COMP 4161 S2/11
Advanced Topics in Software Verification

Assignment 1

This assignment starts on Thursday, 28.7.2011 and is due on Thursday, 04.8.2011, 23:59h. We will accept plain text files, PDF files, and Isabelle theory files (.thy). Submit using `give` on a CSE machine:

```
give cs4161 a1 files
```

For example:

```
give cs4161 a1 a1.thy a1.pdf
```

or

```
give cs4161 a1 a1.thy a1.txt
```

1 β Reduction and Encodings (15+10 marks)

This purpose of this question is to find β normal forms for lambda terms. You are required to justify your answer by providing the set of β reductions leading from the term to its normal form. Each step should only reduce *one* redex (i.e. one reduction per step). Ideally, you would underline the redex being reduced.

(a) Given the encoding of booleans and booleans operations in lambda calculus seen in the lecture, and recalled below:

$\texttt{true} \equiv \lambda x\ y.\ x$
$\texttt{false} \equiv \lambda x\ y.\ y$
$\texttt{if} \equiv \lambda z\ x\ y.\ z\ x\ y$
$\texttt{not} \equiv \lambda x.\ \texttt{if}\ x\ \texttt{false}\ \texttt{true}$
$\texttt{and} \equiv \lambda x\ y.\ \texttt{if}\ x\ y\ \texttt{false}$

show that the β normal form for `(not (and true false))` is `true`.

(b) Given the encoding of natural numbers in lambda calculus (Church Numerals) seen in the lecture, and recalled below:

$0 \equiv \lambda f\ x.\ x$
$1 \equiv \lambda f\ x.\ f\ x$
$2 \equiv \lambda f\ x.\ f\ (f\ x)$
$3 \equiv \lambda f\ x.\ f\ (f\ (f\ x))$
$\texttt{add} \equiv \lambda m\ n.\ \lambda f\ x.\ m\ f\ (n\ f\ x)$

show that the β normal form for (`add` 2 1) is 3.

2 Types (15+10 marks)

This question will require to construct type derivation trees. Each node of the tree should correspond to the application of a *single* typing rule. Ideally,

you would indicate which typing rule is used at each step.

(a) Construct a type derivation tree for the term $\lambda a\ b\ c.\ (a\ b)\ b\ (c\ b)$

(b) If $\Gamma = [IF \leftarrow (bool \Rightarrow \alpha \Rightarrow \alpha \Rightarrow \alpha); a \leftarrow \tau_1; b \leftarrow \tau_2; c \leftarrow \tau_3]$, then under which condition on τ_1, τ_2 and τ_3, is $\Gamma \vdash IF\ a\ b\ c$ well typed? And of which type is it?

3 Unification (10 marks)

Find a unifier (substitution) such that $\lambda\, x\ y\ z.\ ?F\ (x\ y\ z)\ = \lambda\, x\ y\ z.\ x\ (?F\ y)\ z$. Explain your answer.

4 Proofs in Propositional Logic (40 marks)

Prove the following lemmas in Isabelle. You may use only the rules `notI`, `notE`, `conjI`, `conjE`, `disjI1`, `disjI2`, `disjCI`, `disjE`, `impI`, `impE`, `iffI`, `iffE`, and `classical` in single step rule applications with the proof methods `rule`, `erule`, `assumption`, and `case_tac`.
(Recall that the command `thm` *`thmName`* causes Isabelle to print the rule *`thmName`*. This might be useful when deciding which rules to apply to complete the following proofs.)

(a) $(A \wedge B) \longrightarrow (\neg(C \wedge \neg A)) \wedge (C \longrightarrow B)$ (6 marks)

(b) $(\neg(a \longrightarrow b)) = (a \wedge \neg b)$ (6 marks)

(c) $[\![(\neg a) = b]\!] \Longrightarrow (a = (\neg b))$ (6 marks)

(d) $(((f \longrightarrow g) \longrightarrow (h \longrightarrow f)) \longrightarrow g) = ((f \longrightarrow g) \wedge (g \vee h))$ (10 marks)

(e) $(a \longrightarrow b) = (\neg(a \wedge \neg b))$ (6 marks)

(f) $(a \longrightarrow b) = (\neg(a \wedge \neg b))$ again, but only using the lemmas 4(a) to 4(d). (6 marks)

For this question, please submit an Isabelle theory file that contains all required lemmas and is processed without errors by Isabelle 2011. You can (but do not have to) use the template file provided on the lecture home page. End partial/incomplete proofs with the command `sorry` (to be able to use them in the last lemma if needed).