CS720

Logical Foundations of Computer Science

Lecture 9: Inductive propositions

Tiago Cogumbreiro

Inductive propositions

In lectures 7 and 8 we learned to write inductive definitions that compose other propositions (eg, $\wedge$ takes holds two propositions)

> Think about the following statement:

A product $X \times Y$ is to a conjunction $P \wedge Q$, the same way a `list X` is to...?

> Today we define inductive definitions that can "hold" an unbounded number of propositions.

Today we will learn…

- (recursive) inductive definitions
- implementing binary relations
- properties on binary relations

Exercise

Let us define even numbers inductively...

- In the world of propositions, what is a signature of a number being even?

Exercise

Let us define even numbers inductively…

> In the world of propositions, what is a signature of a number being even?

```
Inductive ev: nat → Prop
```

Exercise

Let us define even numbers inductively...

> In the world of propositions, what is a signature of a number being even?

```
Inductive ev: nat → Prop
```

- 0 is even
- If n is even, then $2 + n$ is also even.

Inductively defined even

In Logic, the constructors `ev_0` and `ev_SS` of propositions can be called *inference rules*.

```
Inductive ev: nat → Prop :=
(* Rule 1: *)
| ev_0:
  ev 0
(* Rule 2: *)
| ev_SS: forall n,
  ev n →
(*------------*)
  ev (S (S n)).
```

Which can be typeset as an inductive definition with the following notation:

$$\frac{}{\mathtt{ev}(0)}\,\mathtt{ev\_0} \qquad \frac{\mathtt{ev}(n)}{\mathtt{ev}(\mathtt{S}(\mathtt{S}(n)))}\,\mathtt{ev\_SS}$$

Proving that 4 is even

$$\frac{}{\mathtt{ev}\ 0}\ \mathtt{ev\_0}$$

$$\frac{}{\mathtt{ev}\ 2}\ \mathtt{ev\_SS}$$

$$\frac{}{\mathtt{ev}\ 4}\ \mathtt{ev\_SS}$$

Backward style: From `ev_SS` we can conclude that 4 is even, if we can show that 2 is even, which follows from `ev_SS` and the fact that 0 is even (by `ev_0`).

Forward style: From the fact that 0 is even (`ev_0`), we use theorem `ev_SS` to show that 2 is even; so, applying theorem `ev_SS` to the latter, we conclude that 4 is even.

```
Goal ev 4.
Proof. (* backward style proof *)
  apply eq_SS.
  apply eq_SS.
  apply ev_0.
Qed.

Goal ev 4.
Proof. (* forward style proof *)
  apply (ev_SS 2 (ev_SS 0 ev_0)).
Qed.
```

Reasoning about inductive propositions

```
Theorem evSS : forall n,
  ev (S (S n)) -> ev n.
```

(Done in class.)

Example

```
Goal ~ ev 3.
```

(Done in class.)

Proofs by induction

```
Goal forall n, ev n -> ~ ev (S n).
```

(Done in class.)

Proofs by induction

```
Goal forall n, ev n -> ~ ev (S n).
```

(Done in class.)

> Notice the difference between induction on `n` and on judgment `ev n`.

Relations in Coq

```
Inductive le : nat → nat → Prop :=
  | le_n :
    forall n,
    le n n

  | le_S :
    forall n m,
    le n m →
    le n (S m).
Notation "n ≤ m" := (le n m).
```

$$\frac{}{n \leq n}\,\mathtt{le\_n} \qquad \frac{n \leq m}{n \leq \mathsf{S}\ m}\,\mathtt{le\_S}$$

Exercise

```
Goal 3 ≤ 6.
```

Less-than

```
Definition lt (n m:nat) := le (S n) m.
```

> How do we prove that this definition is correct?

Less-than

```
Definition lt (n m:nat) := le (S n) m.
```

> How do we prove that this definition is correct?

```
Goal n ≤ m ↔ lt n m \/ n = m.
```

Less-than

- How can we define Less-Than inductively?

Less-than

How can we define Less-Than inductively?

```
Inductive lt : nat → nat → Prop :=
  | lt_base :
    forall n,
    lt n (S n)

  | lt_S :
    forall n m,
    lt n m →
    lt n (S m).
Notation "n < m" := (lt n m).
```

How do we prove that this definition is correct?

Exercises on Less-Than

Prove that

1. $<$ is transitive
2. $<$ is irreflexive
3. $<$ is asymmetric
4. $<$ is decidable