Aide-mémoire des tactiques de preuve Coq

Logique propositionnelle minimale

Introduction

```
1 subgoal

  ============================
  forall x : T, P x -> Q x
```

```
intros x.
```

```
subgoal 1 is:

  x : T
  ============================
  P x -> Q x
```

```
intros H.
```

```
subgoal 1 is:

  x : T
  H : P x
  ============================
  Q x
```

Terme de preuve fourni exactement

```
1 subgoal

  a : A
  b : B
  ============================
  A
```

```
exact a.
```

```
Qed.
```

Application (d'hypothèse ou de lemme)

```
1 subgoal

  t : T
  H : forall x : T, P x -> Q x -> R x
  ============================
  R t
```

```
apply H.
```

```
subgoal 1 is:

  t : T
  H : forall x : T, P x -> Q x -> R x
  ============================
  P t
```

```
subgoal 2 is:

  t : T
  H : forall x : T, P x -> Q x -> R x
  ============================
  Q t
```

Propriétés de l'égalité (relation d'équivalence)

Réflexivité

```
1 subgoal

  t : T
  ============================
  t = t
```

```
reflexivity.
```

```
Qed.
```

Réflexivité (modulo calcul)

```
1 subgoal

  ============================
  1 + 1 = 2
```

```
reflexivity.
```

```
Qed.
```

Symétrie

```
1 subgoal

  t1, t2 : T
  ============================
  t1 = t2
```

```
symmetry.
```

```
subgoal 1 is:

  t1, t2 : T
  ============================
  t2 = t1
```

Transitivité

```
1 subgoal

  t1, t2, t : T
  ============================
  t1 = t2
```

```
transitivity t.
```

```
subgoal 1 is:

  t1, t2, t : T
  ============================
  t1 = t
```

```
subgoal 2 is:

  t1, t2, t : T
  ============================
  t = t2
```

Utilisation d'une égalité

Réécriture dans le but

```
1 subgoal

  t1, t2 : T
  E : t1 = t2
  ============================
  P t1
```

```
rewrite E.
```

```
subgoal 1 is:

  t1, t2 : T
  E : t1 = t2
  ============================
  P t2
```

```
rewrite <- E.
```

```
subgoal 1 is:

  t1, t2 : T
  E : t1 = t2
  ============================
  P t1
```

Réécriture dans une hypothèse

```
1 subgoal

  t1, t2 : T
  E : t1 = t2
  H : P t1
  ============================
  G
```

```
rewrite E in H.
```

```
subgoal 1 is:

  t1, t2 : T
  E : t1 = t2
  H : P t2
  ============================
  G
```

```
rewrite <- E in H.
```

```
subgoal 1 is:

  t1, t2 : T
  E : t1 = t2
  H : P t1
  ============================
  G
```

Types inductifs

Soit le code suivant définissant un type inductif

```
Inductive t (p : T1) :=
| C1 (p11 : T11 p) (p12 : T12 p p11)
| C2 (p21 : T21 p) (p22 : t p)
.
```

Application d'un constructeur

```
1 subgoal

  p : T1
  ============================
  t p
```

```
constructor 2.
```

```
subgoal 1 is:

  p : T1
  ============================
  T21 p
```

```
subgoal 2 is:

  p : T1
  ============================
  t p
```

Analyse par cas

```
1 subgoal

  p : T1
  ============================
  forall x : t p, prop p x
```

```
destruct x.
```

```
subgoal 1 is:

  p : T1
  p11 : T11 p
  p12 : T12 p p11
  ============================
  prop p (C1 p p11 p12)
```

```
subgoal 2 is:

  p : T1
  p21 : T21 p
  x : t p
  ============================
  prop p (C2 p p21 x)
```

Induction

```
1 subgoal

  p : T1
  ============================
  forall x : t p, prop p x
```

```
induction x.
```

```
subgoal 1 is:

  p : T1
  p11 : T11 p
  p12 : T12 p p11
  ============================
  prop p (C1 p p11 p12)
```

```
subgoal 2 is:

  p : T1
  p21 : T21 p
  x : t p
  IHx : prop p x
  ============================
  prop p (C2 p p21 x)
```

Égalité et propriétés des constructeurs

Les constructeurs sont disjoints

```
1 subgoal

  ============================
  C1 p1 p11 p12 <> C2 p1 p21 p22
```

```
discriminate.
```

```
Qed.
```

Les constructeurs sont disjoints

```
1 subgoal

  K : C1 p1 p11 p12 = C2 p1 p21 p22
  ============================
  G
```

```
discriminate K.
```

```
Qed.
```

Chaque constructeur est injectif

```
1 subgoal

  a, a' : T21 p1
  b, b' : t p1
  H : C2 p1 a b = C2 p1 a' b'
  ============================
  G
```

```
injection H.
```

```
subgoal 1 is:

  a, a' : T21 p1
  b, b' : t p1
  H : C2 p1 a b = C2 p1 a' b'
  ============================
  b = b' -> a = a' -> G
```

Connecteurs et logique propositionnelle

Introduction du connecteur ET

```
1 subgoal

  ============================
  A /\ B
```

```
split.
```

```
subgoal 1 is:

  ============================
  A
```

```
subgoal 2 is:

  ============================
  B
```

Introduction du connecteur OU (gauche)

```
1 subgoal

  ============================
  A \/ B
```

```
left.
```

```
subgoal 1 is:

  ============================
  A
```

Introduction du connecteur OU (droite)

```
1 subgoal

  ============================
  A \/ B
```

```
right.
```

```
subgoal 1 is:

  ============================
  B
```

Introduction du connecteur existentiel

```
1 subgoal

  e : T
  ============================
  exists x : T, P x
```

```
exists e.
```

```
subgoal 1 is:

  e : T
  ============================
  P e
```

Élimination du connecteur ET

```
1 subgoal

  H : A /\ B
  ============================
  G
```

```
destruct H as [HA HB].
```

```
subgoal 1 is:

  HA : A
  HB : B
  ============================
  G
```

Élimination du connecteur OU

```
1 subgoal

  H : A \/ B
  ============================
  G
```

```
destruct H as [HA|HB].
```

```
subgoal 1 is:

  HA : A
  ============================
  G
```

```
subgoal 2 is:

  HB : B
  ============================
  G
```

Élimination du connecteur existentiel

```
1 subgoal

  H : exists x : T, P x
  ============================
  G
```

```
destruct H as [x Hx].
```

```
subgoal 1 is:

  x : T
  Hx : P x
  ============================
  G
```

Réduction

Dépliage de définition (unfold)

```
1 subgoal

  a : A
  ============================
  ~ ~ A
```

```
unfold not.
```

```
subgoal 1 is:

  a : A
  ============================
  (A -> False) -> False
```

```
intros H.
```

```
subgoal 1 is:

  a : A
  H : A -> False
  ============================
  False
```

```
apply H.
```

```
subgoal 1 is:

  a : A
  H : A -> False
  ============================
  A
```

```
exact a.
```

```
Qed.
```

Simplification du but

```
1 subgoal

  a : A
  l : list A
  ============================
  [a] ++ l = a :: l
```

```
simpl.
```

```
subgoal 1 is:

  a : A
  l : list A
  ============================
  a :: l = a :: l
```

Automatisation

Arithmétique linéaire

```
Require Import Omega.

Lemma example :
      forall (n p : nat), 2 * (n + 1) - n + p > n.
```

```
intros n p; omega.
```

```
Qed.
```

Tautologies propositionnelles

```
Lemma iff_implies :
      forall (P1 P2 : Prop), (P1 <-> P2) <-> ((P1 -> P2) /\ (P2 -> P1)).
```

```
tauto.
```

```
Qed.
```

Tautologies du premier ordre

```
(* Aucun violoniste n'est punk ;
   Or quelques punks sont musiciens ;
   Donc quelques musiciens ne sont pas violonistes. *)
Lemma Fresison :
      forall (T : Type) (violoniste punk musicien : T -> Prop),
      (forall v, violoniste v -> ~ punk v) ->
      (exists p, punk p /\ musicien p) ->
      exists m, musicien m /\ ~ violoniste m.
```

```
firstorder.
```

```
Qed.
```