



Coqatoo

Generating Natural Language Versions of Coq Proofs

Andrew Bedford

Laval University

CoqPL 2018

Motivation

- Proofs can sometimes be hard to understand, particularly for less-experienced users
- CtCoq and its successor Pcoq are no longer available

Example

Input

```
Lemma conj_imp_equiv : forall P Q R:Prop,  
  (P /\ Q -> R) <-> (P -> Q -> R).
```

```
Proof.
```

```
  intros. split. intros H HP HQ. apply H. apply conj.  
  assumption. assumption. intros H HPQ. inversion HPQ. apply  
  H. assumption. assumption.
```

```
Qed.
```

Coqatoo's rewriting algorithm can be decomposed in three steps:

- 1 Information extraction
- 2 Proof tree construction
- 3 Tactic-based rewriting

Step 1: Information extraction

Coqatoo captures the intermediary proof states

```
1 subgoal

=====
forall P Q R : Prop, (P /\ Q -> R) <-> (P -> Q -> R)
```

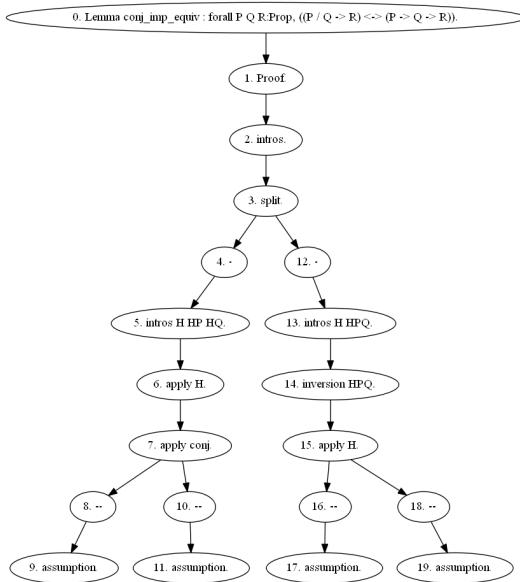
Listing 1: State before executing the first intros tactic

```
1 subgoal

P, Q, R : Prop
=====
(P /\ Q -> R) <-> (P -> Q -> R)
```

Listing 2: State after executing the first intros tactic

Step 2: Proof tree construction



Step 3: Tactic-based rewriting

Example

Output

```
Lemma conj_imp_equiv : forall P Q R:Prop, ((P /\ Q -> R) <=> (P -> Q -> R)).
Proof.
  (* Assume that P, Q and R are arbitrary objects of type Prop. Let us show that (P
  /\ Q -> R) <=> (P -> Q -> R) is true. *) intros.
split.
- (* Case (P /\ Q -> R) -> P -> Q -> R: *)
  (* Suppose that P, Q and P /\ Q -> R are true. Let us show that R is true. *)
  intros H HP HQ.
  (* By our hypothesis P /\ Q -> R, we know that R is true if P /\ Q is true. *)
  apply H.
  apply conj.
  -- (* Case P: *)
    (* True, because it is one of our assumptions. *) assumption.
  -- (* Case Q: *)
    (* True, because it is one of our assumptions. *) assumption.
- (* Case (P -> Q -> R) -> P /\ Q -> R: *)
  (* Suppose that P /\ Q and P -> Q -> R are true. Let us show that R is true. *)
  intros H HPQ.
  (* By inversion on P /\ Q, we know that P, Q are also true. *) inversion HPQ.
  (* By our hypothesis P -> Q -> R, we know that R is true if P and Q are true. *)
  apply H.
  -- (* Case P: *)
    (* True, because it is one of our assumptions. *) assumption.
  -- (* Case Q: *)
    (* True, because it is one of our assumptions. *) assumption.
Qed.
```

Listing 3: Output in annotation mode

Demonstration

Comparison

Disadvantages

- It only works on proofs whose tactics are supported, while the approach of Coscoy et al. worked on any proof.
- It may require additional verifications to ensure that unnecessary information (e.g., an assertion which isn't used) is not included in the generated proof.

Comparison

Advantages

- It enables us to more easily control the size and verbosity of the generated proof (one or two sentences per tactic by default).
- It maintains the order and structure of the user's original proof script; this is not necessarily the case in Coscoy et al.

Future work

- Increase the number of supported tactics
 - Goal: Software Foundations
- Add partial support for automation
- Integration with existing development environments
- Add a LaTeX output mode