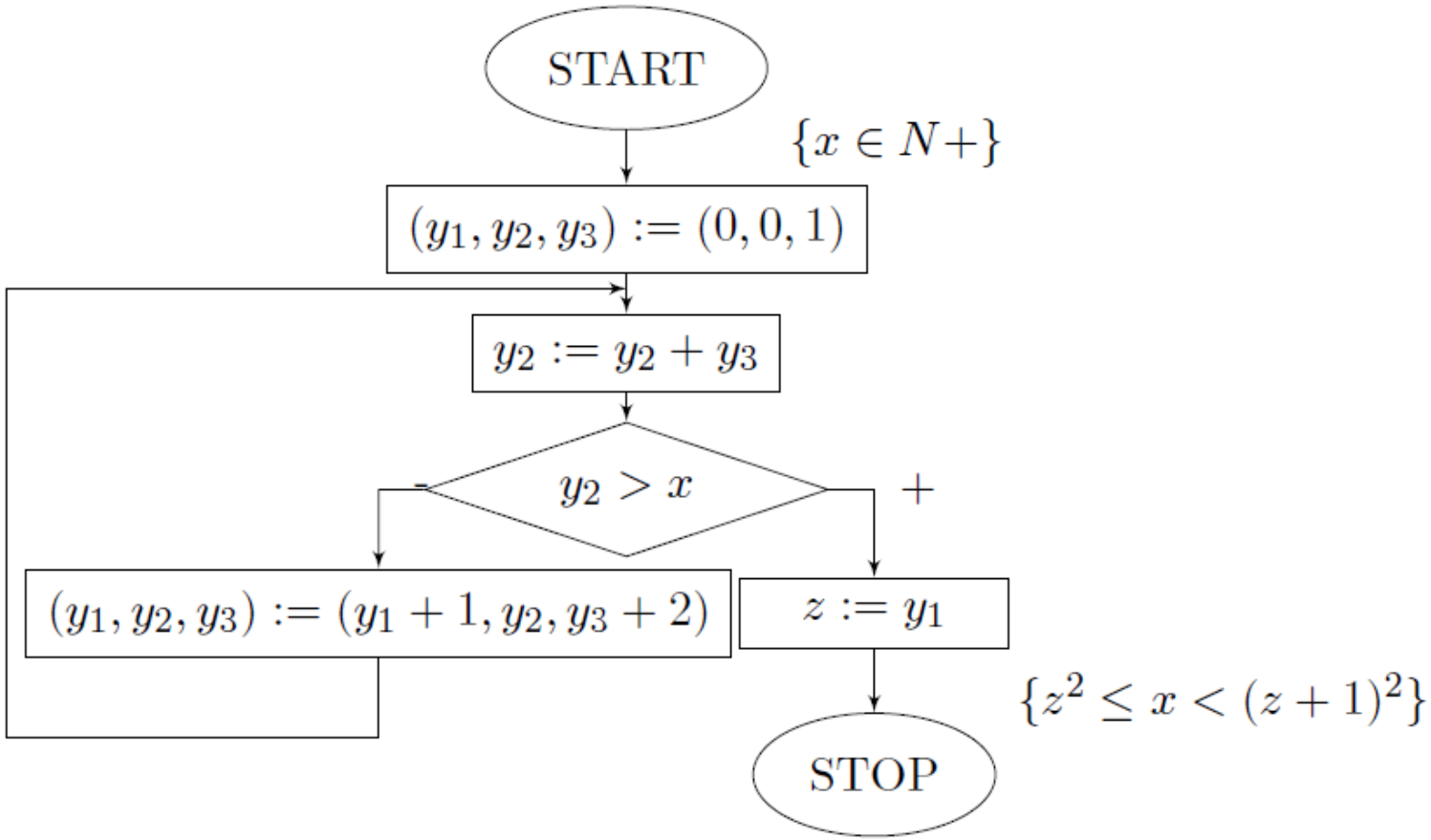
$$f(j) = \begin{cases} max\ \{\ s : s < j \wedge b_1 b_2 ... b_s = b_{j-s+1} ... b_j\ \} \\ 0, \textit{ ak neexistuje } s \geq 1, \end{cases} \tag{7}$$

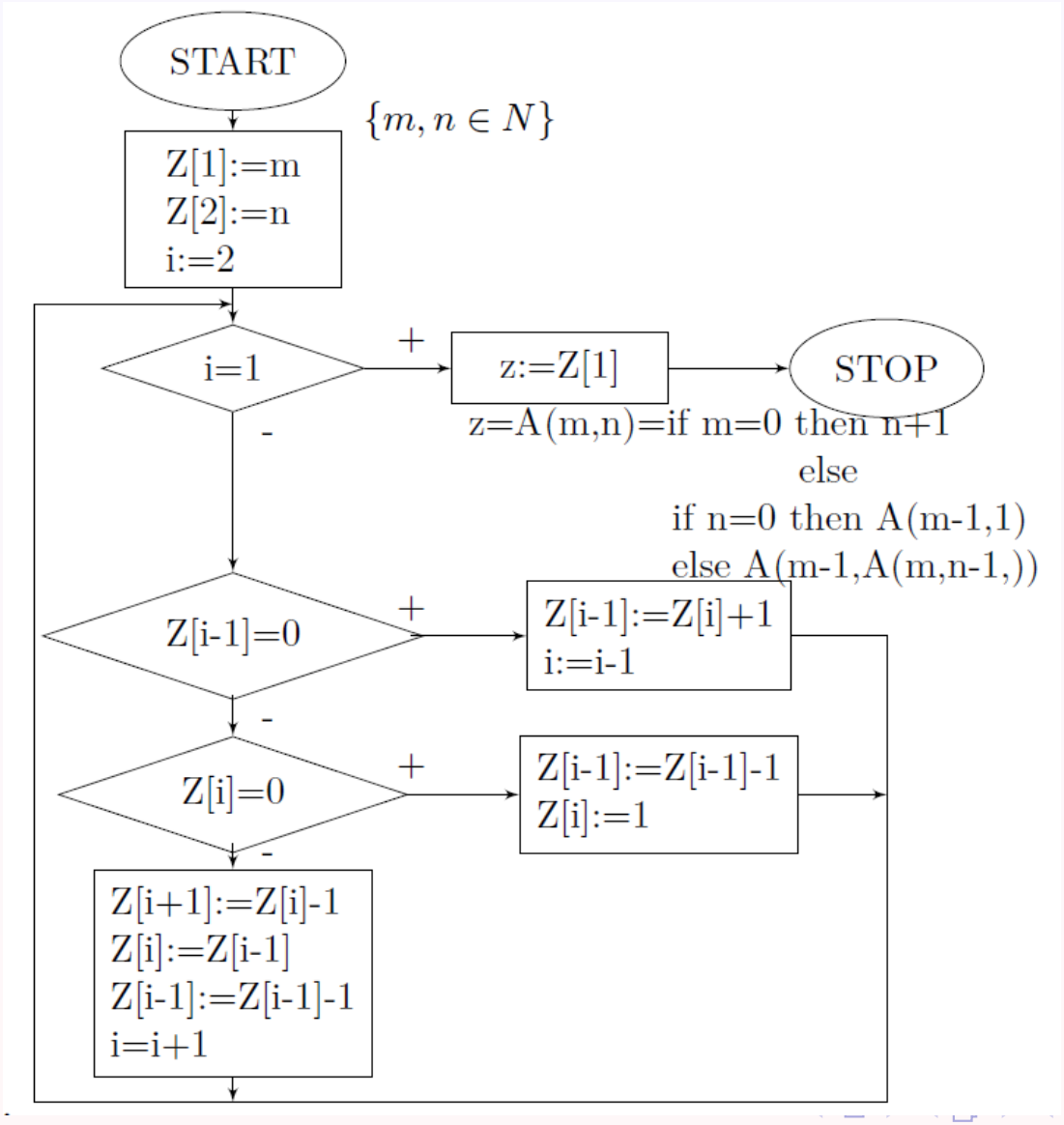
pre $1 \leq j \leq n$ *a reťazec* $b_1 b_2 ... b_n$ *nad abecedou A.*
Vytvorte program, ktorý počíta túto funkciu pre daný reťazec $b_1 b_2 ... b_n$*. Dokážte, že skončí.*

Dokážte správnosť nasledujúcich programov v jazyku vývojových diagramov vzhľadom na vstupné a výstupné podmienky:




START
$\{x[1], ..., x[n] \in R \wedge n \geq 1\}$
(y_1, y_2, y_3):=(x[1],1,2)
$y_3 \leq n$
-
+
$(z_1, z_2) := (y_1, y_2)$
$x[y_3] > y_1$
-
+
STOP
$z_1 = max\{x[1], ..., x[n]\} \wedge z_1 = x[z2]$
$(y_1, y_2) := (x[y_3], y_3)$
$y_3 := y_3 + 1$


START
$\{x \in N+\}$
$(y_1, y_2, y_3) := (0, 0, 1)$
$y_2 := y_2 + y_3$
$y_2 > x$
−
+
$(y_1, y_2, y_3) := (y_1 + 1, y_2, y_3 + 2)$
$z := y_1$
$\{z^2 \leq x < (z+1)^2\}$
STOP


START
$\{m, n \in N\}$
Z[1]:=m
Z[2]:=n
i:=2
i=1
+
z:=Z[1]
STOP
-
z=A(m,n)=if m=0 then n+1
else
if n=0 then A(m-1,1)
else A(m-1,A(m,n-1,))
Z[i-1]=0
+
Z[i-1]:=Z[i]+1
i:=i-1
-
Z[i]=0
+
Z[i-1]:=Z[i-1]-1
Z[i]:=1
-
Z[i+1]:=Z[i]-1
Z[i]:=Z[i-1]
Z[i-1]:=Z[i-1]-1
i=i+1