HAND IN EXAM
Answers recorded on examination paper

Queen's University
Faculty of Arts and Science
School of Computing
CISC422
Final Examination
Friday, Dec 7, 2012
Instructor: J. Dingel

Student id (clearly printed please): _____

Instructions: This examination is **three hours** in length. You are permitted to have one 8.5"x11" data sheet with information on both sides. No other references, calculators, or aids are allowed.

Read all questions carefully. There are 5 questions in this examination, some with several parts. You must answer all questions. Write all your answers in this exam. An extra page is provided at the end for rough work. If you answer a question on a different page, you must indicate where the answer is to be found.

Important notes: Make sure your student id is clearly printed on this page. If the instructor is unavailable in the examination room and doubt exists as to the interpretation of any question, you are urged to submit your answer with a clear statement of any assumptions made.

Good luck!

For marking use only:

Question	Points
1	/16
2	/24
3	/18
4	/12
5	/40
Total	/110

Id: _____

Question 1: Propositional & Predicate Logic, 16 points

Consider the following informal statements:

1. *"For all t1, t2, t3, if t1 is a subtype of t2 and t2 is a subtype of t3, then t1 also is a subtype of t3"*
2. *"Float is a subtype of Number"*
3. *"Boolean is not a subtype of Number"*

a) (6 points) Using the function symbols F, B, and N (all with arity zero) to represent the mentioned types and the predicate symbol st (with arity 2) to represent the subtype relationship, formalize the above statements in predicate logic.

b) (10 points) Using your formalization, formally prove that the following statement follows from the above statements:

"Boolean is not a subtype of Float"

Only use the proof rules given in the courseware notes. Remember to include justifications for each proof step.

Id: _____

Question 2: Alloy, 24 points

Consider the following partial Alloy object model for finite state machines (FSMs).

```
sig State {}                                    // set of states
sig AP {}                                       // set of atomic propositions
sig FSM {
        states : set State                      // the set of reachable states of the FSM
        initial : set State                     // the set of initial states of the FSM
        next : State -> State                   // the transition relation of the FSM
        labels : State -> AP                    // the labelling function of the FSM
}
```

a) (4 points) Write down an Alloy fact `States` that ensures that for all FSMs, the attribute `states` contains all states that are reachable in the FSM from the initial states through the `next` relation.

b) (4 points) Write down an Alloy fact `Initial` that ensures that for all FSMs, the attribute `initial` contains a subset of the reachable states.

c) (4 points) Write down an Alloy fact `Total` that ensures that for all FSMs, the transition relation of the FSM is total.

Id: _____

Question 2 (continued)

For your convenience, here is a copy of the Alloy model from the previous page.

```
sig State {}                                    // set of states
sig AP {}                                       // set of atomic propositions
sig FSM {
        states : set State                      // the set of reachable states of the FSM
        initial : set State                     // the set of initial states of the FSM
        next : State -> State                   // the transition relation of the FSM
        labels : State -> AP                    // the labelling function of the FSM
}
```

d) (4 points) Write an Alloy predicate `Always[M : FSM, p : AP]` that returns true if and only if `p` holds in (i.e., is one of the labels of) every reachable state of `M`.

e) (4 points) We say that a finite state machine M *preserves an atomic proposition* p if the transition relation of M is such that whenever p holds in a state s of M, $p$also holds in all successor states of s. Write an Alloy predicate `Preserved[M : FSM, p : AP]` that returns true if and only if `M` preserves `p`.

f) (4 points) Write an Alloy assertion `Induction` that says that whenever an atomic proposition p is true in all initial states of a finite state machine M, and M preserves p, then p holds in all reachable states of M. You may use any previously defined predicates.

Id: _____

Question 3: CTL, 18 points

For each of the following three pairs of formulas φ, φ', decide if they are equivalent. If they are equivalent, write **"Yes"**; if they are not equivalent, write **"No"** and draw a Kripke structure (i.e., finite state machine) M such that one formula holds in M, but not the other. When drawing M, clearly indicate the initial state and which atomic propositions hold in which states.

1. (6 points) $\varphi = \mathit{true}$ and $\varphi' = (\mathbf{EG}\ p) \rightarrow (\mathbf{AG}\ p)$

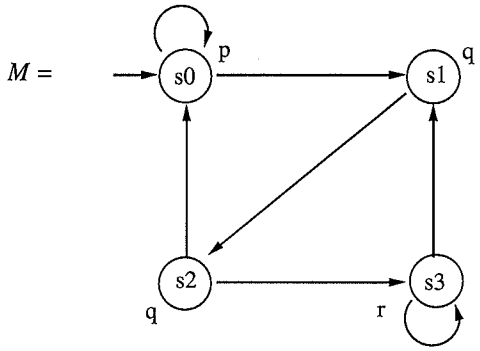
2. (6 points) $\varphi = \mathbf{AG}\ \mathbf{AG}\ p$ and $\varphi' = \mathbf{AG}\ p$

3. (6 points) $\varphi = \mathbf{A}[p\ \mathbf{U}\ q] \vee \mathbf{A}[r\ \mathbf{U}\ q]$ and $\varphi' = \mathbf{A}[(p \vee r)\ \mathbf{U}\ q]$

Id: _____

Question 4: Model checking, 12 points

Consider the following graphical representation of a Kripke structure (i.e., finite state machine) M.



For each of the following four CTL formulas φ decide whether the formula holds in machine M defined above. If your answer is **"No"**, that is, φ does not hold in M, then give a counter example, that is, an execution path in M illustrating the **violation** of φ. Remember that paths are always infinite. Use periods "..." and parentheses to indicate that a sequence of states is repeated infinitely often.

a) (3 points) $\mathbf{AG}\ (q \vee (\neg r) \vee \mathbf{AX}\ q)$

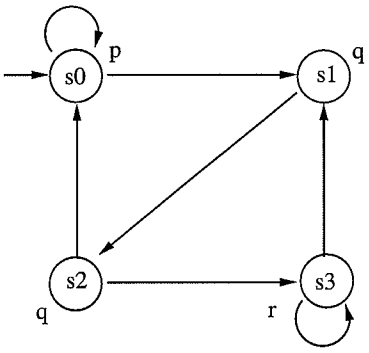
b) (3 points) $\mathbf{AG}\ ((\mathbf{AX}\ q) \rightarrow (\mathbf{AF}\ q))$

Id: _____

Question 4 (continued)

The Kripke structure M from the previous page is repeated here for your convenience.

$M =$



c) (3 points) $\mathbf{AG}\ ((\mathbf{EG}\ r) \rightarrow (\mathbf{AX}\ q))$

d) (3 points) $\mathbf{AX}\ \mathbf{AF}\ r$

Id: _____

Question 5: SMV, 40 points

Consider the following code defining an NuSMV program `main`:

```
MODULE main
VAR
  x :  -100..100;
  p0 :  process Proc(x);
  p1 :  process Proc(x);
ASSIGN
  init(x) := 0;
  -- definition of Proc(x)
  -- given on right
```

```
MODULE Proc(x)
VAR
  t :  -100..100;
  i :  1..10;
  pc :  {1, 2, 3, 4, 5};
ASSIGN
  init(t) := 0;
  init(i) := 1;
  init(pc) := 1;
  -- ASSIGN section
  -- continued on right
```

```
next(t) := case
              pc=1 :  x;
              TRUE : t;
           esac;
next(x) := case
              pc=2 :  t+1;
              TRUE : x;
           esac;
next(i) := case
              pc=3 :  i+1;
              TRUE : i;
           esac;
next(pc) := case
              pc=1 :  2;
              pc=2 :  3;
              pc=3 :  4;
              pc=4 & i<=5 :  1;
              pc=4 & i>5 :  5;
              TRUE : pc;
           esac;
FAIRNESS running
```

a) (4 points) Briefly explain what program `main` is doing. Complement your explanation with pseudocode using a C- or Java-like language.

Id: _____

Question 5 (continued)

```
MODULE main
VAR
  x :  -100..100;
  p0 :  process Proc(x);
  p1 :  process Proc(x);
ASSIGN
  init(x) := 0;
  -- definition of Proc(x)
  -- is given on right
```

```
MODULE Proc(x)
VAR
  t :  -100..100;
  i :  1..10;
  pc :  {1, 2, 3, 4, 5};
ASSIGN
  init(t) := 0;
  init(i) := 1;
  init(pc) := 1;
  -- ASSIGN section is
  -- continued on right
```

```
next(t) := case
              pc=1 :  x;
              TRUE : t;
           esac;
next(x) := case
              pc=2 :  t+1;
              TRUE : x;
           esac;
next(i) := case
              pc=3 :  i+1;
              TRUE : i;
           esac;
next(pc) := case
              pc=1 :  2;
              pc=2 :  3;
              pc=3 :  4;
              pc=4 & i<=5 :  1;
              pc=4 & i>5 :  5;
              TRUE : pc;
           esac;
FAIRNESS running
```

b) (4 points) Let M be the Kripke structure (i.e., Finite State Machine) defined by program `main` (repeated above for your convenience). We represent a single state s of M by a 7-tuple $(x, t_0, i_0, pc_0, t_1, i_1, pc_1)$ where, e.g., x is the value of variable `x`, t_0 is the value of variable `t` of process `p0`, and i_1 is the value of variable `i` of process `p1`, etc. For instance, the initial state of M is represented as $(0, 0, 1, 1, 0, 1, 1)$.

Beginning from the initial state of the machine M, draw the computation tree T of M to a depth of three, i.e., in your tree, there should be three edges on every path from the root to a leaf.

Id: _____

Question 5 (continued)

We say that process `p0` has terminated iff `p0.pc=5`. Similarly for `p1`. Also, we say that program `main` has terminated iff process `p0` and `p1` have terminated.

c) (3 points) Program `main` always eventually terminates. Write down a CTL formula that could be used by NuSMV to verify this fact.

d) (3 points) Once program `main` has terminated, it always remains terminated forever, i.e., it is impossible for it to become "not terminated". Write down a CTL formula that could be used by NuSMV to verify this fact.

e) (2 points) Due to the non-determinism in program `main`, the value of shared variable `x` upon termination is not unique, i.e., different executions of `main` may leave `x` with different values when termination is reached. What is the *largest value* that shared variable `x` can have upon termination of `main`?

f) (4 points) Describe how NuSMV could be used to verify that the number you have written in response to the previous question is indeed the largest value that variable `x` can have at termination of `main`?

g) (4 points) Perhaps surprisingly, the *smallest value* that variable `x` can have at termination is 2. Describe how NuSMV could be used to obtain an execution trace in which program `main` terminates in a state in which `x` has value 2.

Id: _____

Question 5 (continued)

h) (16 points) For each of the following properties φ_1 through φ_5, express it formally in CTL and decide whether or not it is satisfied by program `main` given in Part a) of this question by writing **"Yes"** or **"No"**. No justification necessary.

1. (4 points) φ_1: *"For all states s along every path, it is always the case that if* `pc` *of* `p0` *is equal to 4 in s, then* `pc` *will be equal to 1, 4, or 5 in all immediate successors of s".*

   - φ_1 in CTL:

   - φ_1 true in `main`?:

2. (4 points) φ_2: *"For all states s along every path, if* `x` *is equal to 5 in s, then* `x` *will always be greater or equal to 5 in all future states".*

   - φ_2 in CTL:

   - φ_2 true in `main`?:

3. (4 points) φ_3: *"Along every path it is always the case that if* `pc` *is equal to 2 in process* `p0`*, then* `t` *is equal to* `x` *in* `p0`*."*

   - φ_3 in CTL:

   - φ_3 true in `main`?:

4. (4 points) φ_4: *"There exists a path along which process* `p0` *terminates eventually and process* `p1` *has not yet terminated until then (i.e., there exists a path along which* `p0` *terminates before* `p1` *does)."*

   - φ_4 in CTL:

   - φ_4 true in `main`?:

Id: _____

Scratch sheet: