HAND IN

Answers recorded on exam paper

QUEEN'S UNIVERSITY FINAL EXAMINATION

FACULTY OF ARTS AND SCIENCE

School of Computing

CISC/CMPE 422 and CISC 835 – Professor Juergen Dingel

December 18, 2016

INSTRUCTIONS TO STUDENTS:

This examination is 3 HOURS in length.

There are five questions to this examination, some with several parts.

Please answer all questions in the exam.

The following aids are allowed:
One 8.5"x11" data sheet with information on both sides.

Put your student number on all pages of all answer booklets, including the front.

GOOD LUCK!

PLEASE NOTE:

Proctors are unable to respond to queries about the interpretation of exam questions.
Do your best to answer exam questions as written.

Id: ____________________

Question 1: Predicate Logic (15 points)

Consider the following two predicate logic formulas

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

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

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

Id: ____________________

Question 2: Alloy (35 points)

1. Consider the following Alloy specification M of classes, interfaces, methods, and some of their relationships as typically found in object-oriented programming:

```
module Classes

sig Method {}
sig Interface {
  declares : set Method }
```

```
sig Class {
  implements : set Interface,
  defines : set Method,
  extends : set Class,
  inherits : set Method }
```

a) (3 points) In the space below, draw the graphical representation (i.e., meta model) of M. Make sure you include multiplicity constraints, if any.

b) (4 points) Consider the predicate

```
pred show() {
   some implements
   #Method > 1
}
```

Draw an instance of M that also satisfies all the constraints in `show`, that is, draw an instance that could be created by Alloy in response to executing the command '`run show for 3`' on M.

Id: ____________________

Question 2: Alloy (continued)

2. For your convenience, the Alloy model M from the previous page is repeated here.

```
module Classes

sig Method {}
sig Interface {
  declares : set Method }

sig Class {
   implements : set Interface,
   defines : set Method,
   extends : set Class,
   inherits : set Method }
```

Using the Alloy specification above, express each of the following invariants formally in Alloy.

a) (4 points) *"The* `extends` *relationship does not contain any cycles"*

b) (4 points) *"A class c inherits method m if and only if m is defined by one of the classes extended by c"*

c) (4 points) *"For every class and every interface implemented by the class, all methods declared in the interface are defined in the class"*

d) (4 points) *"Every class c1 extending another class c2 implements all interfaces that c2 implements"*

Id: ____________________

Question 2: Alloy (continued)

3. (12 points) Consider the Alloy specification **test** on the left and the instance satisfying all constraints in **test** produced by the Alloy analyzer on the right.

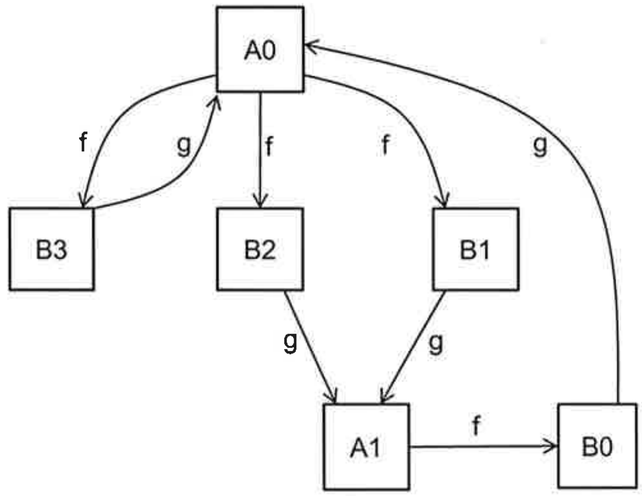
```
module test

sig A {
  f : some B }

sig B {
  g : some A }

pred show() {}

run show for 4
```



For each of the following Alloy expressions and formulas, determine which value the expression or formula evaluates to in the instance on the right and write down that value.

a) `f.g` evaluates to:

b) `some a:A | no a.f` evaluates to:

c) `f = ~g` evaluates to:

d) `some a:A | one a.f` evaluates to:

e) `{a:A | a in f.g.a}` evaluates to:

f) `(A -> B) - f` evaluates to:

Id: ______________________

Question 3: CTL (18 points)

Consider the three pairs of non-equivalent formulas φ_i, φ'_i below. For each pair, find a Kripke Structure that distinguishes them. More precisely, for each pair, draw a Kripke Structure M_i such that one formula holds in M_i, but not the other.
Important: When drawing M_i, make sure that you clearly indicate (1) the initial state of M_i, (2) which atomic propositions occuring φ_i and φ'_i hold in which states of M_i, and (3) which of the two formulas holds in M_i. Also, remember that the transition relation of a Kripke Structure is total.

a) (6 points) $\varphi_1 = \mathbf{AG}\ p$ and $\varphi'_1 = \mathbf{AG}\ \mathbf{AX}\ p$

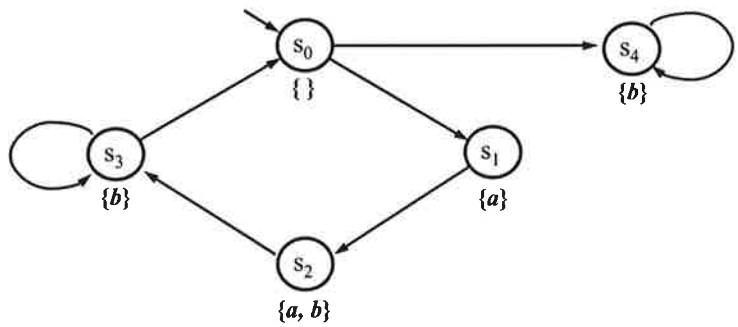
b) (6 points) $\varphi_2 = \mathbf{E}[p\ \mathbf{U}\ \mathbf{AG}\ q]$ and $\varphi'_2 = \mathbf{E}[p\ \mathbf{U}\ q]$

c) (6 points) $\varphi_3 = \mathbf{EF}\ p$ and $\varphi'_3 = \mathbf{EF}\ \mathbf{EX}\ p$

Id: ____________________

Question 4: Model checking (28 points)

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

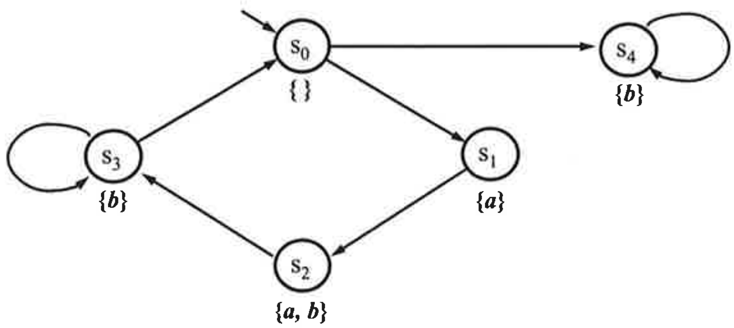


a) (4 points) Define M textually as a 4-tuple $M = (S, S_0, R, L)$. Make sure you define all components of M.

b) (4 points) 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.

Question 4: Model checking (continued)

For your convenience, the Kripke Structure M from the previous page is repeated here.



c) (16 points) In the table below, the rows are labeled with CTL formulas φ_i and the columns are labeled with states s_j of M. For each empty cell (φ_i, s_j) in the table, determine whether or not M in state s_j satisfies formula φ_i, i.e., if $(M, s_j) \models \varphi_i$ or $(M, s_j) \not\models \varphi_i$. Write *"Yes"* into cell (φ_i, s_j), if $(M, s_j) \models \varphi_i$. Write *"No"*, otherwise. For instance, $\mathbf{AX}\ (a \vee b)$ is satisfied in state s_0, but not satisfied in state s_3, i.e., $(M, s_0) \models \mathbf{AX}\ (a \vee b)$ and $(M, s_3) \not\models \mathbf{AX}\ (a \vee b)$.

	s_0	s_1	s_2	s_3	s_4
$\mathbf{AX}\ (a \vee b)$	Yes	Yes	Yes	No	
$\mathbf{A}[b\ \mathbf{U}\ a]$					
$\mathbf{AG\ EX}\ (a \rightarrow b)$					
$\mathbf{EG\ EF}\ (a \wedge b)$					

d) (4 points) Consider the formula $\varphi = \mathbf{AG}\left[(\mathbf{EX}\ (\neg a \wedge \neg b)) \rightarrow \mathbf{AX\ AX}\ \neg a\right]$. Does M satisfy this formula, i.e., does $(M, s_0) \models \varphi$ hold? If so, write *"Yes"* in the space below. If not, write *"No"* and a counterexample, i.e., an execution of M violating the formula.

Id: ____________________

Question 5: SMV (29 points)

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

```
MODULE main
VAR  x : {U, D};
     c : {0, 1, 2, 3};
     p : P(x,c);
ASSIGN
     init(x) := U; init(c) := 0;
     next(x) := case
                  c<3 : {U, D};
                  c=3 : x;
                esac;
```

```
MODULE P(x,c)
ASSIGN
   next(c) := case
                 x=U & c<3 : c+1;
                 x=D & c>0 : c-1;
                 TRUE : c;
              esac;
```

a) (9 points) Draw the finite state machine M that is defined by the code above. Represent a single state s of M by a pair (x, c) where x is the value of variable `x` and c is the value of variable `c`. For instance, the initial state of M is represented as $(U, 0)$. Your drawing should clearly indicate the initial states of M, the reachable states of M, and the transition relation of M.

Id: ____________________

Question 5: SMV (continued)

b) (20 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. A *successor* of a state s is a state that can be reached in *one* step from s.

1. (4 points) φ_1: *"There exists a path along which* `x` *is always equal to* `U`*"*
   - φ_1 in CTL:

2. (4 points) φ_2: *"Along all paths, whenever* `x` *is* `U` *and* `c` *is 1, then* `c` *is 2 is all successor states"*
   - φ_2 in CTL:

3. (4 points) φ_3: *"There exists an execution to a state s such that from s* `c` *will forever always be 3"*
   - φ_3 in CTL:

4. (4 points) φ_4: *"There exists a path along which* `c` *is 2 eventually, and* `c` *is 1 until then"*
   - φ_4 in CTL:

5. (4 points) φ_5: *"Always along all executions, whenever* `c` *is 2 in some state, then* `c` *is different from 2 in all next states"*
   - φ_5 in CTL:

Id: ______________________

Scratch sheet: