

logical verification lecture 11

2011 05 13

prop2 and lambda2

overview: past present future

wat hier?

identity

```
Definition natid : nat -> nat :=  
  fun n : nat => n.
```

```
Definition boolid: bool -> bool :=  
  fun b : bool => b.
```

```
Definition polyid : forall A : Set, A -> A :=  
  fun (A : Set) => fun (a : A) => a.
```

lists

```
Inductive natlist : Set :=  
  natnil : natlist  
| natcons : nat -> natlist -> natlist.
```

```
Inductive boollist : Set :=  
  boolnil : boollist  
| boolcons : bool -> boollist -> boollist.
```

```
Inductive polylist (X : Set) : Set :=  
  polynil : polylist X  
| polycons : X -> polylist X -> polylist X.
```

pairs

```
Inductive natprod : Set :=  
| natpair : nat -> nat -> natprod.
```

```
Inductive boolprod : Set :=  
| boolpair : bool -> bool -> boolprod.
```

```
Inductive polyprod (X Y :Set) : Set :=  
| polypair : X -> Y -> polyprod X Y
```

bintrees

```
Inductive natbintree : Set :=  
  natleaf : natbintree  
| natnode :  
  natbintree -> nat -> natbintree -> natbintree.
```

```
Inductive boolbintree : Set :=  
| boolleaf : boolbintree  
| boolnode :  
  boolbintree -> bool -> boolbintree -> boolbintree.
```

```
Inductive polybintree (X : Set) : Set :=  
  polyleaf : polybintree X  
| polynode :  
  polybintree X -> X -> polybintree X -> polybintree X
```

instantiation

using application

lambda2: examples

$\lambda a : \text{Set}. \lambda x : a. x$

$\lambda a : \text{Prop}. \lambda x : a. x$

$\lambda a : \star. \lambda x : a. x$

$\lambda a : \text{Set}. \lambda x : a. \lambda b : \text{Set}. \lambda y : a \rightarrow b. (y x)$

$\lambda a : \text{Prop}. \lambda x : a. \lambda b : \text{Prop}. \lambda y : a \rightarrow b. (y x)$

$\lambda a : \star. \lambda x : a. \lambda b : \star. \lambda y : a \rightarrow b. (y x)$

$\lambda a : \text{Set}. \lambda b : \text{Set}. \lambda x : a \rightarrow b. \lambda y : a. (x y)$

$\lambda a : \text{Prop}. \lambda b : \text{Prop}. \lambda x : a \rightarrow b. \lambda y : a. (x y)$

$\lambda a : \star. \lambda b : \star. \lambda x : a \rightarrow b. \lambda y : a. (x y)$

lambda2: instantiation

by application and beta-reduction

natural numbers in lambda2

- $N = \prod a: * . a \rightarrow (a \rightarrow a) \rightarrow a$
- $n = \lambda a: * . \lambda o: a . \lambda s: a \rightarrow a . s^n o$
- $\text{successor} = \lambda n: N . \lambda a: * . \lambda o: a . \lambda s: a \rightarrow a . s (n a o s)$
- $\text{successorb} = \lambda n: N . \lambda a: * . \lambda o: a . \lambda s: a \rightarrow a . (n a (s o) s)$

booleans in lambda2

- $B = \Pi a: * . a \rightarrow a \rightarrow a$
- $T = \lambda a: * . \lambda x: a. \lambda y: a. x$
- $F = \lambda a: * . \lambda x: a. \lambda y: a. y$
- $\text{not} = \lambda b: B. \lambda a: * . \lambda u: a. \lambda v: a. (b a v u)$
- $\text{and} = \lambda b: B. \lambda b': B. \lambda a: * . \lambda u: a. \lambda v: a. (b a (b' a u v) v)$
- $\text{or} = \lambda b: B. \lambda b': B. \lambda a: * . \lambda u: a. \lambda v: a. (b a u (b' a u v))$

Curry-Howard-De Bruijn isomorphism

prop1 $\sim \lambda \rightarrow$
pred1 $\sim \lambda P$
prop2 $\sim \lambda 2$

Curry-Howard-De Bruijn isomorphism

back to the examples

$\lambda a : \star . \lambda x : a . x : \prod a : \star . \prod x : a . a$

$\lambda a : \star . \lambda x : a . \lambda b : \star . \lambda y : a \rightarrow b . (y x) : \prod a : \star . \prod x : a . \prod b : \star . \prod y : a \rightarrow b .$

$\lambda a : \star . \lambda b : \star . \lambda x : a \rightarrow b . \lambda y : a . (x y) : \prod a : \star . \prod b : \star . \prod x : a \rightarrow b . \prod y : a .$

Curry-Howard-De Bruijn isomorphism: dynamics

beta-reduction corresponds to detour-elimination_x