EXAM 3 September 20, 2011

This exam consists of five exercises. The available points for each item are written in the margin. You need at least 50 points to pass.

1 Consider the boolean function $f(x,y,z) = x \oplus xy \oplus yz$.

[10] (a) Give a binary decision tree for f with the variable ordering $[x,y,z]$ and use the reduce algorithm to construct an equivalent reduced OBDD.

[10] (b) Determine all minimal adequate subsets of $\{\rightarrow, \oplus, \bot, f\}$.

[4] 2 (a) Apply the unification algorithm to determine a most general unifier of `f(g(X),Y,g(Z))` and `f(Z,g(X),Y)`, if possible.

[6] (b) Write a Prolog-predicate `genvar/4` for storing variable to variable mappings. The predicate takes a variable `X` and a list of variable-pairs `Ps` as input and returns a variable and a list of variable-pairs as output:

- If `(X,Y)` is contained in `Ps` for some `Y`, then `Y` should be returned as variable and the pairs `Ps` are returned unchanged.
- Otherwise, a new variable `Z` should be created and returned, and the pair `(X,Z)` should be added somewhere in the list.

Examples:

- `?- genvar(X1,[(X2,X4),(X1,X7)],Y,Ps)` should yield the answer
  `Y = X7, Ps = [(X2,X4),(X1,X7)]`
- `?- genvar(X1,[(X2,X4),(X3,X7)],Y,Ps)` might yield the answer
  `Y = X9, Ps = [(X2,X4),(X3,X7),(X1,X9)]]`

Hint: Think carefully how to compare variables.

[10] (c) Write a predicate `copyterm/2` which behaves like the built-in predicate `copy_term/2` of Prolog: It should copy a term where all variables are renamed to fresh ones, e.g., the query `?- copyterm(f(g(X),Y,X),Z)` should yield the answer `Z = f(g(X'),Y',X')`.

Hints:

- You can assume that `genvar/4` is available, even if you did not solve the previous exercise.
- You have to use some built-in predicates like `var/1`, `=../2`, `functor/3`, or `arg/3` which can be used to analyze, disassemble and assemble terms.
- You might have to define auxiliary predicates which also take a variable to variable mapping as argument.

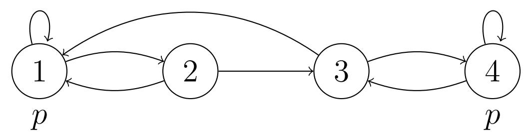
3 For each of the following sequents, either give a natural deduction proof or find a model which does not satisfy it.

[6] (a) $p \wedge \neg p, q \vdash \neg q$

[7] (b) $\exists x\, \exists y\, (P(x) \wedge \neg P(y)), \forall x\, (Q(x) \to P(x)) \vdash \exists x\, \neg Q(x)$

[7] (c) $\exists x\, \exists y\, (P(x) \wedge \neg P(y)), \forall x\, (P(x) \to Q(x)) \vdash \exists x\, \neg Q(x)$

4 Consider the CTL formula $\phi = \mathsf{A}[(p \vee \mathsf{AX}\,\mathsf{EG}\,p)\,\mathsf{U}(\neg p \vee \mathsf{AF}\,\mathsf{AX}\,p)]$ and the following model $\mathcal{M}$:



[7] (a) Apply the CTL model checking algorithm to determine the states of $\mathcal{M}$ which satisfy ϕ.

[7] (b) Provide a CTL formula ψ that is satisfied only in state 1 of $\mathcal{M}$.

[6] (c) In this part we extend CTL with the following *strict future* operator ES:

$$\mathcal{M}, s \models \mathsf{ES}\,\alpha \iff \exists \text{path } s = s_1 \to s_2 \to s_3 \to \cdots\ \exists i \geqslant 2 \text{ such that } \mathcal{M}, s_i \models \alpha$$

Extend the model checking algorithm with instructions for deciding which states should be labelled by formulas of the shape $\mathsf{ES}\,\alpha$.

[20] 5 Determine whether the following statements are true or false. Every correct answer is worth 2 points. For every wrong answer 1 point is subtracted, provided the total number of points is non-negative.

statement
Unifiable terms have a most general unifier.
Every valid sequent has a proof in which the rule modus tollens is used.
The clauses $\{\neg p, q, r, \neg s\}$ and $\{\neg p, \neg q, s\}$ have exactly two resolvents.
The algebraic normal form of the boolean function $f(x,y) = \overline{x} + y$ is $1 \oplus x \oplus xy$.
Executing the Prolog query `?- X =:= 1+2.` produces `No`.
$\exists x\, (\exists x\, \psi \to \phi) \dashv\vdash \exists x\, \psi \to \exists x\, \phi$
SWI-Prolog is free software.
If ϕ and ψ are theorems then $\phi \vdash \psi$ and $\psi \vdash \phi$ are valid.
Tseitin's transformation transforms every propositional formula into an equisatisfiable Horn formula.
A path $s_1 \to s_2 \to \cdots$ in a CTL model is fair with respect to a set $C = \{\psi\}$ consisting of a single fairness constraint if $s_i \models \psi$ for infinitely many i.