Exam Logical Verification

January 16, 2009

There are six (6) exercises.
Answers may be given in Dutch or English. Good luck!

Exercise 1. This exercise is concerned with first-order propositional logic (prop1) and simply typed λ-calculus ($\lambda\!\rightarrow$).

a. Give a proof in prop1 showing that the following formula is a tautology:

$$((A \rightarrow B \rightarrow A) \rightarrow B) \rightarrow B$$

(5 points)

b. Give the type-derivation in $\lambda\!\rightarrow$ corresponding to the proof in 1a.
(5 points)

c. Give closed inhabitants in simply typed λ-calculus of the following types:

$$(A \rightarrow A \rightarrow B) \rightarrow A \rightarrow B$$
$$(A \rightarrow B \rightarrow C) \rightarrow B \rightarrow A \rightarrow C$$
$$((B \rightarrow A \rightarrow B) \rightarrow A) \rightarrow A$$

(5 points)

Exercise 2. This exercise is concerned with first-order predicate logic (pred1) and λ-calculus with dependent types (λP).

a. Give a proof in pred1 with a $\forall$-detour showing that the following formula is a tautology:

$$\forall x.\, (P(x) \rightarrow (\forall y.\, P(y) \rightarrow A) \rightarrow A)$$

(5 points)

b. Give the λP-term corresponding to the formula in 2a.
(5 points)

c. Give a closed inhabitant in λP of the answer to 2b.
(5 points)

Exercise 3. This exercise is concerned with second-order propositional logic (prop2) and polymorphic λ-calculus ($\lambda 2$).

a. Give a proof in prop2 showing that the following formula is a tautology:

$$(\forall c. ((a \to b \to c) \to c)) \to a$$

(5 points)

b. Give the $\lambda 2$-type corresponding to the formula of 3a.

(5 points)

c. Give a closed inhabitant in $\lambda 2$ of the answer to 3b.

(5 points)

Exercise 4. This exercises is concerned with encodings.

a. The data-type of booleans is encoded in $\lambda 2$ as follows:

$$\mathsf{Bool} = \Pi a : *. \, a \to a \to a$$

Give two different closed inhabitants of Bool that can be used as encodings of *true* and *false*.

(5 points)

b. The disjunction $\mathsf{Or}\, A\, B$ is encoded in $\lambda 2$ as follows:

$$\mathsf{Or}\, A\, B = \Pi c : *. \, (A \to c) \to (B \to c) \to c$$

Assume $P : A$ and use P to give an inhabitant of $\mathsf{Or}\, A\, B$.

(5 points)

c. Assume the following setting in Coq:

```
(* prop representing the propositions is declared as a Set *)
Parameter prop : Set.

(* implication on prop is a binary operator *)
Parameter imp : prop -> prop -> prop.

(* T expresses if a proposion in prop is valid
   if (T p) is inhabited then p is valid
   if (T p) is not inhabited then p is not valid *)
Parameter T : prop -> Prop.
```

Give the type of `imp_introduction`, the variable that models the introduction rule for `imp`.

(5 points)

Exercise 5. This definition is concerned with inductive data-types in Coq.

a. Give the definition of an inductive data-type `two` with exactly two elements.

   (5 points)

b. Give the induction principle for `two_ind`, for your data-type from 5a.

   (5 points)

c. Give the definition of an inductive data-type `natpair` of pairs of natural numbers. (You can use the data-type `nat` of natural numbers.)

   (5 points)

Exercise 6. This exercise is concerned with inductive predicates in Coq.

a. Complete the following definition of conjunction:

```
Inductive and (A : Prop) (B : Prop) : Prop :=
```

   (4 points)

b. Consider the following inductive predicates:

```
Inductive ev : nat -> Prop :=
| ev0 : ev 0
| evS : forall n:nat , odd n -> ev (S n)
with odd : nat -> Prop :=
| oddS : forall n:nat , ev n -> odd (S n) .
```

   Give inhabitants of the following:

```
ev 0
odd 1
ev 2
```

   (6 points)

c. Complete the following definition of the inductive predicate `even`:

```
Inductive even : nat -> Prop :=
| even0  :
| evenSS :
```

   (5 points)

The final note is (the total amount of points plus 10) divided by 10.