HAND IN EXAM
Answers recorded on examination paper

Queen's University
Faculty of Arts and Science
School of Computing

CISC422 Final Examination

Monday, April 16, 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	/12
2	/41
3	/18
4	/16
5	/38
Total	/125

Id: ________________________

Question 1: (Predicate Logic, 12 points)

Consider the following two predicate logic formulas

$$\begin{aligned}\varphi_1 &= \forall x.\big(P(x) \to \neg\exists y.Q(f(x,y))\big)\\ \varphi_2 &= \forall x.\neg\exists y.\big(P(x) \to Q(f(x,y))\big)\end{aligned}$$

where x and y are variables, P and Q are predicate symbols of arity 1, and f is a function symbol of arity 2.

φ_1 and φ_2 are not equivalent. More precisely, φ_1 does not imply φ_2. Find a model $\mathcal{M}$, such φ_1 holds in $\mathcal{M}$, but φ_2 does not hold in $\mathcal{M}$. Make sure you provide a **complete** definition of your model $\mathcal{M}$.

Id: ____________________

Question 2: (Alloy, 41 points)

Consider the following Alloy specification of aspects of the game Curling.

```
module Curling
   sig Team {
     stones :  set Stone,
     opponent :  Team
   }
   sig Winner extends Team {}
   sig Stone {
     // stones closer to the centre
     closerThan :  set Stone
   }
   // some stones are in the "house"
   sig HouseStone extends Stone {}
   // some stones in the house are "points"
   sig Point extends HouseStone {}
   // helper function for readability
   fun closer[s1,s2 :  Stone] {
     s2 in s1.closerThan
   }
```

```
fact CurlingFacts1 {
  // F1:  ?
  some t1 :  Team | some t2 :  (Team-t1) |
                                   no (Team-t1-t2)

  // F2:  No team is its own opponent
  all t :  Team | t != t.opponent

  // F3:  A stone belongs to exactly one team
  all s :  Stone | one t :  Team | s in t.stones

  // F4:  A stone s is not closer than s
  all s :  Stone | !closer[s,s]

  // F5:  ?
  all s1,s2 :  Stone | closer[s1,s2] & closer[s2,s1]
                                              => s1=s2
}
```

1. (10 points) In the space below, draw the Alloy class diagram (a.k.a., meta model) corresponding to the Alloy specification above. Make sure your diagram contains all signatures, relations, and multiplicity constraints enforced by the specification. Remember that not necessarily all constraints expressed in a textual Alloy specification can be also be expressed in an Alloy class diagram.

2. (6 points) Briefly and informally, but nonetheless precisely, explain the properties enforced by the facts `F1` and `F5`.

Question 2 (Alloy, continued)

```
module Curling
   sig Team {
     stones : set Stone,
     opponent : Team
   }
   sig Winner extends Team {}
   sig Stone {
     // stones closer to the centre
     closerThan : set Stone
   }
   // some stones are in the "house"
   sig HouseStone extends Stone {}
   // some stones in the house are "points"
   sig Point extends HouseStone {}
   // helper function for readability
   fun closer[s1,s2 : Stone] {
     s2 in s1.closerThan
   }
```

```
fact CurlingFacts1 {
  // F1: ?
  some t1 : Team | some t2 : (Team-t1) |
                              no (Team-t1-t2)

  // F2: No team is its own opponent
  all t : Team | t != t.opponent

  // F3: A stone belongs to exactly one team
  all s : Stone | one t : Team | s in t.stones

  // F4: A stone s is not closer than s
  all s : Stone | !closer[s,s]

  // F5: ?
  all s1,s2 : Stone | closer[s1,s2] & closer[s2,s1]
                                            => s1=s2
}
```

3. (5 points) Complete the fact below such that it expresses the statement: *"Given a pair of different stones, one is closer than the other"*.

   `fact Q1.3 {`

4. (5 points) Complete the fact below such that it expresses the statement: *"If a stone s1 is closer than s2, and s2 is in the House, then s1 is in the House, too"*.

   `fact Q1.4 {`

5. (5 points) Complete the function below such that for each team `t` it returns the stone of team `t` (if any) that is closer to the button than all other stones of team `t`.

   `fun closest[t : Team] : set Stone {`

Question 2 (Alloy, continued)

```
module Curling
   sig Team {
     stones : set Stone,
     opponent : Team
   }
   sig Winner extends Team {}
   sig Stone {
     // stones closer to the centre
     closerThan : set Stone
   }
   // some stones are in the "house"
   sig HouseStone extends Stone {}
   // some stones in the house are "points"
   sig Point extends HouseStone {}
   // helper function for readability
   fun closer[s1,s2 : Stone] {
     s2 in s1.closerThan
   }
```

```
fact CurlingFacts1 {
  // F1: ?
  some t1 : Team | some t2 : (Team-t1) |
                               no (Team-t1-t2)

  // F2: No team is its own opponent
  all t : Team | t != t.opponent

  // F3: A stone belongs to exactly one team
  all s : Stone | one t : Team | s in t.stones

  // F4: A stone s is not closer than s
  all s : Stone | !closer[s,s]

  // F5: ?
  all s1,s2 : Stone | closer[s1,s2] & closer[s2,s1]
                                            => s1=s2
}
```

6. (5 points) Using the function from the previous question, complete the fact below such that it expresses: *"Any stone hs in the house is a point iff there exists a team t to which hs belongs and hs is closer than the closest stone of t's opponent"*.

   `fact Q2.6 {`

7. (5 points) Complete the fact below such that it expresses: *"A team is a winner iff there exists at least one point that belongs to it"*.

   `fact Q2.7 {`

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 pts) $\varphi \;=\; (\mathbf{EX}\; p) \wedge (\mathbf{EX}\; q)$ and $\varphi' \;=\; (\mathbf{EX}\; p \wedge q)$

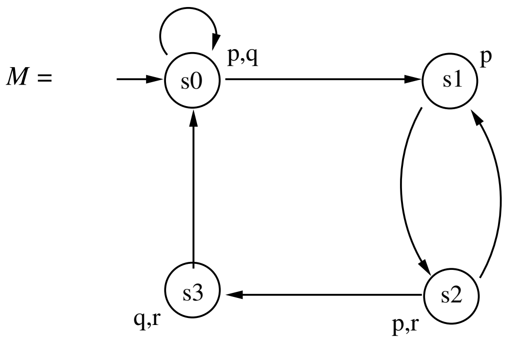
2. (6 pts) $\varphi \;=\; (\mathbf{AF}\; p) \vee (\mathbf{AF}\; q)$ and $\varphi' \;=\; (\mathbf{AF}\; p \vee q)$

3. (6 pts) $\varphi \;=\; \mathbf{E}[p\; \mathbf{U}\; q] \wedge \mathbf{E}[p\; \mathbf{U}\; r]$ and $\varphi' \;=\; \mathbf{E}[p\; \mathbf{U}\; q \wedge r]$

Id: ______________________

Question 4: (Model checking, 16 points)

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



1. (4 pts) Define M textually as a 4-tuple $M = (S, S_0, R, L)$.

2. (4 pts) Beginning from state s_0, unwind M into its corresponding computation tree T. Draw T to a depth of 4. More precisely, you must show all computation paths of M starting at s_0 up to length 4, where the length of a path is the number of edges on it.

Id: ______________________

Question 4: (Model checking, continued)

4. (8 pts) For each of the following CTL formulas φ decide whether the formula holds in machine M defined on the previous page. 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) $\mathbf{AG}(r \rightarrow \mathbf{AF}q)$

   (b) $\mathbf{AG}\big(p \rightarrow \mathbf{AX}\ \mathbf{AX}(p \vee r)\big)$

   (c) $\mathbf{AG}\big((q \wedge \mathbf{AX}p) \rightarrow \mathbf{EG}p\big)$

   (d) $\mathbf{A}\big[r\ \mathbf{U}\ \mathbf{EG}q\big]$

Id: ______________________

Question 5: (SMV, 38 pts)

Consider the following SMV code.

```
MODULE main
VAR
  x :  {1, 2, 3};
  y :  {a, b};
  z :  boolean;
ASSIGN
  init(x) := 1;                         init(y) := a;                           init(z) := 0;
  next(x) := case                       next(y) := case                         next(z) := case
                x=1 :  {1, 2};                        (x=1 | x=3) :  a;                       y=b :  !z;
                x=2 :  3;                             1 :  b;                                 1 :  z;
                x=3 :  1;                         esac;                                    esac;
             esac;
```

1. (9 pts) Let M be the Kripke structure (i.e., Finite State Machine) defined by the SMV code above. We represent a single state s of M by a triple (x, y, z) where x is the value of variable `x`, y is the value of variable `y`, and z is the value of variable `z`. For instance, the initial state of M is represented as $(1, a, 0)$ (or, $(1, a, \mathit{false})$).

   Beginning from the initial state of the machine M, draw the computation tree T of M. Explore every path in T until you hit a state that is already contained in T; in other words, all leaf nodes in T must contain states that also occur higher up in T (i.e., in non-leaf nodes).

   Hint: Your tree T should contain at least four and not more than ten different states.

Id: ____________________

Question 5: (SMV, continued)

2. (4 pts) Using computation tree T from the previous question as a guide, draw the Kripke structure M that is defined by the SMV code above. Your drawing should clearly indicate the initial states of M, the reachable states of M, and the transition relation of M. You don't need to show the labelling function.

Id: ______________________

Question 5: (SMV, continued)

3. (25 pts) For each of the following properties φ_1 through φ_5, express it formally in CTL and decide whether or not it is true in your finite state machine M given in Part 1 of this question by writing **"Yes"** or **"No"**. No justification necessary.

   1. (5 pts) φ_1: "Along every path it is always the case that if `x` is 1 or 2, then `y` is equal to `a`".
      - φ_1 in CTL:
      - φ_1 true in M?:
   2. (5 pts) φ_2: "For all states s along every path, if `x` is equal to 1 in s, then s has a successor in which `x` is 1 and another successor in which `x` is 2".
      - φ_2 in CTL:
      - φ_2 true in M?:
   3. (5 pts) φ_3: "Along every path it is always the case that `x` is 3 if and only if `y` is equal to `b`".
      - φ_3 in CTL:
      - φ_3 true in M?:
   4. (5 pts) φ_4: "There exists a path along which `x` is always equal 1, `y` is always equal to `a`, and `z` never holds".
      - φ_4 in CTL:
      - φ_4 true in M?:
   5. (5 pts) φ_5: "Along every path and in every state, we always eventually reach a state in which `x` equal to 1 and `z` doesn't hold".
      - φ_5 in CTL:
      - φ_5 true in M?:

Id: ______________________

Scratch sheet: