

Táticas e automação de provas

Introdução ao assistente de provas Coq

Rodrigo Ribeiro

Combinadores de táticas

- Combinador secuencial.

```
Theorem seq_comb  
: forall (A B : Prop),  
    (((A -> B) -> B) -> B) -> A -> B.
```

Proof.

```
  intros A B HImp Ha ; apply HImp ;  
  intros Hab ; apply Hab ; assumption.
```

Qed.

Combinadores de táticas

- Composição generalizada

Theorem chain

```
: forall (A B C : Prop), (A -> B -> C) ->  
  (A -> B) -> A -> C.
```

Proof.

```
  intros A B C Habc Hab Ha ; apply Habc ;  
    [ assumption | apply Hab ; assumption ].
```

Qed.

Combinadores de táticas

Theorem orElse_example

```
: forall (A B C D : Prop), (A -> B) -> C ->  
  ((A -> B) -> C -> (D -> B) -> D) -> A -> D.
```

Proof.

```
  intros A B C D Hab Hc H Ha ;  
    apply H ; (assumption || intro H1) ;  
      apply Hab ; assumption.
```

Qed.

Táticas Adicionais

- ▶ constructores
- ▶ clear
- ▶ congruence
- ▶ intuition
- ▶ omega

Tática auto

- ▶ Prova qualquer conclusão que pode ser resolvida usando `assumption`, `intros` e `apply` até uma profundidade máxima.
- ▶ Valor padrão de profundidade 5.
- ▶ Execução de `auto` nunca resulta em erro.
 - ▶ Resolve completamente a conclusão ou não a modifica.

Tática auto

```
Theorem auto_example1
: forall (A B C D : Prop), (A -> B) -> C ->
  ((A -> B) -> C -> (D -> B) -> D) -> A -> D.
Proof.
  auto.
Qed.
```

Hint databases

- ▶ Conjunto de teoremas a serem utilizados pela tática `auto`.
- ▶ Comandos
 - ▶ Hint Resolve `thm1 ... thmn`.
 - ▶ Hint Constructors `t1 ... tn`.
 - ▶ Hint Unfold `df1 ... dfn`.

Exemplo

```
Inductive Plus : nat -> nat -> nat -> Prop :=  
| PlusZero  
  : forall m, Plus 0 m m  
| PlusSucc  
  : forall n m r,  
    Plus n m r ->  
    Plus (S n) m (S r).
```

Exemplo

Example plus_4_3_auto : Plus 4 3 7.

Proof.

repeat constructor.

Qed.

Exemplo

Hint Constructors Plus.

Example plus_4_3_auto : Plus 4 3 7.

Proof.

 auto.

Qed.

Programando táticas com Ltac

- ▶ Construções para casamento de padrão sobre o estado de prova.

```
Ltac break_if :=  
  match goal with  
  | [ |- if ?X then _ else _ ] => destruct X  
end.
```

Programando táticas com Ltac

► Exemplo

```
Theorem hmm : forall (a b c : bool),  
  if a  
  then if b  
        then True  
        else True  
  else if c  
        then True  
        else True.
```

Proof.

```
  intros; repeat break_if; constructor.  
Qed.
```

Programando táticas com Ltac

- ▶ context patterns

```
Ltac break_if_inside :=  
  match goal with  
  | [ |- context[if ?X then _ else _] ] => destruct X  
  end.
```

Programando táticas com Ltac

► Exemplo

```
Theorem hmm2 : forall (a b : bool),  
  (if a then 42 else 42) = (if b then 42 else 42).
```

```
Proof.
```

```
  intros; repeat break_if_inside; reflexivity.
```

```
Qed.
```

Programando táticas com Ltac

- ▶ “Combo” repeat + match goal

```
Ltac simple_tauto :=  
  repeat match goal with  
    | [ H : ?P |- ?P ] => exact H  
    | [ |- True ] => constructor  
    | [ |- _ /\ _ ] => constructor  
    | [ |- _ -> _ ] => intro  
    | [ H : False |- _ ] => destruct H  
    | [ H : _ /\ _ |- _ ] => destruct H  
    | [ H : _ \/ _ |- _ ] => destruct H  
    | [ H1 : ?P -> ?Q, H2 : ?P |- _ ] =>  
      apply H1 in H2  
  end.
```


Programando táticas com Ltac

► Exemplo

```
Lemma simple_example
  : forall A B C, (A -> B) -> (B -> C) -> A -> C.
Proof.
  simple_tauto.
Qed.
```