Interactive Formal Verification
5: Logic in Isabelle

Tjark Weber
(Slides: Lawrence C Paulson)
Computer Laboratory
University of Cambridge

Logical Frameworks

Logical Frameworks

- A formalism to represent other formalisms

Logical Frameworks

- A formalism to represent other formalisms
- Support for *natural deduction*

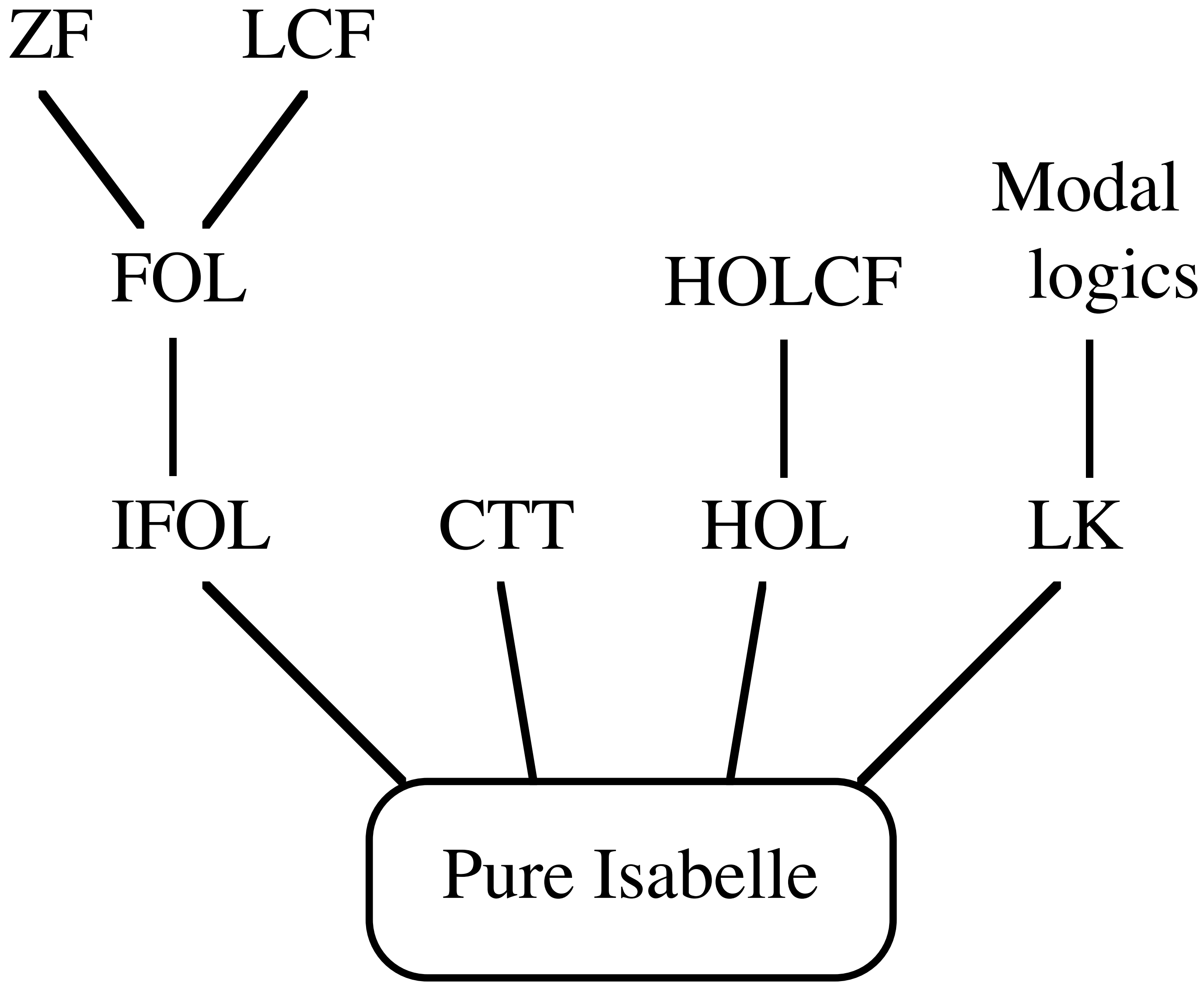
Logical Frameworks

- A formalism to represent other formalisms
- Support for *natural deduction*
- A common basis for implementations

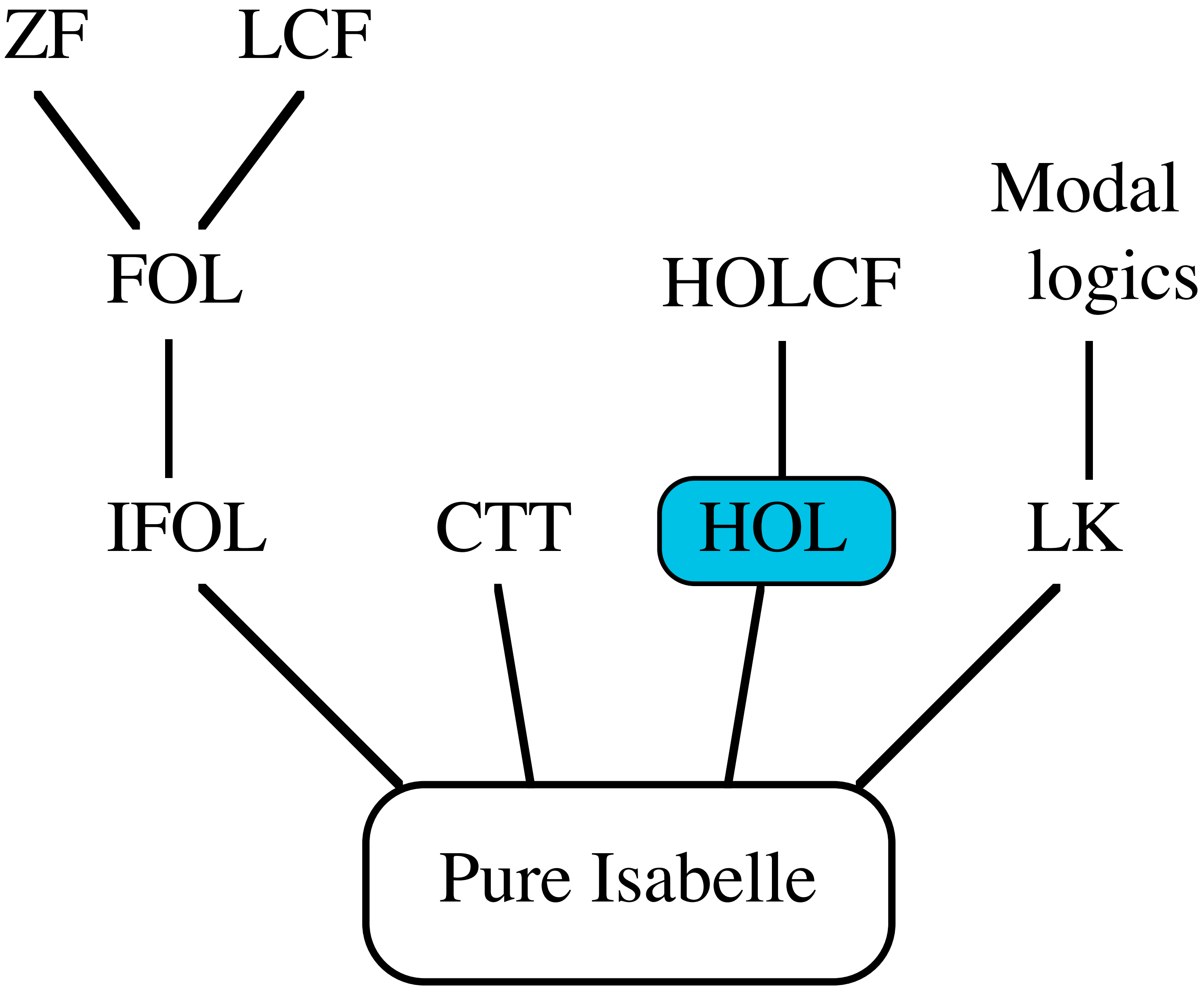
Logical Frameworks

- A formalism to represent other formalisms
- Support for *natural deduction*
- A common basis for implementations
- Type theories are commonly used, but Isabelle uses a simple meta-logic whose main primitives are
  - ⇒ (implication)
  - Λ (universal quantification)

Isabelle's Family of Logics



Isabelle's Family of Logics



ZF

LCF

FOL

HOLCF

Modal logics

IFOL

CTT

HOL

LK

Pure Isabelle

Natural Deduction in Isabelle

Natural Deduction in Isabelle

$$\frac{P \quad Q}{P \wedge Q}$$

```
P ⇒ (Q ⇒ P ∧ Q))
```

Natural Deduction in Isabelle

$$\frac{P \quad Q}{P \wedge Q}$$

```
P ⇒ (Q ⇒ P ∧ Q))
```

$$\frac{P \wedge Q}{P}$$

```
P ∧ Q ⇒ P
```

Natural Deduction in Isabelle

$$\frac{P \quad Q}{P \wedge Q}$$

```
P ⇒ (Q ⇒ P ∧ Q))
```

$$\frac{P \wedge Q}{P}$$

```
P ∧ Q ⇒ P
```

$$\frac{P \wedge Q}{Q}$$

```
P ∧ Q ⇒ Q
```

Natural Deduction in Isabelle

$$\frac{P \quad Q}{P \wedge Q}$$

```
P ⇒ (Q ⇒ P ∧ Q))
```

$$\frac{P \wedge Q}{P}$$

```
P ∧ Q ⇒ P
```

$$\frac{P \wedge Q}{Q}$$

```
P ∧ Q ⇒ Q
```

$$\frac{P \rightarrow Q \quad P}{Q}$$

```
P→Q ⇒ (P ⇒ Q)
```

Meta-implication

Meta-implication

- The symbol ⇒ (or ==>) expresses the relationship between premise and conclusion

Meta-implication

- The symbol ⇒ (or ==>) expresses the relationship between premise and conclusion
- ... and between subgoal and goal.

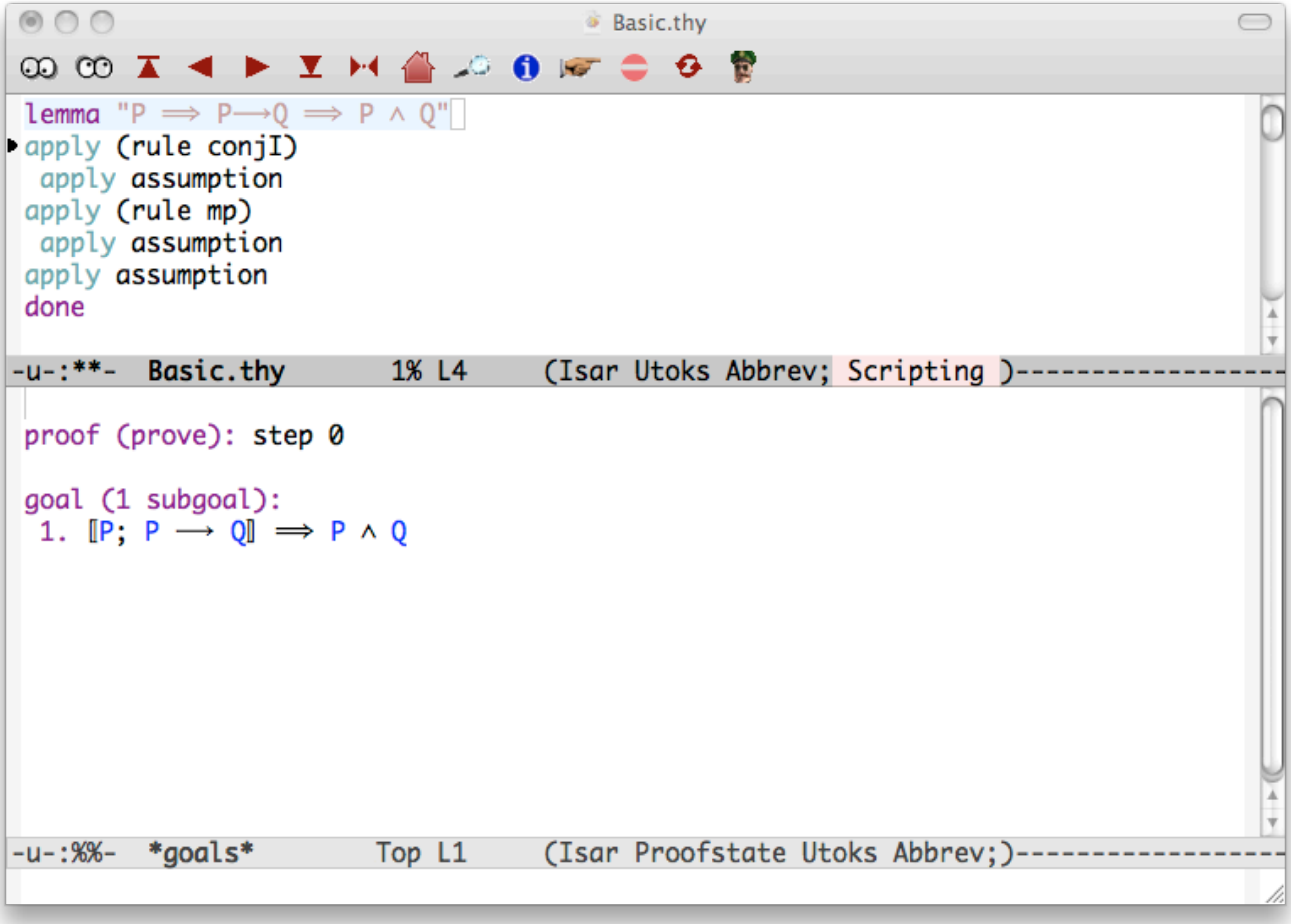
Meta-implication

- The symbol ⇒ (or ==>) expresses the relationship between premise and conclusion
- ... and between subgoal and goal.
- It is distinct from →, which is not part of Isabelle's underlying logical framework.

Meta-implication

- The symbol ⇒ (or ==>) expresses the relationship between premise and conclusion
- ... and between subgoal and goal.
- It is distinct from →, which is not part of Isabelle's underlying logical framework.
- `P⇒(Q⇒R)` is abbreviated as `⟦P;Q⟧ ⇒ R`

A Trivial Proof



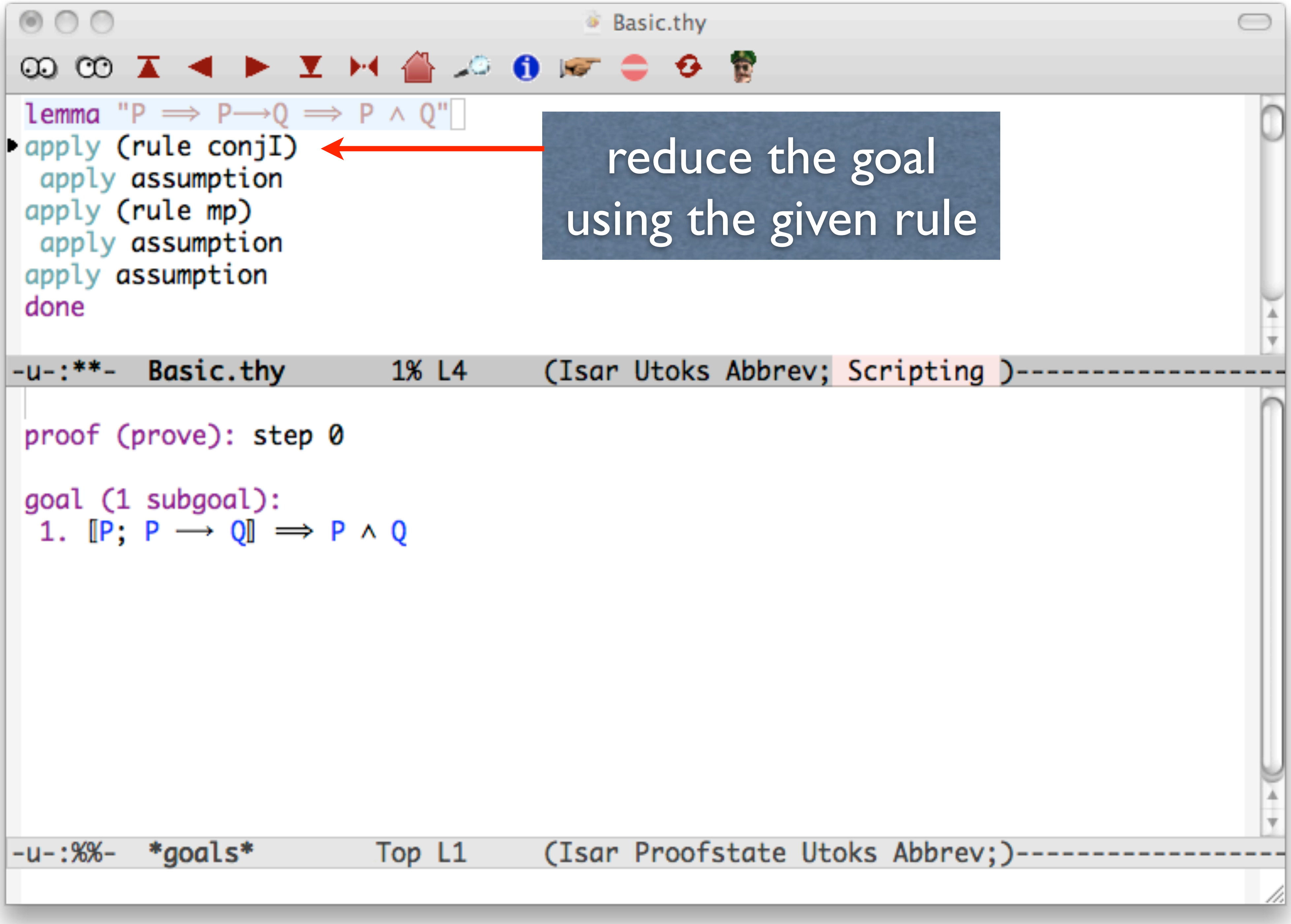
```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

```
-u-:**-  Basic.thy       1% L4      (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 0

goal (1 subgoal):
 1. ⟦P; P ⟶ Q⟧ ⟹ P ∧ Q
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
```

A Trivial Proof



```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

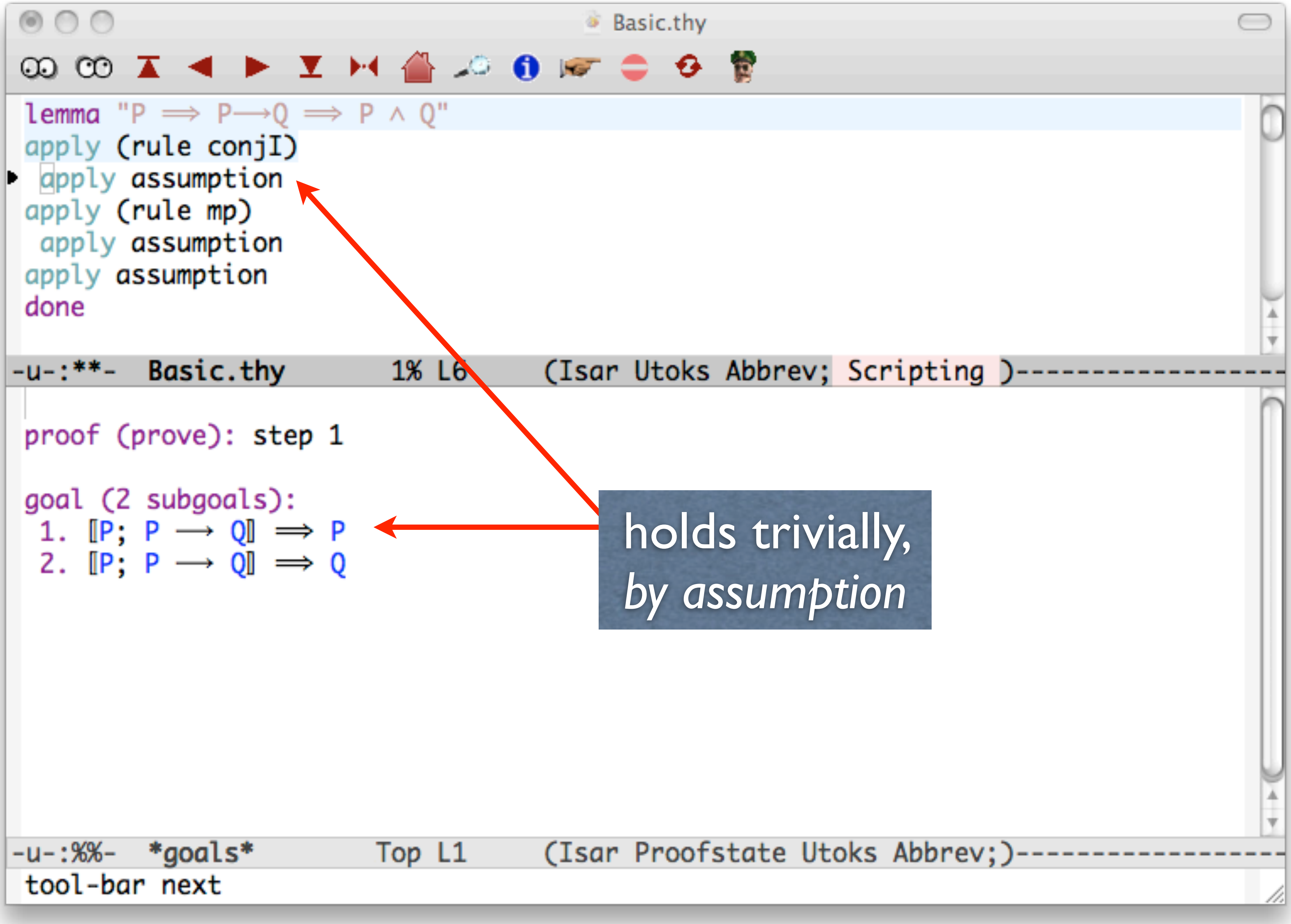
```
-u-:**-  Basic.thy        1% L4       (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 0

goal (1 subgoal):
 1. ⟦P; P ⟶ Q⟧ ⟹ P ∧ Q
```

```
-u-:%%-  *goals*         Top L1       (Isar Proofstate Utoks Abbrev;)-----------------
```

reduce the goal using the given rule

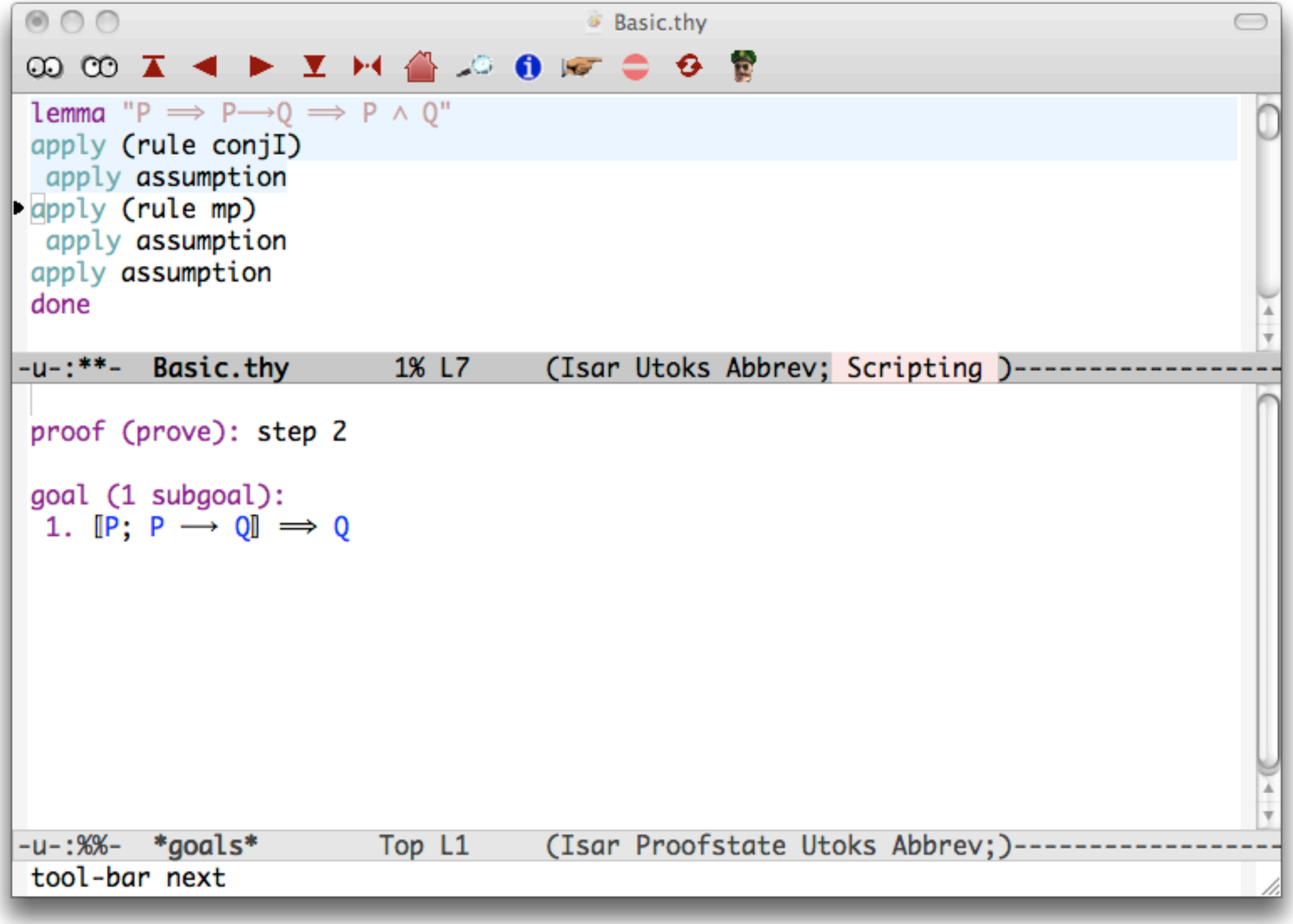
Proof by Assumption



```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
-u-:**-  Basic.thy      1% L6      (Isar Utoks Abbrev; Scripting )-----------------

proof (prove): step 1

goal (2 subgoals):
 1. ⟦P; P ⟶ Q⟧ ⟹ P
 2. ⟦P; P ⟶ Q⟧ ⟹ Q

-u-:%%-  *goals*       Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

holds trivially, *by assumption*

Proof by Assumption



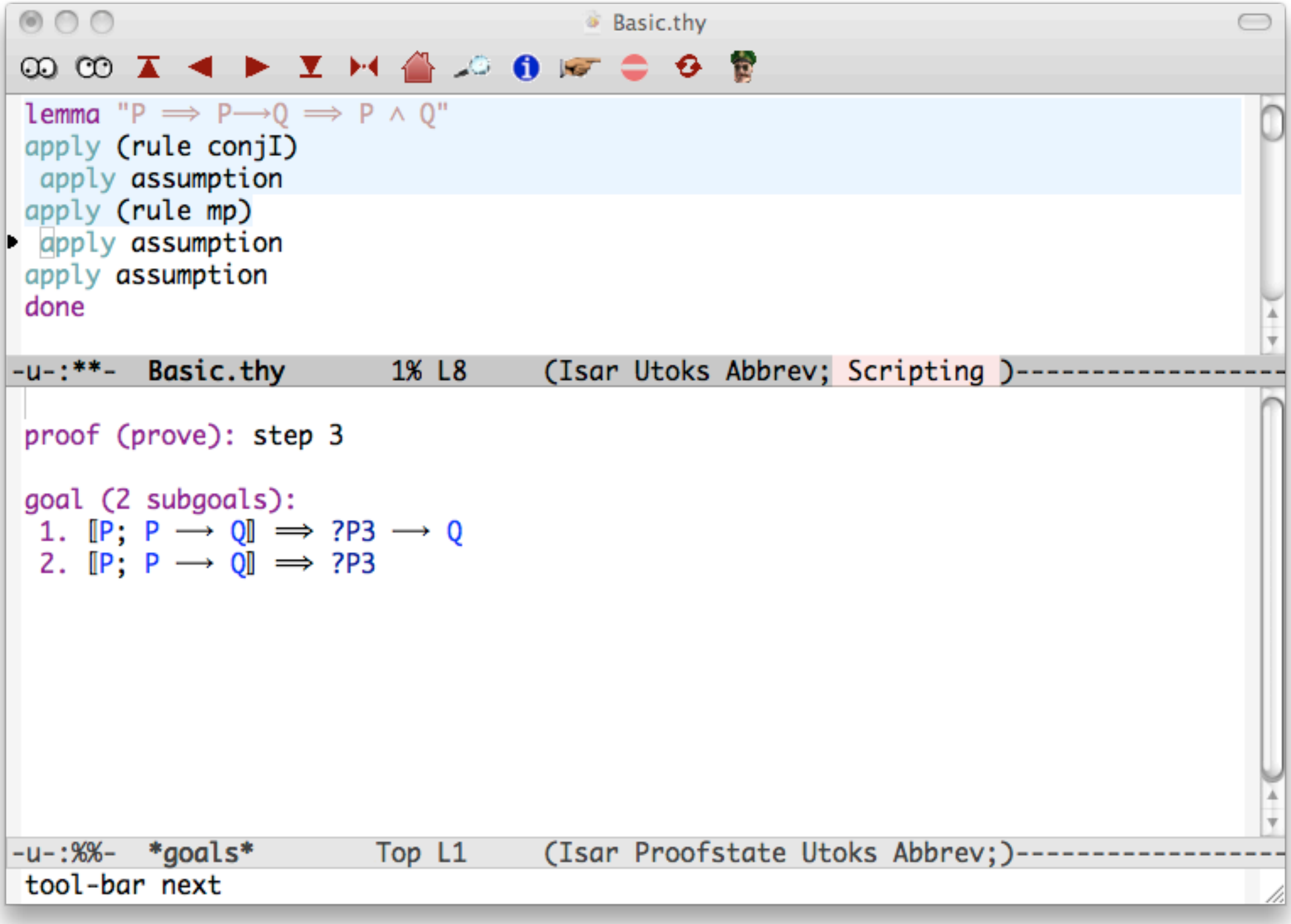
```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

```
-u-:**-  Basic.thy      1% L7      (Isar Utoks Abbrev; Scripting )------------------
```

```
proof (prove): step 2

goal (1 subgoal):
 1. ⟦P; P ⟶ Q⟧ ⟹ Q
```

```
-u-:%%-  *goals*       Top L1      (Isar Proofstate Utoks Abbrev;)------------------
tool-bar next
```

Unknowns in Subgoals



Basic.thy

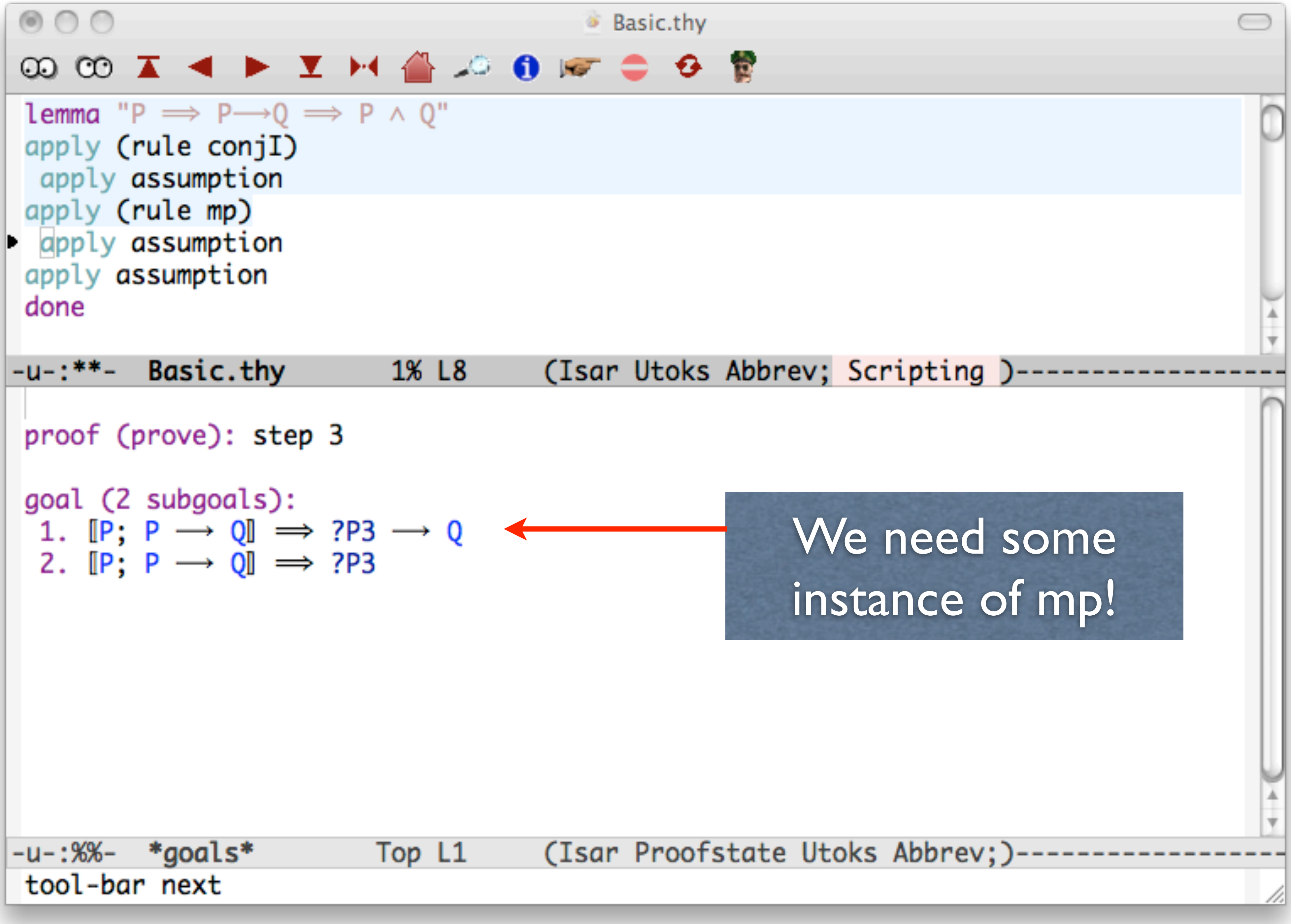
```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

```
-u-:**-  Basic.thy      1% L8      (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 3

goal (2 subgoals):
 1. ⟦P; P ⟶ Q⟧ ⟹ ?P3 ⟶ Q
 2. ⟦P; P ⟶ Q⟧ ⟹ ?P3
```

```
-u-:%%-  *goals*       Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Unknowns in Subgoals



```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

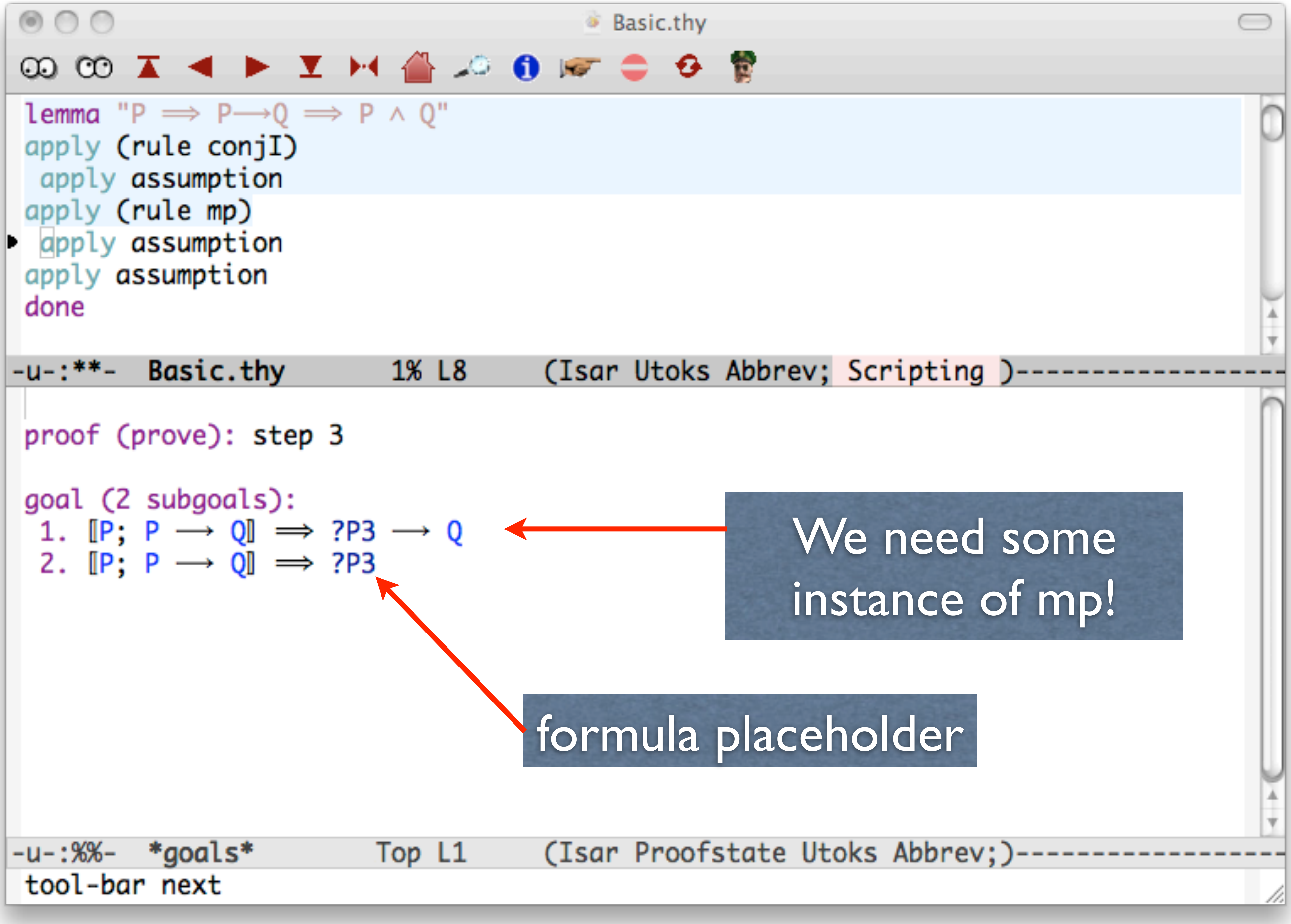
```
-u-:**-  Basic.thy       1% L8      (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 3

goal (2 subgoals):
 1. ⟦P; P ⟶ Q⟧ ⟹ ?P3 ⟶ Q
 2. ⟦P; P ⟶ Q⟧ ⟹ ?P3
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

We need some instance of mp!

Unknowns in Subgoals



```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

```
-u-:**-  Basic.thy      1% L8     (Isar Utoks Abbrev; Scripting )-----------------
```

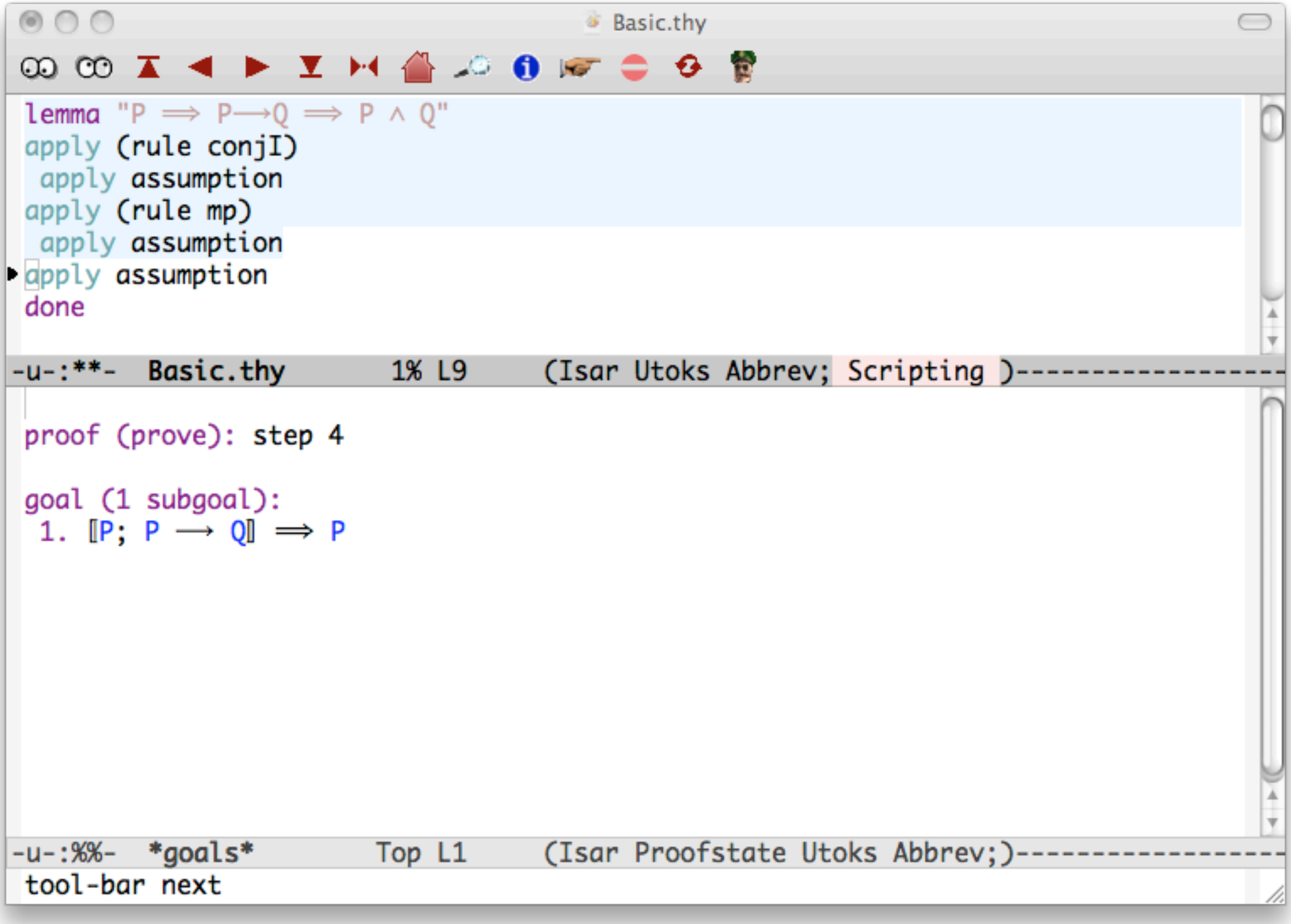
```
proof (prove): step 3

goal (2 subgoals):
 1. ⟦P; P ⟶ Q⟧ ⟹ ?P3 ⟶ Q
 2. ⟦P; P ⟶ Q⟧ ⟹ ?P3
```

```
-u-:%%-  *goals*      Top L1     (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

We need some instance of mp!

formula placeholder

Unknowns and Unification



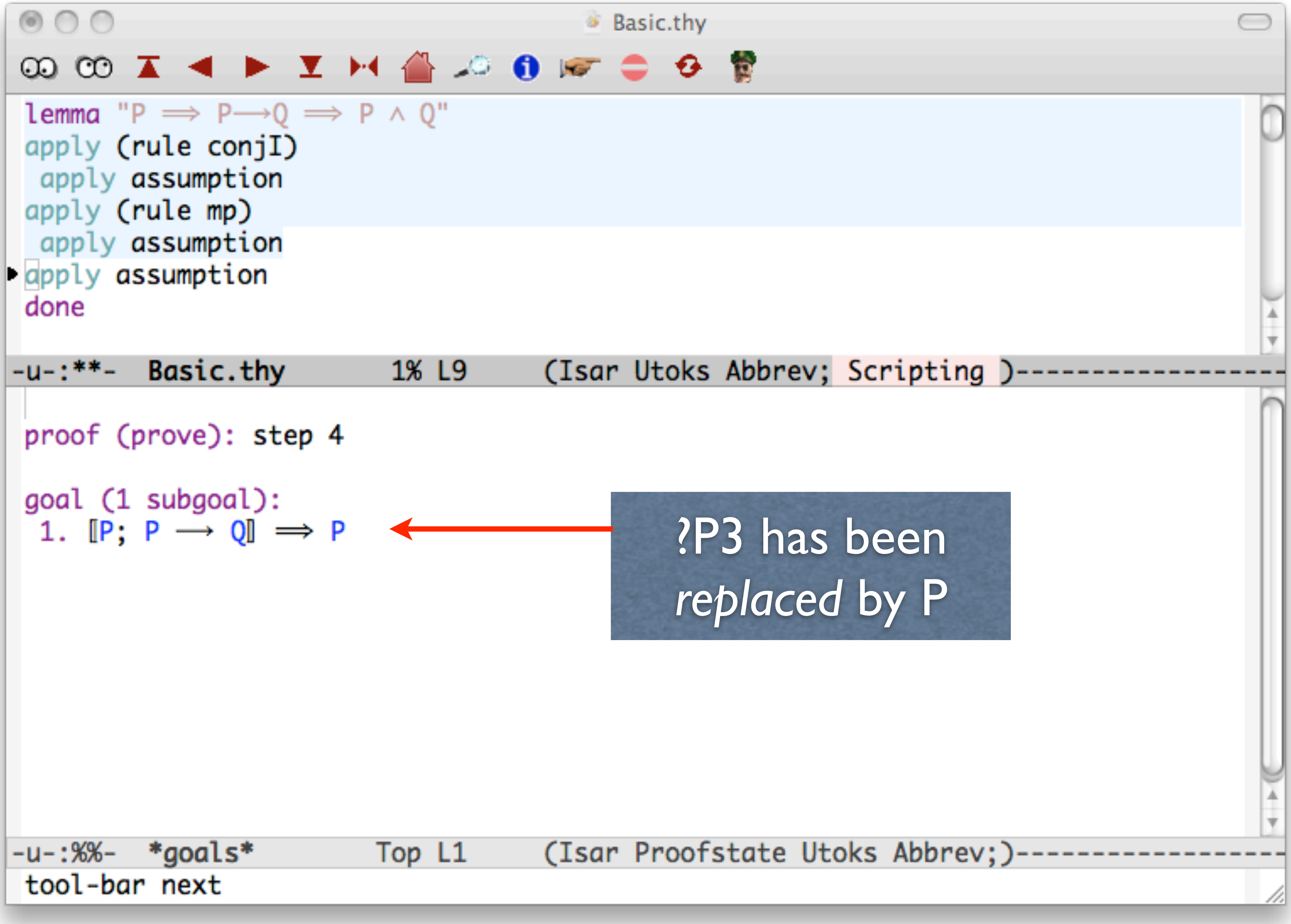
```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

```
-u-:**-  Basic.thy      1% L9      (Isar Utoks Abbrev; Scripting )
```

```
proof (prove): step 4

goal (1 subgoal):
 1. ⟦P; P ⟶ Q⟧ ⟹ P
```

```
-u-:%%-  *goals*        Top L1     (Isar Proofstate Utoks Abbrev;)
tool-bar next
```

Unknowns and Unification



```
lemma "P ⟹ P⟶Q ⟹ P ∧ Q"
apply (rule conjI)
 apply assumption
apply (rule mp)
 apply assumption
apply assumption
done
```

```
-u-:**-  Basic.thy      1% L9      (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 4

goal (1 subgoal):
 1. ⟦P; P ⟶ Q⟧ ⟹ P
```

```
-u-:%%-  *goals*       Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

?P3 has been *replaced* by P

Discharging Assumptions

Discharging Assumptions

$$\frac{\begin{array}{c}[P]\\ \vdots \\ Q\end{array}}{P \to Q}$$

```
(P ⇒ Q) ⇒ P⟶Q
```

Discharging Assumptions

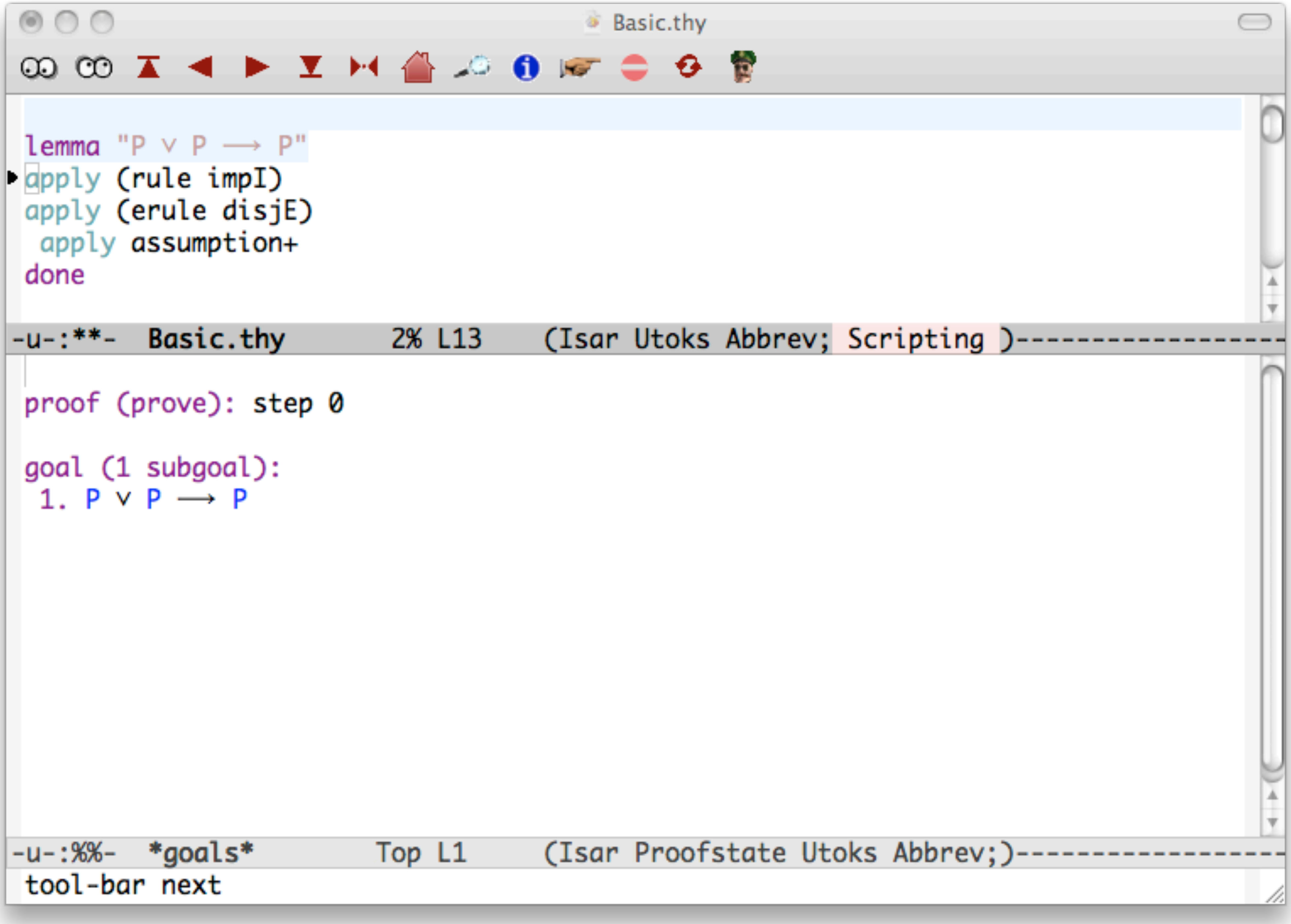
$$\frac{\begin{matrix}[P]\\ \vdots \\ Q\end{matrix}}{P \to Q}$$

```
(P ⇒ Q) ⇒ P⟶Q
```

$$\frac{P \lor Q \qquad \begin{matrix}[P]\\ \vdots\\ R\end{matrix} \qquad \begin{matrix}[Q]\\ \vdots\\ R\end{matrix}}{R}$$

```
⟦P ∨ Q; P⇒R; Q⇒R⟧ ⇒
          R
```

A Proof using Assumptions



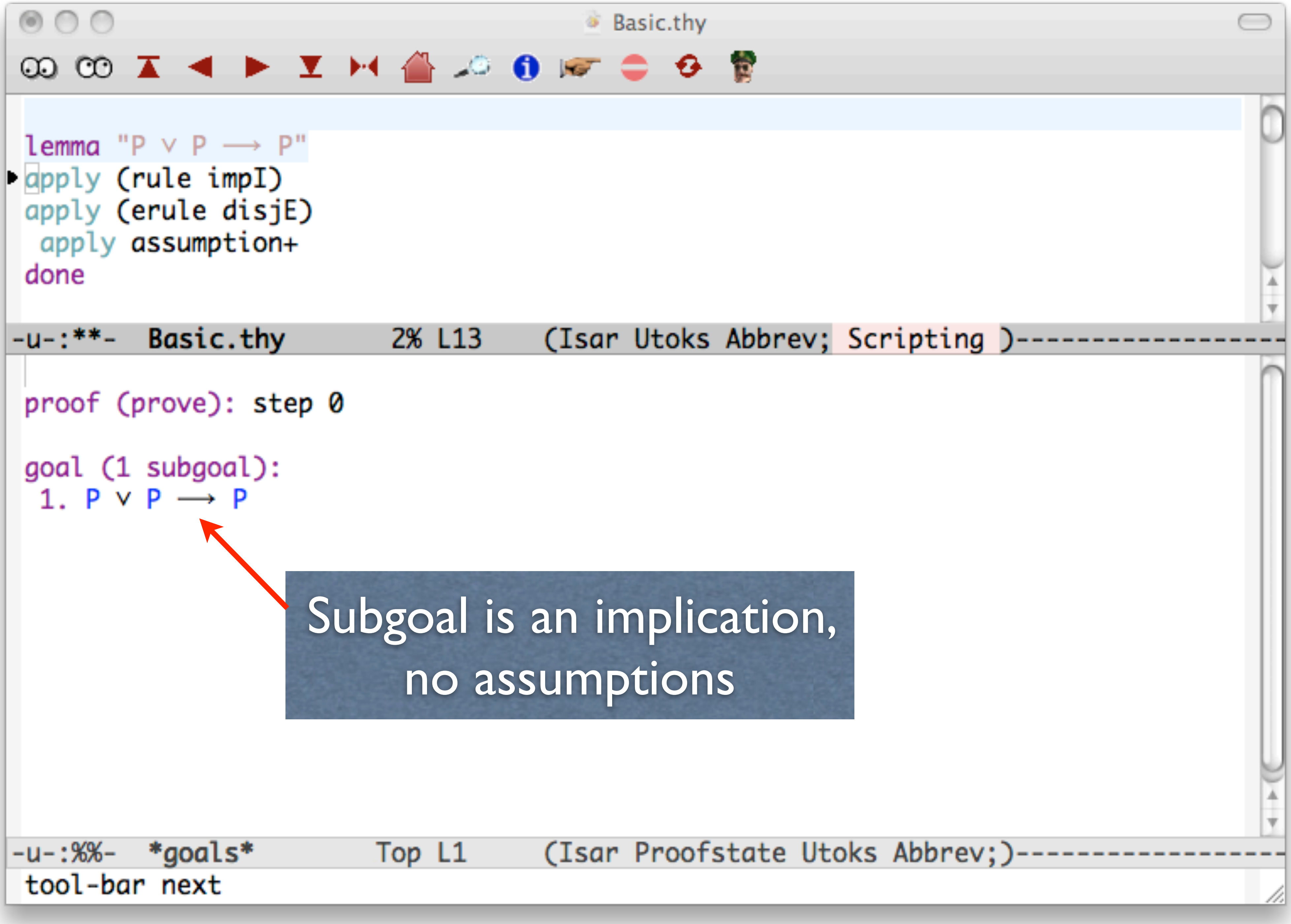
```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

```
-u-:**-  Basic.thy        2% L13     (Isar Utoks Abbrev; Scripting )------------------
```

```
proof (prove): step 0

goal (1 subgoal):
 1. P ∨ P ⟶ P
```

```
-u-:%%-  *goals*         Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

A Proof using Assumptions



```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

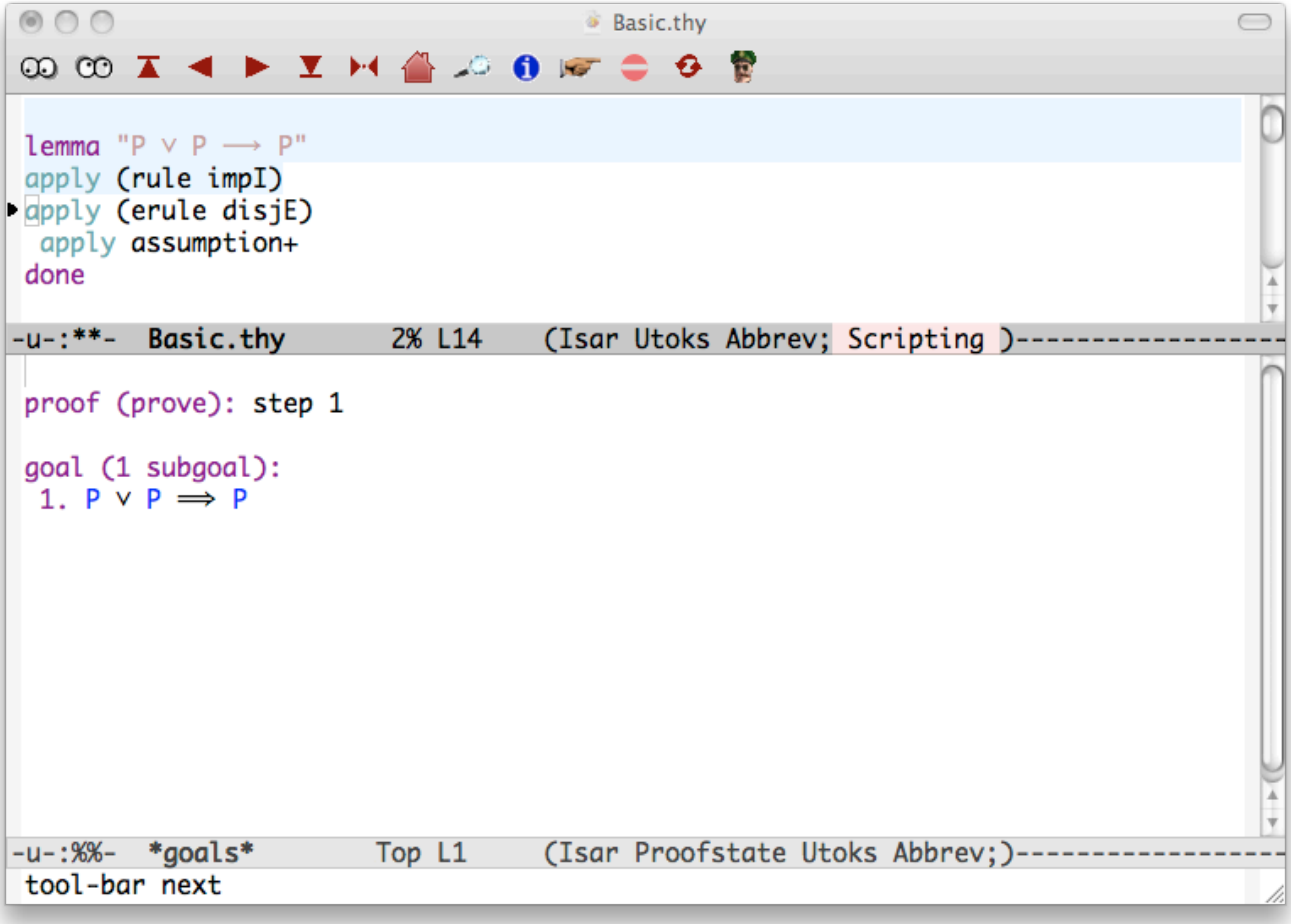
```
-u-:**-  Basic.thy        2% L13     (Isar Utoks Abbrev; Scripting )------------------
```

```
proof (prove): step 0

goal (1 subgoal):
 1. P ∨ P ⟶ P
```

```
-u-:%%-  *goals*         Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Subgoal is an implication, no assumptions

After Implies-Introduction



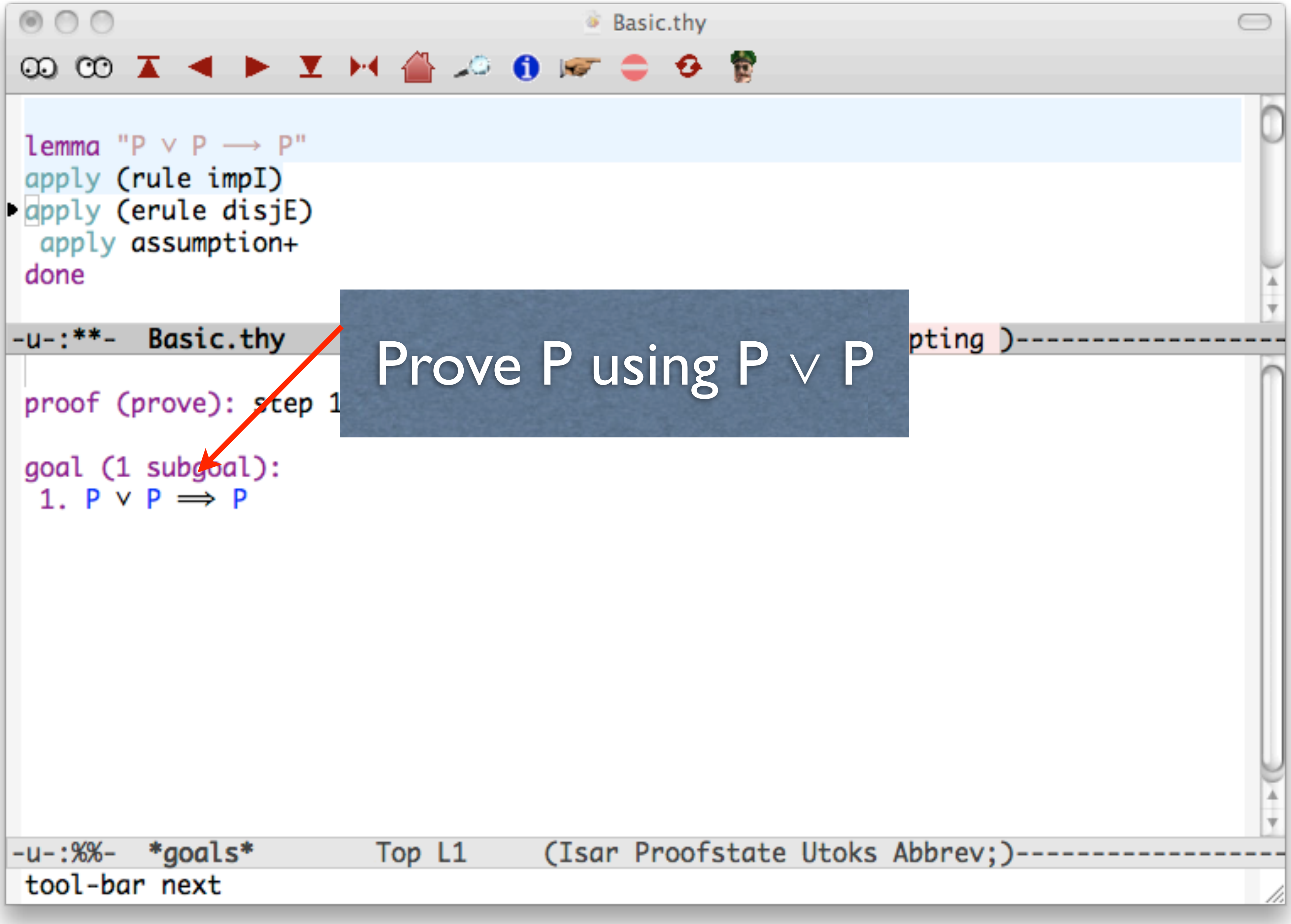
```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

```
-u-:**-  Basic.thy       2% L14     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 1

goal (1 subgoal):
 1. P ∨ P ⟹ P
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

After Implies-Introduction



```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

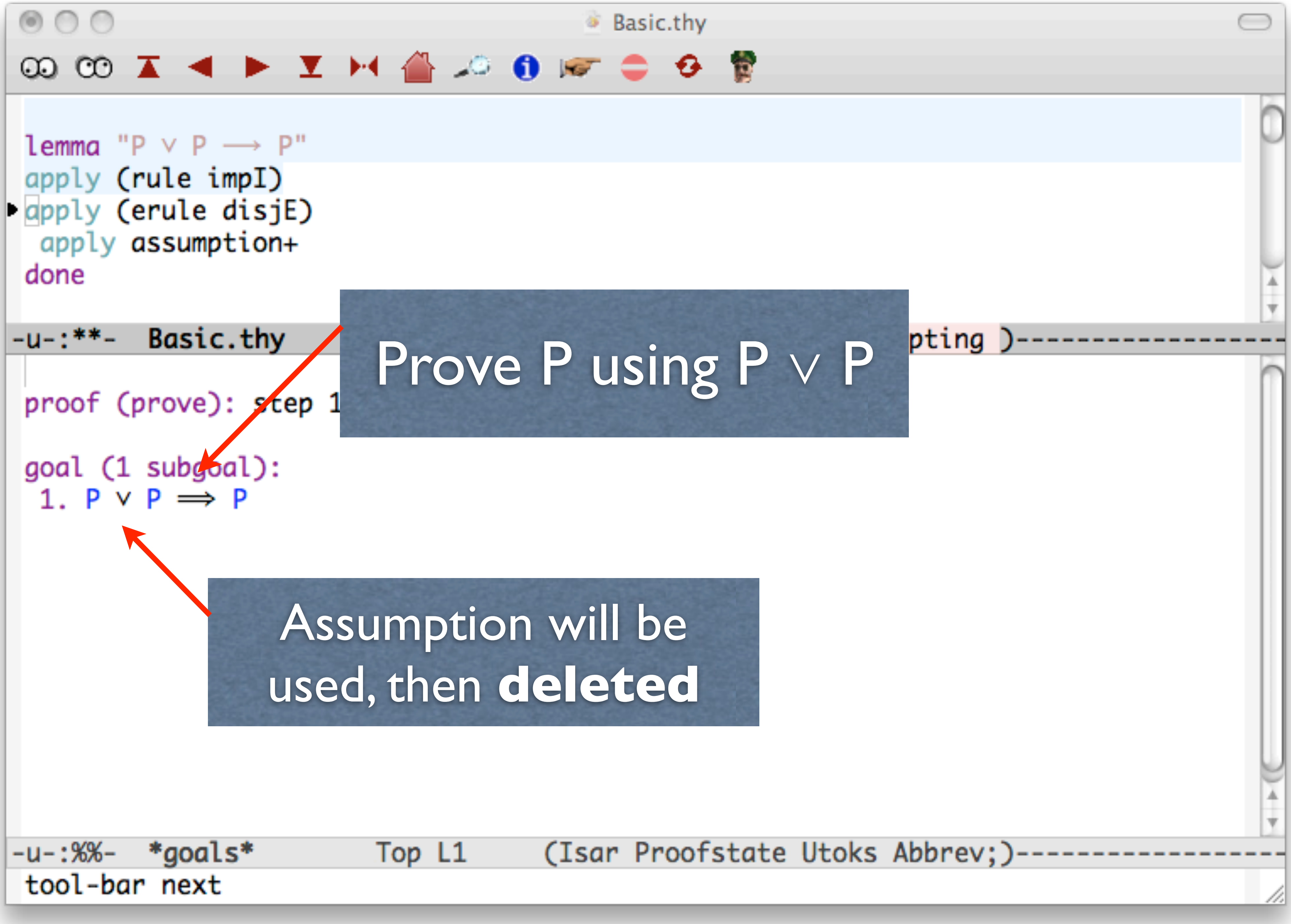
```
-u-:**-  Basic.thy
```

```
proof (prove): step 1

goal (1 subgoal):
 1. P ∨ P ⟹ P
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Prove P using P ∨ P

After Implies-Introduction



```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

```
-u-:**-  Basic.thy                                                    pting )-----------------
```

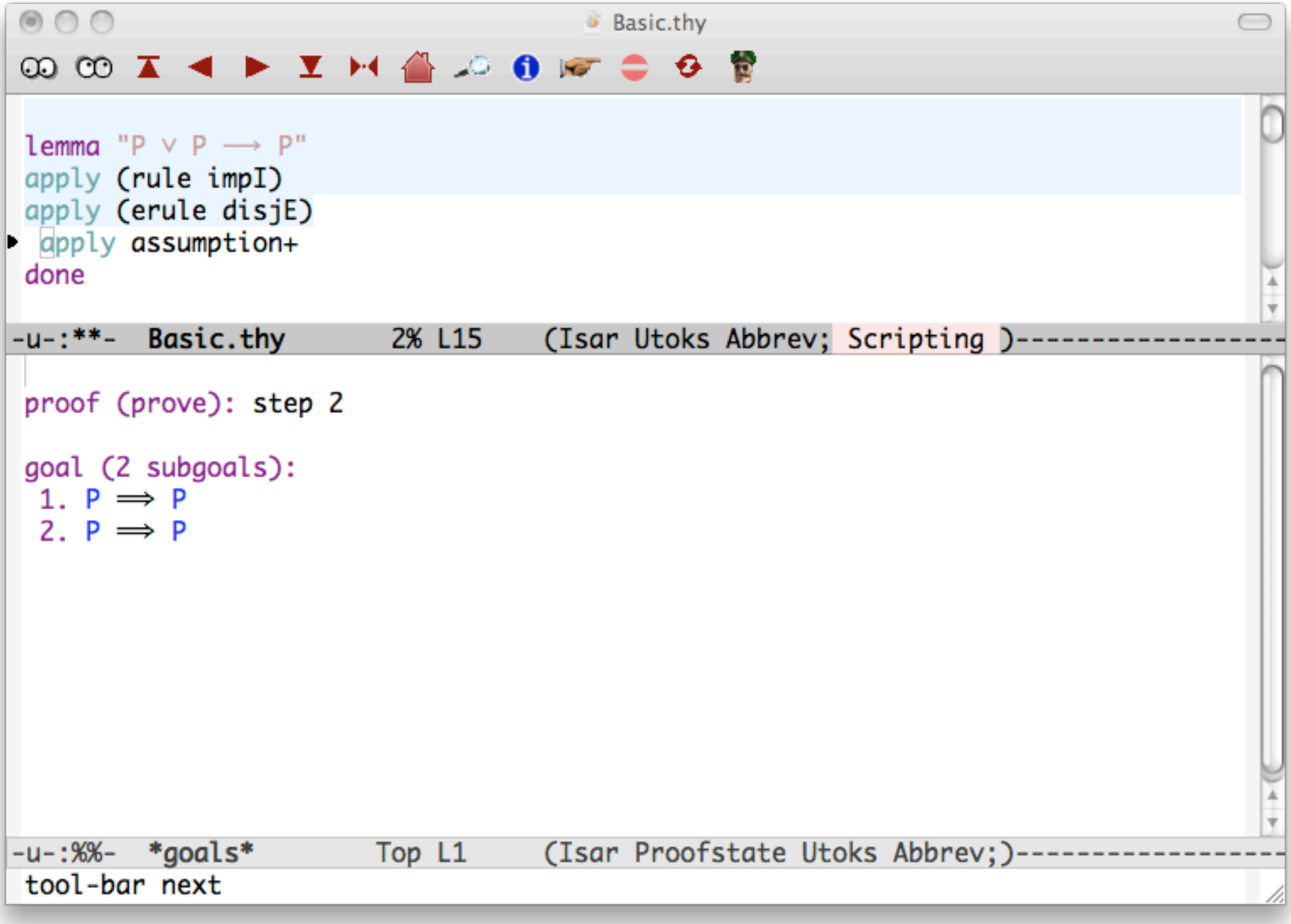
```
proof (prove): step 1

goal (1 subgoal):
 1. P ∨ P ⟹ P
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Prove P using P ∨ P

Assumption will be used, then **deleted**

Disjunction Elimination



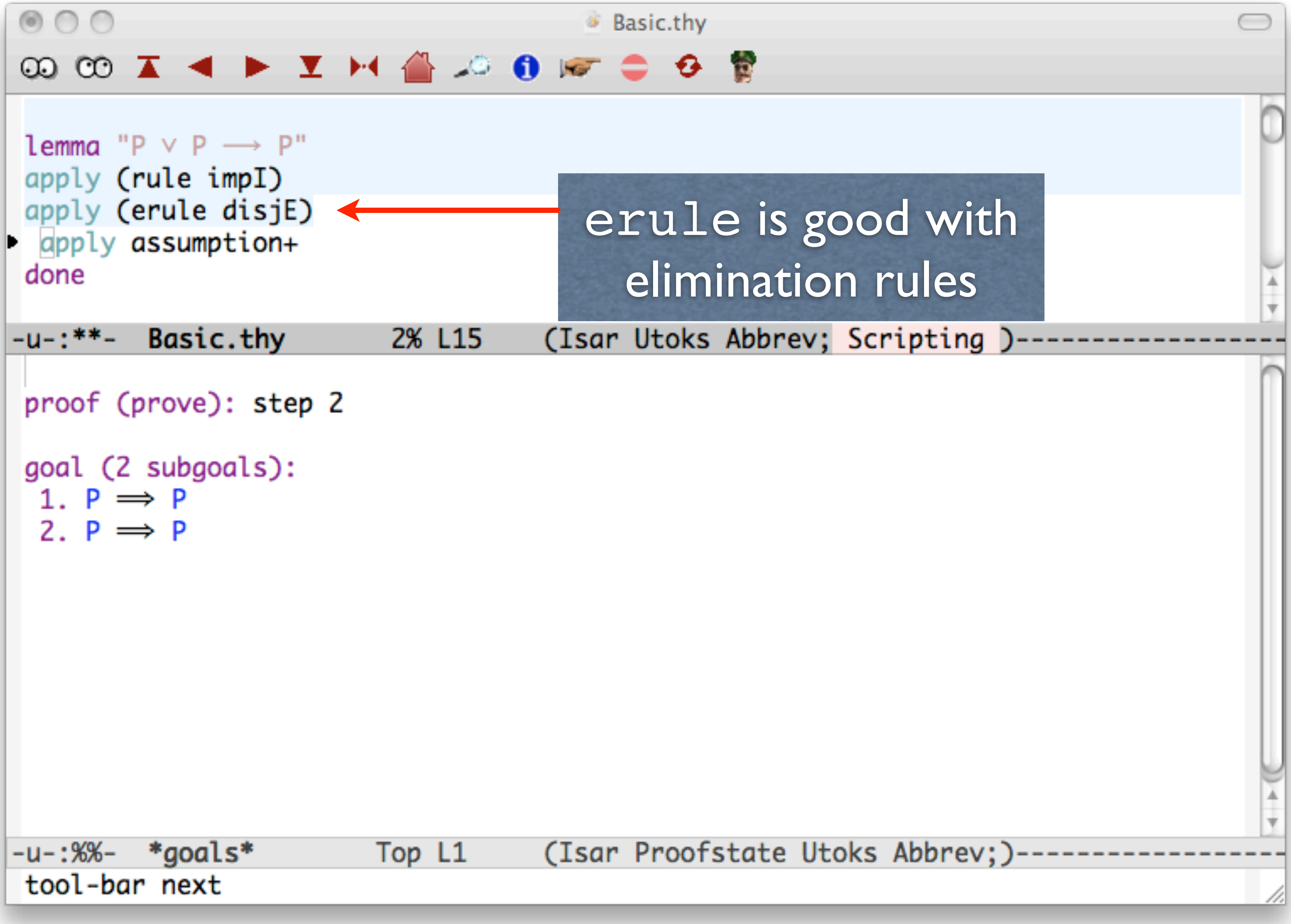
```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

```
-u-:**-  Basic.thy      2% L15     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 2

goal (2 subgoals):
 1. P ⟹ P
 2. P ⟹ P
```

```
-u-:%%-  *goals*       Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Disjunction Elimination



```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

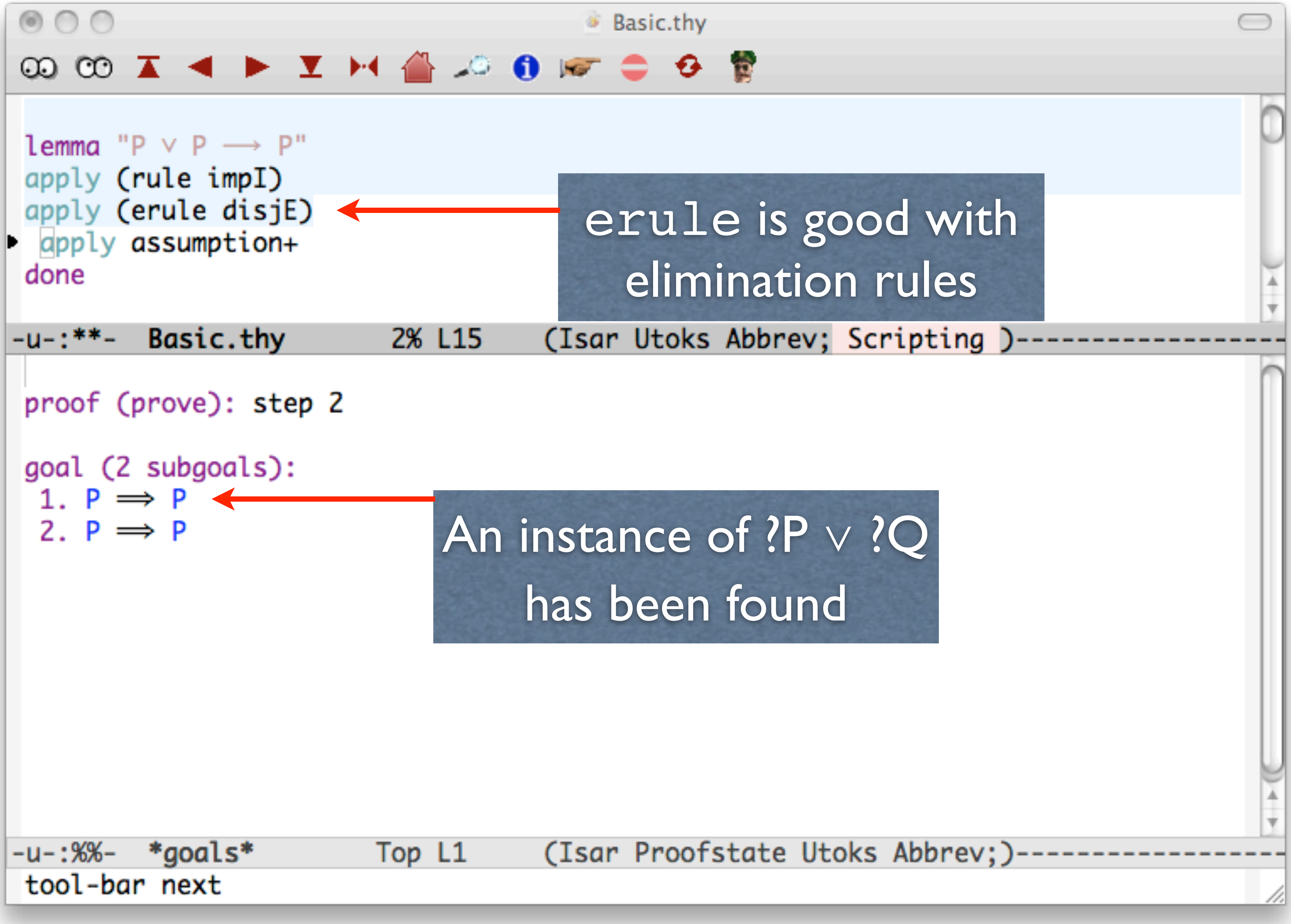
erule is good with elimination rules

```
-u-:**-  Basic.thy       2% L15     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 2

goal (2 subgoals):
 1. P ⟹ P
 2. P ⟹ P
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Disjunction Elimination



```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
apply assumption+
done
```

```
-u-:**-  Basic.thy       2% L15     (Isar Utoks Abbrev; Scripting )------------------
```

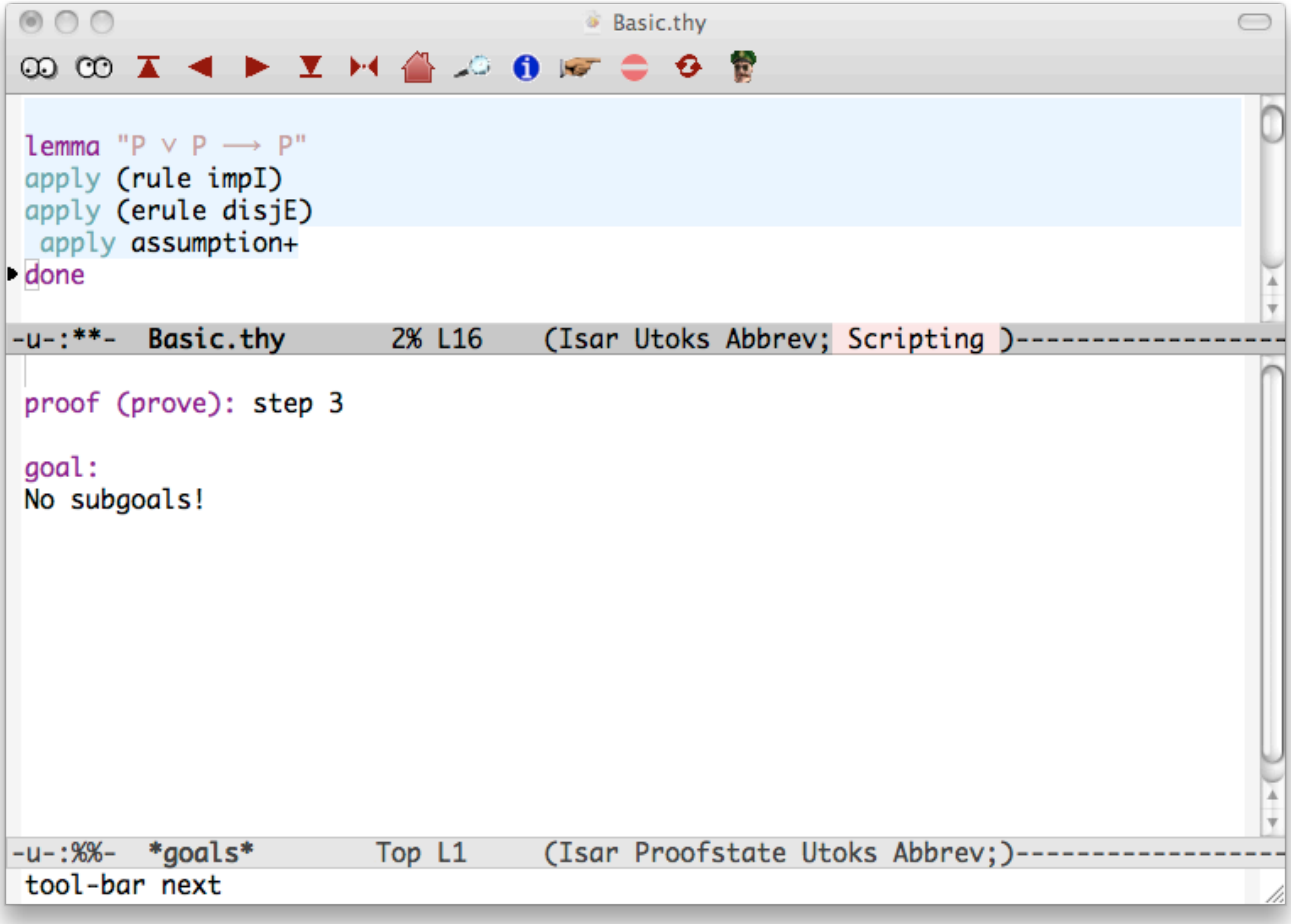
```
proof (prove): step 2

goal (2 subgoals):
 1. P ⟹ P
 2. P ⟹ P
```

```
-u-:%%-  *goals*        Top L1     (Isar Proofstate Utoks Abbrev;)------------------
tool-bar next
```

erule is good with elimination rules

An instance of ?P ∨ ?Q has been found

The Final Step



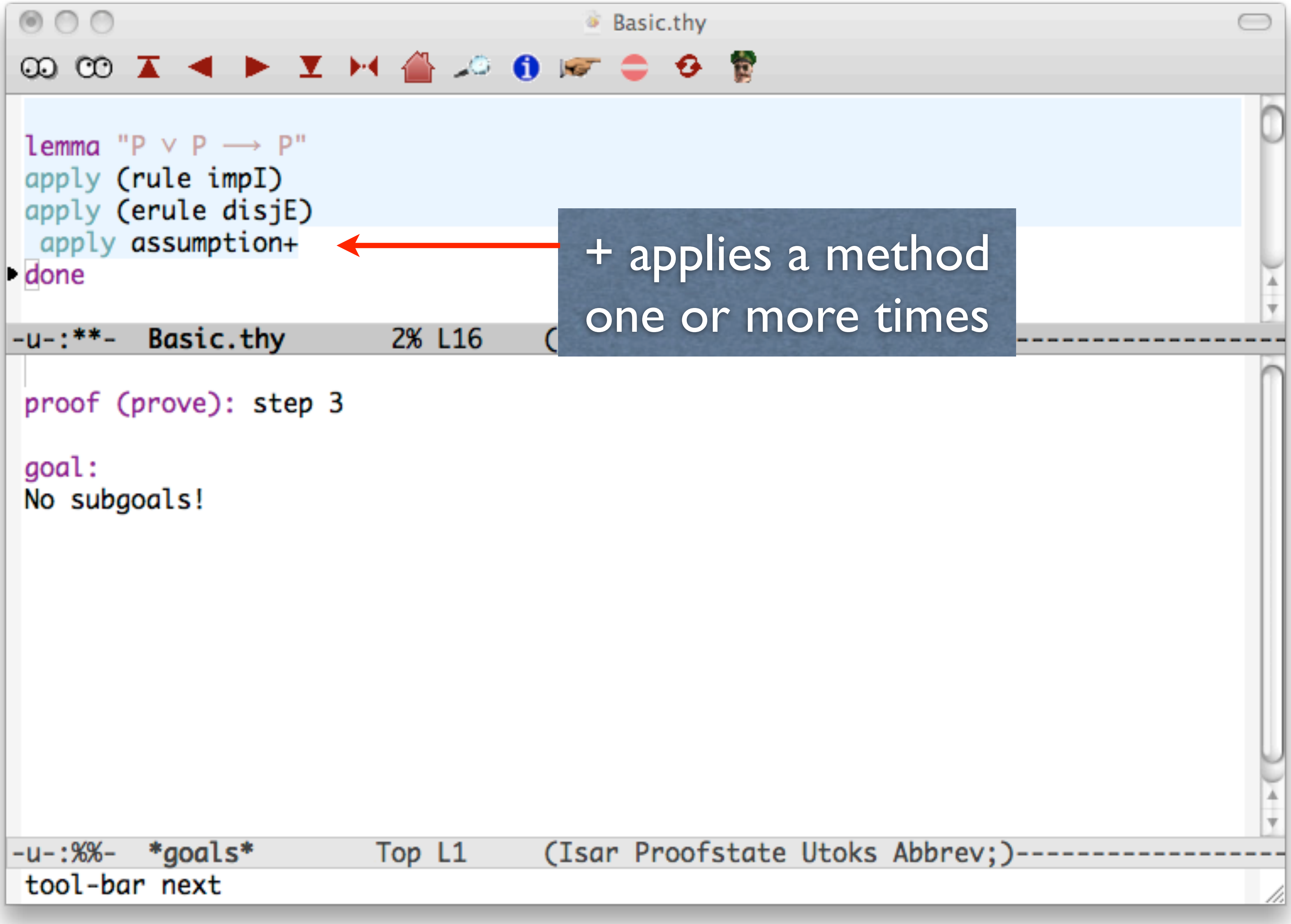
```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

```
-u-:**-  Basic.thy       2% L16     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 3

goal:
No subgoals!
```

```
-u-:%%-  *goals*        Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

The Final Step



```
lemma "P ∨ P ⟶ P"
apply (rule impI)
apply (erule disjE)
 apply assumption+
done
```

+ applies a method one or more times

```
-u-:**-  Basic.thy      2% L16     (
```

```
proof (prove): step 3

goal:
No subgoals!
```

```
-u-:%%-  *goals*       Top L1      (Isar Proofstate Utoks Abbrev;)-----------------
tool-bar next
```

Quantifiers

Quantifiers

$$\frac{P(t)}{\exists x.\, P(x)}$$

`P(x) ⇒ ∃x.P(x)`

Quantifiers

$$\frac{P(t)}{\exists x.\, P(x)}$$

```
P(x) ⇒ ∃x.P(x)
```

$$\frac{\exists x.\, P(x) \qquad \begin{array}{c}[P(x)]\\ \vdots \\ Q\end{array}}{Q}$$

```
⟦∃x.P(x); ⋀x.P(x)⇒Q⟧ ⇒
           Q
```

Quantifiers

$$\frac{P(t)}{\exists x.\, P(x)}$$

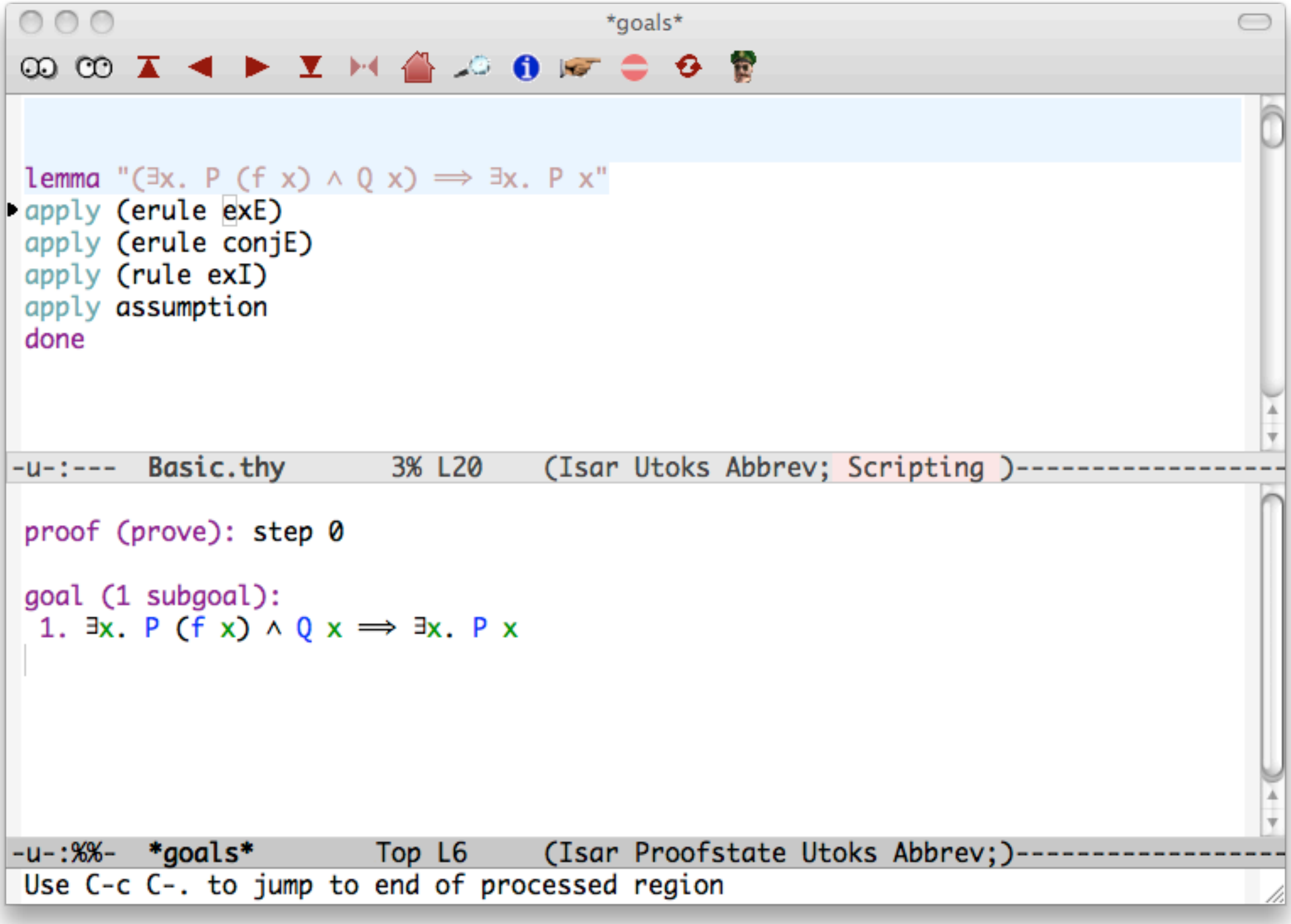
```
P(x) ⇒ ∃x.P(x)
```

$$\frac{\exists x.\, P(x) \qquad \begin{array}{c}[P(x)]\\ \vdots \\ Q\end{array}}{Q}$$

```
⟦∃x.P(x); ⋀x.P(x)⇒Q⟧ ⇒
    Q
```

meta-universal quantifier
states the variable condition

A Tiny Quantifier Proof



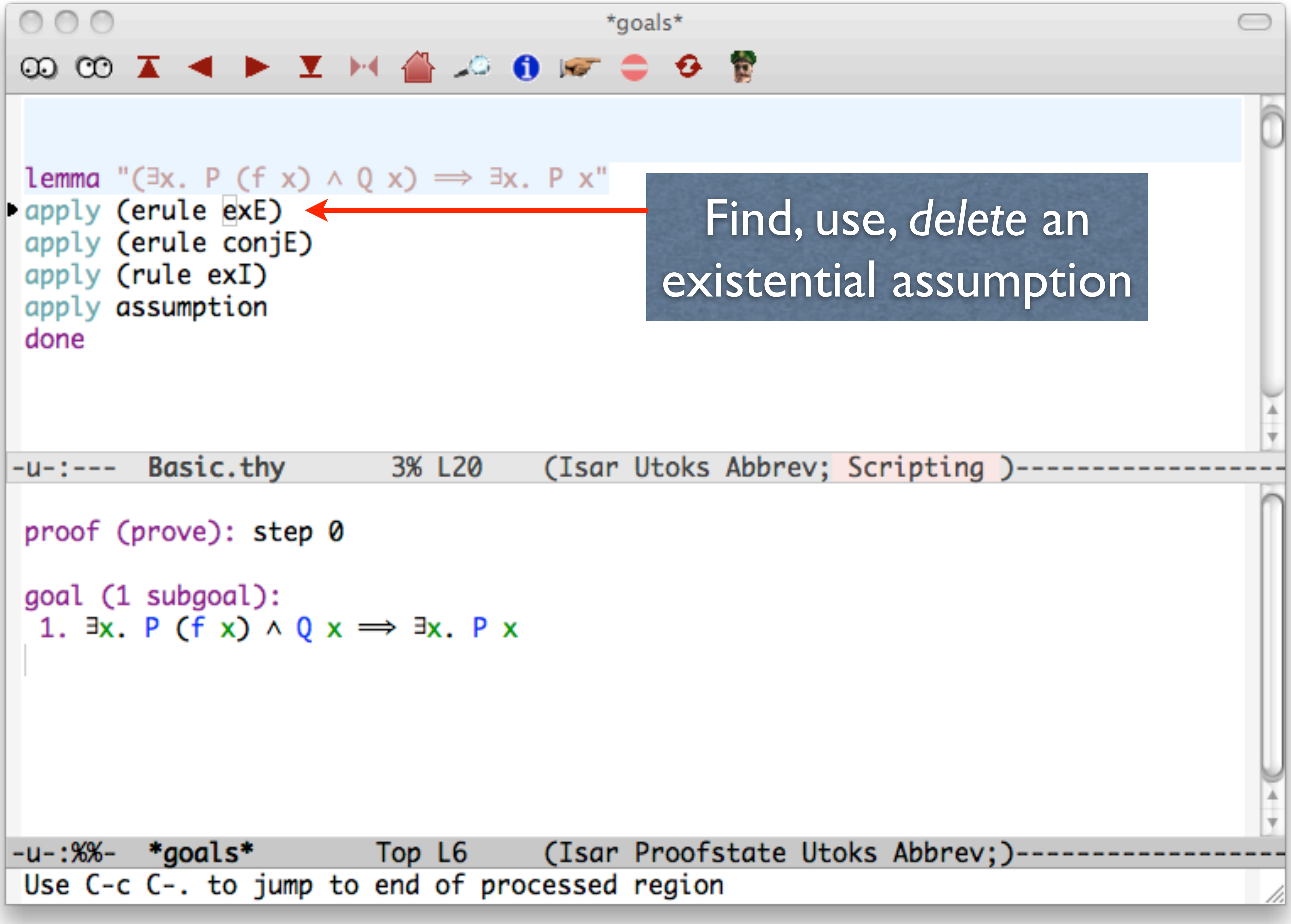
```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

```
-u-:---  Basic.thy       3% L20     (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 0

goal (1 subgoal):
 1. ∃x. P (f x) ∧ Q x ⟹ ∃x. P x
```

```
-u-:%%-  *goals*        Top L6      (Isar Proofstate Utoks Abbrev;)----------------
Use C-c C-. to jump to end of processed region
```

A Tiny Quantifier Proof



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

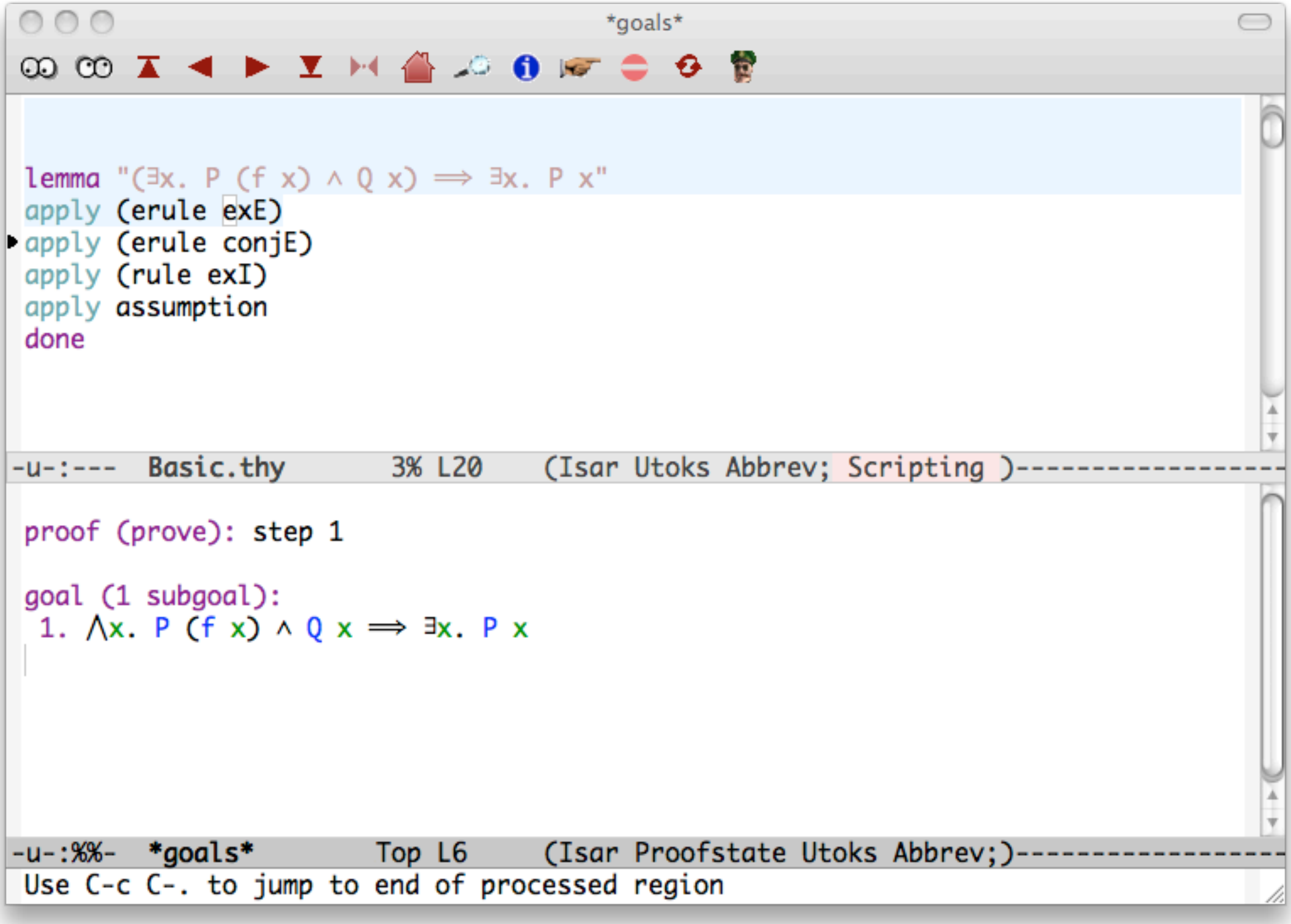
Find, use, *delete* an existential assumption

```
-u-:---  Basic.thy       3% L20     (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 0

goal (1 subgoal):
 1. ∃x. P (f x) ∧ Q x ⟹ ∃x. P x
```

```
-u-:%%-  *goals*         Top L6     (Isar Proofstate Utoks Abbrev;)----------------
Use C-c C-. to jump to end of processed region
```

Conjunction Elimination



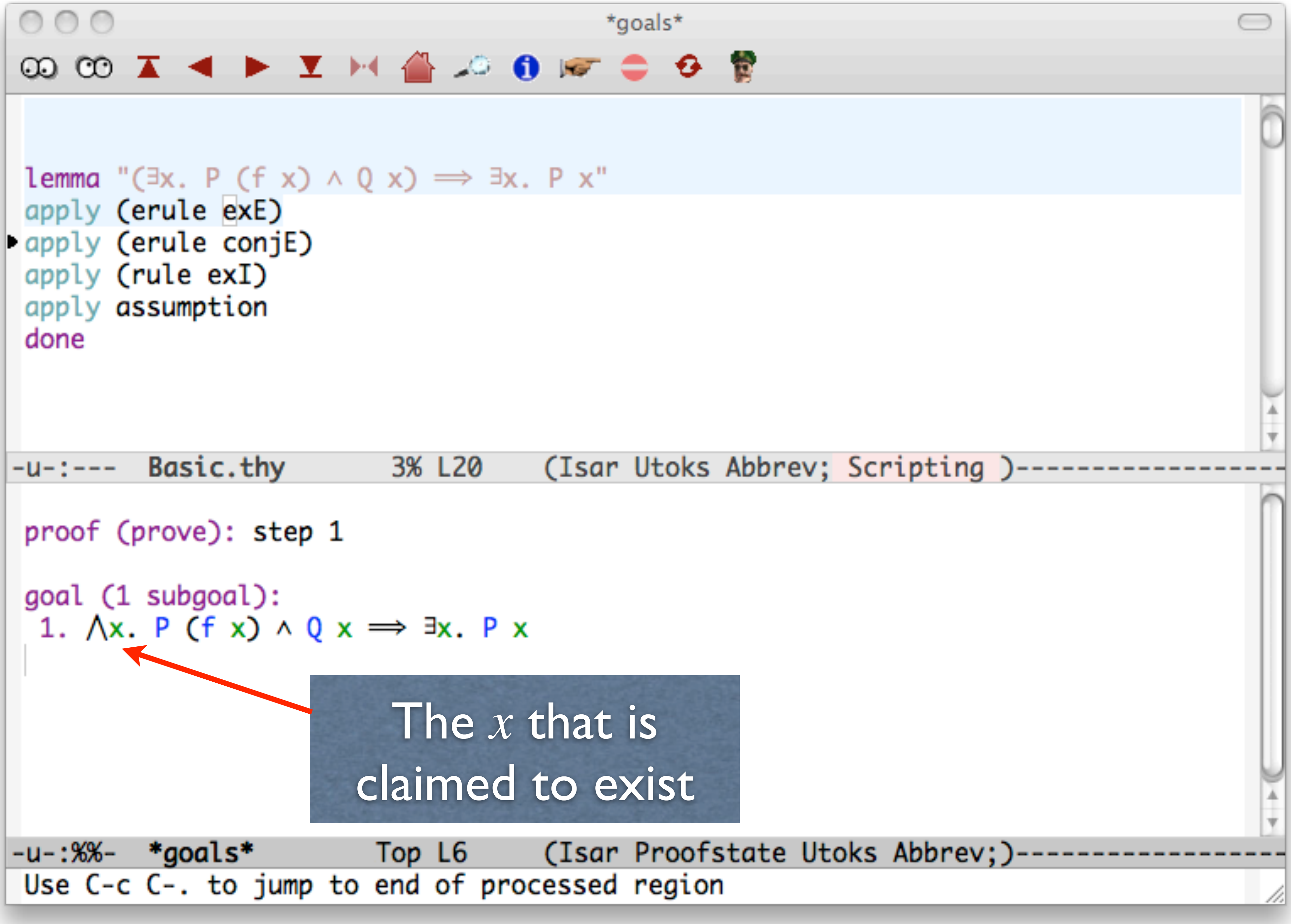
```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

```
-u-:---  Basic.thy     3% L20    (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 1

goal (1 subgoal):
 1. ⋀x. P (f x) ∧ Q x ⟹ ∃x. P x
```

```
-u-:%%-  *goals*      Top L6     (Isar Proofstate Utoks Abbrev;)----------------
Use C-c C-. to jump to end of processed region
```

Conjunction Elimination



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

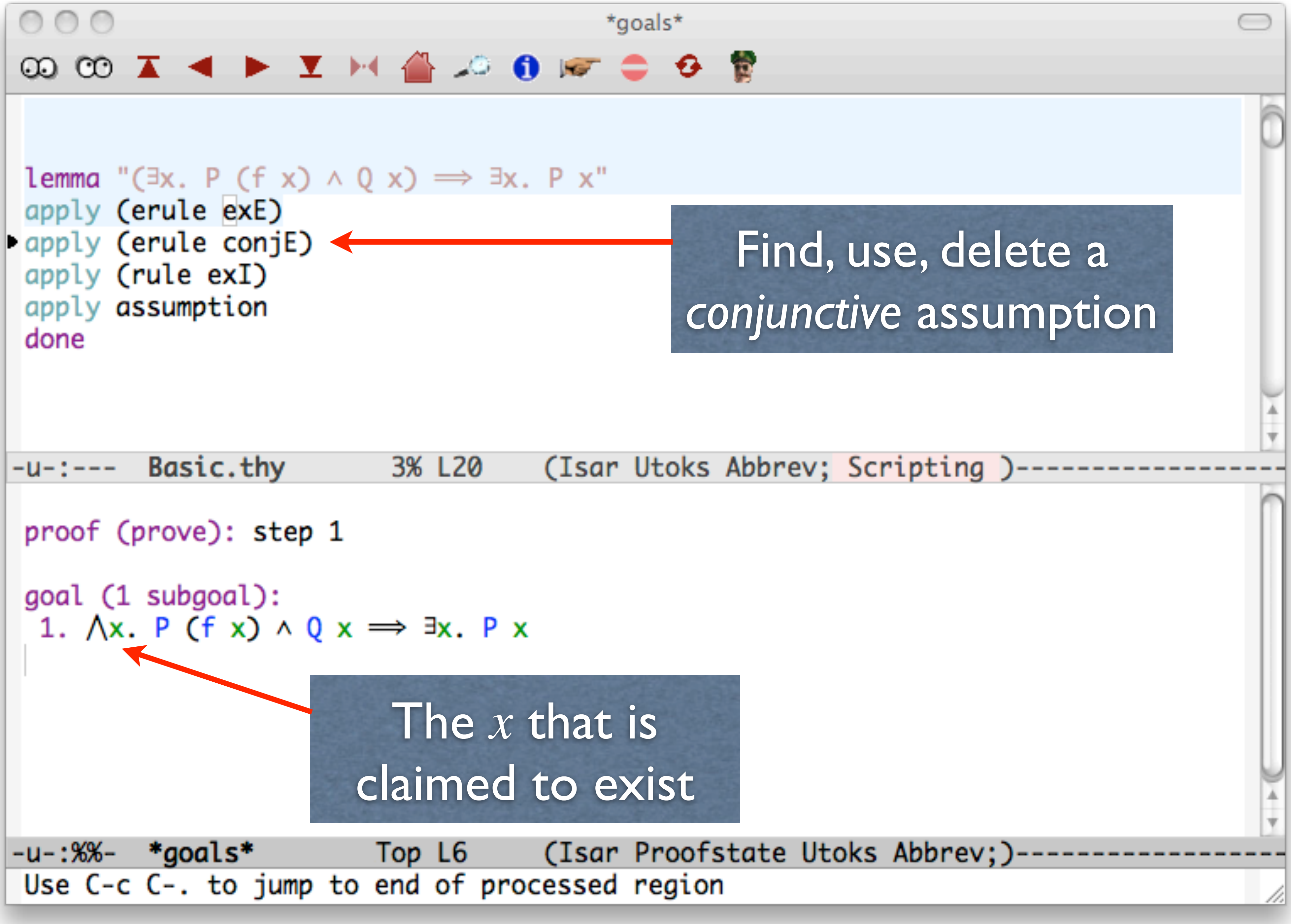
```
-u-:---  Basic.thy      3% L20     (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 1

goal (1 subgoal):
 1. ⋀x. P (f x) ∧ Q x ⟹ ∃x. P x
```

```
-u-:%%-  *goals*        Top L6     (Isar Proofstate Utoks Abbrev;)----------------
Use C-c C-. to jump to end of processed region
```

The x that is claimed to exist

Conjunction Elimination



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

Find, use, delete a *conjunctive* assumption

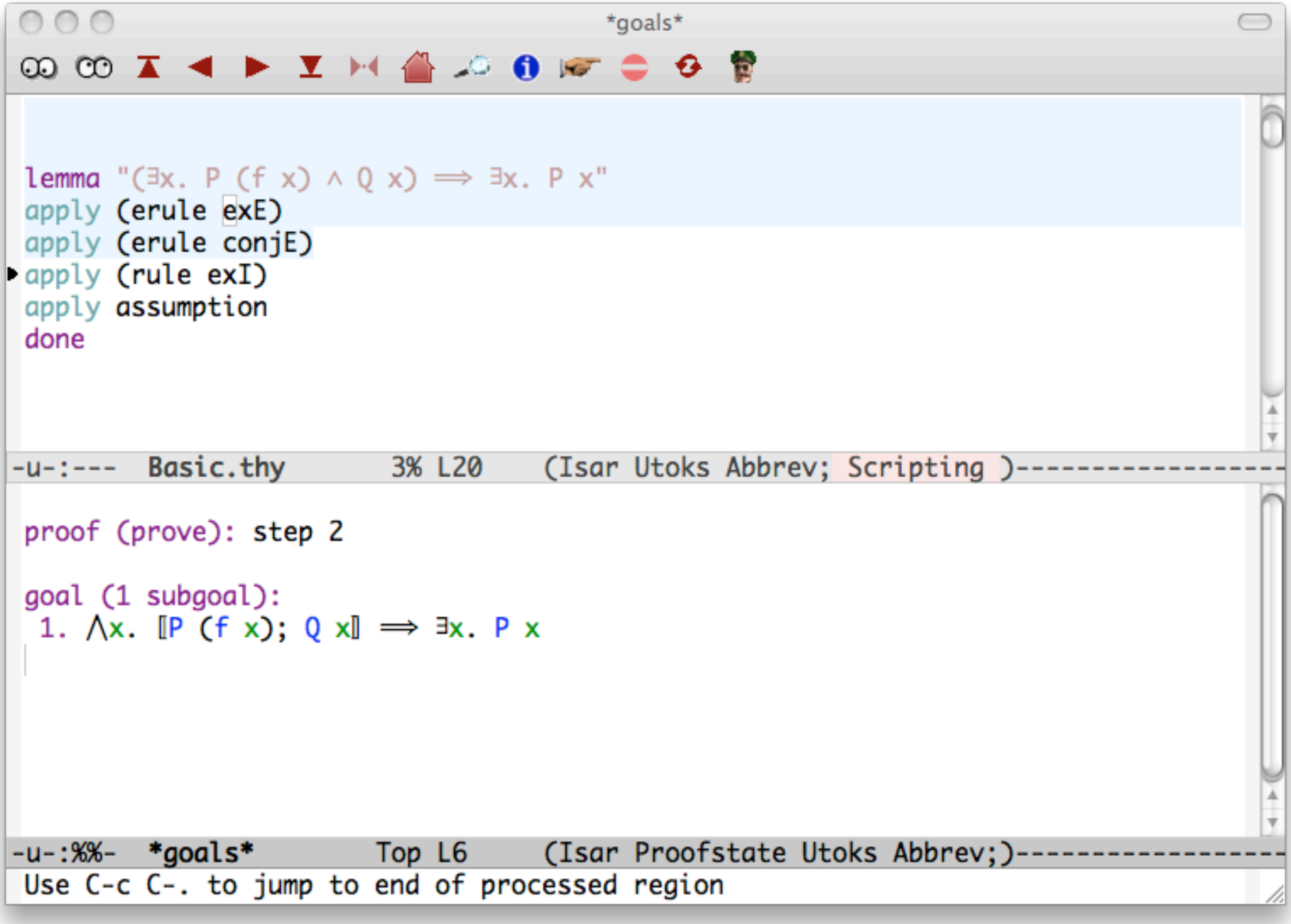
```
-u-:---  Basic.thy      3% L20     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 1

goal (1 subgoal):
 1. ⋀x. P (f x) ∧ Q x ⟹ ∃x. P x
```

The x that is claimed to exist

```
-u-:%%-  *goals*        Top L6     (Isar Proofstate Utoks Abbrev;)-----------------
Use C-c C-. to jump to end of processed region
```

Now for ∃-Introduction



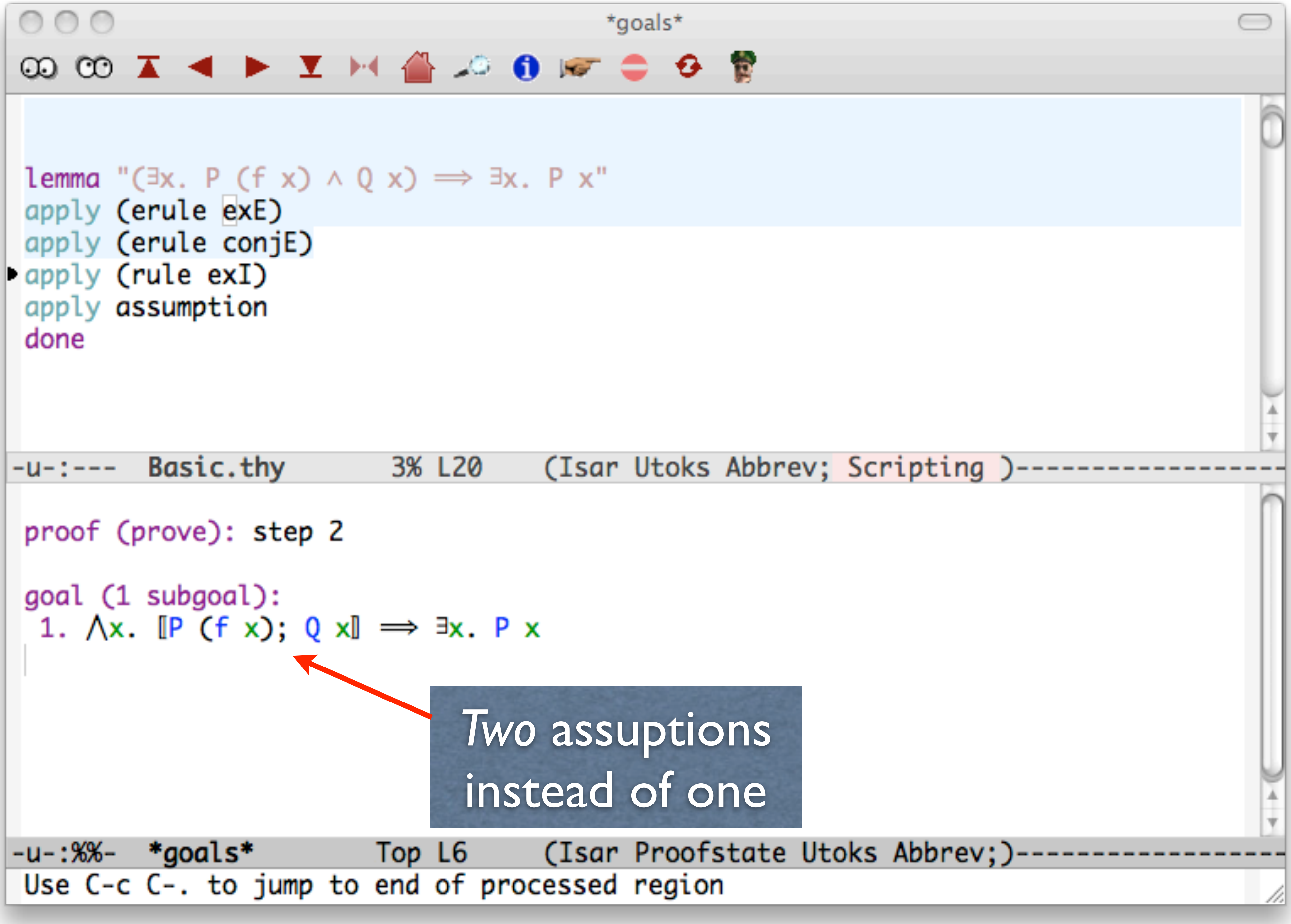
```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

```
-u-:---  Basic.thy       3% L20     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 2

goal (1 subgoal):
 1. ⋀x. ⟦P (f x); Q x⟧ ⟹ ∃x. P x
```

```
-u-:%%-  *goals*        Top L6      (Isar Proofstate Utoks Abbrev;)-----------------
Use C-c C-. to jump to end of processed region
```

Now for ∃-Introduction



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

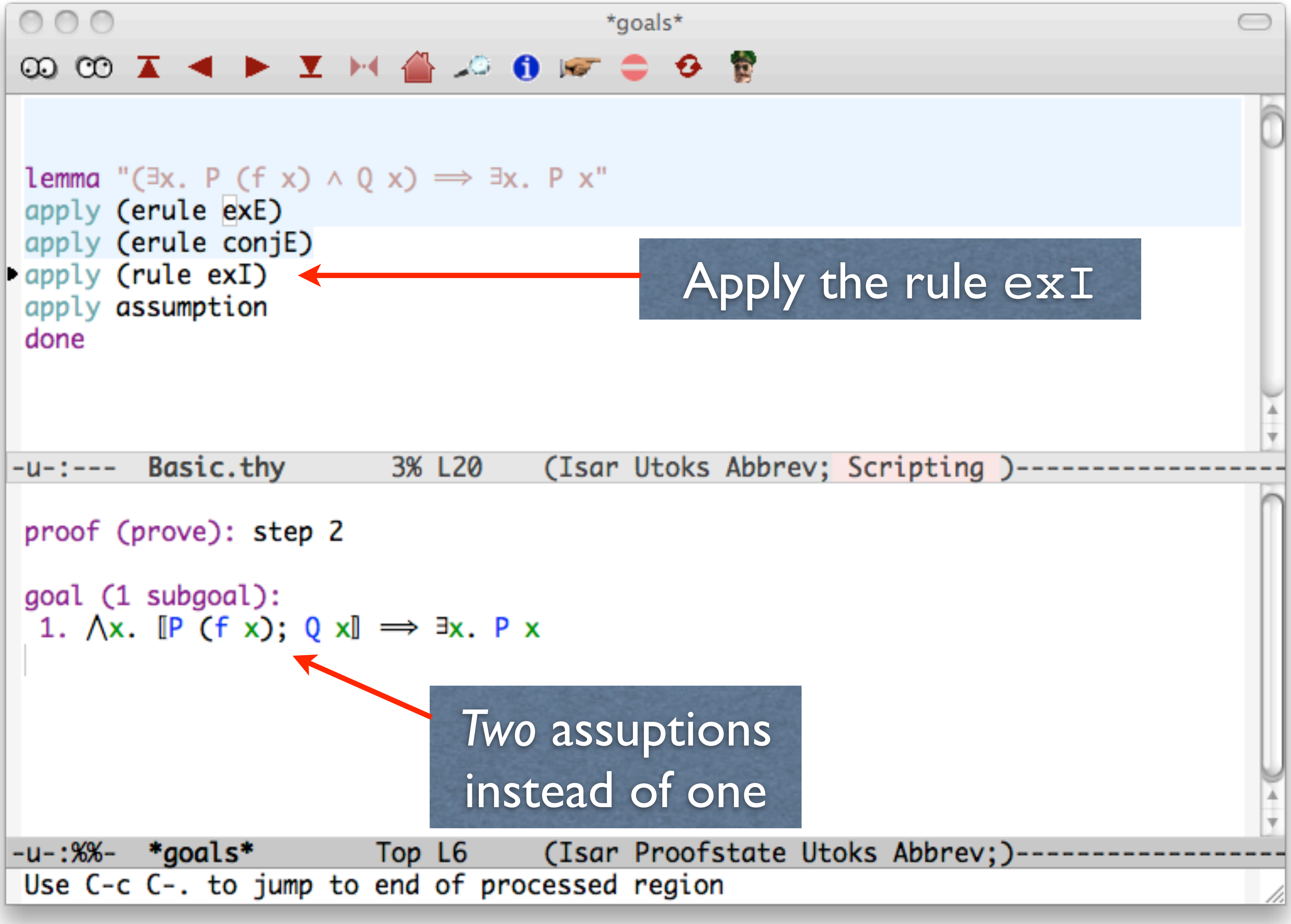
```
-u-:---  Basic.thy      3% L20     (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 2

goal (1 subgoal):
 1. ⋀x. ⟦P (f x); Q x⟧ ⟹ ∃x. P x
```

```
-u-:%%-  *goals*       Top L6      (Isar Proofstate Utoks Abbrev;)-----------------
Use C-c C-. to jump to end of processed region
```

Two assuptions instead of one

Now for ∃-Introduction



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

```
-u-:---  Basic.thy      3% L20     (Isar Utoks Abbrev; Scripting )-----------------
```

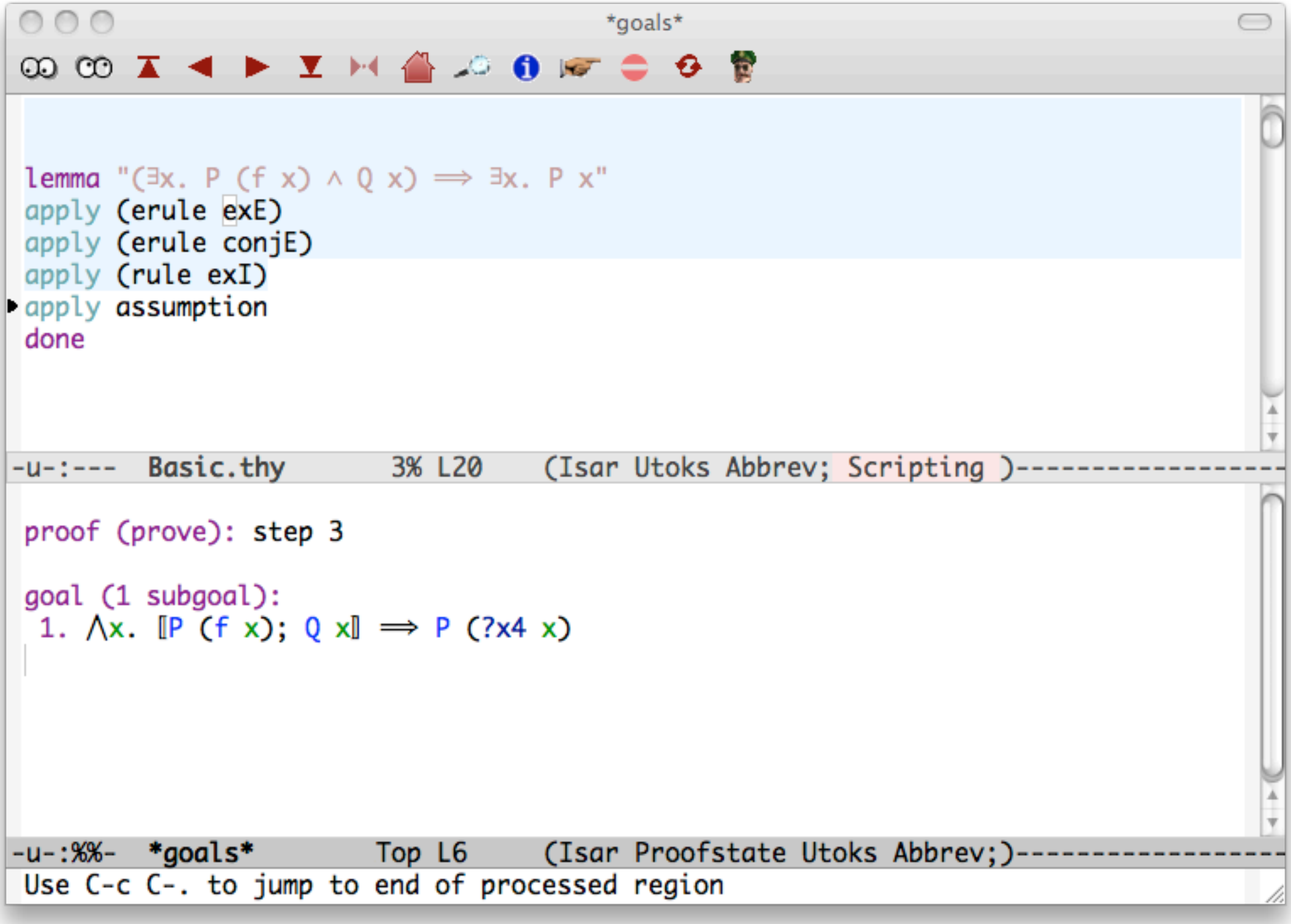
```
proof (prove): step 2

goal (1 subgoal):
 1. ⋀x. ⟦P (f x); Q x⟧ ⟹ ∃x. P x
```

```
-u-:%%-  *goals*        Top L6     (Isar Proofstate Utoks Abbrev;)-----------------
Use C-c C-. to jump to end of processed region
```

Apply the rule exI

Two assuptions instead of one

An Unknown for the Witness



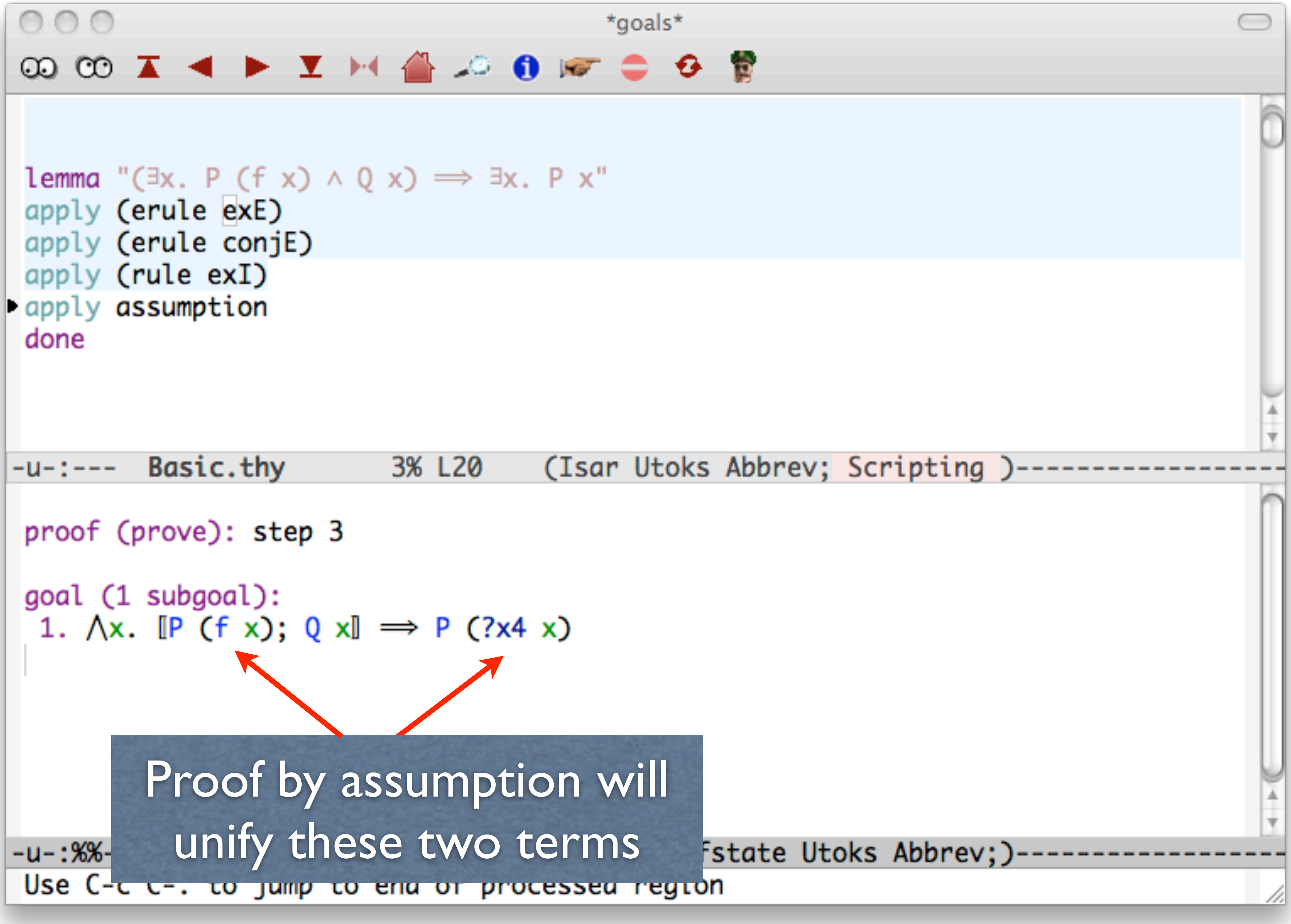
```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

```
-u-:---  Basic.thy       3% L20     (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 3

goal (1 subgoal):
 1. ⋀x. ⟦P (f x); Q x⟧ ⟹ P (?x4 x)
```

```
-u-:%%-  *goals*        Top L6      (Isar Proofstate Utoks Abbrev;)-----------------
Use C-c C-. to jump to end of processed region
```

An Unknown for the Witness



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

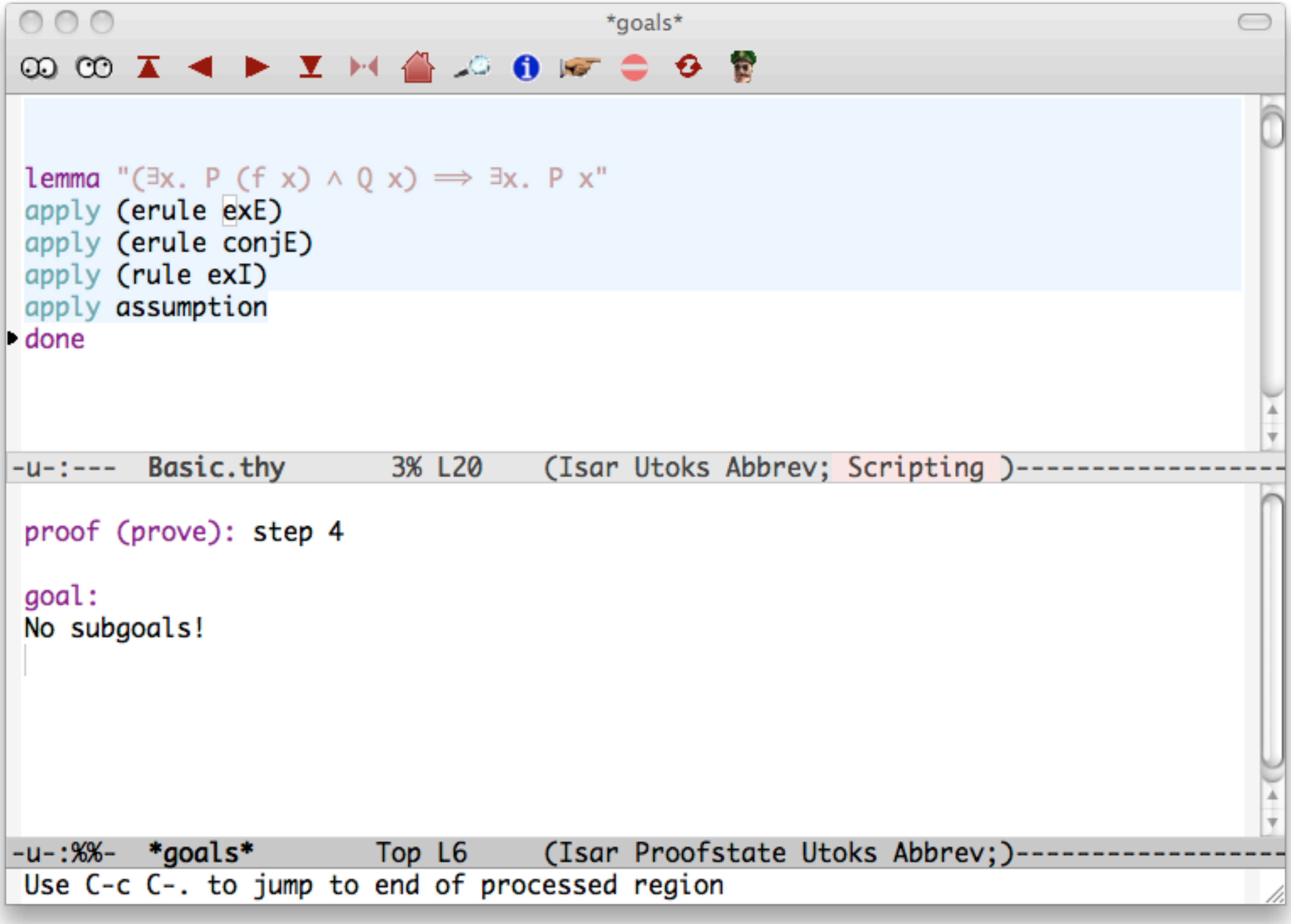
```
-u-:---  Basic.thy       3% L20     (Isar Utoks Abbrev; Scripting )-----------------
```

```
proof (prove): step 3

goal (1 subgoal):
 1. ⋀x. ⟦P (f x); Q x⟧ ⟹ P (?x4 x)
```

```
-u-:%%-                  fstate Utoks Abbrev;)-----------------
Use C-c C-. to jump to end of processed region
```

Proof by assumption will unify these two terms

Done!



```
lemma "(∃x. P (f x) ∧ Q x) ⟹ ∃x. P x"
apply (erule exE)
apply (erule conjE)
apply (rule exI)
apply assumption
done
```

```
-u-:---  Basic.thy       3% L20      (Isar Utoks Abbrev; Scripting )----------------
```

```
proof (prove): step 4

goal:
No subgoals!
```

```
-u-:%%-  *goals*        Top L6       (Isar Proofstate Utoks Abbrev;)----------------
Use C-c C-. to jump to end of processed region
```