CS-E4110 Concurrent Programming
Autumn 2016 - Tutorials
Computational Model and Critical Section

2016-09-13

Task 1 (Ben-Ari, 2.4)

For positive values of K, what are the possible final values of n in the following algorithm? Show how these values are produced.

Algorithm 2.10: Incrementing and decrementing	
integer n ← 0	
p	**q**
integer temp	integer temp
p1: do K times	q1: do K times
p2: temp ← n	q2: temp ← n
p3: n ← temp + 1	q3: n ← temp − 1

Task 2 (Ben-Ari, 2.6)

Consider the following algorithm where each of ten processes executes the statements with i set to a different number in 1,..., 10.

Algorithm 2.16: Concurrent algorithm A
integer array [1..10] C ← ten *distinct* initial values integer array [1..10] D
integer myNumber, count
p1: myNumber ← C[i]
p2: count ← number of elements of C less than myNumber
p3: D[count + 1] ← myNumber

1. What does the algorithm do?
2. What would happen if D in line p3 were replaced by C?
3. What would happen if the array C were initialised with values that are not all distinct? Correct the algorithm to take care of this case. If your solutions requires mutual exclusion, clearly mark the critical section.

Task 3 (Ben-Ari, 3.2)

Construct the state diagram for the following algorithm. Use it so show that mutual exclusion holds for the algorithm. You may abreviate the algorithm to reduce the number of states and state transitions.

Algorithm 3.8: Third attempt	
boolean wantp ← false, wantq ← false	
p	**q**
loop forever	loop forever
p1: non-critical section	q1: non-critical section
p2: wantp ← true	q2: wantq ← true
p3: await wantq = false	q3: await wantp = false
p4: critical section	q4: critical section
p5: wantp ← false	q5: wantq ← false

Task 4 (Ben-Ari, 3.6)

(Manna and Pnueli 1995) Prove that mutual exclusion is satisfied by the following algorithm for the critical section problem:

Algorithm 3.14: Manna-Pnueli algorithm	
integer wantp ← 0, wantq ← 0	
p	**q**
loop forever	loop forever
p1: non-critical section	q1: non-critical section
p2: if wantq = −1	q2: if wantp = −1
wantp ← −1	wantq ← 1
else wantp ← 1	else wantq ← −1
p3: await wantq ≠ wantp	q3: await wantp ≠ − wantq
p4: critical section	q4: critical section
p5: wantp ← 0	q5: wantq ← 0

Note that the `if` statement is atomic. Find a scenario showing that the algorithm is not correct if the `if` statement is not atomic.