TBO (INF02598)
PROVA 0 : A REVANCHE
SOLUÇÃO E NOTAS

RAUL H.C. LOPES

1. Introdução

A soluão e as notas da prova seguem.

1.1. **Vetores.** Vale a notação de vetores apresentada em sala com $S.0..m-1$, representando um vetor de m elementos indexado a partir de 0, e $S.i$, o elemento de índice i.

1.2. **Ordem crescente e decrescente.** Dado que S é um vetor de $m > 0$ inteiros, assuma as seguintes definições:

$$increasing.S.l.u \triangleq \forall i, j : l \leq i < j < u : S.i < S.j$$

$$decreasing.S.l.u \triangleq \forall i, j : l \leq i < j < u : S.i > S.j$$

$$bitonic.S.l.u \triangleq \exists i : l \leq i < u : increasing.S.l.i \wedge decreasing.S.i.u$$

Assuma as seguintes abreviações:

$$S(l..u) \uparrow \triangleq\ increasing.S.l.u$$

$$S(l..u) \downarrow \triangleq\ decreasing.S.l.u$$

$$S(l..u) \updownarrow \triangleq\ bitonic.S.l.u$$

$$setof.S, l, u \triangleq\ \{S.i : l \leq i < u\}$$

Note que $setof.S, l, u$ é uma abreviação para $setof(S, l, u)$.

2. As questões

Questão 1. *Dados inteiros $x \geq 0$ e $y > 0$, derive algoritmos para calcular inteiros q e r que serão respectivamente quociente e resto da divisão inteira de x por y.*

Solução.

$division(x,\ y,\ q,\ r)\ \ is$
$\{x, y \in Int \wedge x > 0 \wedge y > 0\}$
$[\![$
initially $q{=}0 \wedge r{=}x$

do $\{P\}$
$r \geq y \ \rightarrow\ r, q, := r-y, q+1$
od
$\{P \wedge \neg B\}$
$\{R\}$
$]\!]$

FIGURA 1. Algoritmo de divisao inteira

Se o objetivo consiste em calcular resto e quociente da divisão inteiro, então ele pode ser traduzido em calcular

$$R \stackrel{\triangle}{=}\ 0 \leq r < y \wedge x = qy + r$$

o que, por processo de weakening*, dá*

$$x = qy + r$$

Sem mais truques e usando o método que o "tio" ensinou lá no primário, conte quantas vezes é possível subtrair y de x, o que sugere que no início:

$$q = 0 \wedge r = x$$

e a invariante

$$P \stackrel{\triangle}{=}\ x > 0 \wedge y > 0 \wedge 0 \leq r \leq x \wedge r = x - qy$$

r sempre decresce e é variável derivada:

$$T \stackrel{\triangle}{=}\ r$$

r deve ser decrementado de y e não pode ser menor do que zero, o que dá a guarda do loop e a transição:

$$B \stackrel{\triangle}{=}\ r \geq y$$

$$S \stackrel{\triangle}{=}\ r, q := r - y, q + 1$$

O algoritmo, complicadíssimo, segue na figura 1.

A correção do algoritmo segue trivialmente dos fatos provados a seguir.

- *P é válido inicialmente. Prova trivial.*

- $P \wedge B \Rightarrow \mathbf{wp}(S, P)$

$$
\begin{array}{ll}
& \mathbf{wp}(S, R) \\
= & \\
& x > 0 \wedge y > 0 \wedge 0 \leq r - y \leq x \wedge r - y = x + (q-1)y \\
= & \langle \textit{porque} (q-1)y = qy - y \rangle \\
& x > 0 \wedge y > 0 \wedge 0 \leq r - y \wedge r - y \leq x \wedge r = x + qy \\
\Leftarrow & \langle \textit{porque } r - y \geq 0 \wedge y > 0 \rangle \\
& x > 0 \wedge y > 0 r \geq 0 \wedge r - y \leq x \wedge r = x + qy \\
\Leftarrow & \langle \textit{porque } y > 0 \wedge r \leq x \Rightarrow r - y \leq x \rangle \\
& P \wedge B
\end{array}
$$

- $P \wedge \neg B \Rightarrow R$

$$
\begin{array}{ll}
& R \\
= & \\
& 0 \leq r < y \wedge x = qy + r \\
= & \langle \textit{aritmética} \rangle \\
& r = x - qy \wedge \wedge 0 \leq r \wedge r < y \\
\Leftarrow & \langle \textit{aritmética } e \text{ weakening} \rangle \\
& P \wedge r < y \\
= & \\
& P \wedge B
\end{array}
$$

- $P \wedge B \Rightarrow T > 0$

$$
\begin{array}{ll}
& T > 0 \\
= & r > 0 \\
\Leftarrow & \langle \textit{aritmética } e \text{ weakening} \rangle \\
& r \geq y \wedge y > 0 \\
\Leftarrow & \langle \textit{weakening} \rangle \\
& P \wedge B
\end{array}
$$

- $P \wedge B \Rightarrow T > T'$

Note $T' = r - y$*, é valor* T *após a transição.*

$$\begin{aligned} T &> T' \\ =& r > r - y \\ =& y > 0 \\ \Leftarrow& P \wedge B \end{aligned}$$

- *Eventualmente o loop termina, quando* $\neg B$ *é válido.*
  Trivial: $T = r$ *decresce, já provado,* y *não muda, eventualmente* $r < y$*, que é* $\neg B$*.*

□

Questão 2. *Dado um vetor* $S(0..m-1)$ *inteiros, sendo* $m > 0$*, derive um algoritmo não recursivo que use um único loop para atualizar* S *de tal forma que* $S.i$ *contenha a soma do prefixo de tamanho* i *dos elementos iniciais. Por exemplo, se* S *tem inicialmente os elementos*

$$< 5, -1, 9, 10, 1, 7, 8 >$$

ao final S *conterá:*

$$< 0, 5, 4, 13, 23, 24, 31 >$$

Por exemplo, $S.3$ *contém ao final a soma:* $0 + 5 + (-1) + 9$*.*

Solução.

O objetivo consiste em estabelecer estabelecer o prescan *de* S*, que é definido pela propriedade* R*:*

$$R = \forall j : 0 \leq j < m : S.j = (+k : 0 \leq k < j : B.k)$$

Generalizando R*, pela subsituição da constante* m *por uma variável* i*, e definindo os limites de* i*, obtém-se a primeira parte da invariante:*

$$P_0 \stackrel{\triangle}{=} \ 0 \leq i \leq m \wedge \forall j : 0 \leq j < i : S.j = (+k : 0 \leq k < j : B.k)$$

Como cada $S.i$ *acumula a soma dos elementos que precediam* i *na seqüência inicial, um artifíciol trivial consiste em manter em alguma variável auxiliar o valor dessa soma:*

$$P_1 \stackrel{\triangle}{=} x = (+k : 0 \leq k < i : B.k)$$

A invariante do loop é P*, é a conjunção das duas condições com ofato de que* $S(i..m-1)$ *não foi alterado.*

$$P \stackrel{\triangle}{=} \ P_0 \wedge P_1 \wedge S(i..m-1) = B(i..m-1)$$

A variável derivada T *é óbvia:*

$$T \stackrel{\triangle}{=} m - i$$

$prescan(S, n)$ is
$\{S(0..m-1), m \in Int \wedge m > 0\}$
$[\![$ ***Var*** $i, x \backslash in$ ***Int***
initially $i{=}0 \wedge x{=}0 \wedge B{=}S$

do $\{P\}$
$m{-}i \neq 0 \rightarrow S.i, x, i{:=}x, S.i, i{+}1$
od
$\{P \wedge \neg B\}$
$\{R\}$
$]\!]$

FIGURA 2. Algoritmo de divisao inteira

A guarda do loop e a transição são dados pro B_0 *e* S_0*:*

$$B_0 \stackrel{\triangle}{=} m - i \neq 0$$

$$S_0 \stackrel{\triangle}{=} S.i, x, i := x, x + S.i, i + 1$$

O algoritmo aparece na figura 2. Note que B *é usado apenas como variável lógica.*

A correção segue trivialmente dos seguintes fatos:

- $P \wedge \neg B_0 \Rightarrow R$

$$\begin{array}{ll} & R \\ = & \\ & i = m \wedge \forall j : 0 \leq j < i : S.j = (+k : 0 \leq k < j : B.k) \\ \Leftarrow & \\ & P \wedge \neg B_0 \end{array}$$

- $P \wedge B \Rightarrow T > 0$

$$\begin{array}{ll} & T > 0 \\ = & \\ & m - i > 0 \\ \Leftarrow & \langle porque\ i \leq m \wedge m - i \neq 0 \rangle \\ & P \wedge B_0 \end{array}$$

- $P \wedge B_0 \Rightarrow T > T'$

$$\begin{aligned} & T > T' \\ =& m - i > m - (i+1) \\ \Leftarrow& P \wedge B_0 \end{aligned}$$

- $P \wedge B_0 \Rightarrow \mathbf{wp}(S_0, P)$

$$\begin{aligned} & \mathbf{wp}(S_0, P) \\ =& \\ & 0 \leq i+1 \leq m \\ & \wedge \forall j : 0 \leq j < i+1 : S'.j = (+k : 0 \leq k < j : B.k) \\ & \wedge x + S.i = (+k : 0 \leq k < i+1 : B.k) \\ & \wedge S'(i+1..m-1) = B(i+1..m-1) \\ =& \langle \textit{assumindo } S(0..i-1) = S'(0..i-1) \rangle \\ & 0 \leq i+1 \leq m \\ & \wedge \forall j : 0 \leq j < i : S.j = (+k : 0 \leq k < j : B.k) \\ & \wedge S'.i = (+k : 0 \leq k < i : B.k) \\ & \wedge x = (+k : 0 \leq k < i : B.k) \\ & \wedge S.i = B.i \\ & \wedge S'(i+1..m-1) = B(i+1..m-1) \\ =& \langle S'.i = x \wedge S'(i+1..m-1) = S(i+1..m-1) \rangle \\ & 0 \leq i+1 \leq m \\ & \wedge \forall j : 0 \leq j < i : S.j = (+k : 0 \leq k < j : B.k) \\ & \wedge x = (+k : 0 \leq k < i : B.k) \\ & \wedge x = (+k : 0 \leq k < i : B.k) \\ & \wedge S.i = B.i \\ & \wedge S(i+1..m-1) = B(i+1..m-1) \\ \Leftarrow& \langle 0 \leq i \leq m \wedge m \neq i \Rightarrow 0 \leq i+1 \leq m \rangle \\ & P \wedge B_0 \end{aligned}$$

□

Questão 3. *Dada um vetor $S(0..m-1)$ de $m > 0$ inteiros distintos,tal que*

$$\textit{bitonic.S.0.m}$$

contrua um algoritmo não-recursivo que use um único loop para calcular Z tal que

- *Z é um vetor de m inteiros que representa uma permutação de S.*
- *$increasing.Z.0.m$*

Solução.

O objetivo consiste em estabelecer R

$$R \triangleq\ setof.Z, 0, m = setof.S, 0, m \wedge Z(0..m) \uparrow$$

Os seguinte teorema é fundamental para a solução.

Teorema 1. $0 \leq i < j \leq m \wedge S.i < S.j \wedge S(0..m) \updownarrow \Rightarrow S.i > S(i-1)$

Prova.

Assuma, em argumento de contradição, que $S.i < S(i-1)$. Então, porque a seqüência é bitônica, $S(i-1..m) \downarrow$. Porque $i < j$, $S.i > S.j$, o que contradiz umas das premissas do teorema.

Um algoritmo para ordenar S deve repetir m passos de seleção do menor (não selecionado previamente) elemento e sua transferência para Z. Dada a organização de S, $S(0..m) \updownarrow$, o menor dos elementos de S pode estar no início ou no fim da seqüência, e isso se repetirá a cada seleção do próximo menor. Isso sugere consumir S pelos dois flancos simultaneamente, respeitando:

- *$S(0..i-1)$ é porção esquerda de S já consumida;*
- *$S(j+1..m-1)$ é porção direita de S já consumida;*
- *$Z(0..k-1)$ contém os elementos já transferidos de S*

Isso sugere a invariante P

$$\begin{aligned} P \triangleq & 0 \leq i \leq m \wedge 0 \leq j \leq m \wedge 0 \leq k \leq m \\ & \wedge S(0..i) \uparrow \\ & \wedge S(j+1..m) \downarrow \\ & \wedge Z(0..k) \uparrow \\ & \wedge setof.Z, 0, k = setof.S, 0, i \cup setof.S, j+1, m \end{aligned}$$

A variável derivada T conta o número de elementos a transferir

$$T \triangleq\ j - i + 1$$

A guarda do loop fica

$$G \triangleq j - i \geq 0$$

$merge(S,m)\ \ is$
$\{S(0..m-1)\mathbf{Int} \wedge m > 0 \wedge S \updownarrow 0m\}$
$[\![\textbf{\textit{Var}}\ \ i\ ,\ j\ ,\ k \backslash\ in\ \ \textbf{\textit{Int}}$
$\quad \textbf{\textit{initially}}\ \ i{=}0 \wedge j{=}m{-}1 \wedge k{=}0$

$\quad \textbf{\textit{do}}\ \{P\}$
$\qquad j{-}i \geq 0\ \rightarrow\ \textbf{\textit{if}}$
$\qquad\qquad\qquad S\,.\,i \leq S\,.\,j\ \rightarrow\ Z\,.\,k\ ,\ i\ ,\ k := S\,.\,i\ ,\ i+1, k+1$
$\qquad\qquad\qquad [\!]\ S\,.\,j\ <\ S\,.\,i\ \rightarrow\ Z\,.\,k\ ,\ i\ ,\ k := S\,.\,j\ ,\ j-1, k+1$
$\qquad\qquad \textbf{\textit{fi}}$
$\quad \textbf{\textit{od}}$
$\quad \{P \wedge \neg G\}$
$\quad \{R\}$
$]\!]$

FIGURA 3. Algoritmo de merge

Cada transição do loop transfere S.i ou S.j definido pelas seguintes guardas e comandos respectivos:

$$B_0 \triangleq S.i \leq S.j \qquad S_0 \triangleq\ Z.k, i, k := S.i, i+1, k+1$$
$$B_1 \triangleq S.j < S.i \qquad S_1 \triangleq\ Z.k, j, k := S.j, j-1, k+1$$

O algorimo completo aparece na figura 3. A correção do algoritmo é segue dos seguintes fatos.

- $B_0 \vee B_1$ *é sempre verdadeiro, o que garante que a transição* ***if*** *dentro do loop não bloqueia.*
- $P \wedge G \wedge B_0 \Rightarrow T > 0$

$$\begin{aligned} & T > 0 \\ =& j - i + 1 > 0 \\ =& j - i \geq 1 \\ \Leftarrow& G \\ \Leftarrow& P \wedge G \wedge B_0 \end{aligned}$$

- $P \wedge B_1 \wedge G \Rightarrow T > 0$
  Análogo ao anterior.

- $P \wedge G \wedge B_0 \Rightarrow T > T'$ *Assumindo que* S_0 *seja executado.*

$$\begin{aligned} & T > T' \\ =& j - i + 1 > j - (i+1) + 1 \\ =& j - i + 1 > j - i \\ =& true \\ \Leftarrow & P \wedge G \wedge B_0 \end{aligned}$$

- $P \wedge G \wedge B_1 \Rightarrow T > T'$
  Análogo ao anterior.
- $P \wedge \neg G \Rightarrow R$

$$\begin{aligned} & R \\ = & \\ & i + 1 = j \wedge k = m \\ & \wedge\, setof.Z, 0, k = setof.S, 0, i \cup setof.S, j + 1, m \\ & \wedge\, Z(0..k) \uparrow \\ \Leftarrow & P \wedge \neg G \end{aligned}$$

- $P \wedge G \wedge B_0 \Rightarrow \mathbf{wp}(S_0, P)$

$$\begin{aligned} & \mathbf{wp}(S_0, P) \\ = & \\ & 0 \leq i + 1 \leq m \wedge 0 \leq j \leq m \wedge 0 \leq k + 1 \leq m \\ & \wedge\, S(0..i+1) \uparrow \\ & \wedge\, S(j+1..m) \downarrow \\ & \wedge\, Z'(0..k+1) \uparrow \\ & \wedge\, setof.Z', 0, k+1 = setof.S, 0, i+1 \cup setof.S, j+1, m \\ = & \\ & 0 \leq i + 1 \leq m \wedge 0 \leq j \leq m \wedge 0 \leq k + 1 \leq m \\ & \wedge\, S(0..i) \uparrow \wedge S(i-1) < S.i \\ & \wedge\, S(j+1..m) \downarrow \\ & \wedge\, Z(0..k) \uparrow \wedge Z(k-1) < Z'.k = S.i \\ & \wedge\, setof.Z, 0, k = setof.S, 0, i \cup setof.S, j+1, m \\ & \wedge\, Z'.k = S.i \Leftarrow \qquad \langle por\ Z(k-1) = S(i-1) < S.i \\ & \quad e\ por\ teorema\ teorema\ 1 \rangle \\ & P \wedge G \wedge B_0 \end{aligned}$$

Aluno	Q1	Q2	Q3	Prova
Evelin Amorin	9	6	0	5
Frederico Franzoni	10	10	0	6,7
Leonardo Contadini	3	3	0	2
Wallace Rocha	6	3	0	3
Aline Martins	3	3	0	2
Raphael Santos	6	3	0	3
Almir Fernandes	3	0	0	1
Diego Velasco	3	0	0	1
Lucio Ribeiro	0	0	0	0
Viviane Ventura	3	3	0	2
Bruno Segrini	3	3	0	2
Bruno Monteiro	3	3	0	2
Camilo Grobério	3	3	0	2
Nuno Rasseli	3	6	0	3
Mauro Pinto	3	3	0	2
Giancarlo Del Piero	0	0	0	0
Rômulo Douto	0	0	0	0
Aerthon Cabral	0	0	0	0
Celestino Borges	0	0	0	0

FIGURA 4. Notas da prova 0

- $P \wedge G \wedge B_1 \Rightarrow \mathbf{wp}(S_1, P)$
  Análogo ao anterior.

□

3. AS NOTAS

As notas da prova aparecem na figura 4.