Programación Concurrente
Práctico 3: Sincronización y Progreso

J. Blanco, N. Wolovick

17 de septiembre de 2001

1. Considere un multiprograma con 1 componente de tipo A y 47 componentes de tipo B.

Pre: $s = 0$	
A: $*[s := s + 1]$	B: $*$[**if** $K \leq s \rightarrow$ **skip fi**; $s := s - 1$]

   Derive para que valores de K se cumple el invariante del sistema $0 \leq s$.

2. Para cada uno de los siguientes programas, analice sus propiedades de progreso. Suponga atomicidad en todas las asignaciones y sentencias guardadas. Suponga también que la composición secuencial tiene menor precedencia que la paralela.

   (a) $\{\neg b\}\ b := true \parallel$ **if** b **skip fi**; $b := false\ \{\neg b\}$

   (b) $\{\neg b\}\ b := true \parallel b := false$; **if** b **skip fi**$\{b\}$

   (c) $\{\neg b\}\ b := true; b := false \parallel$ **if** b **skip fi**$\{?\}$

   (d) $\{\neg b \wedge \neg c\}\ b := true$; **if** c **skip fi** $\parallel\ \ c := true$; **if** b **skip fi**$\{b \wedge c\}$

3. Demuestre progreso en los siguientes multiprogramas

(a)

Pre: $true$	
A: $v := q$; **if** $v = p \rightarrow$ **skip fi**; $v := q$	B: $v := p$; **if** $v = q \rightarrow$ **skip fi**; $v := p$
Post: ?	

(b)

Pre: $x = y$	
A: $*$[**if** $x \neq y \rightarrow$ **skip fi**; $x := x + 1$]	B: $*$[**if** $x = y \rightarrow$ **skip fi**; $y := y + 1$]
Inv: ?	

4. El libro [AO91] plantea la siguiente regla para demostrar *deadlock freedom* (individual y total) para programas paralelos con variables compartidas y sincronización.

   Considere un programa paralelo $S \equiv [S_1 \parallel \ldots \parallel S_n]$.

   (a) Una tupla $(R_1, \ldots, R_n)$ de sentecias se denomina *deadlock potencial de* S si se dan las siguientes condiciones
   - Para todo $i \in \{1, \ldots, n\}$, R_i es una sentencia guardada en la componente S_i o el símbolo E que denota la sentencia vacía y representa la terminación de S_i,
   - para algún $i \in \{1, \ldots, n\}$, R_i es una sentencia guardada.

   (b) Dada un esquema de prueba estandard de corrección parcial para no interferencia $\{p_i\}S_i^*\{q_i\}$, $i \in \{1, \ldots, n\}$, con cada deadlock potencial de S asociamos una tupla $(r_1, \ldots, r_n)$ de aserciones, poniendo para $i \in \{1, \ldots, n\}$:
   - $r_i \equiv pre(R_i) \wedge \neg B$ si $R_i \equiv \textbf{if } B \to S \textbf{ fi}$
   - $r_i \equiv q_i$ si $R_i \equiv E$.

   Si podemos demostrar $\neg \bigwedge_{i=1}^{n} r_i$ para todas las tuplas de aserciones $(r_1, \ldots, r_n)$, ninguno de los deadlocks potenciales pueden ocurrir, estableciendo *deadlock freedom*.

   Utilizando esta regla demuestre ausencia de deadlock para

   (a) el ejercicio (3a) de éste práctico,

   (b) el algoritmo de *Concurrent Vector Writing* derivado en clase.

5. Demuestre el progreso de las N componentes del multiprograma

Pre: $\langle \forall q :: x.q = 0 \rangle$
Comp.p: $*[\textbf{if } \langle \forall q :: x.p \leq x.q \rangle \to \textbf{skip fi};$ $x.p := x.p + 1$ $]$

6. Esta es la versión 3 de la derivación del algoritmo *Concurrent Vector Writing*.

Pre: $i = 0 \wedge j = 0$	
A: **do** $i \neq N \to$ $x.i := 0;$ $\{x.i = 0?\}$ $i := i + 1$ **od** $\{i = N\}$	A: **do** $j \neq N \to$ $\{i \leq j?N_0\}$ $x.j := 1;$ $j := j + 1$ **od** $\{j = N\}$
Inv: $\langle \forall k : 0 \leq k < i : x.k = 0 \rangle$	
Post: $\langle \forall k : 0 \leq k < N : x.k = 0 \rangle$	

   (a) Muestre que si continuamos la derivación fortaleciendo ligeramente la aserción de la Nota 0 a $\{i < j\}$, el multiprograma resultante tiene riesgo de deadlock.

   (b) Realice una derivación alternativa, asegurando la Nota 0 con un skip guardado en B.

7. (WF99-6) Dado el siguiente multiprograma muestre:

Pre: $x = 0 \wedge y = 0 \wedge \neg p \wedge \neg q$	
A: $p := true$; $x := x + 1$; **if** $1 \leq y \rightarrow y := y - 1$ **fi** $\{q\}$	A: $q := true$; $y := y + 1$; **if** $1 \leq x \rightarrow x := x - 1$ **fi** $\{p\}$

   (a) corrección de la anotación,

   (b) progreso,

   (c) invariancia de $0 \leq x \leq 1 \wedge 0 \leq y \leq 1$,

   (d) la precondición $x = 0 \wedge y = 0$,

   (e) que si reemplazamos la sentecia guardada de la componente A por la de menor grado de atomicidad **if** $1 \leq y \rightarrow$ **skip fi**; $y := y - 1$, la corrección se mantiene.

8. (WF99-9) *"No siempre debilitar las guardas de un* ***if*** *o fortalecer las de un* ***do*** *ayuda al progreso".*

   (a) Muestre la terminación del siguiente programa:

Pre: $true$	
A: **do** $x \vee y \rightarrow x := false$ **od**	B: **do** $x \vee y \rightarrow y := false$ **od**

   (b) Pero la terminación no se garantiza si cambiamos a ...

Pre: $true$	
A: **do** $y \rightarrow x := false$ **od**	B: **do** $x \rightarrow y := false$ **od**

   (c) Analice la situación para esta última modificación.

Pre: $true$	
A: **do** $y \rightarrow x := false$ **od**; $x := false$	B: **do** $x \rightarrow y := false$ **od**; $y := false$

9. (WF99-21) Considere un multiprograma con 23 componentes de tipo A y 47 de tipo B.

Pre: $s = 0$	
A: $*[s := s + 1]$	B: $*[s := s - 1]$

   Sincronice las componentes usando solo skips guardados para que $0 \leq s \leq 100$ sea un invariante del sistema.

10. (WF99-38) Sincronice las 3 componentes del siguiente multiprograma de manera tal que la postcondición implique $x = 0 \wedge y = 1 \wedge z = 2$.

Pre: $true$		
A: $x := 0$; $y := 0$; $z := 0$	A: $x := 1$; $y := 1$; $z := 1$	A: $x := 2$; $y := 2$; $z := 2$