UNITY

Aufgabe 11: Sattelpunkt einer Matrix

Sei $\mathcal{A}[0 \ldots N-1, 0 \ldots N-1]$ eine Matrix von Integer-Werten. Ein Element $A[u,v]$ nennt man einen *Sattelpunkt* der Matrix, genau dann wenn gilt:

$$\begin{array}{rcl} A[u,v] & = & \langle \min i : 0 \leq i < N :: A[i,v] \rangle \wedge \\ A[u,v] & = & \langle \max j : 0 \leq j < N :: A[u,j] \rangle, \end{array}$$

wenn es also das kleinste Element seiner Spalte und das größte seiner Zeile ist.
Geben Sie ein Programm an, das bestimmt, ob eine gegebene Matrix einen Sattelpunkt besitzt, d.h. berechnen Sie den Wert der boolschen Variable *sp*, gegeben durch:

$$\begin{array}{rcl} sp & = & \langle \exists u, v : 0 \leq u < N \wedge 0 \leq v < N :: \\ & & A[u,v] = \langle \min i : 0 \leq i < N :: A[i,v] \rangle \wedge \\ & & A[u,v] = \langle \max j : 0 \leq j < N :: A[u,j] \rangle \\ & & \rangle. \end{array}$$

Aufgabe 12: Leser-Schreiber Problem

Wir betrachten ein Programm, `user`, das zwei Integervariablen nr,nw enthält. Für diese gilt für eine gegebene Konstante N:

$$\textbf{invariant } 0 \leq nr \leq N \wedge 0 \leq nw \leq N. \tag{1}$$

Initial gilt $nr = nw = 0$. Die Variablen nr und nw werden nur in folgenden Anweisungen in `user` benutzt:

{`startread`}	$nr := nr + 1$
{`endread`}	$nr := nr - 1$
{`startwrite`}	$nw := nw + 1$
{`startwrite`}	$nw := nw - 1$

Diese Anweisungen dürfen andere Variablen verändern und können mit Bedingungen (if-clause) versehen sein. Außerdem kann `user` weitere Anweisungen und Variablen haben.
Damit eine weitere Invariante gültig ist, ist es notwendig die Anweisungen von `user` zu modifizieren:

$$\textbf{invariant } nw \leq q \wedge (nr = 0 \vee nw = 0). \tag{2}$$

Dieses Problem ist als *Leser-Schreiber* Problem bekannt. Die Variablen nr, nw beinhalten die Anzahl von Prozessen, die auf eine Datei lesend bzw. schreibend zugreifen. Eine Leseoperation wird begonnen durch das Ausführen der `startread`-Anweisung und beendet mit einem `endread`; `startwrite` und `endwrite` klammern eine Schreiboperation. Die Anzahl der Prozesse im System ist N, was wir Invariante (1) entnehmen können. Invariante (2) wird notwendig durch die Eigenheiten der Lese- und Schreiboperationen: Eine beliebige Anzahl von Leseoperationen kann nebenläufig ausgeführt werden, während eine Schreiboperation nicht nebenläufig zu einer Leseoperation oder einer weiteren Schreiboperation ausgeführt werden kann.

Lösungsvorschlag Aufgabe 11

Alle Indices i, j, u, v seien über einem Definitionsbereich $0 \ldots N-1$ definiert. Für jedes u, v gilt dabei:

$$\langle \min i :: A[i, v] \rangle \leq A[u, v] \leq \langle \max j :: A[u, j] \rangle$$

$A[u, v]$ ist daher genau dann ein Sattelpunkt, wenn gilt:

$$\langle \min i :: A[i, v] \rangle \geq \langle \max j :: A[u, j] \rangle$$

Wir erhalten daher:

$$sp = \langle \exists u, v :: \langle \min i :: A[i, v] \rangle \geq \langle \max j :: A[u, j] \rangle \rangle$$

Sei $X[v]$ gegen als $\langle \min i :: A[i, v] \rangle$ und $Y[u]$ als $\langle \max j :: A[u, j] \rangle$. Dann ist

$$\begin{aligned} sp &= \langle \exists u, v :: X[v] \geq Y[u] \rangle \\ &= (\langle \max v :: X[v] \rangle \geq \langle \min u :: Y[u] \rangle) \end{aligned}$$

Zur Berechnung von sp kann damit folgendes Programm angegeben werden:

Programm `Sattelpunkt`
  declare X, Y : array$[0 \ldots N-1]$of integer
  always
    $\langle \| v :: X[v] = \langle \min i :: A[i, v] \rangle \rangle$
  $\|$ $\langle \| u :: Y[u] = \langle \max j :: A[u, j] \rangle \rangle$
  $[\!]$ $sp = (\langle \max v :: X[v] \rangle \geq \langle \min u :: Y[u] \rangle)$
end { Sattelpunkt}

Auf einer Monoprozessormaschinee kann das Programm mit einem Aufwand $O(N^2)$ ausgeführt werden, auf einer Parallelmaschine mit $O(N^2)$ Prozessoren mit $O(\log N)$

Lösungsvorschlag Aufgabe 12

Mit,

$$0 \leq nr \leq N \wedge 0 \leq nw \leq N,$$

ist folgendes leicht zu sehen:

$$[nw \leq q \wedge (nr = 0 \vee nw = 0)] \equiv [(nr + N \times nw) \leq N].$$

Deshalb implementieren wir die Invariante $(nr + N \times nw) \leq N$ statt der Invariante (2). Definiere der Übersicht halber eine Variable t:

$$t = N - (nr + N \times nw),$$

mit der Forderung:

$$\textbf{invariant } t \geq 0. \tag{3}$$

Ein Reduzieren von nr oder nw vergrößert t, `startread`- und `endread`-Anweisungen erhalten deshalb $t \geq 0$. Eine `startread`-Anweisung erhöht nr um 1, dadurch wird t um 1 erniedrigt. Deshalb kann eine `startread`-Anweisung nur unter der

Voraussetzung $t \geq 1$ ausgeführt werden. Eine ähnliche Herleitung begründet, daß eine `startwrite`-Anweisung die Vorbedingung $t \geq N$ benötigt.

Modifiziertes Programm `user`

declare	t : integer	
always	$t = N - (nr + N \times nw)$	
assign		
{`startread`}	$nr := nr + 1$	`if` $t \geq 1$
$\|$ {`endread`}	$nr := nr - 1$	
$\|$ {`startwrite`}	$nw := nw + 1$	`if` $t \geq N$
$\|$ {`startwrite`}	$nw := nw - 1$	

end { user}