Open Boek Tentamen
Concurrent en Distributed Programming 213530

Dinsdag 8 April 2008, 9:00-12:30

University of Twente
department of
computer science

1.1. Translate the following natural language propositions into Linear Time Temporal Logic.
Explain your choices for atomic propositions and how you chose for ambiguous interpretations.
(a) Whenever the critical section is entered it must have been locked before and will be unlocked afterwards.
(b) The stoplight is first green, then yellow, then red, and then green again, etc.
(c) A and B may only hold when also C holds.
(d) A and B hold only in states where *afterwards* C holds.
(e) If A holds infinitely often, B will hold eventually.

10 marks

1.2. Give a realistic, meaningful example in natural language for a proposition that requires nested while-operators when translated to Linear Time Temporal Logic.
Give also the Linear Time Temporal formalisation of your proposition.
("Realistic, meaningful" does not mean a proposition with "a's", "b's" and "c's".)

3 marks

1.3. Proof or disprove the following
(a) $\Box\Diamond A \wedge \Box\Diamond B \Rightarrow \Diamond(A \wedge B)$
(b) $\Box A \;\mathcal{U}\; \Box B \Rightarrow \Diamond\Box(A \wedge B) \vee \Box\neg A$
(c) $A \;\mathcal{U}\; B \Rightarrow \Diamond\ (A \wedge \bigcirc B)$
(d) $(\Diamond A \;\mathcal{U}\; \Box B) \wedge \neg A \Rightarrow \Diamond(A \wedge B)$

12 marks

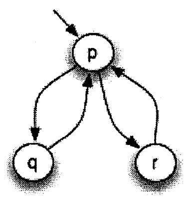
2.1. Explain the concepts of weak fairness and strong fairness using the example of a student who wants to talk to a lecturer and knocks at the lecturer's door for that purpose.

3 marks

2.2. Why do we often not consider unfair computations when we verify concurrent programs?

3 marks

2.3. Give examples for unfair computations in the following transition diagram. How many unfair computations are there?



3 marks

3. Consider the producer-consumer problem, where one process produces hamburgers, one process produces milkshakes, and a third process may take a pair of a hamburger and a milkshake together. Give a semaphore solution to this problem. Explain your solution in natural language (full sentences).

6 marks

boolean wantp:=false, wantq:=false integer turn:=1	
p	q
loop forever p1: non-critical section p2: wantp:=true p3: while wantq p4: if turn=2 p5: wantp:=false p6: await turn==1 p7: wantp:=true p8: critical section p9: turn:=2 p10: wantp:=false	loop forever q1: non-critical section q2: wantq:=true q3: while wantp q4: if turn=1 q5: wantq:=false q6: await turn==2 q7: wantq:=true q8: critical section q9: turn:=1 q10: wantq:=false

4.1 Proof that the following formulae are invariants of the algorithm above:

(i) $turn=1 \lor turn=2$

(ii) $p3..5 \lor p8..10 \Leftrightarrow wantp$

(iii) $q3..5 \lor q8..10 \Leftrightarrow wantq$

and use them to prove that mutual exclusion holds.

8 marks

4.2.1 Draw the transition graph for the algorithm below:
4.2.2 Mark an unfair path. Why is this an unfair path?
4.2.3. What is the reachable state space?
4.2.4. Which path leads to a deadlock?
4.2.5. Is mutual exclusion satisfied? Give an argument using the transition graph.

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

10 marks

5. In a group of people each person has a telephone available. Each person knows the telephone numbers of a few other people of the group, but no one knows all of them. The group wants to make an appointment for a meeting. There are three plans available, plan A, plan B, and plan C, and each person supports one plan.

5.1. Describe a distributed algorithm in natural language that finds out what the majority vote of the group is (which plan has most supporters). In the end, everybody has to know what the majority plan is.

5.2. What makes the algorithm that you suggest a typical "distributed" algorithm? Are there elements in your algorithm that are characteristic to non-distributed algorithms?

5.3. Give a Promela or Uppaal model describing your algorithm from 5.1. Explain the data structures you use.

5.4. What are Temporal Logic properties that prove the correctness of your model from 5.3. ?

15 marks