Interactive Computer Theorem Proving

Lecture 2: Propositional and First-Order Logic

CS294-9
August 31, 2006
Adam Chlipala
UC Berkeley

Liberal use of **natural language**

What is a Proof?

Theorem: For all n, $2\sum_{i \in 0..n} i = n(n+1)$

Ad-hoc idea of **which high-level argument techniques** are valid

Proof: By **induction on *n***.

Base case ($n = 0$):

$$2\sum_{i \in 0..n} i = 0 = 0(0+1)$$

Inductive case (n = n' + 1):

IH: $2\sum_{i \in 0..n'} i = n'(n'+1)$

$$2\sum_{i \in 0..n} i = 2(n'+1) + 2\sum_{i \in 0..n'} i$$

$$= 2(n'+1) + n'(n' + 1)$$

$$= (n' + 1)(n' + 2)$$

All sorts of zany **notations**

“Obvious steps” whose obviousness is in the eye of the beholder

Getting Warmer....

Statements	Reasons
1. Segment AD bisects segment BC.	1. Given.
2. Segments AM and MD are congruent.	2. When a segment is bisected, the two resulting segments are congruent.
3. Segment BC bisects segment AD.	3. Given.
4. Segments BM and CM are congruent.	4. When a segment is bisected, the two resulting segments are congruent.
5. Angles AMB and DMC are congruent.	5. Vertical angles are congruent.
6. Triangles ABM and DCM are congruent.	6. SAS postulate (2, 4, 5).

Coq Version

```
Fixpoint sum (n : nat) : nat :=
  match n with
    | O => O
    | S n => S n + sum n
  end.

Theorem sum_equals : forall n, 2 * sum n = n * (n + 1).
  induction n.

  trivial.

  defn sum.
  rewrite mult_plus_distr_l.
  rewrite IHn.
  ring_nat.
Qed.
```

Under the Hood….

```
sum_equals =
fun n : nat =>
nat_ind (fun n0 : nat => 2 * sum n0 = n0 * (n0 + 1))
  (refl_equal (0 * (0 + 1)))
  (fun (n0 : nat) (IHn : 2 * sum n0 = n0 * (n0 + 1)) =>
   eq_ind_r (fun n1 : nat => n1 = S n0 * (S n0 + 1))
     (eq_ind_r (fun n1 : nat => 2 * S n0 + n1 = S n0 * (S n0 + 1))
        (eq_ind
           (interp_cs plus mult 1 0
              (Node_vm n0 (Empty_vm nat) (Empty_vm nat))
              (Cons_monom 2 Nil_var
                 (Cons_monom 3 (Cons_var End_idx Nil_var)
                    (Cons_varlist
                       (Cons_var End_idx (Cons_var End_idx Nil_var))
                       (Nil_monom nat)))))
           (fun n1 : nat => (1 + 1) * (1 + n0) + n0 * (n0 + 1) = n1)
           (sym_eq
              (spolynomial_simplify_ok nat plus mult 1 0 nateq
                 (Node_vm n0 (Empty_vm nat) (Empty_vm nat)) NatTheory
                 (SPplus
                    (SPmult (SPplus (SPconst 1) (SPconst 1))
                       (SPplus (SPconst 1) (SPvar nat End_idx)))
                    (SPmult (SPvar nat End_idx)
                       (SPplus (SPvar nat End_idx) (SPconst 1))))))
           ((1 + n0) * (1 + n0 + 1))
           (spolynomial_simplify_ok nat plus mult 1 0 nateq
              (Node_vm n0 (Empty_vm nat) (Empty_vm nat)) NatTheory
              (SPmult (SPplus (SPconst 1) (SPvar nat End_idx))
                 (SPplus (SPplus (SPconst 1) (SPvar nat End_idx)) (SPconst 1)))))
        IHn) (mult_plus_distr_l 2 (S n0) (sum n0))) n
```

Propositional Logic

$$p ::= \top \mid \bot \mid P \mid p \to p \mid p \land p \mid p \lor p \mid \neg p$$

$P \to P$

P	$P \to P$
0	**1**
1	**1**

$P \land Q \to P \lor Q$

P	Q	$P \land Q \to P \lor Q$
0	**0**	**1**
0	**1**	**1**
1	**0**	**1**
		1

OK, now let's check

$$A \land B \land C \land D \land E \land F \to D$$

Natural Deduction: "and"

$$\frac{A \qquad B}{A \wedge B}\wedge I$$

$$\frac{A \wedge B}{A}\wedge E1 \qquad \frac{A \wedge B}{B}\wedge E2$$

$$\frac{}{\top}\top I$$

Example

$$\dfrac{\dfrac{}{\top}\top I \qquad \dfrac{}{\top}\top I}{\top \wedge \top}\wedge I$$

Example

Given: $\mathcal{D} : A \wedge B$

$$\dfrac{\dfrac{\mathcal{D}}{B}\wedge E2 \quad \dfrac{\mathcal{D}}{A}\wedge E1}{B \wedge A}\wedge I$$

Natural Deduction: “implies”

$$\frac{A \to B \qquad A}{B}\ {\to}E$$

$$\frac{\begin{array}{c} x : A \\ \vdots \\ B \end{array}}{A \to B}\ {\to}I$$

On the board:

$(A \to B) \to (B \to C) \to (A \to C)$

Example

$$\frac{\begin{array}{c} x : A \\ \vdots \\ x \end{array}}{A \to A}\ {\to}I$$

Example

$$\frac{\begin{array}{c} x : A \to B \\ \vdots \\ \dfrac{\begin{array}{c} y : A \\ \vdots \\ \dfrac{x \qquad y}{B}\ {\to}E \end{array}}{A \to B}\ {\to}I \end{array}}{(A \to B) \to A \to B}\ {\to}I$$

Natural Deduction: “or”

$$\frac{A}{A \vee B}\vee\mathrm{I1} \qquad\qquad \frac{B}{A \vee B}\vee\mathrm{I2}$$

$$\frac{A \vee B \quad \begin{array}{c} x : A \\ \vdots \\ C \end{array} \quad \begin{array}{c} y : B \\ \vdots \\ C \end{array}}{C}\vee\mathrm{E}$$

Example

$$\begin{array}{c} x : A \vee A \\ \vdots \\ \dfrac{\dfrac{x \quad \begin{array}{c} y : A \\ \vdots \\ y \end{array} \quad \begin{array}{c} z : A \\ \vdots \\ z \end{array}}{A}\vee\mathrm{E}}{A \vee A \to A}\to\mathrm{I} \end{array}$$

On the board:

$(A \to C) \to (B \to C) \to A \vee B \to C$

Natural Deduction: "false" and "not"

$$\frac{\perp}{A}\ \perp E$$

$$\frac{\begin{array}{c} x : A \\ \vdots \\ \perp \end{array}}{\neg A}\ \neg I \qquad\qquad \frac{\neg A \quad A}{B}\ \neg E$$

On the board: $\neg\perp$ *On the board:* $\neg\neg\top$

We can derive the "not" rules by defining:

$$\neg A \equiv A \rightarrow \perp$$

And now in Coq…

(natded.v)

First-Order Logic

$$t ::= \boldsymbol{x} \mid \mathbf{f(t, \ldots, t)}$$

$$p ::= \top \mid \bot \mid p \to p \mid p \wedge p \mid p \vee p \mid \neg p$$

$$\mid \mathbf{P(t, \ldots, t)} \mid \mathbf{t = t} \mid \forall \boldsymbol{x}\mathbf{, p} \mid \exists \boldsymbol{x}\mathbf{, p}$$

Examples

$\forall x, \mathrm{P}(x) \to \exists y, \mathrm{Q}(x, y)$

$\forall x, \mathrm{f}(x) = \mathrm{g}(x) \to x = \mathrm{c} \vee x = \mathrm{h}(x, \mathrm{c})$

- **Bad news:** We're dealing with infinite universes, so there is no clear analogue of truth tables.
- **Good news:** We can extend natural deduction to work for first-order logic.

Natural Deduction: “forall”

$$\frac{\forall x, A}{A\{x := t\}}\ \forall E$$

$$\frac{\begin{matrix} y \\ \vdots \\ A\{x := y\} \end{matrix}}{\forall x, A}\ \forall I$$

Natural Deduction: “exists”

$$\frac{A\{x := t\}}{\exists x, A}\ \exists I$$

$$\frac{\exists x, A \qquad \begin{array}{c} y \quad A\{x := y\} \\ \vdots \\ B \end{array}}{B}\ \exists E$$

Natural Deduction: “equals”

$$\frac{}{t = t}\ {=}\mathrm{I}$$

$$\frac{t1 = t2 \qquad\qquad A}{A\{t1 := t2\}}\ {=}\mathrm{E}$$

Conclusion

- All code is on the web site.
- HW1 is posted
  - Propositional & first-order logic
  - Due before the start of next lecture
- Next lecture: Data structures