UPPSALA UNIVERSITET   LABORATORY EXERCISE, part 1
Matematiska institutionen   Tillämpad logik, ht-08
Erik Palmgren   2007-11-17

Laboratory exercises, part 1

The first part of the laboratory exercises aims to get you acquainted with the proof assistant
Coq and to make some simple proofs with its help. You may work in groups of up to three
persons. Hand in finished and annotated proofs at the latest December 4, 2008.

Coq is freely available for different computer platforms. See the links on the course web
page which points to the download pages for Coq and tutorials on Coq. The tutorial *The
Coq Proof Assistant - a Tutorial* (October 2008) by Gérard Huet, Gilles Kahn and Christine
Paulin-Mohring is recommended reading. Consult also the online reference manual.

The Coq system is started under Linux or Mac OS X by typing `coqtop` in a terminal win-
dow. Below is an example where we prove $\neg(\exists x \in D)P(x) \to (\forall x \in D)\neg P(x)$ in intuitionistic
predicate calculus. First P is declared to be a predicate over D, or a *propositional function*
on D. Then the formula we want to prove is entered, the *goal formula*.

```
$ coqtop
Welcome to Coq 8.1pl2 (Oct. 2007)

Coq < Section Predicate_calculus.
Coq < Variable D:Set.
D is assumed

Coq < Variable P:D -> Prop.
P is assumed

Coq < Goal   ~ (exists x:D, P(x)) -> (forall x:D,  ~P(x)).
1 subgoal

  D : Set
  P : D -> Prop
  ============================
   ~ (exists x : D, P x) -> forall x : D, ~ P x
```

One of the general tactics is `intros` which can apply several logical introduction rules in
succession. Below we do $\to$-introduction, $\forall$-introduction and $\neg$-introduction in one sweep.
The variables `H d H2` are requested names for the resulting assumptions.

```
Unnamed_thm < intros H d H2.
```

```
1 subgoal

  D : Set
  P : D -> Prop
  H : ~ (exists x : D, P x)
  d : D
  H2 : P d
  ============================
   False
```

The goal is now to prove a contradiction, which can be done by applying the hypothesis *H*. This in turn gives a goal which is an existential statement. The element of *D* which we are looking for is probably *d* so we make an existential introduction: `exists d.` The new goal can then be proved by referring to the assumption.

```
Unnamed_thm < apply H.
1 subgoal

  D : Set
  P : D -> Prop
  H : ~ (exists x : D, P x)
  d : D
  H2 : P d
  ============================
   exists x : D, P x

Unnamed_thm < exists d.
1 subgoal

  D : Set
  P : D -> Prop
  H : ~ (exists x : D, P x)
  d : D
  H2 : P d
  ============================
   P d

Unnamed_thm < assumption.
Proof completed.
```

The finished proof object can now be inspected:

```
Unnamed_thm < Show Proof.
LOC:
Subgoals
Proof: fun (H : ~ (exists x : D, P x)) (d : D) (H2 : P d) =>
       H (ex_intro (fun x : D => P x) d H2)
```

The commands `Abort`, `Restart`, `Undo` etc. are useful when your proof goes on the wrong track. It is very useful to have a text editor running on the side to record your commands. There is also a special Emacs module dedicated to this, with lots of sophisticated support and tools, called Proof General.

Problems 1

Prove the following propositions in predicate calculus.

(i) $(A \wedge B \to C) \to (A \to (B \to C))$. Here A, B, C are propositions. The goal formula is written

   `(A /\ B -> C) -> (A -> (B -> C))`

(ii) $(\exists x \in D)P(x) \vee (\exists x \in D)Q(x) \to (\exists x \in D)(P(x) \vee Q(x))$. Here P and Q are propositional functions on D. The goal formula is written

   `(exists x:D, P(x)) \/ (exists x:D, Q(x)) -> (exists x:D, P(x) \/ Q(x))`

(iii) $((\exists x \in D)P(x) \to C) \leftrightarrow (\forall x \in D)(P(x) \to C)$. Here C is a proposition. Note that $A \leftrightarrow B$ is defined as $(A \to B) \wedge (B \to A)$ so using `split` at some point is a good tactic. Equivalence $\leftrightarrow$ is rendered `<->` .

(iv) A predicate R of two variables on D — that is a binary relation on D — may conveniently be regarded as a propositional function with the declaraion `Variable R: D -> D -> Prop.` We may also consider relations between different sets. To prepare for a proof of the theorem

$$(\exists y \in E)(\forall x \in D)R(x, y) \to (\forall x \in D)(\exists y \in E)R(x, y).$$

we execute the following commands

```
Coq < Section Predicate_calculus.
Coq < Variables D E : Set.
Coq < Variable R : D -> E -> Prop.
Coq < Theorem EAAE : (exists y:E, forall x:D, R x y)
Coq <                  -> (forall x:D, exists y:E, R x y).
```

The command `Theorem EAAE:` works the same as `Goal` but gives the proposition to be proved the name EAAE.

Complete the above proof! You need to use the tactics `elim` at some point.

(v) The following exercise is to make a formal proof of so-called *well-founded induction* from ordinary complete induction on natural numbers (cf. Exercise sheet 3 problem 12).

Enter the following commands into Coq (the computer savvy cuts and paste from the (electronic) text document).

```
Section Predicate_calculus.
Variable N:Set.
Variable zero:N.
Variable S:N -> N.
Hypothesis Peano3: (forall x y:N, S x = S y -> x=y).
Hypothesis Peano4: (forall x:N, ~ (S x = zero)).
Variable Lt:N -> N -> Prop.
Hypothesis LtAx1a: (forall x y: N, (Lt x (S y)) -> x=y \/ (Lt x y)).
Hypothesis LtAx1b: (forall x y: N, x=y \/ (Lt x y) -> (Lt x (S y))).
Hypothesis LtAx1c: (forall x y: N, (Lt (S x) (S y)) -> (Lt x y)).
Hypothesis LtAx1d: (forall x y: N, (Lt x y) -> (Lt (S x) (S y))).
Hypothesis LtAx2a: (forall x:N, ~ (Lt x zero)).
Hypothesis LtAx2b: (forall x:N, (Lt zero (S x))).
Hypothesis LtAx3: (forall x:N, Lt x (S x)).
Hypothesis IND: (forall P:N ->Prop,
      (P zero) -> (forall n:N, (P n) -> (P (S n))) -> (forall n:N, P n)).
Definition UpTo (Q:N -> Prop) (n:N) : Prop  := (forall m: N, Lt m n -> Q m).
Theorem WFIND: (forall Q:N -> Prop,
    (forall n:N, (forall m:N, Lt m n -> Q m) -> Q n) -> (forall n:N, Q n)).
```

For reasoning about equality (=) there are several built-in tactics. A simple but useful one is `replace`.

The definition `UpTo` may be handy in the proof or as a hint. Given a predicate Q on N it (`UpTo` Q) is the predicate P given by

$$P(n) = (\forall m : N)(m < n \rightarrow Q(m))$$

Complete the proof! (Are all axioms (hypotheses) necessary for the theorem?)

(vi) With the same hypotheses as above formulate and prove in Coq the following

$$(\forall n, m : N)(n < m \vee n = m \vee m < n).$$

You may find it convenient to introduce and prove some lemmas. In the proof of this there is a decision procedure for equality on natural numbers hidden. If the theorem is proved using Coq's own version of numbers, one can directly extract Haskell or ML program that makes such decisions.