Midterm 2010 : CS 4271: Critical Systems and their Verification

1 hour 15 minutes

Instructions to Candidates

- Answer **ALL** questions.
- Answers must be written in the space provided in this booklet; **otherwise they will not be graded**.
- All answers **MUST** come with the correct explanations. There is no credit for guessing. A correct answer without the correct explanation will receive no marks.
- This is an **OPEN BOOK** examination. You are allowed to bring in any books/lecture notes etc.
- You can ask for extra sheets for rough work.
- **PLEASE WRITE YOUR MATRICULATION NUMBER BELOW**.

MATRICULATION NO.:

(This portion is reserved for the examiner's use only)

Question		Marks
Question A	3	
Question B	4	
Question C	3	
Question D	3	
Question E	3	
Question F	2	
Question G	2	
TOTAL	20	

A. 3 marks

Are the following Linear Time Temporal Logic (LTL) formulae equivalent? If yes, give a proof. If not, construct examples to show that they are not equivalent.

$\neg$ $\mathbf{FG}p$ and $\mathbf{G}(\neg p \vee \mathbf{XF}\neg p)$

You can assume that p is an atomic proposition.

Answer: The two formulae are equivalent. Since Fp is satisfied by a trace π iff (a) either the first state of π satisfies p, or (b) one of the second or later states of π satisfies p this results in the following equivalence

$$\mathbf{F}p = p \vee \mathbf{XF}p$$

Now, $\neg\mathbf{FG}p = \mathbf{GF}\neg p$. Using the above equivalence of $\mathbf{F}$ p we get

$\neg\mathbf{FG}p = \mathbf{GF}\neg p = \mathbf{G}(\neg p \vee \mathbf{XF}\neg p)$.

B. 4 marks Consider the following program with two processes, which are composed *asynchronously*. Assume that initially `x == y == 0`.

```
x = 1          a = x
y = 1          b = y
```

What are the possible values of `a` and `b` when the program terminates? For each of these possible values draw a trace that will generate it.

Answer:

1. a == b == 0

```
               a = x
               b = y
   x = 1
   y = 1
```

2. a == b == 1

```
   x = 1
   y = 1
               a = x
               b = y
```

3. a == 0, b == 1

```
               a = x
   x = 1
   y = 1
               b = y
```

4. a == 1, b == 0

```
   x = 1
               a = x
               b = y
   y = 1
```

C. 3 marks

Are the two following Linear time Temporal Logic (LTL) formula equivalent ? If yes, give a proof. If not, construct example traces to show that they are not equivalent. You can assume that φ is an arbitrary LTL formula.

$G(\varphi \Rightarrow X\varphi)$

$G(\varphi \Rightarrow G\varphi)$

Answer: Consider a path π satisfying $\mathbf{G}(\varphi \Rightarrow \mathbf{X}\varphi)$. Let k be an index ≥ 0 such that $\pi^k \models \varphi$ where π^k denotes the suffix of π starting from position k. Then, clearly $\pi^{k+1} \models \varphi$. Again this means $\pi^{k+2} \models \varphi$. A simple induction on i is able to establish that for any natural number i we must have $\pi^{k+i} \models \varphi$. This means $\pi \models G(\varphi \Rightarrow G\varphi)$.

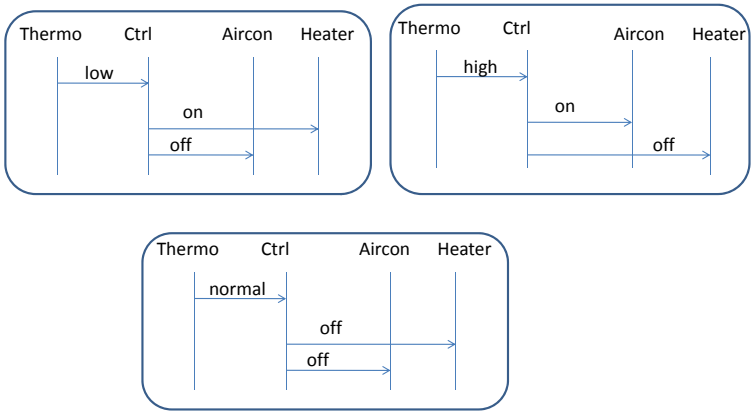
The proof in the other direction is trivial and follows from the definition of G and X operators. If a path satisfies $\mathbf{G}(\varphi \Rightarrow \mathbf{G}\varphi)$ — let k be an index ≥ 0 such that $\pi^k \models \varphi$ where π^k denotes the suffix of π starting from position k. Then, clearly $\pi^{k+1} \models \varphi$. Thus, $\pi \models G(\varphi \Rightarrow X\varphi)$.

D. 3 marks

Consider a system consisting of temperature controller, a thermostat, an air-conditioning unit and a heater unit. When the controller receives temperature-high signal from the thermostat, it sends an on signal to the air-conditioning unit, and an off signal to the heater unit. When the controller receives temperature-low signal from the thermostat, it sends an on signal to the heater unit, and an off signal to the air-conditioning unit. If the controller receives a normal signal from the thermostat, it turns off both units.

Construct multiple Sequence Diagrams showing sample behaviors of the above-mentioned reactive system. You can only get full credit if your collection of Sequence Diagrams is detailed enough to cover as much of the above requirements as possible.

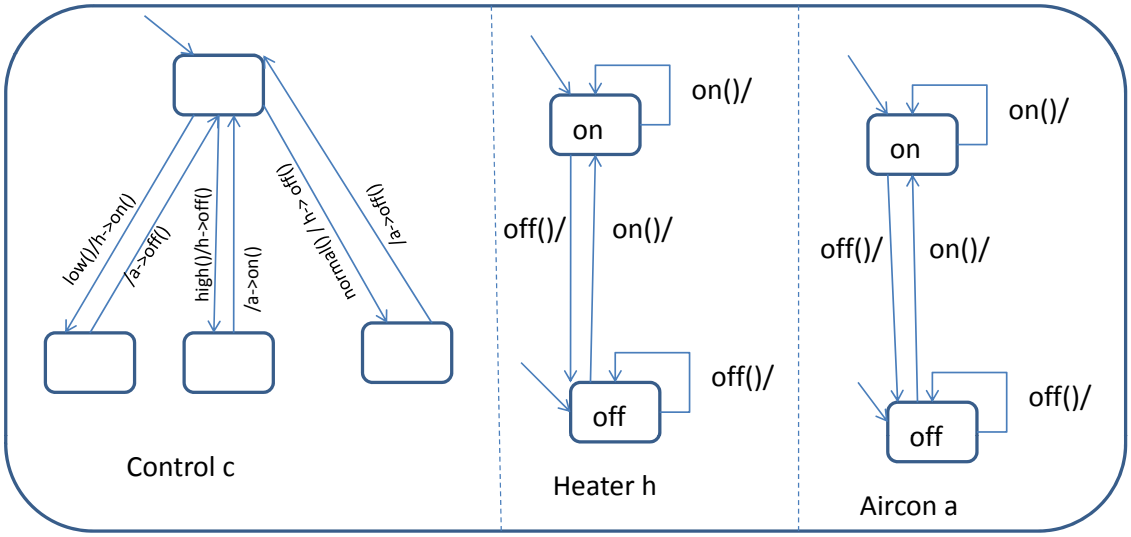
Answer:



E. 3 marks

Construct the overall behavior of the system in Question D as one single UML State Diagram. The thermostat can be treated as external environment. All other components are considered to be part of the "system".

Answer:

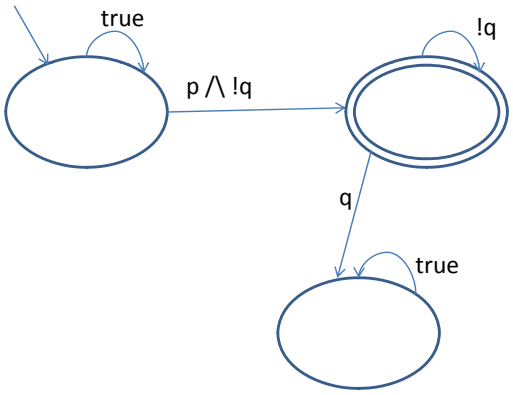


F. 2 marks

Suppose we want to verify the LTL formula $G(p \Rightarrow Fq)$, for a concurrent system Sys where p, q are atomic propositions. As per the LTL model checking algorithm discussed in class, a property automata will be synchronously composed with the state machine of Sys. What is the property automata that will be synchronously composed with the state machine of Sys in this case?

Answer:

The negation of the property is $\neg(G(p \Rightarrow Fq)$, that is, $\neg G(\neg p \vee Fq)$, that is, $F(\neg(\neg p \vee Fq))$, that is, $F(p \wedge \neg Fq)$, that is, $F(p \wedge G(\neg q))$.



G. 2 marks

```
int x = 0;

while(1){
    x = x + 1;
}
```

Can you use model checking to prove the LTL property $F(\texttt{x} == \texttt{649})$. Why or why not?

Answer: The conventional answer is "no" - because model checking is restricted to finite state systems. Thus, if we try to construct the state space of the above program it will not terminate since the value of x comes from an unbounded domain (integer) and also x does assume unboundedly many values.

One can try to argue (with quite a stretch) that the answer is "yes". This is the case, if we do *not* try to construct the state space prior to traversal - but rather construct and traverse it on-the-fly. In this case, we can have a bounded representation of x (say 32 bits) – and construct the state space *during* its traversal. In this case, we will encounter the value x == 649 as we are constructing/traversing the state space. However, we still need to reason explicitly that this program has only one execution trace and that is why when we encounter x == 649 in that execution trace, our proof is complete. This piece of reasoning is, strictly speaking, not being done inside the model checking search which is only trying to search for counter-examples rather than proofs.

END OF PAPER