dConc Aflevering 6 - DA6

Christian Zhuang-Qing Nielsen

201504624, christian@czn.dk

December 17, 2016

MULTIPLE CHOICE

1: According to Amdahl's law, what is the speedup of a computation scheduled on 10 processors rather than one processor, if 75% of the computation can be executed in parallel?

At least 3 but less than 5.

2: What is the meaning of the phrase MyNiceLock has 0-bounded waiting, where MyNiceLock is an imple- mentation of a Lock?

MyNiceLock is first-come-first-served

3: Consider the following suggested implementation of a lock for two threads, written in Promela. Which properties does it satisfy?

mutual exclusion but not freedom from starvation

4: Which of the following is true?

The Bakery lock is first-come-first-served but the Filter lock is not.

5: How can we formalize in LTL the statement "the value of the variable v will eventually stabilize at 100"?

$FG(v == 100)$

6: Consider the following four LTL formulae:
$f_1 : FGFp$
$f_2 : GFGp$
$g_1 : FGp$
$g_2 : GFp$
Which of the following statements is true?

$f_1 \not\equiv g_1$ *and* $f_2 \not\equiv g_2$

7: Consider the following statement in a Promela program. What is its effect, if neither g1, g2 nor g3 are true?

the statement is blocked until at least one of the guards evaluates to true

8: Consider the following Promela program. Which of the following two statements are true?

Both A and B are true

9: Consider the following Promela program. What is the set of possible values for the variable n when the program has terminated?

1,0,1

10: Consider the transition system and the LTL formulae. Which of the LTL formulae are true statements about the initial (bottom left) state?

g is true but not f

11: Which -language does the following Büchi automaton over the alphabet {0,1,2} recognize?

The language of -words so that for all occurrences of the letter "1", there is a subsequenct occurrence of the letter "2".

12: A semaphore is an object with two fields. What are the types of the two fields?

boolean and integer

13: What happens when a thread succeeds in incrementing a semaphore?

If other threads are blocked at the semaphore, one of them is unblocked

14: Consider the following pseudocode of the same style as in "The little book of semaphores", implementing a special kind of lock using a semaphore.

In does not satisfy mutual exclusion. Instead, it allows three threads (but not more) to hold the lock at the same time

15: How many shared variables does the Test-and-set lock use?

1

16: The Test-and-set lock has a disadvantage compared to some other locks we have seen. Which one?

It does not guarantee freedom from starvation

17: What is a thread pool?

The current set of enabled threads on a uniprocessor system

18: Consider the following DAG of tasks, with each task requiring one unit of time to be executed on a single processor, and the arcs indicating dependencies among tasks. What is the difference between the worst and the best possible execution time (measured in the same unit) if the tasks are scheduled greedily to two processors?

0

19: What is optimistic synchronization?

A technique for avoiding to lock a concurrent object in certain situations.

20: What is the definition of a wait-free method?

A method that executes a bounded number of steps, independent of the number of other threads

Regular

21: Sketch a proof that it is not possible to make a correct implementation of a lock for two threads satisfying mutual exclusion and freedom from deadlock using only one shared read-write variable.

For to tråde A og B, så kan de hvis de kun har én fælles variabel deadlocke hvis:

$write_A(flag[A] = true)$ og $write_B(flag[B] = true)$

sker før at

$read_A(flag[B])$ og $read_B(flag[A])$.

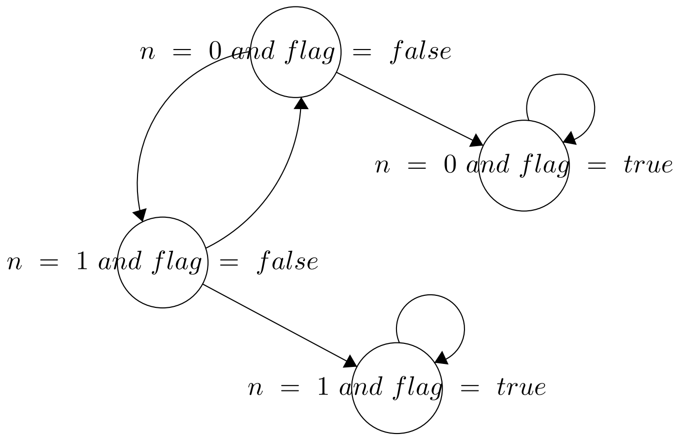
I det tilfælde hvor begge tråde sætter deres eget flag til true før de læser tilstanden af den anden tråds flag vil trådene ende med at vente på hinanden for evigt, og vi står tilbage med en deadlock.

22: Consider the following Promela program.

Explain why the program terminates in all weakly fair executions.

Et system er weakly fair hvis at en given tråd på et tidspunkt altid for lov til at forsætte. I dette tilfælde vil tråden q sætte *flag = true;*, således at programmet terminerer.

Draw the transition system corresponding to the program, as a directed graph (draw states as nodes and transitions as arcs).


$n = 0$ and $flag = false$
$n = 0$ and $flag = true$
$n = 1$ and $flag = false$
$n = 1$ and $flag = true$

In the directed graph, indicate the path corresponding to a non-terminating (but not weakly fair) execution.

Kan ses på figur 1 på side 5.

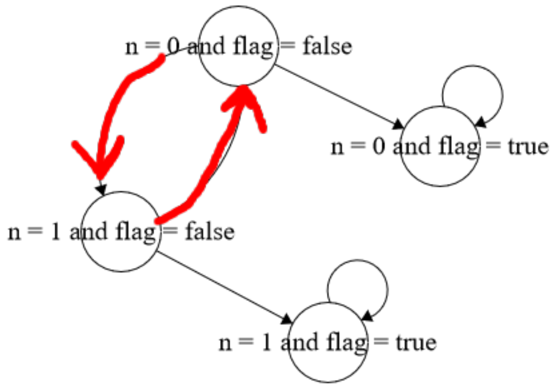
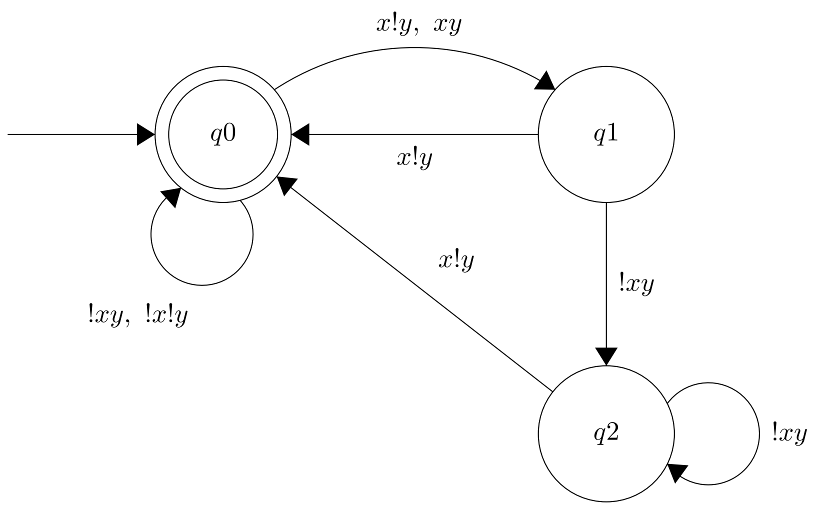


Figure 1: Det tilfælde hvor programmet sidder fast i else-løkken af p.

23: Construct a Büchi automaton (draw the automaton) corresponding to the LTL formula: $x \rightarrow X(yUx)$. That is, the automaton should accept exactly the trails of those paths that satisfy the formula.


$$\begin{tikzcd}
\arrow[r] & q0 \arrow[rr, bend left, "x!y,\ xy"] \arrow[loop below, "!xy,\ !x!y"] & & q1 \arrow[ll, "x!y"'] \arrow[d, "!xy"] \\
& & & q2 \arrow[ull, "x!y"] \arrow[loop right, "!xy"]
\end{tikzcd}$$

24: Sketch, with pseudocode, a construction of a non-reusable barrier for n threads, using semaphores. The threads should lock themselves at the barrier until all n threads have arrived there.

```
Semaphore mutex = Semaphore(1);
Semaphore barrier = Semaphore(0);
int count = 0;

...

mutex.wait()
count += 1
mutex.signal()

if count == n
    barrier.signal()

barrier.wait()
barrier.signal()

...
```

25: Explain why the test-and-test-and-set lock is faster than the test-and-set lock on a bus-based architecture when many threads are in contention over the lock.

TAS bruger enormt meget bus-trafik hver gang der bliver ændret i shared data, hvorimod TATAS kun bruger en (stor) mængde bus-trafik når den CPU releaser låsen (hvor de alle så kæmper om den).