Logical Verification 2016–2017
Vrije Universiteit Amsterdam
Lecturer: dr. J. C. Blanchette

Resit exam
Thursday 6 July 2017, 15:15–18:00, WN-M143
7 problems, 90 points
Answers may be given in English or Dutch

Problem 1. First-order predicate logic (3+3+3+3+3 points)

Are the following formulas tautologies of intuitionistic first-order predicate logic? Justify briefly.

a. $(\exists x.\, \mathsf{A}(x) \wedge \mathsf{B}(x)) \rightarrow (\exists x.\, \mathsf{A}(x)) \vee (\exists x.\, \mathsf{B}(x))$

Answer:

Yes. We can use the x from the hypothesis as witness on either side of the disjunction.

b. $(\exists x.\, \mathsf{A}(x)) \wedge (\exists x.\, \mathsf{B}(x)) \rightarrow (\exists x.\, \mathsf{A}(x) \wedge \mathsf{B}(x))$

Answer:

No. There are no guarantees that the same witness works for both A and B.

c. $\exists x.\, \mathsf{D}(x) \rightarrow \forall y.\, \mathsf{D}(y)$

Answer:

No. This is the drinker's paradox: It is valid classically, but not intuitionistically.

d. $(\neg\neg a \rightarrow a) \rightarrow \exists x.\, \mathsf{D}(x) \rightarrow \forall y.\, \mathsf{D}(y)$

Answer:

No. We still have the drinker's paradox (despite the spurious precondition, about some a we don't care about).

e. $\forall a.\, a \rightarrow \neg\neg a$

Answer:

No because the formula is not first-order.

Problem 2. Simply typed λ-calculus (5+5+3 points)

Consider the following formula of minimal second-order propositional logic:

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

a. Give a proof tree in natural deduction for the above formula, with each inference step annotated with the name of the inference rule.

Answer:

$$\dfrac{\dfrac{\dfrac{\dfrac{[\forall b.\,(a \to b) \to b]^{h_1}}{(a \to a) \to a}\,\mathrm{E}\forall \qquad \dfrac{[a]^{h_2}}{a \to a}\,\mathrm{I}[h_2]{\to}}{a}\,\mathrm{E}{\to}}{(\forall b.\,(a \to b) \to b) \to a}\,\mathrm{I}[h_1]{\to}}{\forall a.\,(\forall b.\,(a \to b) \to b) \to a}\,\mathrm{I}\forall$$

b. Give the polymorphic λ-term associated with the above proof tree according to the Curry–Howard–De Bruijn correspondence. Make sure to indicate the types of all bound variables.

Answer:

$\lambda a : *.\, \lambda h_1 : (\Pi b : *.\, (a \to b) \to b).\, (h_1\, a)\, (\lambda h_2 : a.\, h_2)$.

c. Give the type of the polymorphic λ-term you specified above.

Answer:

$\Pi a : *.\, (\Pi b : *.\, (a \to b) \to b) \to a$.

Problem 3. Inductive predicates (3+4+4 points)

Consider the following type of polymorphic trees:

```
Inductive tree (A : Set) : Set :=
  leaf : A -> tree A
| inner : tree A -> tree A -> tree A.
```

a. Complete the following inductive definition of a predicate `every` A P that tests whether a tree of A elements consists only of elements satisfying P:

```
Inductive every (A : Set) (P : A -> Prop) : tree A -> Prop :=
  ...
```

Answer:

```
  every_leaf : forall x, P x -> every _ P (leaf x)
| every_inner : forall l r, every _ P l -> every _ P r -> every _ P (inner _ l r).
```

b. Define an inductive predicate `is_mirrored` A t u that tests whether the tree u is a mirrored version of the tree t.

Answer:

```
Inductive is_mirrored (A : Set) : tree A -> tree A -> Prop :=
  is_mirrored_leaf : is_mirrored _ (leaf _ x) (leaf _ x)
| is_mirrored_inner : forall l1 r1 l2 r2, is_mirrored _ l1 r2 ->
    is_mirrored _ l2 r1 -> is_mirrored _ (inner _ l1 r1) (inner _ l2 r2).
```

c. Define an inductive predicate `elem` A x that tests whether a tree with labels of type A contains the element $x : A$.

Answer:

```
Inductive elem (A : Set) (x : A) : tree A -> Prop :=
  elem_leaf : elem _ x (leaf _ x)
| elem_inner : forall l r, elem _ x l \/ elem _ x r -> elem _ x (inner _ l r).
```

Problem 4. Inductive datatypes (4+2+4+4 points)

The set of *arithmetic expressions* **AE** is defined inductively as follows:

- $0 \in \mathbf{AE}$;
- $1 \in \mathbf{AE}$;
- if $e, e' \in \mathbf{AE}$, then $e + e' \in \mathbf{AE}$;
- if $e, e' \in \mathbf{AE}$, then $e \times e' \in \mathbf{AE}$.

Note that this definition introduces purely syntactic objects. Accordingly, $0+1$ is not the same expression as $1+0$ or 1.

a. Define an inductive datatype, `aexp`, of arithmetic expressions in Coq.

Answer:

```
Inductive aexp : Set :=
  Zero : aexp
| One : aexp
| Plus : aexp -> aexp -> aexp
| Mult : aexp -> aexp -> aexp.
```

b. Represent the expression $(1 + 0) \times 1$ using your Coq datatype.

Answer:

```
Mult (Plus One Zero) One
```

c. What is the induction principle associated with `aexp` (i.e., the type of `aexp_ind`)?

Answer:

```
forall P : aexp -> Prop,
  P Zero ->
  P One ->
  (forall l r, P l -> P r -> P (Plus l r)) ->
  (forall l r, P l -> P r -> P (Mult l r)) ->
  (forall a. P a)
```

d. Define a Coq function `eval` that, given an arithmetic expression, computes its value, interpreting $0, 1, +, \times$ in the usual way for natural numbers.

Answer:

```
Fixpoint eval (a : aexp) : nat :=
  match a with
    Zero -> 0
  | One -> 1
```

```
| Plus l r -> eval l + eval r
| Mult l r -> eval l * eval r
end.
```

Problem 5. Weak and strong normalization (2+3 points)

a. What is the definition of weak normalization for β-reduction?

Answer:

For all terms M, there exists a term N such that $M \to_\beta^* N$ and N is in normal form (i.e., irreducible).

b. Prove or disprove (informally but convincingly): If $M\,N$ is strongly normalizing, then M and N are strongly normalizing.

Answer:

The proof is by contraposition. Assume that there exists an infinite chain

$$M \longrightarrow_\beta M_1 \longrightarrow_\beta M_2 \longrightarrow_\beta \cdots$$

(or similarly starting from N). We can build the infinite chain

$$M\,N \longrightarrow_\beta M_1\,N \longrightarrow_\beta M_2\,N \longrightarrow_\beta \cdots$$

QED.

Problem 6. Polymorphic λ-calculus (3+3+2+2+2+2+4+4 points)

Consider the following impredicative characterizations:

$$\mathbf{N} \equiv \Pi a : *.\ a \to (a \to a) \to a$$
$$\mathbf{N}^* \equiv \Pi a : *.\ (a \to a) \to a$$

$$\mathtt{Z} \equiv \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ z$$
$$\mathtt{S} \equiv \lambda n : \mathbf{N}.\ \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ s\,(n\,a\,z\,s)$$

a. How many different inhabitants (in closed normal form) does the type $\mathbf{N}$ have? Justify briefly.

Answer:

Countably infinitely many.

b. How many different inhabitants (in closed normal form) does the type $\mathbf{N}^*$ have? Justify briefly.

Answer:

Zero.

c. What is the type of $\mathtt{Z}$?

Answer:

$\mathbf{N}$.

d. What is the type of $\mathtt{S}$?

Answer:

$\mathbf{N} \to \mathbf{N}$.

e. What is the type of $\mathbf{N}$?

Answer:

$*$.

f. What is the type of $\mathbf{N}^*$?

Answer:

$*$.

g. Give a closed term $\mathtt{plus}$ such that $\mathtt{plus}\,(\mathtt{S}^m\,\mathtt{Z})\,(\mathtt{S}^n\,\mathtt{Z}) = \mathtt{S}^{m+n}\,\mathtt{Z}$, where $\mathtt{S}^m$ denotes iterated application (e.g., $\mathtt{S}^2\,\mathtt{Z} \equiv \mathtt{S}\,(\mathtt{S}\,\mathtt{Z})$).

Answer:

$\mathtt{plus} \equiv \lambda m : \mathbf{N}.\ \lambda n : \mathbf{N}.\ \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ n\,a\,(m\,a\,z\,s)\,s.$

h. Show that $\mathtt{plus}\,(\mathtt{S}\,\mathtt{Z})\,(\mathtt{S}\,\mathtt{Z}) = \mathtt{S}\,(\mathtt{S}\,\mathtt{Z})$.

Answer:

$$
\begin{aligned}
& \texttt{plus}\,(\texttt{S}\,\texttt{Z})\,(\texttt{S}\,\texttt{Z}) \\
= {} & (\lambda m : \mathbf{N}.\ \lambda n : \mathbf{N}.\ \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ n\,a\,(m\,a\,z\,s)\,s)\,(\texttt{S}\,\texttt{Z})\,(\texttt{S}\,\texttt{Z}) \\
= {} & \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ \texttt{S}\,\texttt{Z}\,a\,(\texttt{S}\,\texttt{Z}\,a\,z\,s)\,s \\
= {} & \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ s\,(\texttt{Z}\,a\,(s\,(\texttt{Z}\,a\,z\,s))\,s) \\
= {} & \lambda a : *.\ \lambda z : a.\ \lambda s : a \to a.\ s\,(s\,z)
\end{aligned}
$$

Problem 7. Dependent types (5+5 points)

a. Which of the following type judgments belong to λP, the λ-calculus with dependent types?

$$a : * \vdash a : *$$
$$a : * \vdash \Box : *$$
$$a : * \vdash * : \Box$$
$$\mathsf{nat} : *,\ \mathsf{natlist\_dep} : \mathsf{nat} \to * \vdash (\Pi n : \mathsf{nat}.\ \mathsf{natlist\_dep}\ n) : *$$
$$\mathsf{list} : * \to *,\ \mathsf{nil} : (\Pi a : *.\ \mathsf{list}\ a) \vdash (\lambda a : *.\ \mathsf{nil}\ a) : (\Pi a : *.\ \mathsf{list}\ a)$$

Answer:

The first, third, and fourth ones. (The second one is nonsense. The fifth one is in λ2 but not in λP.)

b. Give a formal derivation of the type judgment

$$a : *,\ z : a \vdash (\lambda y : a.\ z) : a \to a$$

in λP.

Answer:

Let $\Gamma \equiv a : *,\ z : a$.

$$\dfrac{\dfrac{}{\dfrac{\Gamma \vdash * : \Box}{\Gamma \vdash a : *}} \qquad \dfrac{}{\dfrac{\Gamma,\ y : a \vdash * : \Box}{\Gamma,\ y : a \vdash z : a}}}{\Gamma \vdash (\lambda y : a.\ z) : a \to a}$$

The grade for the exam is the total amount of points divided by 10, plus 1, rounded to the nearest multiple of 0.5 (avoiding 5.5).