Robust, Semi-Intelligible Isabelle Proofs from ATP Proofs

Steffen Smolka
Advisor: Jasmin Blanchette

ITPs vs. ATPs

ITPs

well suited for
large formalizations

but

require intensive
manual labor

ATPs

fully automatic

but

no proof
management

ITPs

Isabelle

well suited for
large formalizations

but

require intensive
manual labor

**Sledge-
hammer**

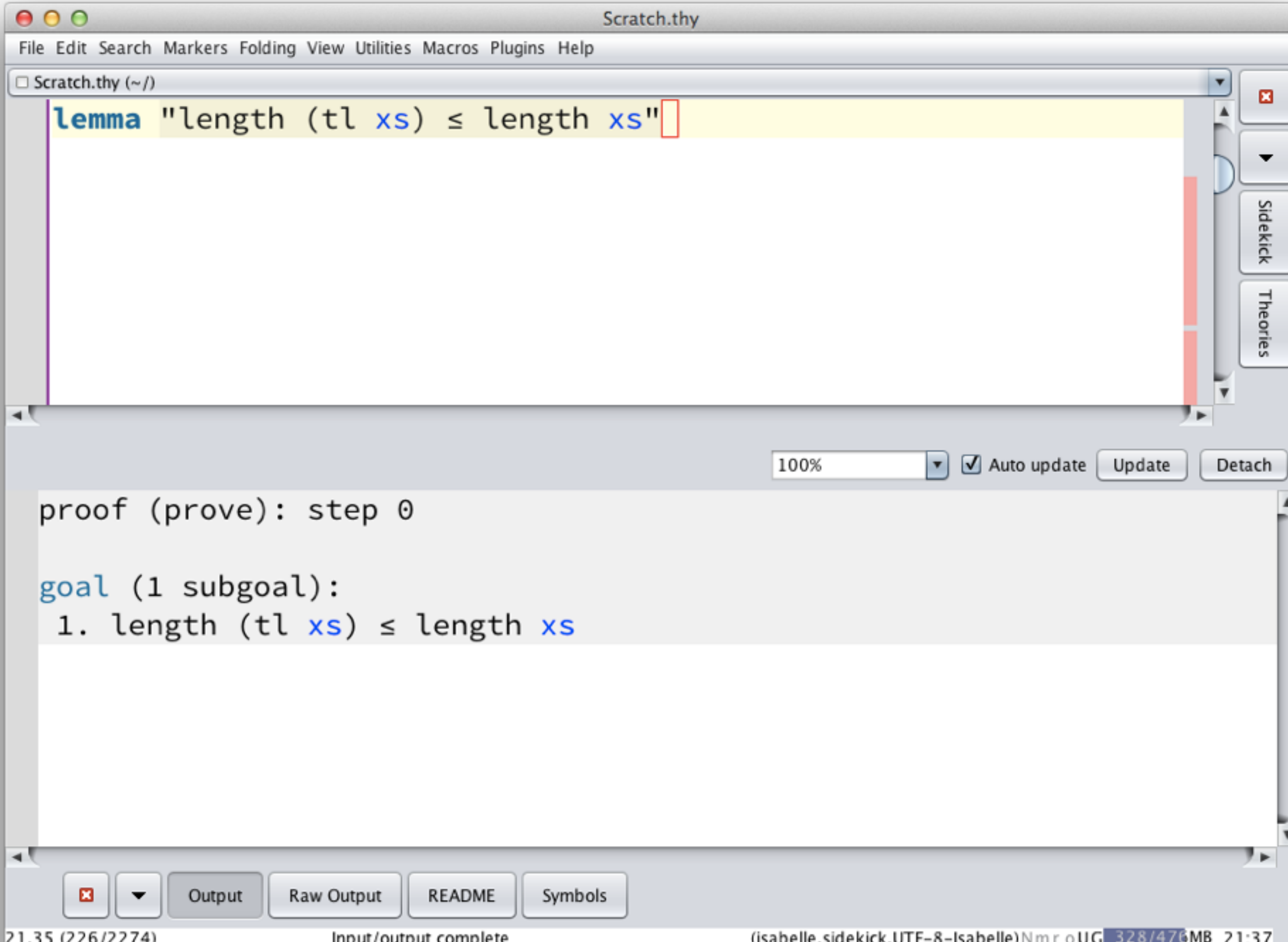
ATPs

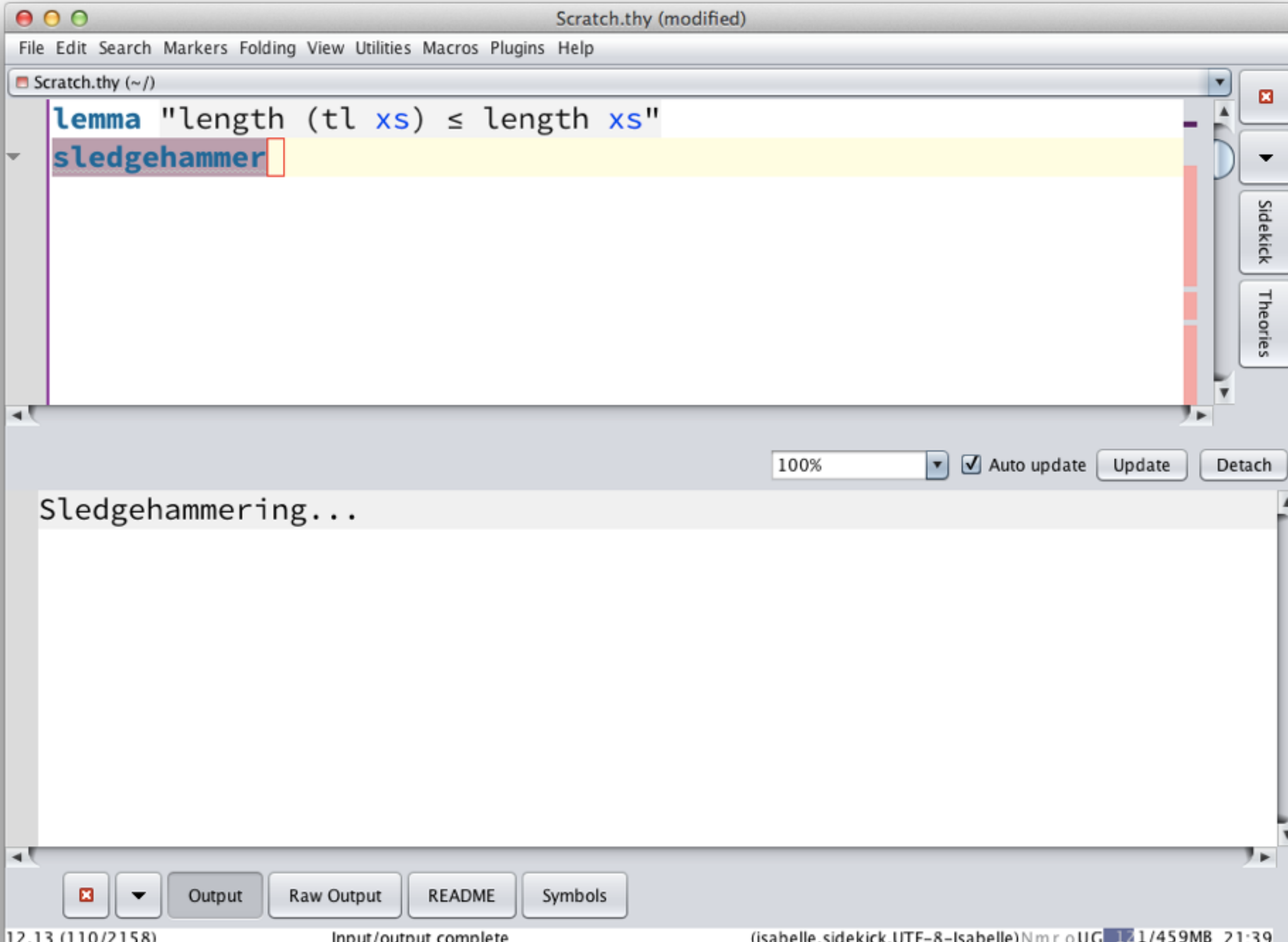
SPASS Vampire
LEO-II E
SATALLAX

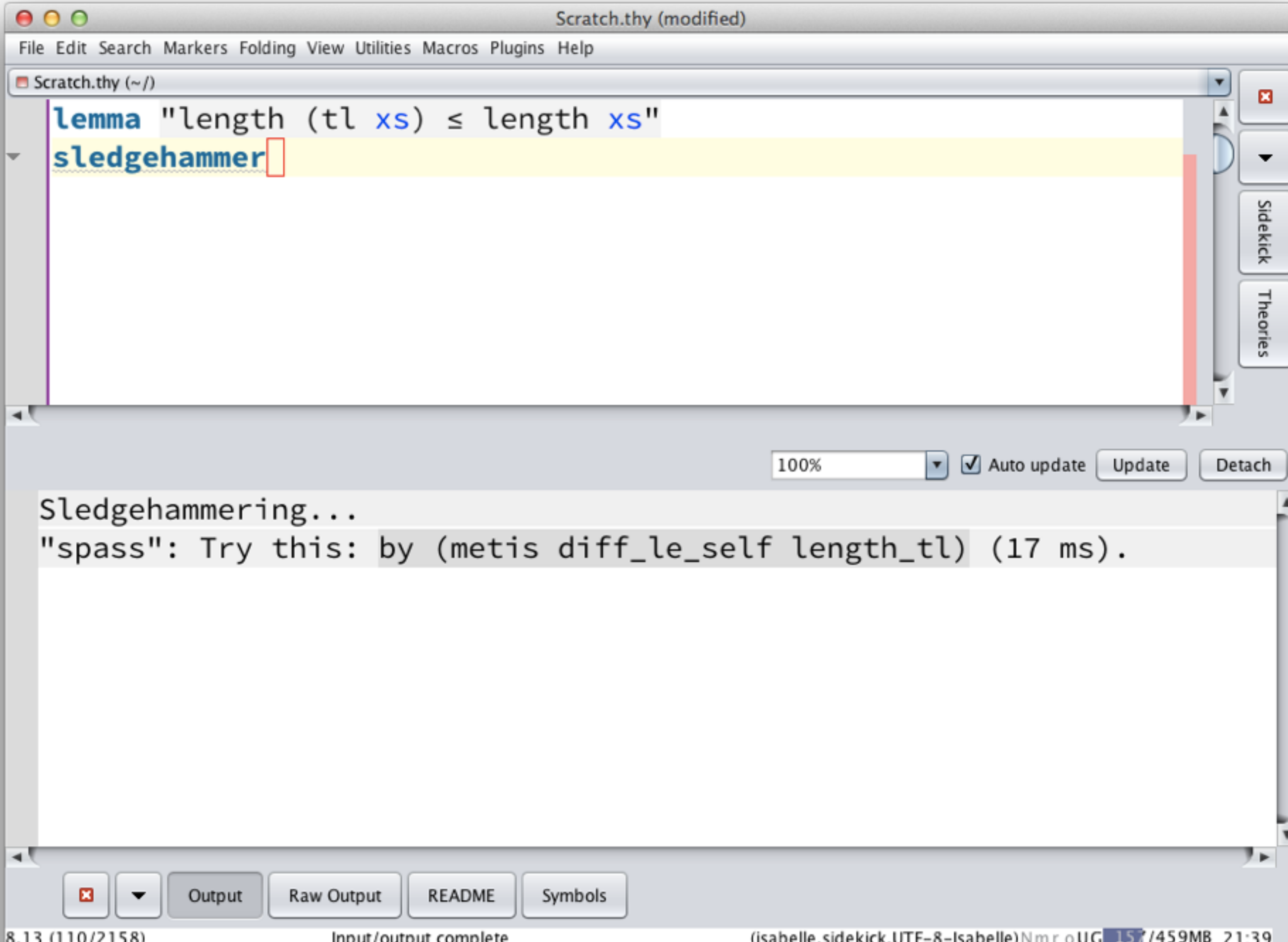
fully automatic

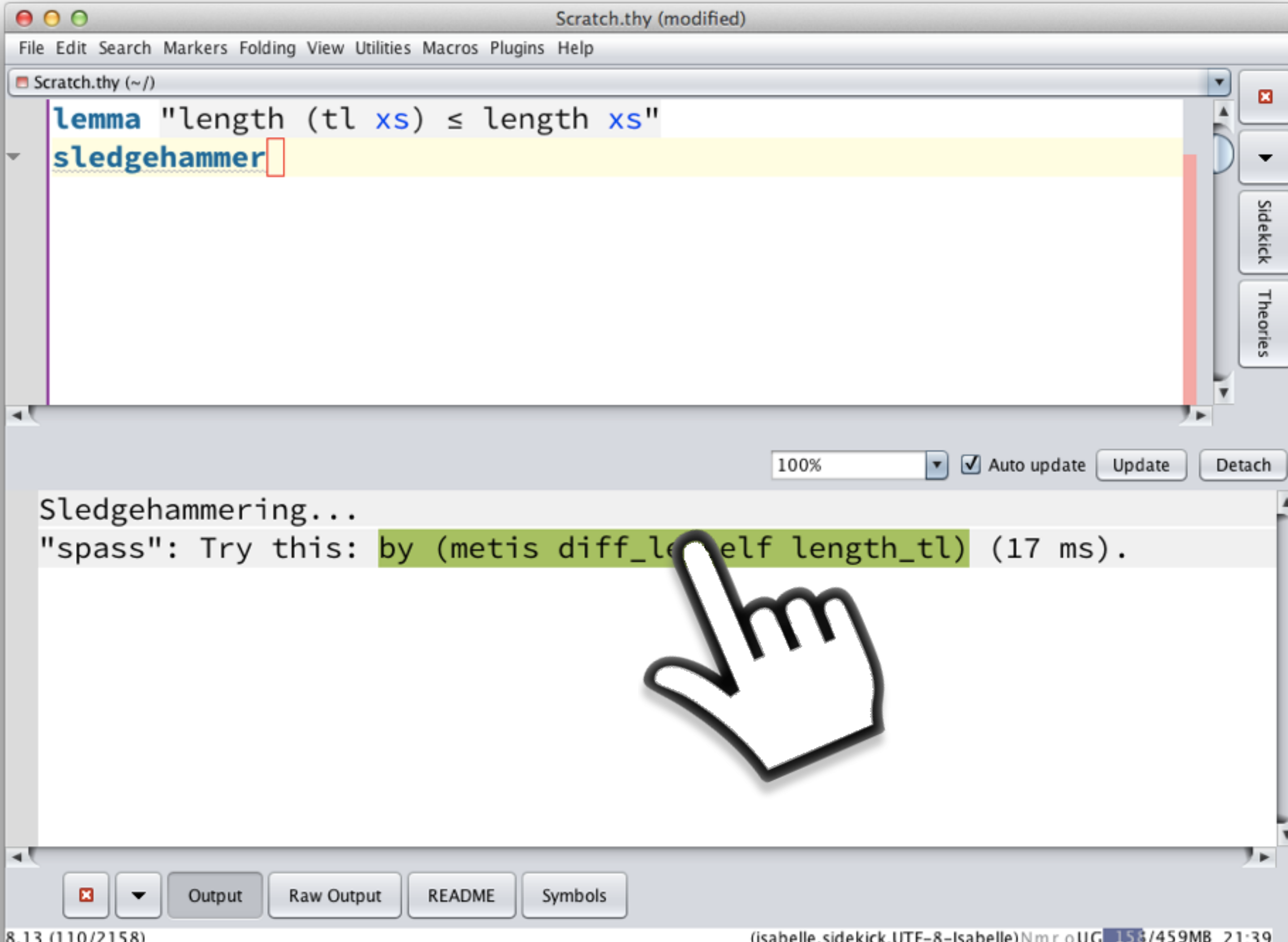
but

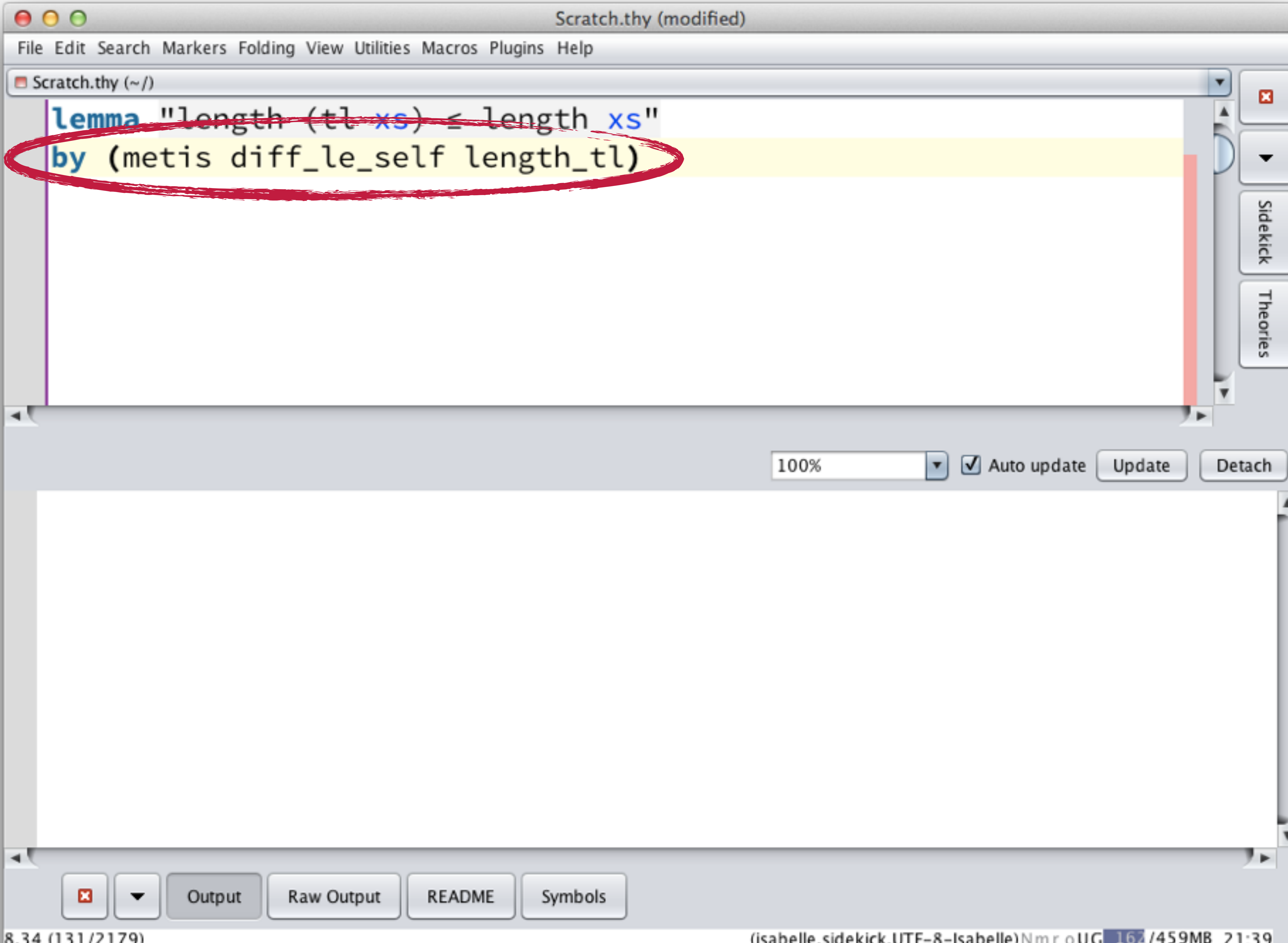
no proof
management


Scratch.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
Scratch.thy (~/)
lemma "length (tl xs) ≤ length xs"
Sidekick
Theories
100%
Auto update
Update
Detach
proof (prove): step 0
goal (1 subgoal):
1. length (tl xs) ≤ length xs
Output
Raw Output
README
Symbols
21,35 (226/2274)
Input/output complete
(isabelle,sidekick,UTF-8-Isabelle)
328/476MB
21:37


Scratch.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
Scratch.thy (~/)
lemma "length (tl xs) ≤ length xs"
sledgehammer
Sidekick
Theories
100%
Auto update
Update
Detach
Sledgehammering...
Output
Raw Output
README
Symbols
12,13 (110/2158)
Input/output complete
(isabelle,sidekick,UTF-8-Isabelle)Nmr oUG
1/459MB 21:39


Scratch.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
Scratch.thy (~/)
lemma "length (tl xs) ≤ length xs"
sledgehammer
Sidekick
Theories
100%
Auto update
Update
Detach
Sledgehammering...
"spass": Try this: by (metis diff_le_self length_tl) (17 ms).
Output
Raw Output
README
Symbols
8,13 (110/2158)
Input/output complete
(isabelle,sidekick,UTF-8-Isabelle)
/459MB 21:39


Scratch.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
Scratch.thy (~/)
lemma "length (tl xs) ≤ length xs"
sledgehammer
Sidekick
Theories
100%
Auto update
Update
Detach
Sledgehammering...
"spass": Try this: by (metis diff_le_self length_tl) (17 ms).
Output
Raw Output
README
Symbols
8,13 (110/2158)
(isabelle,sidekick,UTF-8-Isabelle)
21:39


Scratch.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
Scratch.thy (~/)
lemma "length (tl xs) ≤ length xs"
by (metis diff_le_self length_tl)
Sidekick
Theories
100%
Auto update
Update
Detach
Output
Raw Output
README
Symbols
8,34 (131/2179)
(isabelle,sidekick,UTF-8-Isabelle)Nmr oUG 162/459MB 21:39

Exploit ATPs,

but don't trust them.

LCF Principle (Robin Milner):

Have all proofs checked by the inference kernel.

$\Longrightarrow$ ATP proofs must be **reconstructed** in Isabelle.

Approach A: Metis One-Liners

```
lemma "length (tl xs) ≤ length xs"
by (metis diff_le_self length_tl)
```

proof method

lemmas

Approach A: Metis One-Liners

```
lemma "length (tl xs) ≤ length xs"
by (metis diff_le_self length_tl)
```

proof method **lemmas**

external ATPs: find proof given 100s of facts

Metis: re-find proof given only necessary facts

Approach A: Metis One-Liners

```
lemma "length (tl xs) ≤ length xs"
by (metis diff_le_self length_tl)
```

proof method **lemmas**

external ATPs: find proof given 100s of facts

Metis: re-find proof given only necessary facts

+ usually fast and reliable
+ lightweight
- cryptic
- sometimes slow (several seconds)
- on avg. 5% "loss"

Approach B: Detailed Isar Proofs

```
lemma "length (tl xs) ≤ length xs"
proof -
  have "⋀x1 x2. (x1::nat) - x2 - x1 = 0 - x2"
    by (metis comm_monoid_diff_class.diff_cancel diff_right_commute)
  hence "length xs - 1 - length xs = 0"
    by (metis zero_diff)
  hence "length xs - 1 ≤ length xs"
    by (metis diff_is_0_eq)
  thus "length (tl xs) ≤ length xs"
    by (metis length_tl)
qed
```

Approach B: Detailed Isar Proofs

```
lemma "length (tl xs) ≤ length xs"
proof -
  have "⋀x1 x2. (x1::nat) - x2 - x1 = 0 - x2"
    by (metis comm_monoid_diff_class.diff_cancel diff_right_commute)
  hence "length xs - 1 - length xs = 0"
    by (metis zero_diff)
  hence "length xs - 1 ≤ length xs"
    by (metis diff_is_0_eq)
  thus "length (tl xs) ≤ length xs"
    by (metis length_tl)
qed
```

+ faster than one-liners

+ 100% reconstruction (in principle)

+ self-explanatory

– technically more challenging

Challenge 1:

Resolution proofs are by contradiction

"sin against mathematical exposition" (Knuth et al. 1989)

→ Jasmin Blanchette

Challenge 2:

Skolemization - introduce new symbols during proof

Challenge 3:

Type Annotations - make Isabelle understand its own output

Challenge 4:

Preplay & Optimization - test and improve proofs

Challenge 2:

Skolemization

$$\frac{\forall X.\ \exists Y.\ p(X,\ Y)}{\forall X.\ p(X,\ \mathbf{y}(X))}\ \text{Skolemization}$$

Signature is extended

$$\frac{\forall X.\ \exists Y.\ p(X, Y)}{\forall X.\ p(X, \mathbf{y}(X))}$$ Skolemization

Signature is extended

$$\frac{\forall X.\ \exists Y.\ p(X, Y)}{\exists \mathbf{y}.\ \forall X.\ p(X, \mathbf{y}(X))}$$ Ax. of Choice

$$\frac{\forall X.\ \exists Y.\ p(X, Y)}{\forall X.\ p(X, y(X))}\quad \text{Skolemization}$$

Signature is extended

$$\frac{\forall X.\ \exists Y.\ p(X, Y)}{\exists y.\ \forall X.\ p(X, y(X))}\quad \text{Ax. of Choice}$$

Signature is extended

obtain **y** *where* ∀X. p(X, **y**(X))

<steps with **reduced** sig.>

$$\frac{\forall X.\ \exists Y.\ p(X,\ Y)}{\exists y.\ \forall X.\ p(X,\ y(X))}\ \text{Ax. of Choice}$$

<steps with **extended** sig.>

<steps with **extended** sig.>

$$\frac{\exists \mathbf{y}.\ \forall X.\ p(X, \mathbf{y}(X))}{\forall X.\ \exists Y.\ p(X, Y)}\ \text{Ax. of Choice}$$

<steps with **reduced** sig.>

<steps with **extended** sig.>

$$\frac{\forall \mathbf{y}.\ \exists X.\ \neg p(X,\ \mathbf{y}(X))}{\exists X.\ \forall Y.\ \neg p(X,\ Y)}\ \text{Ax. of Choice}$$

<steps with **reduced** sig.>

<steps with **extended** sig.>

$$\frac{\forall y.\ \exists X.\ \neg p(X,\ y(X))}{\exists X.\ \forall Y.\ \neg p(X,\ Y)} \quad \text{Contrap. of Ax. of Choice}$$

<steps with **reduced** sig.>

<steps with **extended** sig.>

$$\frac{\forall \mathbf{y}.\ \exists X.\ \neg p(X,\ \mathbf{y}(X))}{\exists X.\ \forall Y.\ \neg p(X,\ Y)}$$

Contrap. of Ax. of Choice

<steps with **reduced** sig.>

{ fix **y**

<steps with **extended** sig.>

have ∃X. ¬p(X, **y**(X)) **}**

hence ∃X. ∀Y. ¬p(X, Y)

<steps with **reduced** sig.>

Challenge 3:

Type Annotations

Make Isabelle understand its own output

$$2^{\texttt{nat}} \;+^{\texttt{nat}\to\texttt{nat}\to\texttt{nat}}\; 2^{\texttt{nat}} \;=^{\texttt{nat}\to\texttt{nat}\to\texttt{bool}}\; 4^{\texttt{nat}}$$

$$\Big\downarrow \text{print}$$

$$2 + 2 = 4$$

$$2^{\mathtt{nat}} \;+^{\mathtt{nat\to nat\to nat}}\; 2^{\mathtt{nat}} \;=^{\mathtt{nat\to nat\to bool}}\; 4^{\mathtt{nat}}$$

$\Big\downarrow$ print

$$2 + 2 = 4$$

$\Big\downarrow$ parse

$$2^{\alpha} \;+^{\alpha\to\alpha\to\alpha}\; 2^{\alpha} \;=^{\alpha\to\alpha\to \mathtt{bool}}\; 4^{\alpha}$$

where $\alpha{:}\mathtt{numeral}$

Un-provable

$$2^{\mathtt{nat}}\ +^{\mathtt{nat}\to\mathtt{nat}\to\mathtt{nat}}\ 2^{\mathtt{nat}}\ =^{\mathtt{nat}\to\mathtt{nat}\to\mathtt{bool}}\ 4^{\mathtt{nat}}$$

$$\Big\downarrow \ \text{print}$$

$$(2:\mathtt{nat})\ (+:\mathtt{nat}\to\mathtt{nat}\to\mathtt{nat})\ (2:\mathtt{nat})$$
$$(=:\mathtt{nat}\to\mathtt{nat}\to\mathtt{bool})\ (4:\mathtt{nat})$$

$$\Big\downarrow \ \text{parse}$$

$$2^{\mathtt{nat}}\ +^{\mathtt{nat}\to\mathtt{nat}\to\mathtt{nat}}\ 2^{\mathtt{nat}}\ =^{\mathtt{nat}\to\mathtt{nat}\to\mathtt{bool}}\ 4^{\mathtt{nat}}$$

$$2^{\mathtt{nat}} \ +^{\mathtt{nat}\to\mathtt{nat}\to\mathtt{nat}} \ 2^{\mathtt{nat}} \ =^{\mathtt{nat}\to\mathtt{nat}\to\mathtt{bool}} \ 4^{\mathtt{nat}}$$

$$\Big\downarrow \ \text{print}$$

$$2 + 2 = 4$$

$$\Big\downarrow \ \text{parse}$$

$$2^{\alpha} \ +^{\alpha\to\alpha\to\alpha} \ 2^{\alpha} \ =^{\alpha\to\alpha\to\mathtt{bool}} \ 4^{\alpha}$$

where $\alpha{:}\mathtt{numeral}$

$$2^{\mathtt{nat}} +^{\mathtt{nat \to nat \to nat}} 2^{\mathtt{nat}} =^{\mathtt{nat \to nat \to bool}} 4^{\mathtt{nat}}$$

$$\Big\downarrow \text{print}$$

`(2:nat) + 2 = 4`

$$\Big\downarrow \text{parse}$$

$$2^{\mathtt{nat}} +^{\mathtt{nat \to nat \to nat}} 2^{\mathtt{nat}} =^{\mathtt{nat \to nat \to bool}} 4^{\mathtt{nat}}$$

Goal: Calculate a set of annotations that is

(A) **Complete:** reparsing term must not change its type

(B) **Minimal:** annotations must impair readability as little as possible

$$\mathbf{f}^{\mathtt{nat\to int\to bool}}\ \ \mathbf{x}^{\mathtt{nat}}\ \ \mathbf{y}^{\mathtt{int}}$$

type erasure
≈ printing

$$\mathbf{f}^{-}\ \ \mathbf{x}^{-}\ \ \mathbf{y}^{-}$$

type inference
≈ parsing

$$\mathbf{f}^{\alpha\to\beta\to\gamma}\ \ \mathbf{x}^{\alpha}\ \ \mathbf{y}^{\beta}$$

matching

$$\sigma = \{\ \alpha\mapsto\mathtt{nat},\ \beta\mapsto\mathtt{int},\ \gamma\mapsto\mathtt{bool}\ \}$$

$$\mathbf{f}^{\mathtt{nat\to int\to bool}}\ \ \mathbf{x}^{\mathtt{nat}}\ \ \mathbf{y}^{\mathtt{int}}$$

$\downarrow$ **type erasure** ≈ printing

$$\mathbf{f}^{-}\ \ \mathbf{x}^{-}\ \ \mathbf{y}^{-}$$

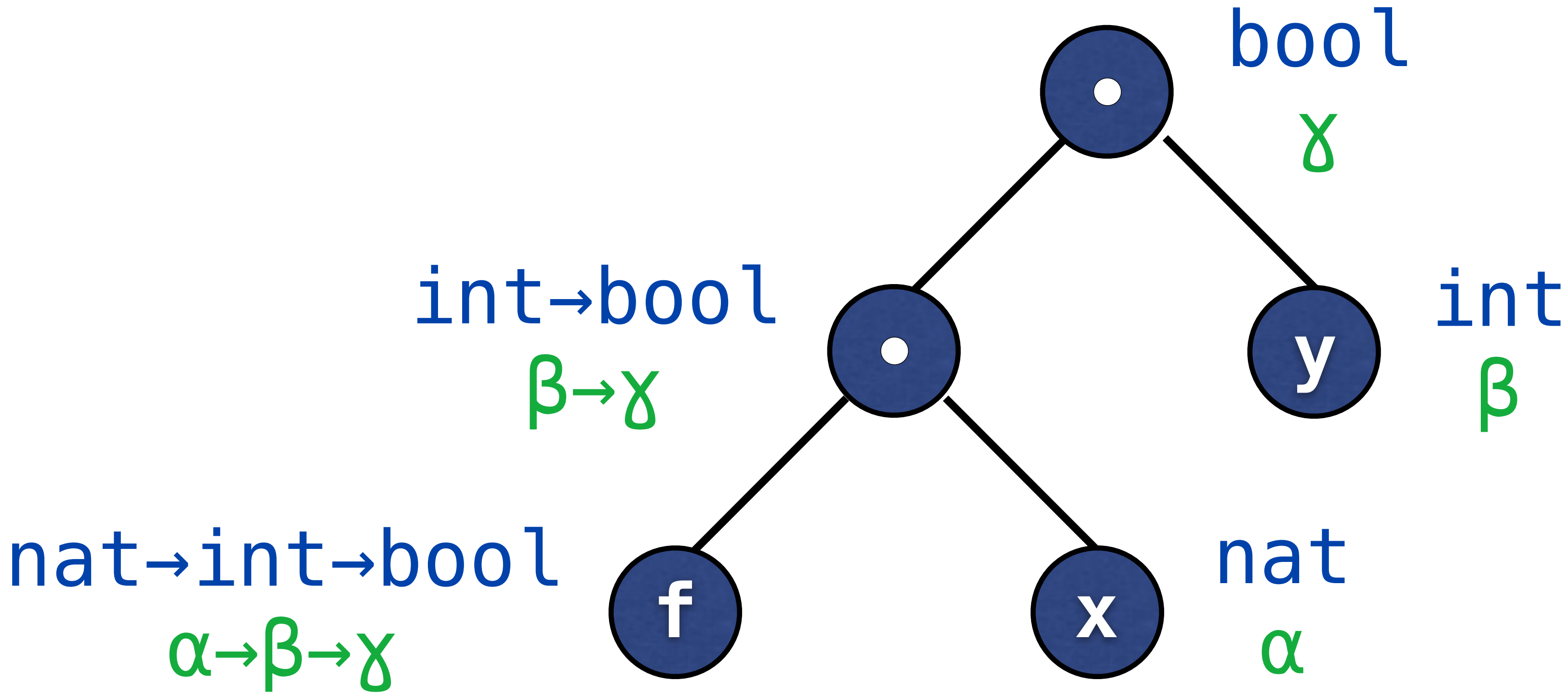
$\downarrow$ **type inference** ≈ parsing

$$\mathbf{f}^{\alpha\to\beta\to\gamma}\ \ \mathbf{x}^{\alpha}\ \ \mathbf{y}^{\beta}$$

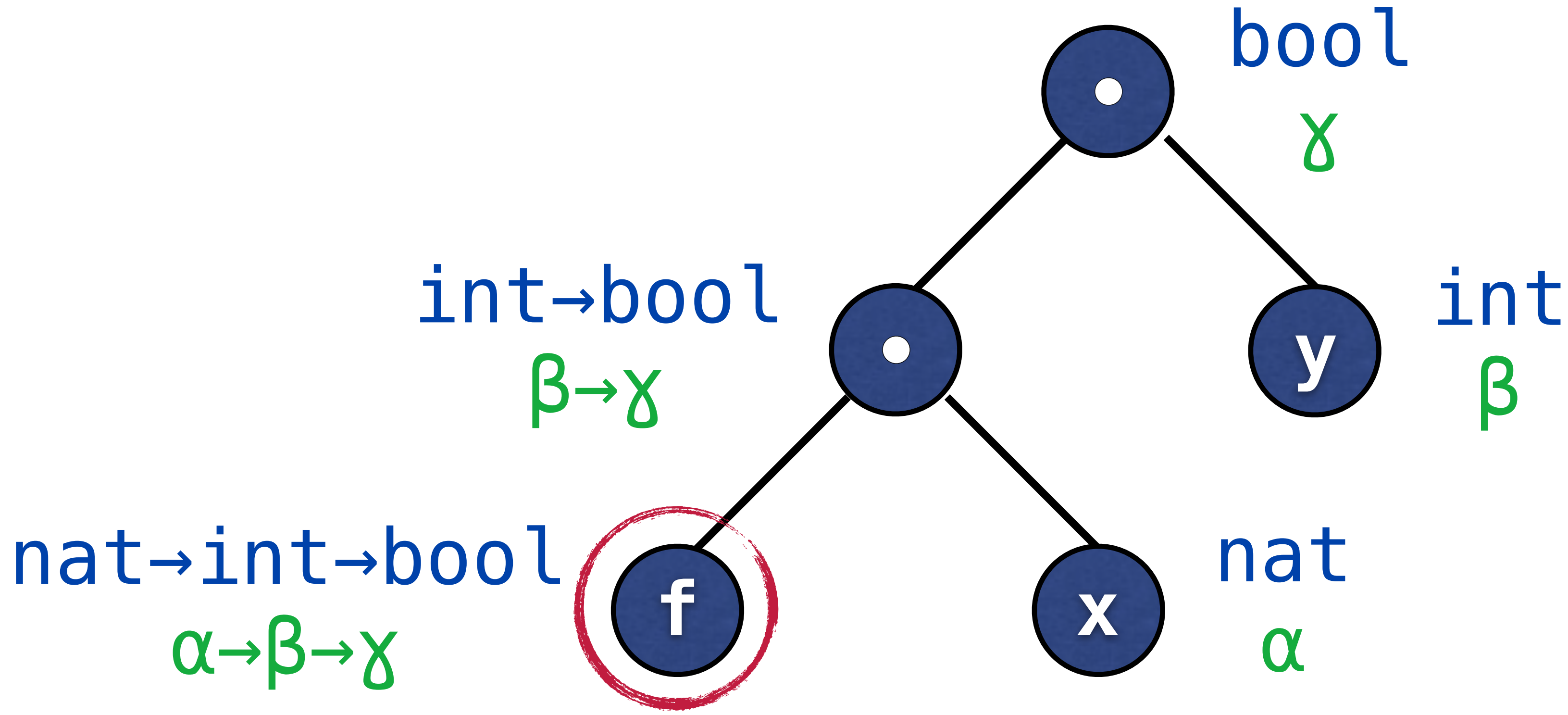
$\downarrow$ **matching**

$$\sigma = \{\ \alpha\mapsto \mathtt{nat},\ \beta\mapsto \mathtt{int},\ \gamma\mapsto \mathtt{bool}\ \}$$

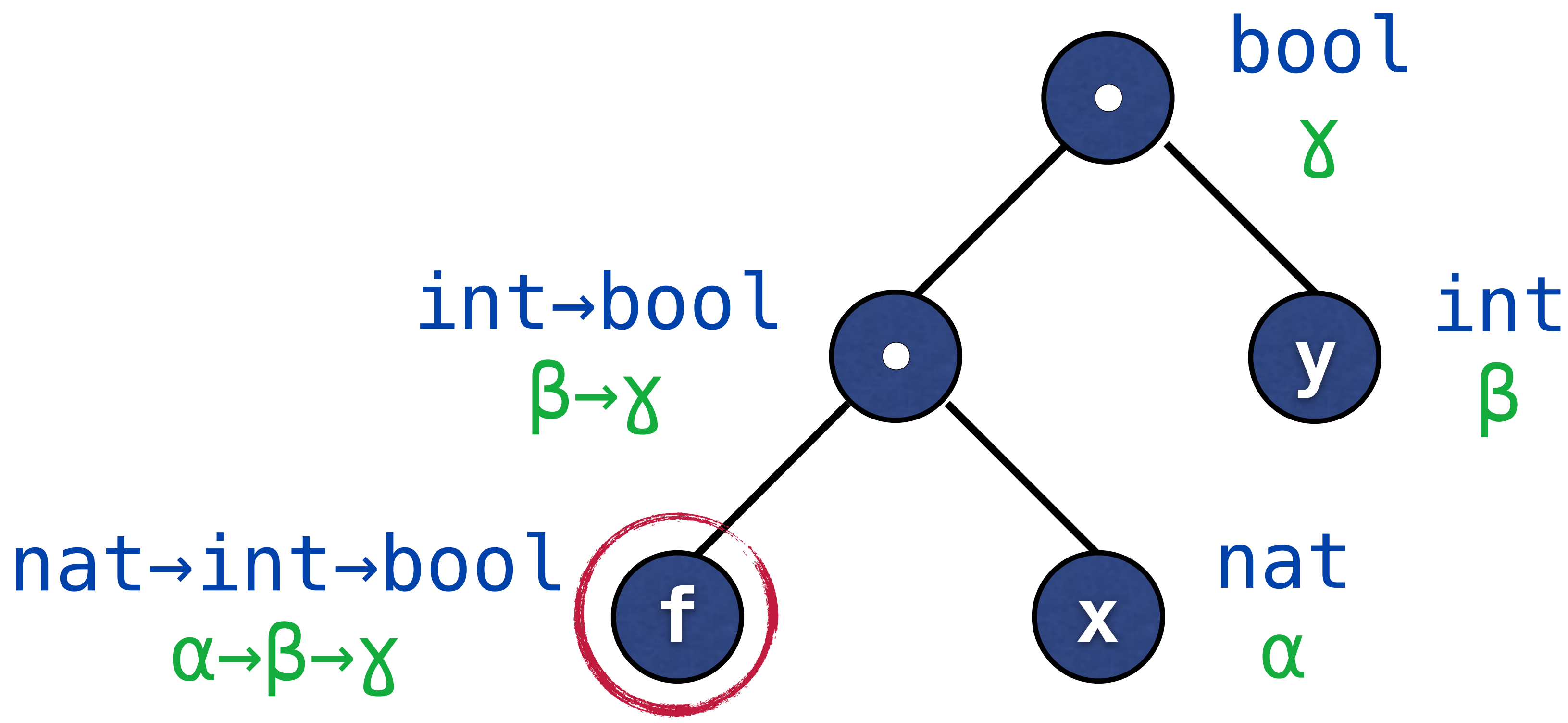
Set of ann. complete IFF it covers Dom(σ)



f x y

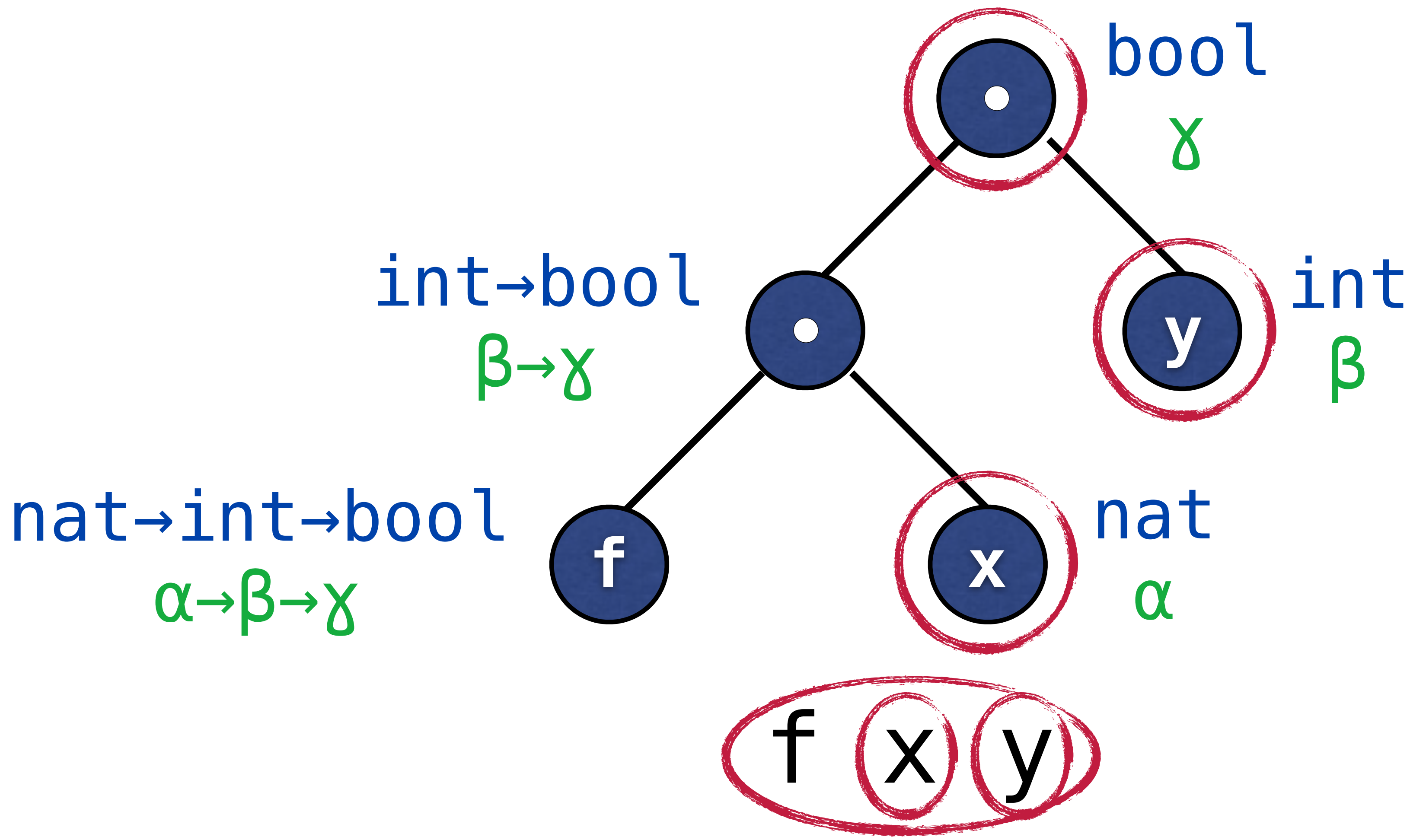


f x y

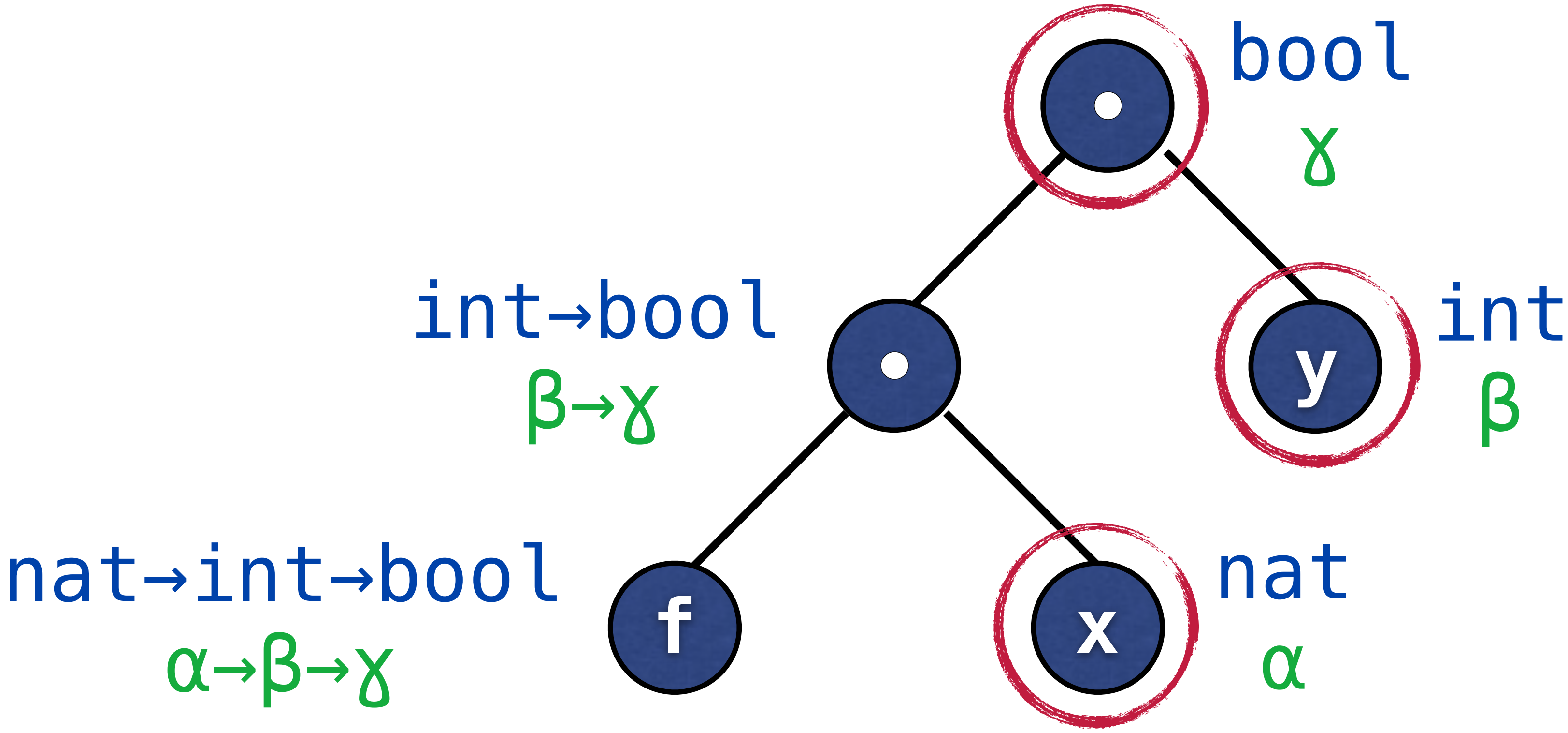


f x y

(f:nat→int→bool) x y

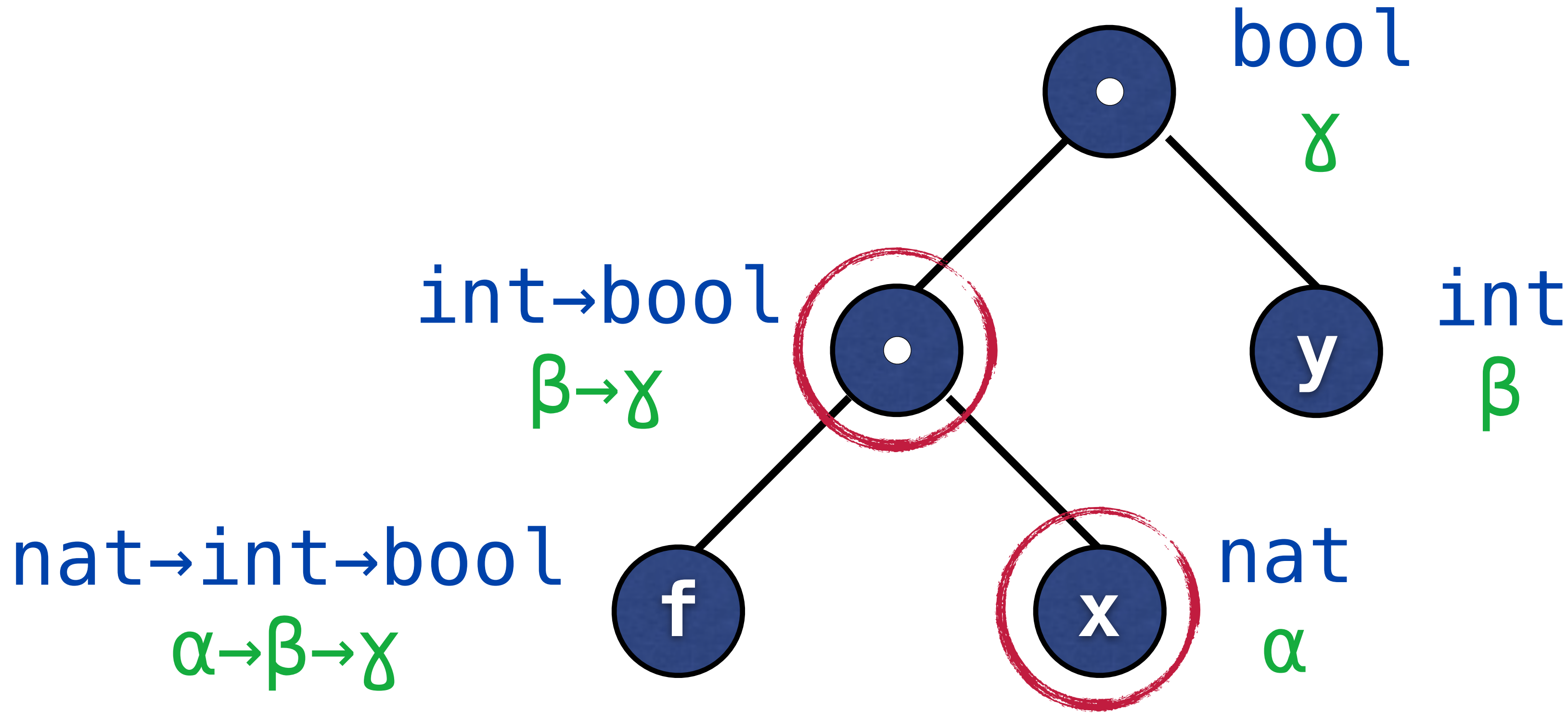


(f:nat→int→bool) x y



f x y

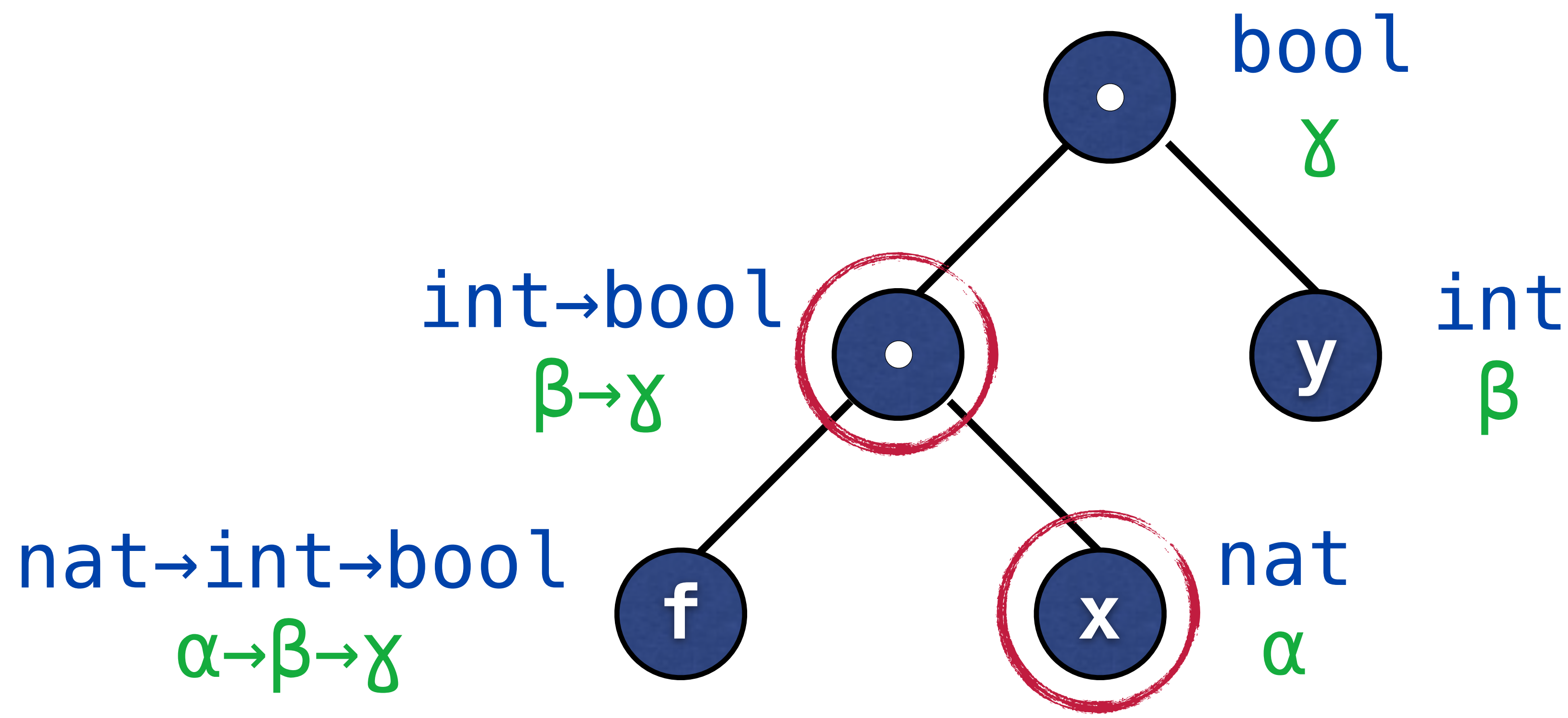
```
(f:nat→int→bool) x y

f (x:nat) (y:int) :bool
```



f x y

(f:nat→int→bool) x y

f (x:nat) (y:int) :bool

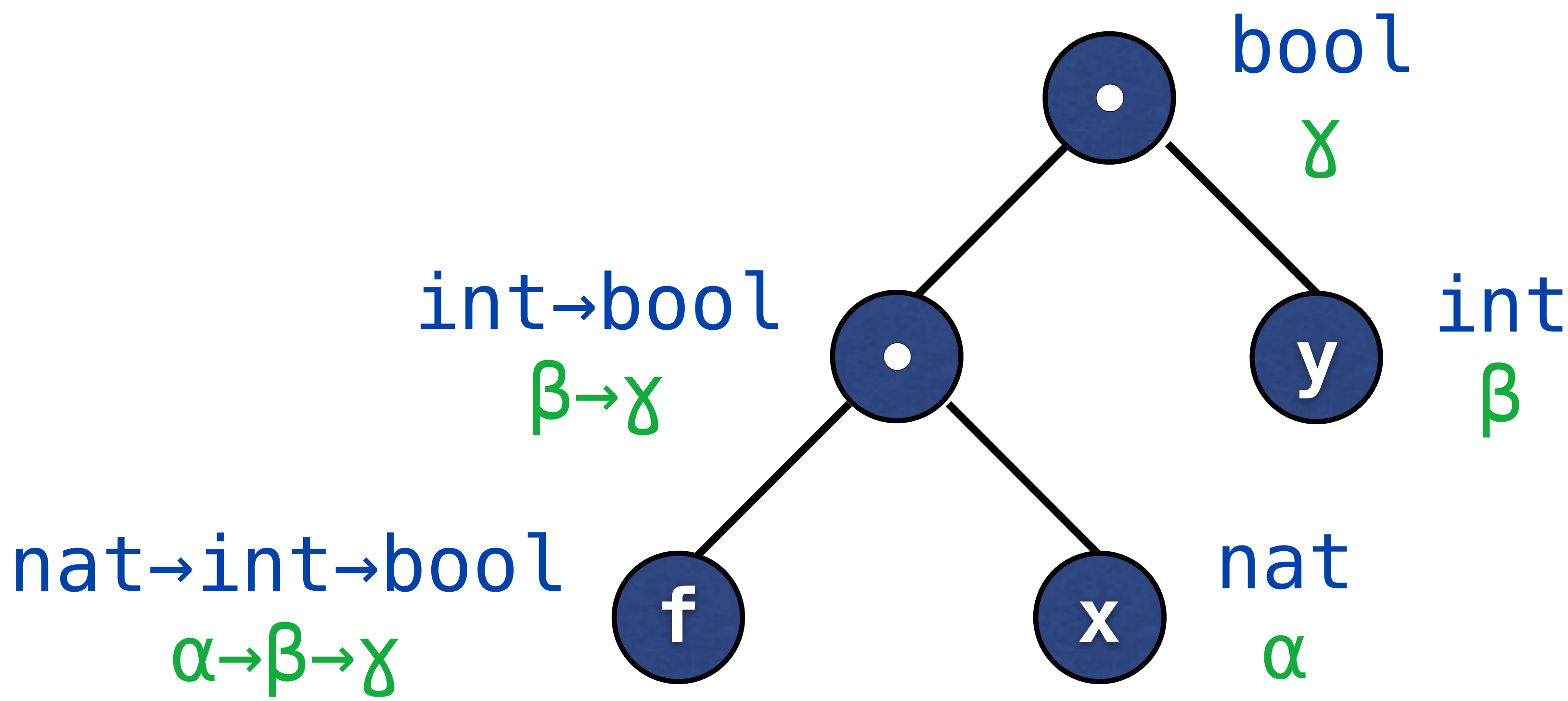


f x y

(f:nat→int→bool) x y

f (x:nat) (y:int) :bool

(f (x:nat) :int→bool) y



f x y

(f:nat→int→bool) x y

f (x:nat) (y:int) :bool

(f (x:nat) :int→bool) y

Which set of annotations is the best?

How do we compute it efficiently?

Which set of annotations is the best?

`cost of t`$^{\tau}$ `:=`

Which set of annotations is the best?

cost of $\mathbf{t}^{\tau}$:=

(size of τ, → small annotations

size of t, → small annotated terms

postindex of t^{τ}) → annotations at the beginning

Which set of annotations is the best?

```
cost of tτ :=
  (size of τ,          → small annotations
   size of t,          → small annotated terms
   postindex of tτ)    → annotations at the beginning
```

≤ lexiographically

\+ component-wise

How do we compute it efficiently?

How do we compute it efficiently?

Instance of **Weighted Set Cover Problem**:

- Finite Universe U → Dom($\boldsymbol{\sigma}$)
- Family $S \subseteq 2^U$ → Possible Annotations

How do we compute it efficiently?

Instance of **Weighted Set Cover Problem**:

- Finite Universe U → Dom($\boldsymbol{\sigma}$)
- Family $S \subseteq 2^U$ → Possible Annotations
- Find $\{U_1,\ldots,U_n\} \subseteq S$ such that
  - $U_1 \cup \ldots \cup U_n = U$ → Completeness
  - cost $\{U_1,\ldots,U_n\}$ minimal → Readability

SCP is **NP-complete** $\implies$ settle for Approximation

Reverse-Greedy Alg. calculates local min:

- start with all annotations
- repeatedly remove the most expensive superfluous annotation

Challenge 4:

Preplay & Optimization

Proof Preplay

Generated proofs are only useful if they...

- work

Proof Preplay

Generated proofs are only useful if they...

- work
- are reasonably fast

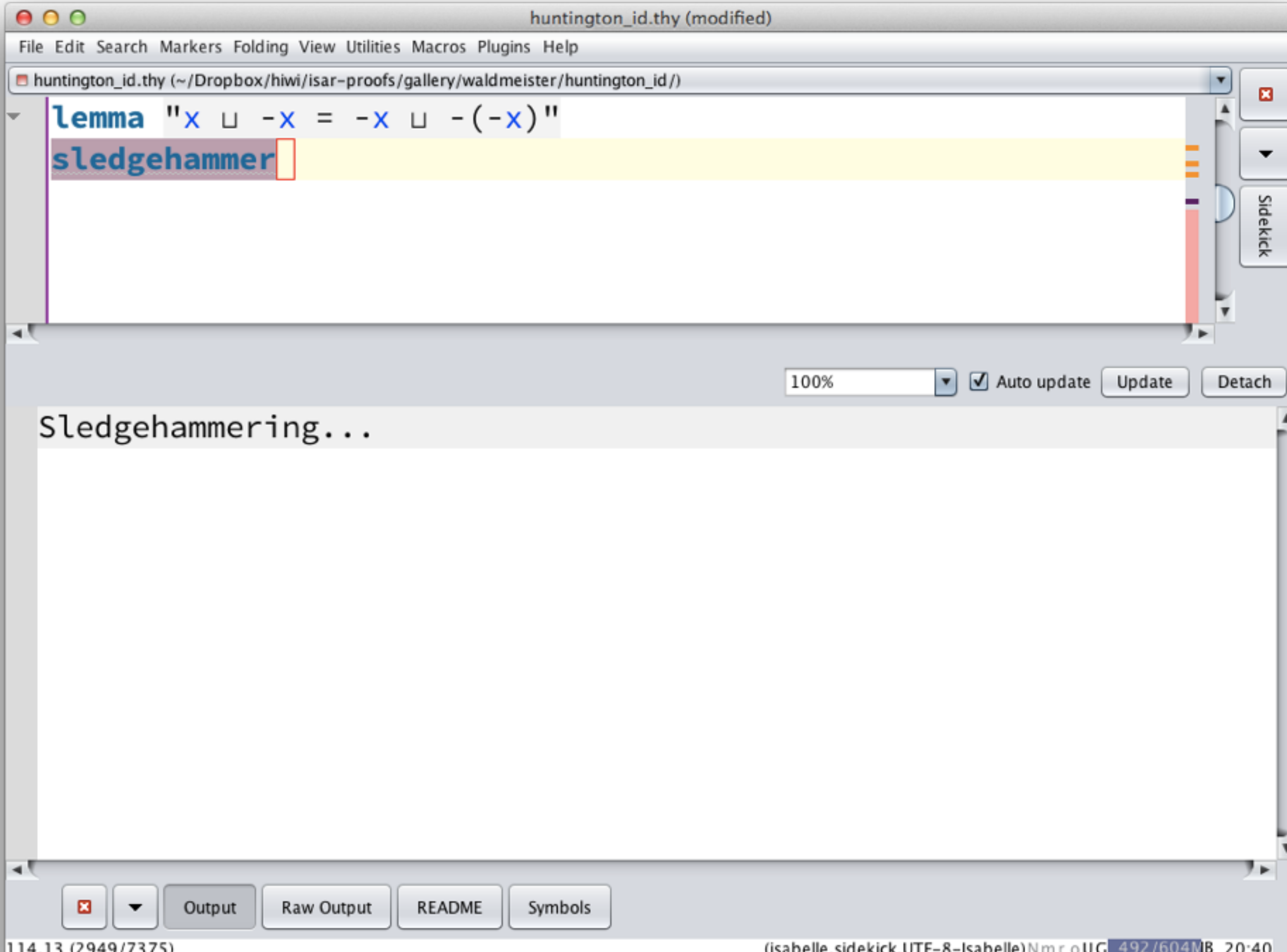
Proof Preplay

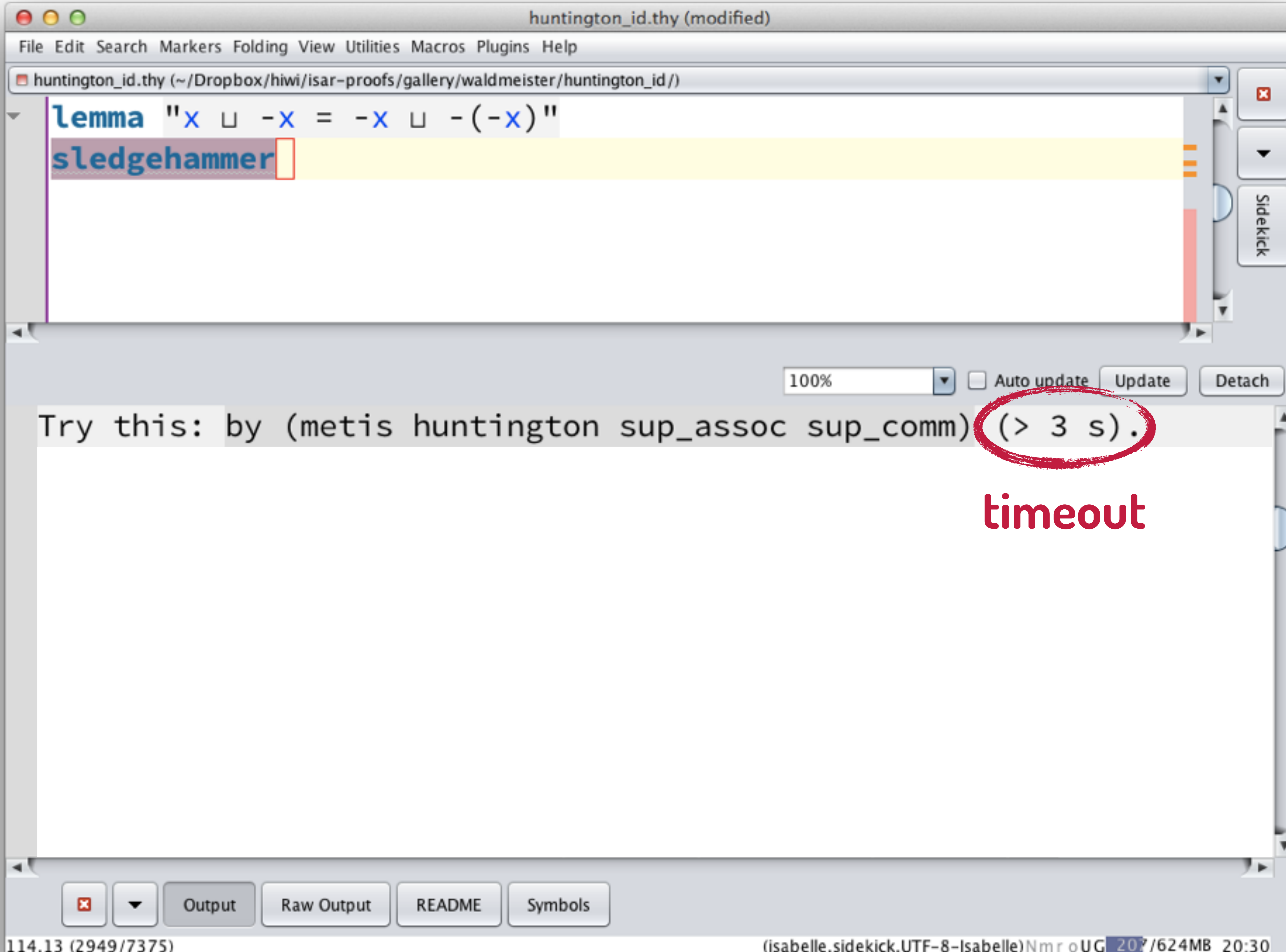
Generated proofs are only useful if they...

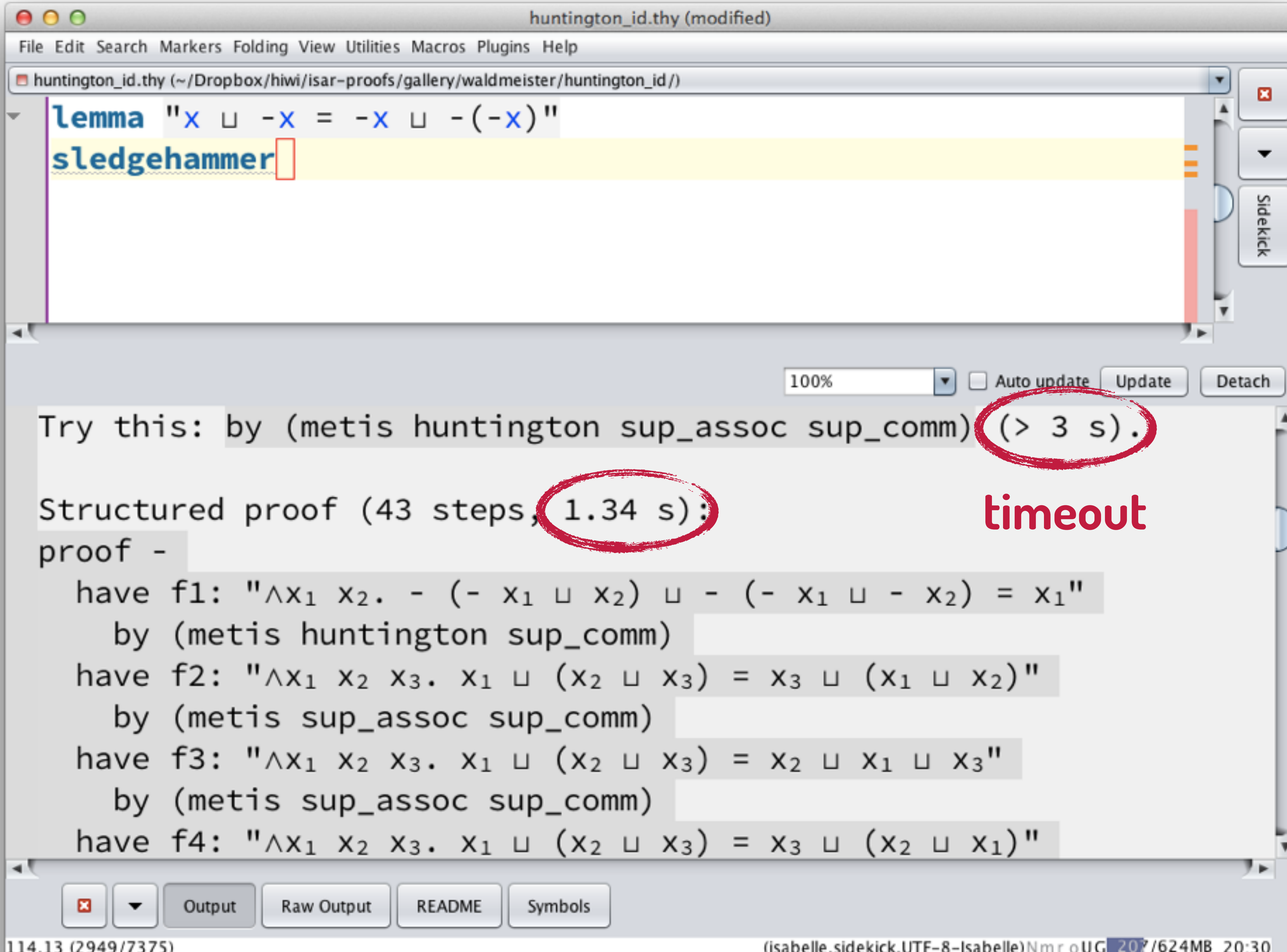
- work
- are reasonably fast

Let the computer find out!

$\rightarrow$ Present proofs with "preplay" information


huntington_id.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
huntington_id.thy (~/Dropbox/hiwi/isar-proofs/gallery/waldmeister/huntington_id/)
lemma "x ⊔ -x = -x ⊔ -(-x)"
sledgehammer
Sidekick
100%
Auto update
Update
Detach
Sledgehammering...
Output
Raw Output
README
Symbols
114,13 (2949/7375)
(isabelle,sidekick,UTF-8-Isabelle)Nmr oUG 492/604MB 20:40


huntington_id.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
huntington_id.thy (~/Dropbox/hiwi/isar-proofs/gallery/waldmeister/huntington_id/)
lemma "x ⊔ -x = -x ⊔ -(-x)"
sledgehammer
Sidekick
100%
Auto update
Update
Detach
Try this: by (metis huntington sup_assoc sup_comm) (> 3 s).
timeout
Output
Raw Output
README
Symbols
114,13 (2949/7375)
(isabelle,sidekick,UTF-8-Isabelle)Nmr oUG
/624MB 20:30


huntington_id.thy (modified)
File Edit Search Markers Folding View Utilities Macros Plugins Help
huntington_id.thy (~/Dropbox/hiwi/isar-proofs/gallery/waldmeister/huntington_id/)
lemma "x ⊔ -x = -x ⊔ -(-x)"
sledgehammer
Sidekick
100%
Auto update
Update
Detach
Try this: by (metis huntington sup_assoc sup_comm) (> 3 s).
Structured proof (43 steps, 1.34 s):
proof -
have f1: "⋀x1 x2. - (- x1 ⊔ x2) ⊔ - (- x1 ⊔ - x2) = x1"
by (metis huntington sup_comm)
have f2: "⋀x1 x2 x3. x1 ⊔ (x2 ⊔ x3) = x3 ⊔ (x1 ⊔ x2)"
by (metis sup_assoc sup_comm)
have f3: "⋀x1 x2 x3. x1 ⊔ (x2 ⊔ x3) = x2 ⊔ x1 ⊔ x3"
by (metis sup_assoc sup_comm)
have f4: "⋀x1 x2 x3. x1 ⊔ (x2 ⊔ x3) = x3 ⊔ (x2 ⊔ x1)"
timeout
Output
Raw Output
README
Symbols
114,13 (2949/7375)
(isabelle,sidekick,UTF-8-Isabelle)
/624MB 20:30

Approach A: Feed proof text to Isabelle

+ close to reality

- expensive

- no timings for individual steps

~~**Approach A:** Feed proof text to Isabelle~~

~~+ close to reality~~

~~– expensive~~

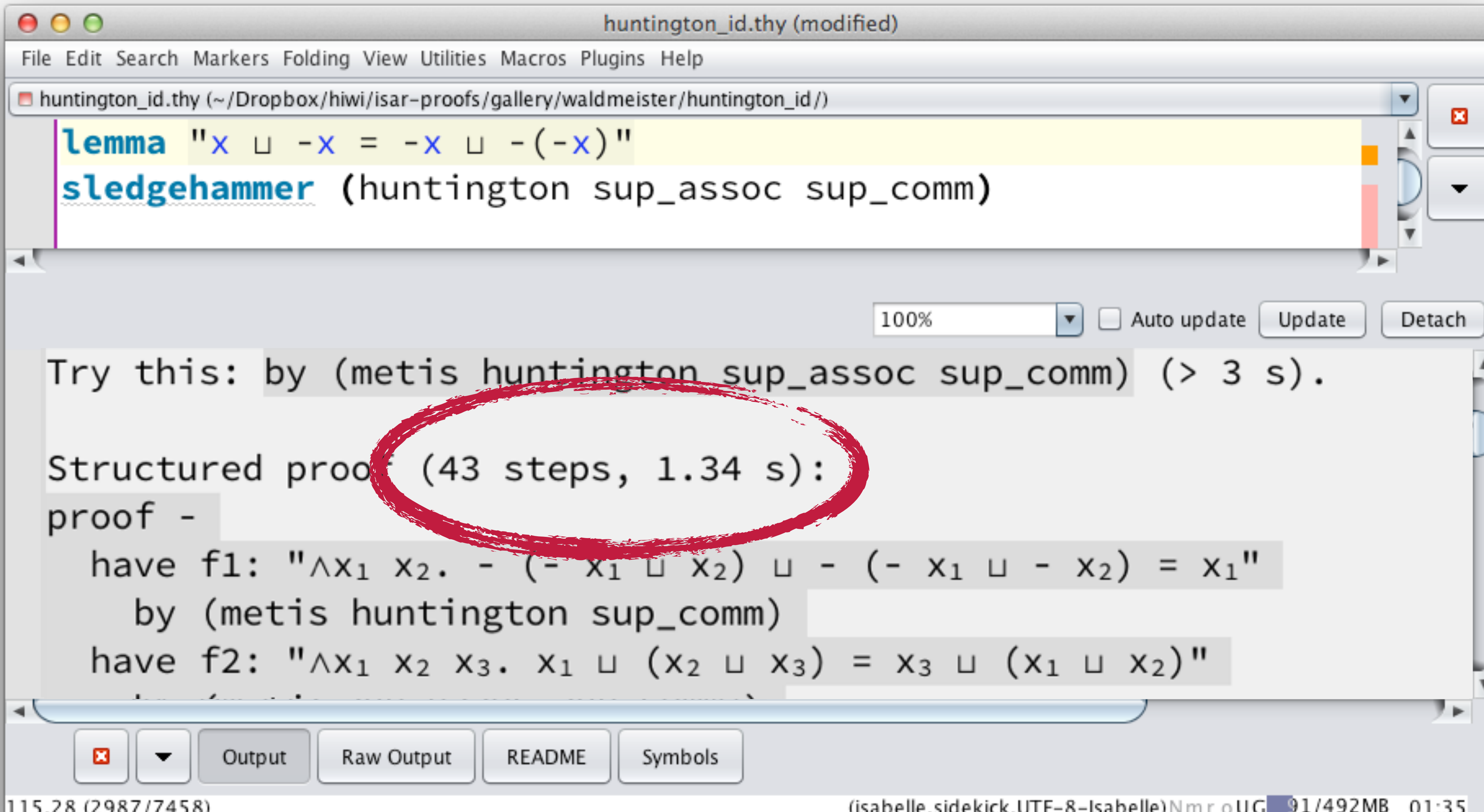
~~– no timings for individual steps~~

Approach B: Simulate replay on ML-level

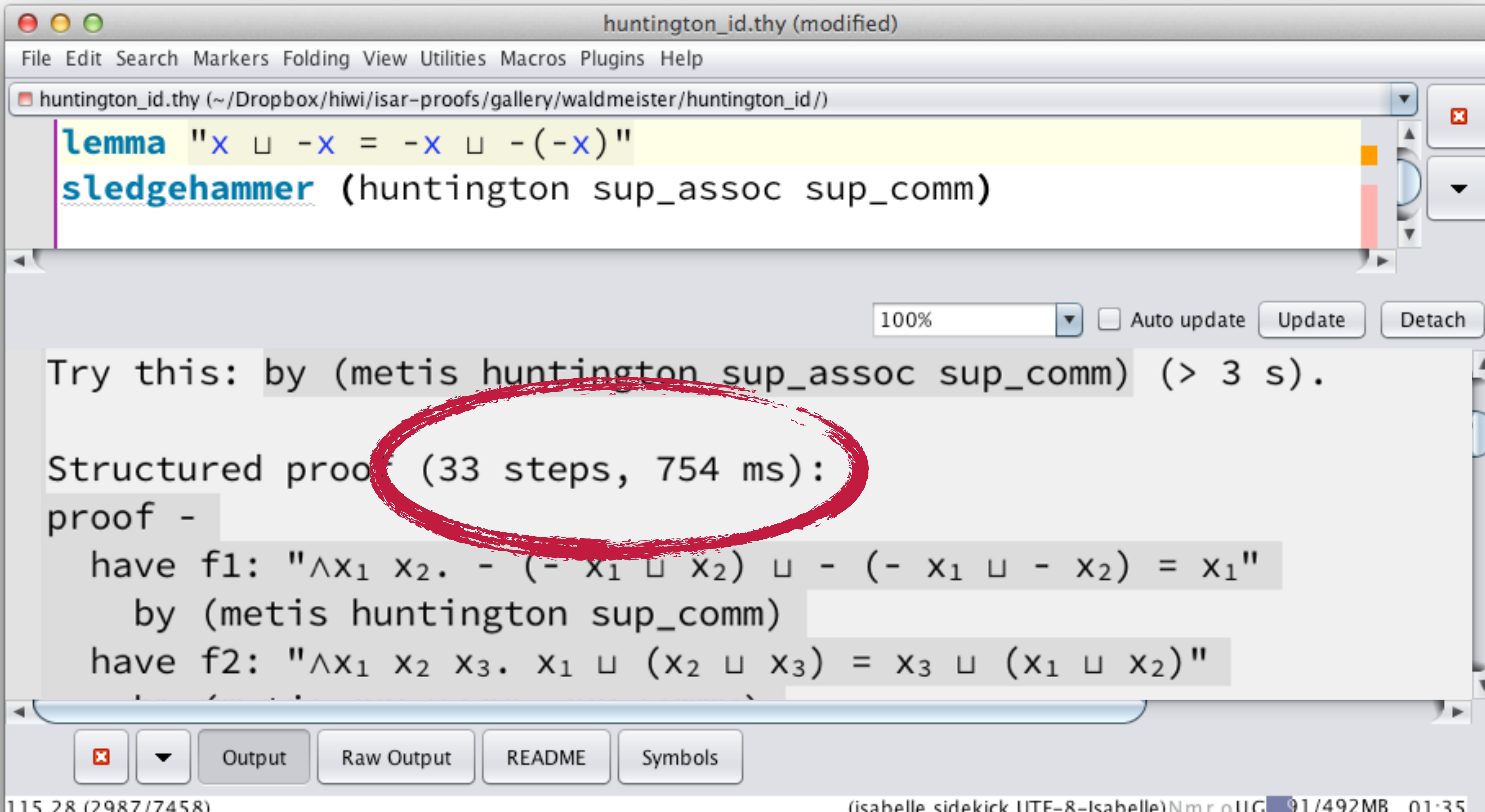
– not the real thing (no printing, no parsing)

+ timings for each step

Proof Compression



Proof Compression



$$A1 \vdash I$$

$$I, A2 \vdash C$$

$$
\begin{array}{l} \mathtt{A1} \vdash \mathtt{I} \\ \mathtt{I},\ \mathtt{A2} \vdash \mathtt{C} \end{array} \xrightarrow{\text{merger}} \mathtt{A1},\mathtt{A2} \vdash \mathtt{C}
$$

$$\texttt{A1} \vdash \texttt{I}$$

$$\texttt{I, A2} \vdash \texttt{C} \xrightarrow{\text{merger}} \texttt{A1,A2} \vdash \texttt{C}$$

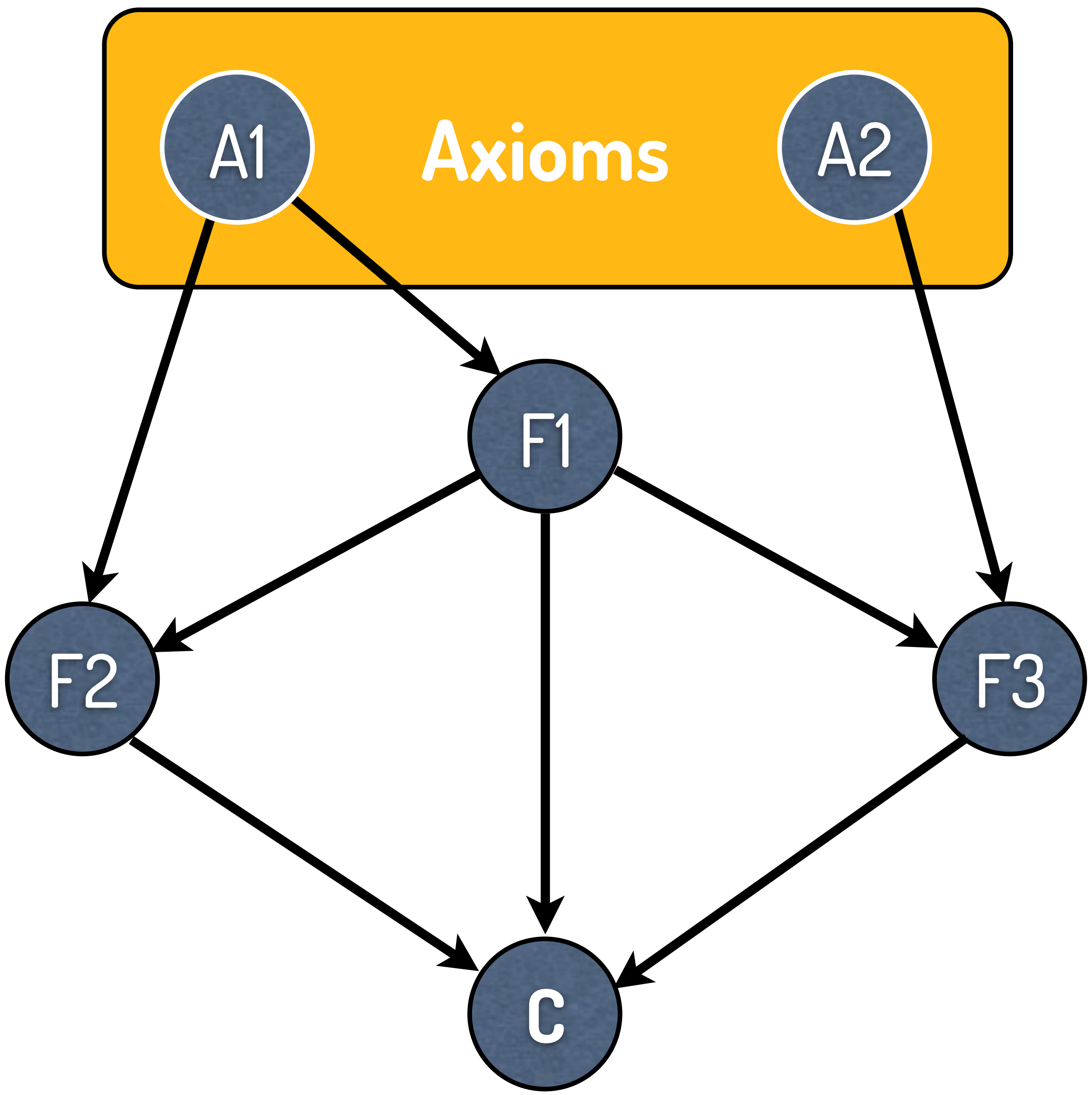
$$t1 + t2 \geq t' \quad ?$$

Does merger save time?

$$\begin{array}{l} \mathtt{A1} \vdash \mathtt{I} \\ \mathtt{I},\ \mathtt{A2} \vdash \mathtt{C} \end{array} \xrightarrow{\text{merger}} \mathtt{A1,A2} \vdash \mathtt{C}$$

$$(t1 + t2)(1+\text{bonus}) \geq t' \quad ?$$

Does merger save time?

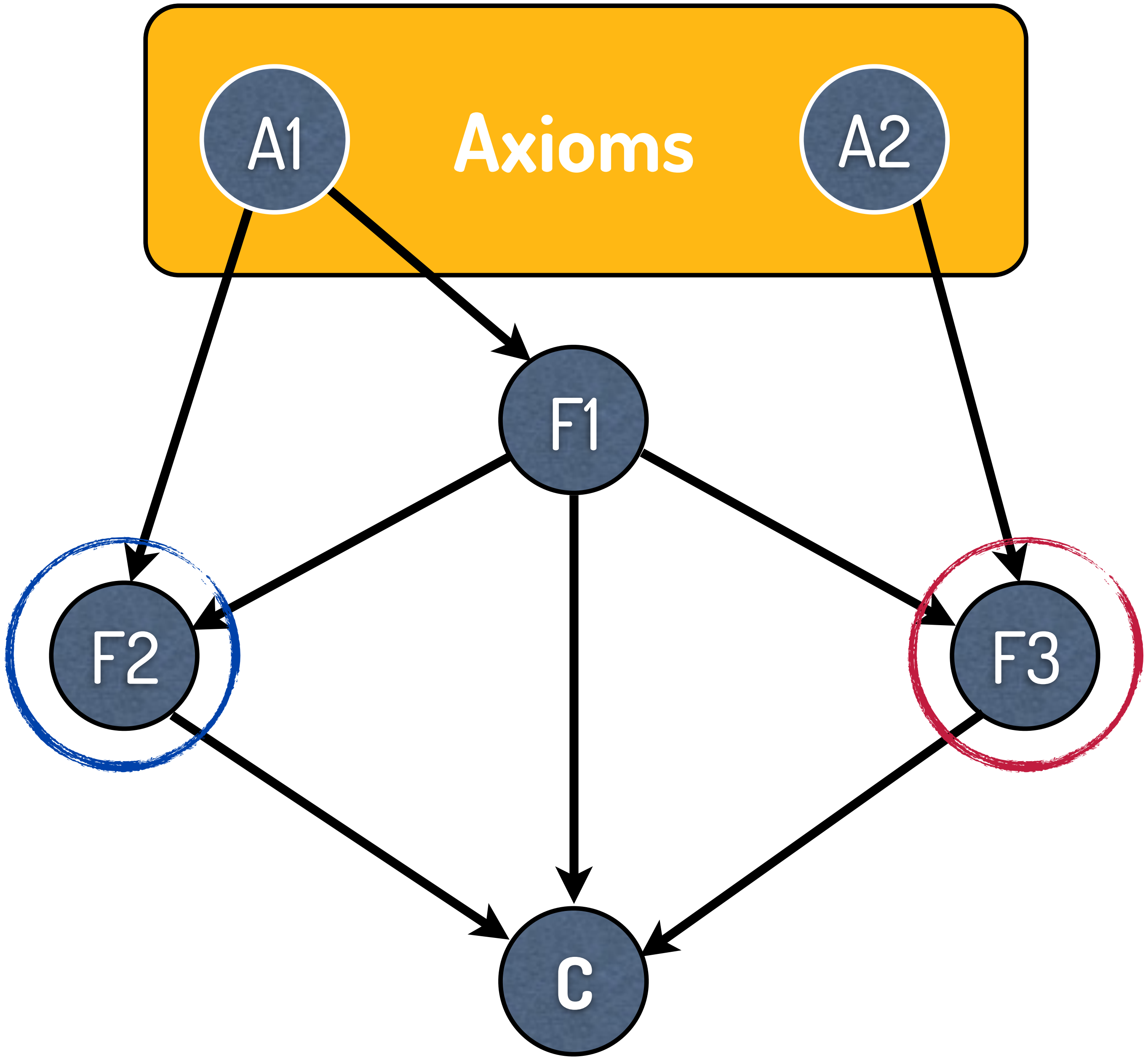


$$A1 \vdash F1$$

$$A1, F1 \vdash F2$$

$$A2, F1 \vdash F3$$

$$F1, F2, F3 \vdash C$$

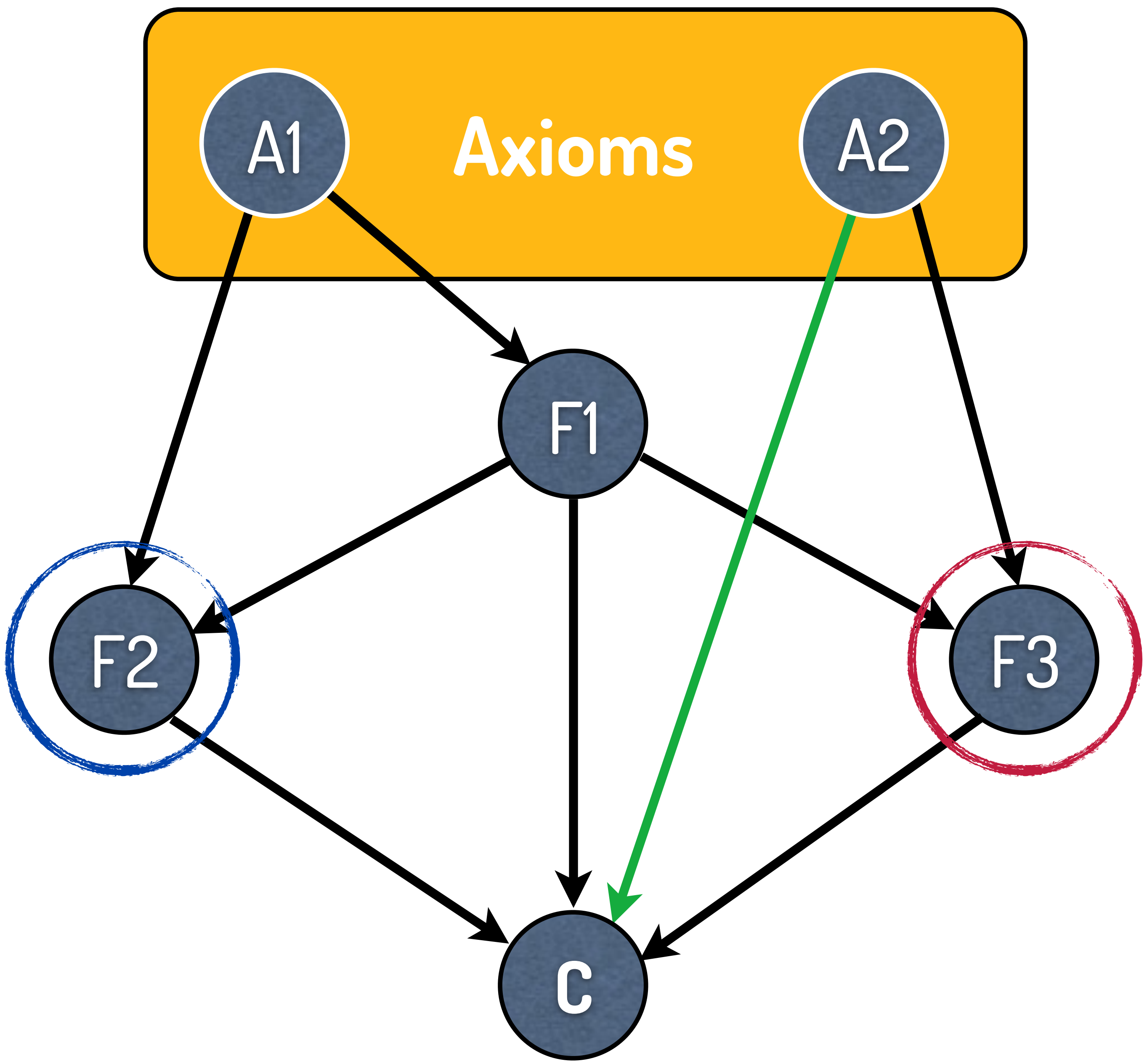


A1 ⊢ F1

A1,F1 ⊢ ***F2***

A2,F1 ⊢ ***F3***

F1,***F2***,***F3*** ⊢ C

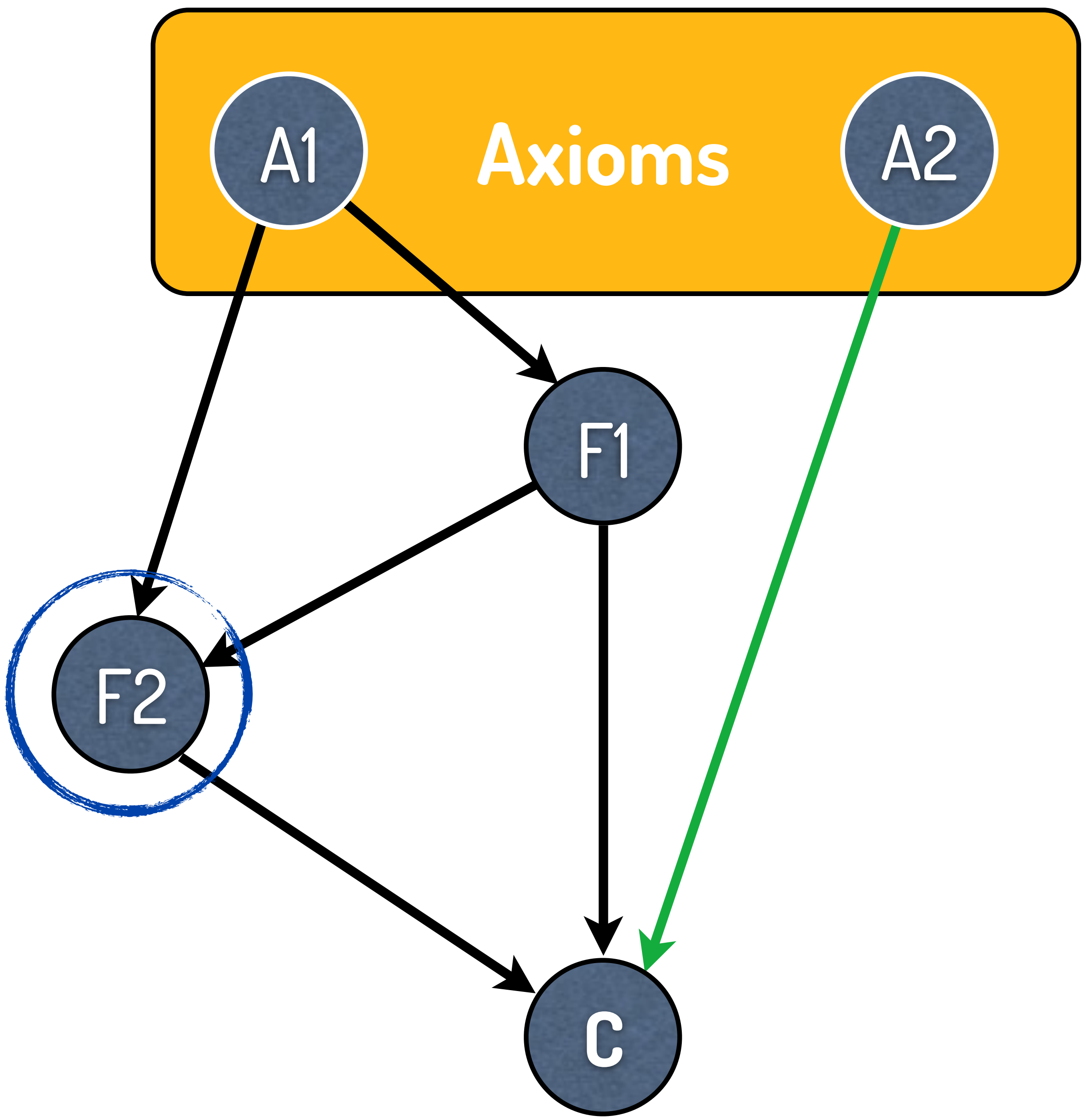


$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

$A2, F1 \vdash \boldsymbol{F3}$

$F1, \boldsymbol{F2}, \boldsymbol{F3} \vdash C$



$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

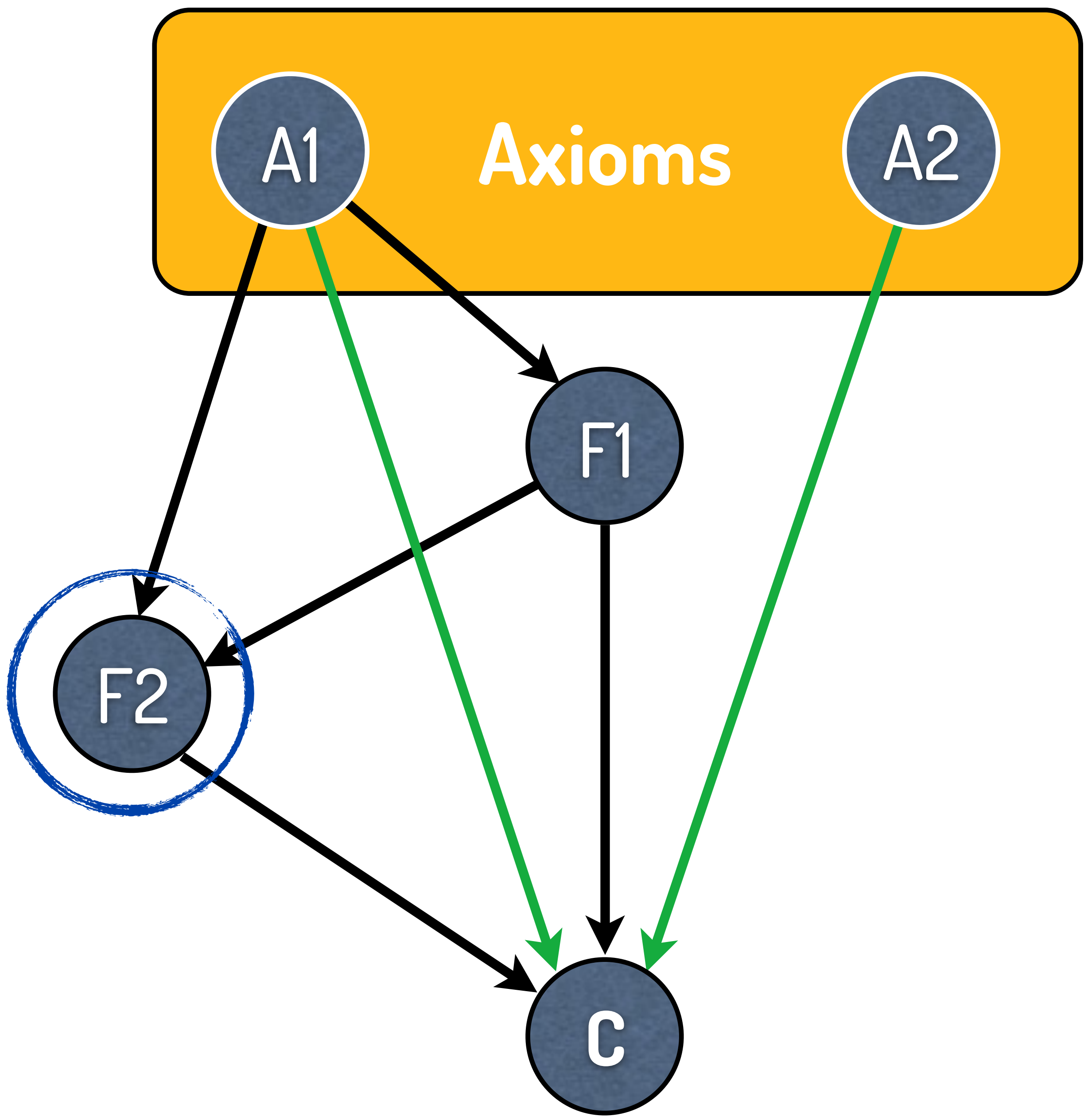
$A2, F1 \vdash \boldsymbol{F3}$

$F1, \boldsymbol{F2}, \boldsymbol{F3} \vdash C$

$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

$F1, \boldsymbol{F2}, A2 \vdash C$



$A1 \vdash F1$

$A1, F1 \vdash \color{blue}{\mathit{F2}}$

$A2, F1 \vdash \color{red}{\mathit{F3}}$

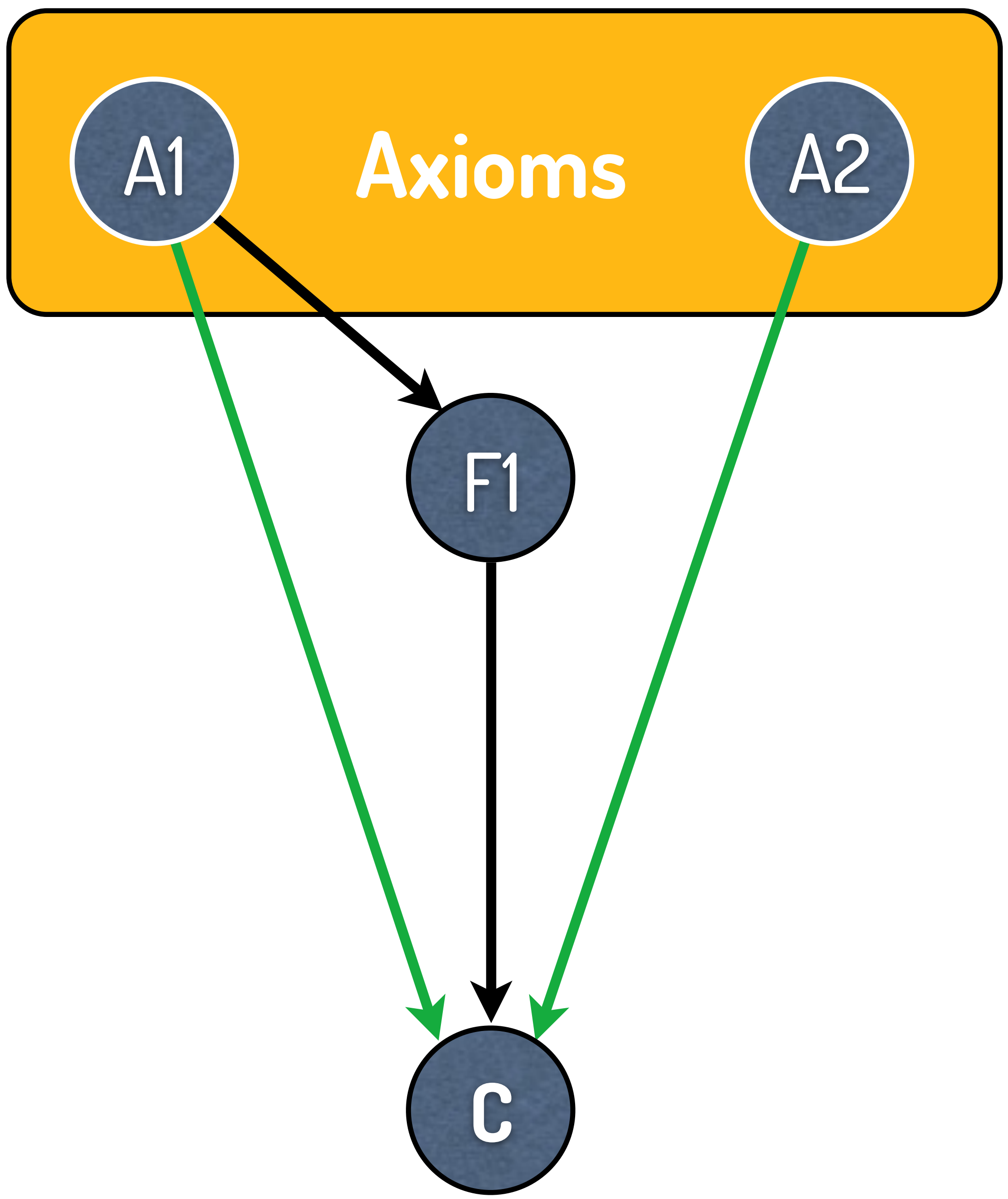
$F1, \color{blue}{\mathit{F2}}, \color{red}{\mathit{F3}} \vdash C$

$\Downarrow$

$A1 \vdash F1$

$A1, F1 \vdash \color{blue}{\mathit{F2}}$

$F1, \color{blue}{\mathit{F2}}, \color{green}{A2} \vdash C$



$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

$A2, F1 \vdash \boldsymbol{F3}$

$F1, \boldsymbol{F2}, \boldsymbol{F3} \vdash C$

↓

$A1 \vdash F1$

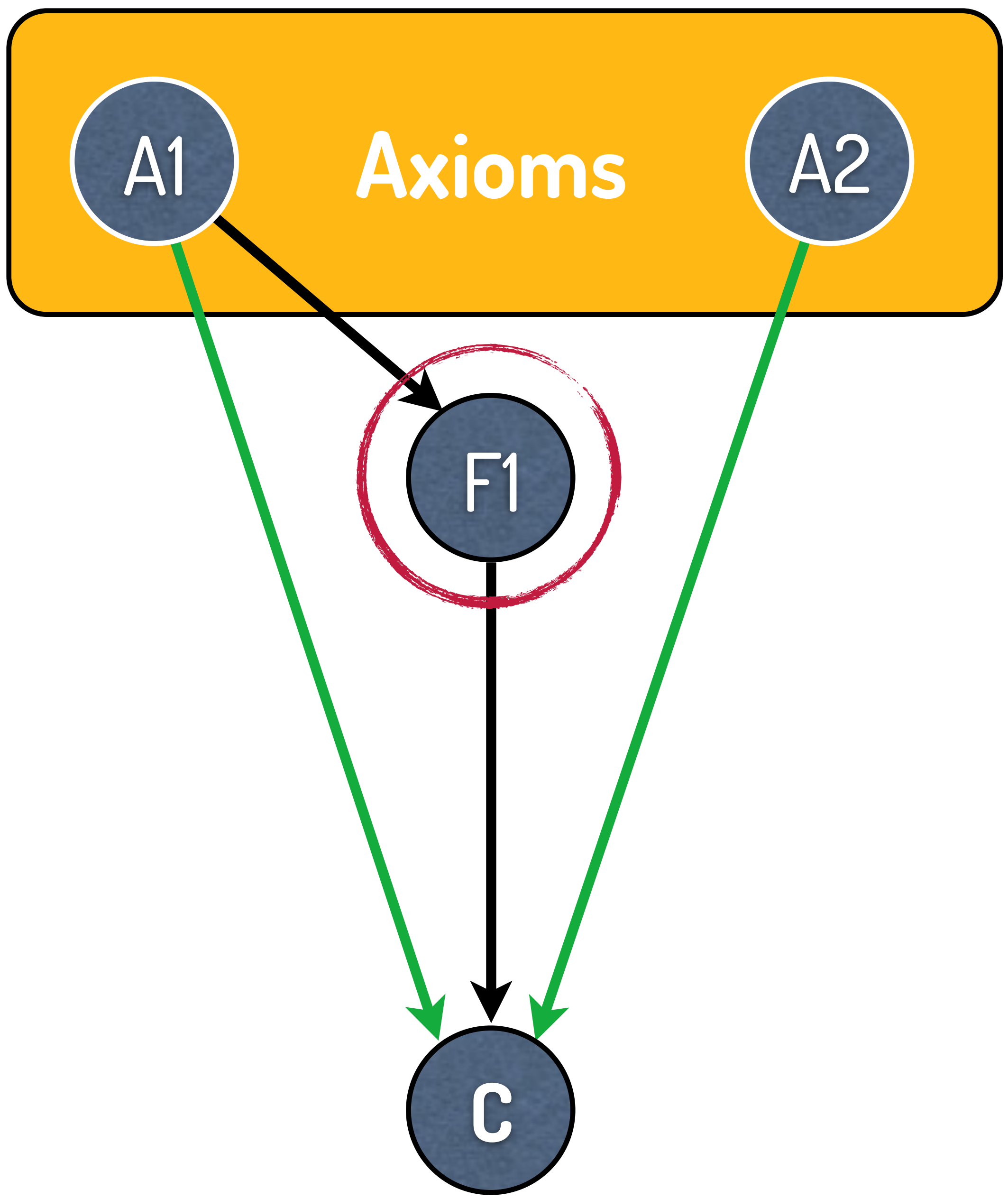
$A1, F1 \vdash \boldsymbol{F2}$

$F1, \boldsymbol{F2}, A2 \vdash C$

↓

$A1 \vdash F1$

$F1, A1, A2 \vdash C$



$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

$A2, F1 \vdash \boldsymbol{F3}$

$F1, \boldsymbol{F2}, \boldsymbol{F3} \vdash C$

$\Downarrow$

$A1 \vdash F1$

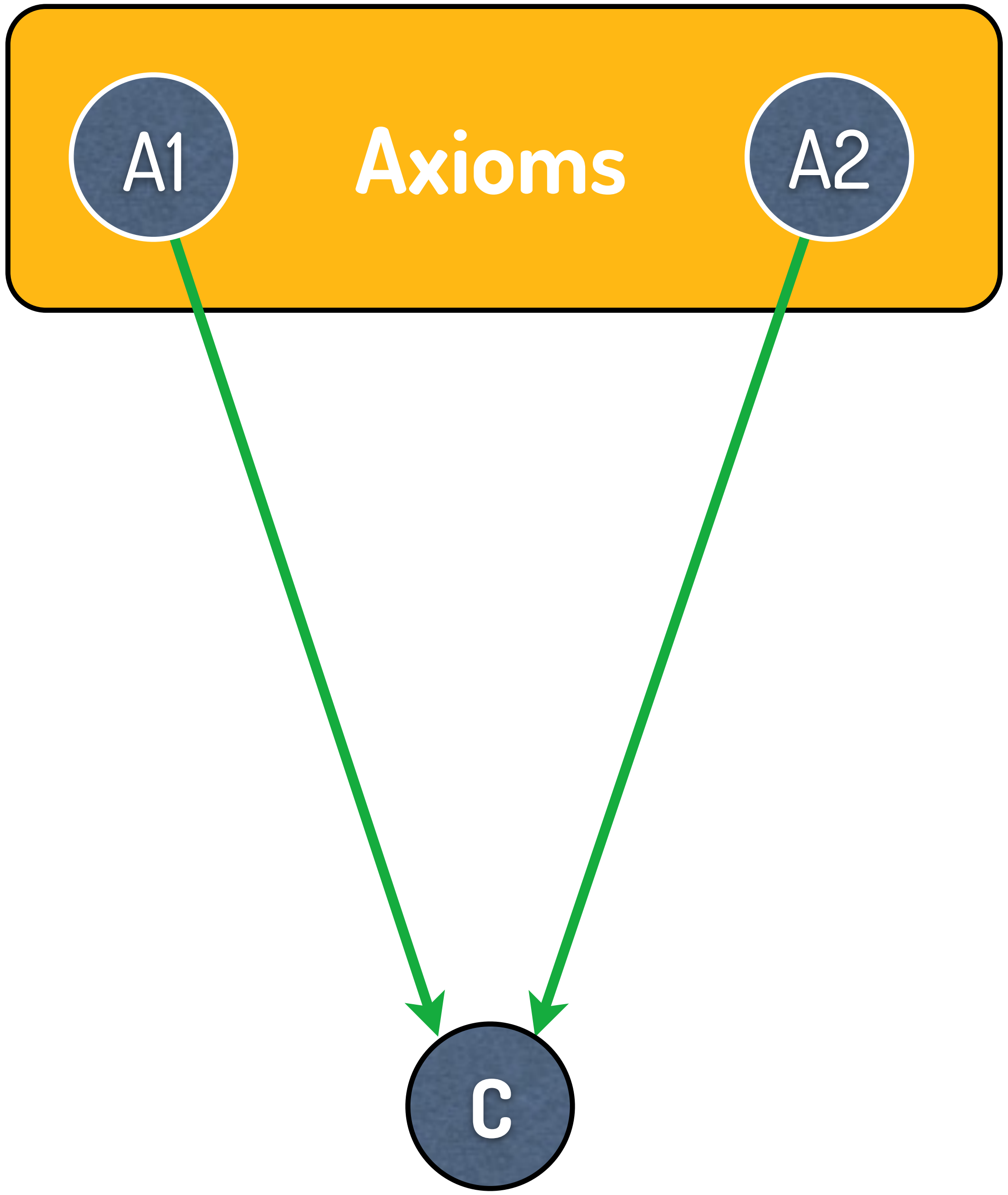
$A1, F1 \vdash \boldsymbol{F2}$

$F1, \boldsymbol{F2}, A2 \vdash C$

$\Downarrow$

$A1 \vdash \boldsymbol{F1}$

$\boldsymbol{F1}, A1, A2 \vdash C$



$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

$A2, F1 \vdash \boldsymbol{F3}$

$F1, \boldsymbol{F2}, \boldsymbol{F3} \vdash C$

$\downarrow$

$A1 \vdash F1$

$A1, F1 \vdash \boldsymbol{F2}$

$F1, \boldsymbol{F2}, A2 \vdash C$

$\downarrow$

$A1 \vdash \boldsymbol{F1}$

$A1, A2 \vdash C \leftarrow \boldsymbol{F1}, A1, A2 \vdash C$

Stop when given compression factor is reached

Stop when given compression factor is reached

Eliminate “large” steps first

Stop when given compression factor is reached

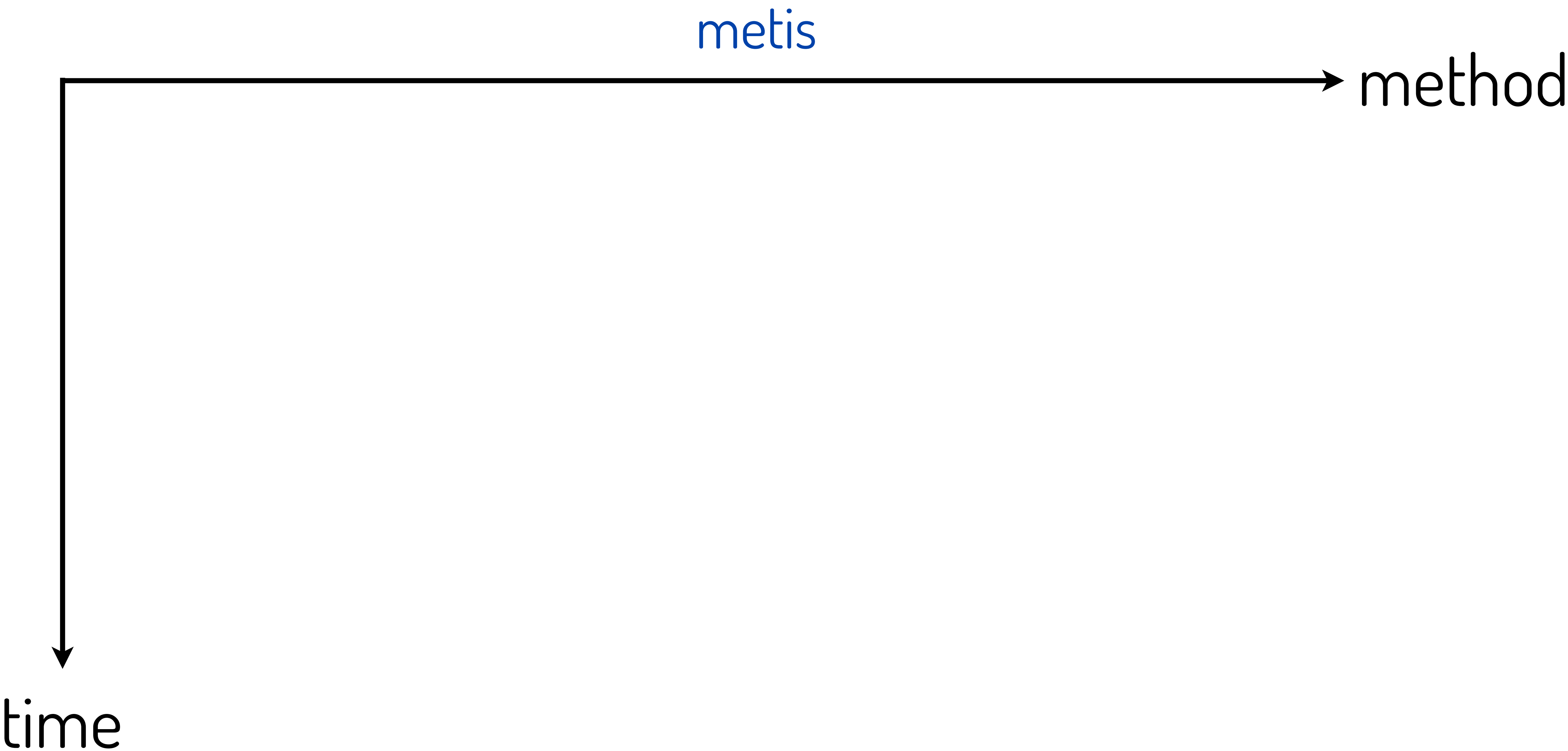
Eliminate “large” steps first

Generalizations:

- eliminate subproofs
- eliminate steps with k successors

Beyond Metis

“Sledgehammer Try0”

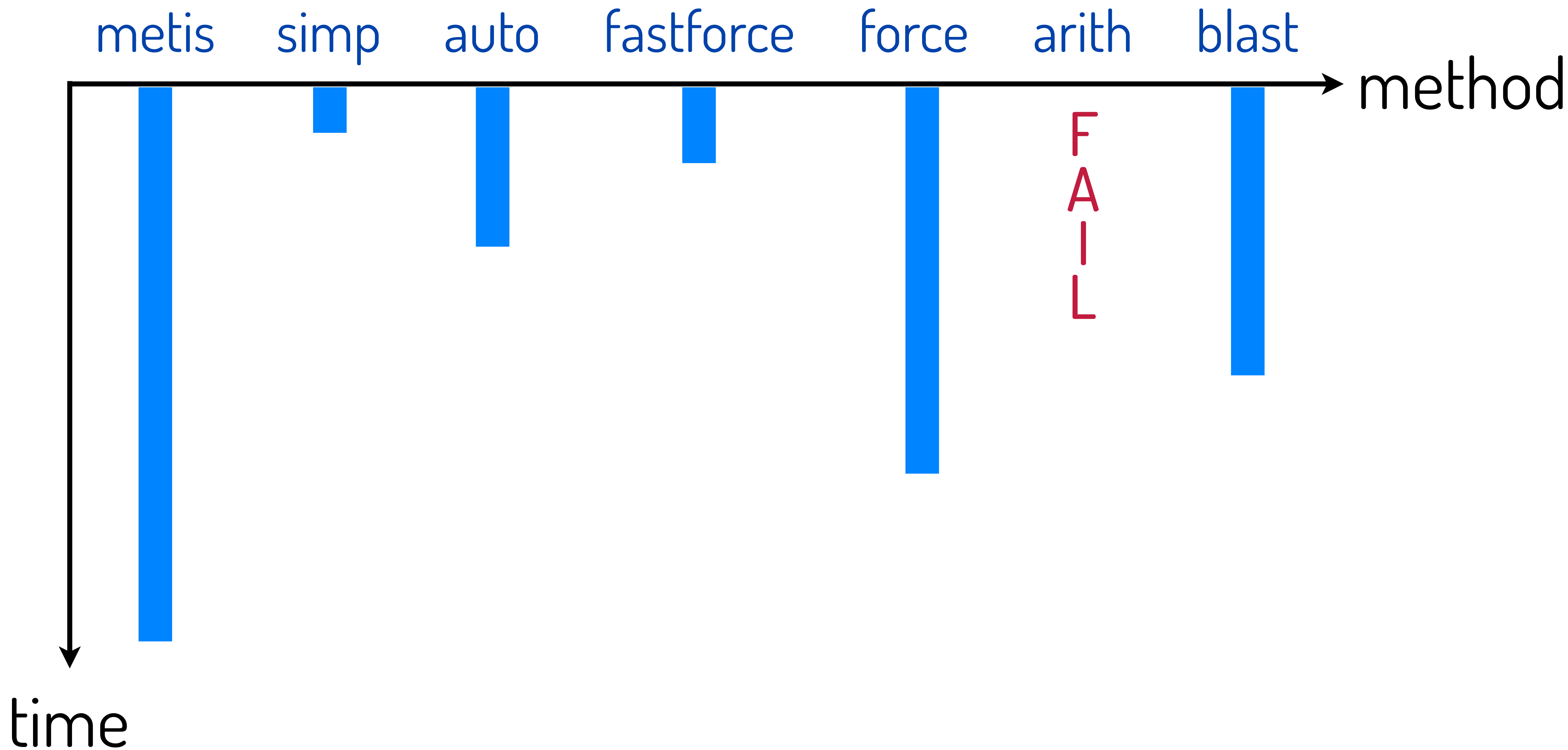


Beyond Metis

“Sledgehammer Try0”

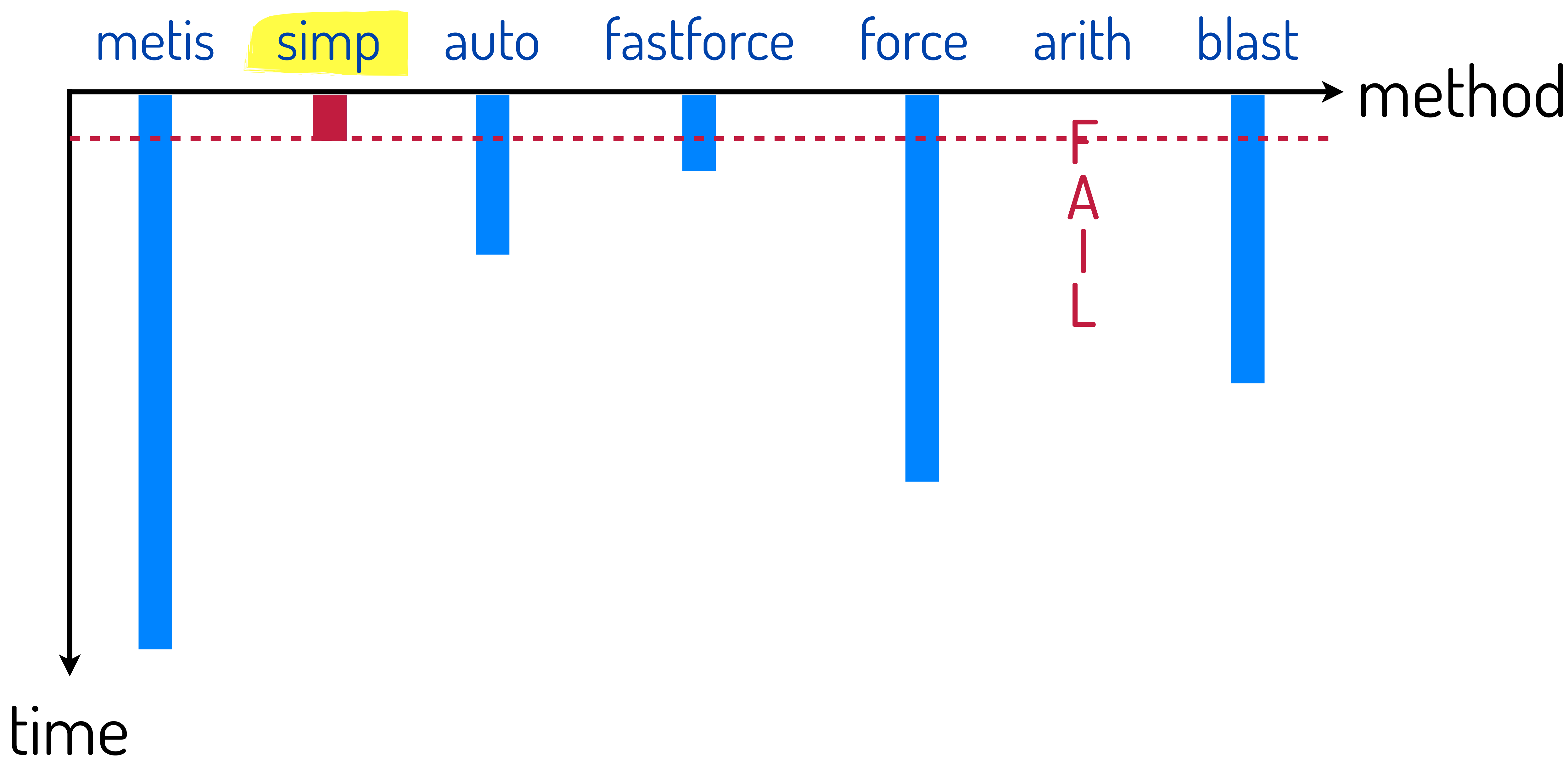
Beyond Metis

“Sledgehammer Try0”



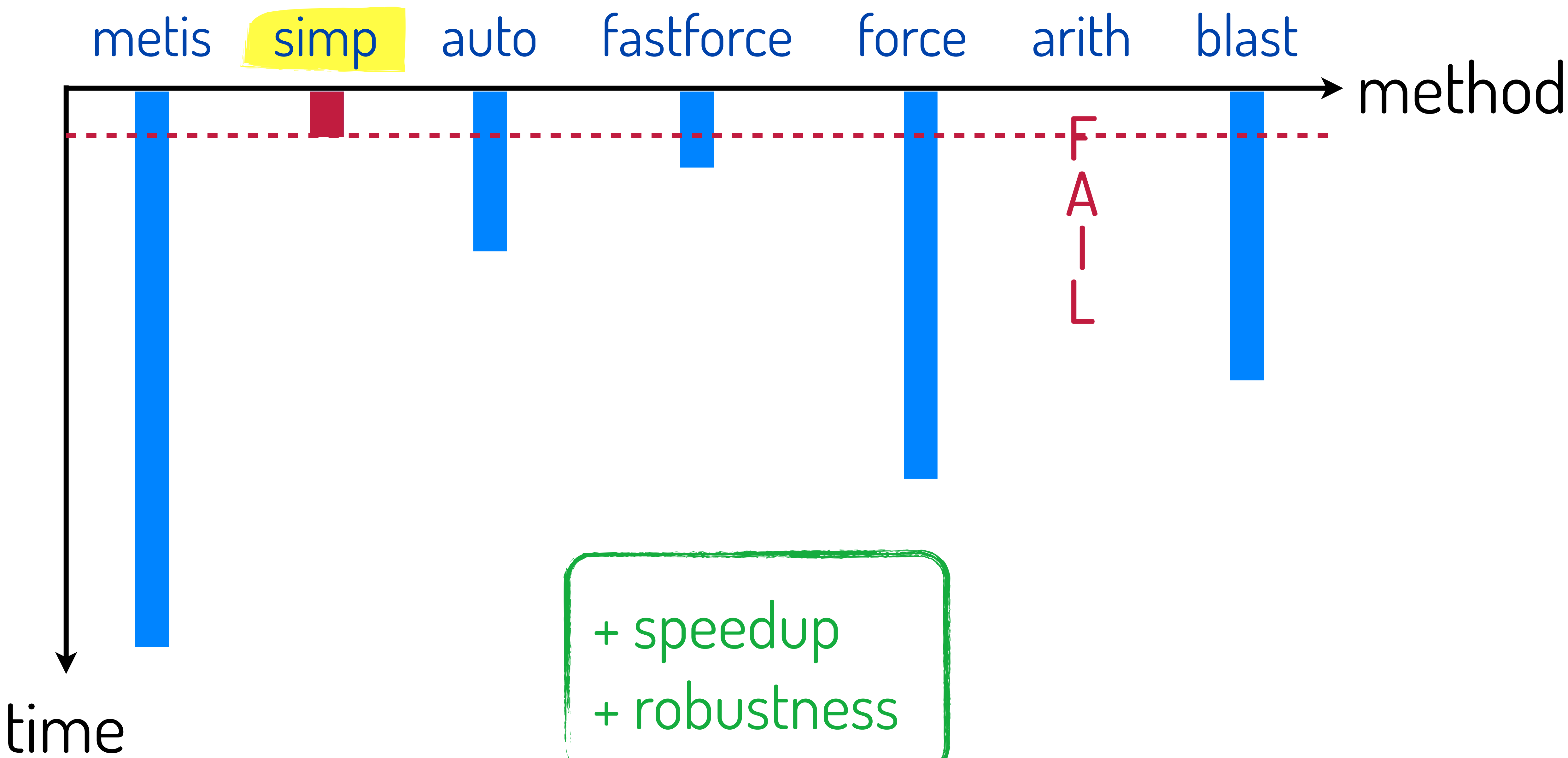
Beyond Metis

“Sledgehammer Try0”



Beyond Metis

"Sledgehammer Try0"



+ speedup
+ robustness

Fact Minimization

Metis knows nothing.
Simp, Auto, ... know about lists, numbers, ...

Fact Minimization

Metis knows nothing.

Simp, Auto, ... know about lists, numbers, ...

```
... using g1 g2 l1 l2 by metis
```

$\downarrow$

```
... using    g2       by simp
```

Fact Minimization

Metis knows nothing.

Simp, Auto, ... know about lists, numbers, ...

```
... using g1 g2 l1 l2 by metis
                ↓
... using    g2       by simp
```

+ may eliminate intermediate steps
+ speedup

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

Sledgehammer

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"

by (metis a0 a1 add_less_cancel_left b0
comm_monoid_add_class.add.right_neutral
comm_monoid_mult_class.mult.left_neutral
comm_semiring_1_class.normalizing_semiring_rules(24)
diff_add_cancel pos_add_strict real_mult_less_iff1)
```

798 ms

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

Sledgehammer

- Isar Proof

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

74 ms

```
proof -
  have "⋀x2 x1. (x2::real) + (x1 - x2) = x1"
    by (metis comm_semiring_1_class.normalizing_semiring_rules(24) diff_add_cancel)
  hence f1: "⋀x1 x2 x3. (x1::real) < x2 - x3 ∨ ¬ x3 + x1 < x2"
    by (metis add_less_cancel_left)
  have f2: "⋀x1 x2. (x1::real) * x2 < x2 ∨ ¬ 0 < x2 ∨ ¬ x1 < 1"
    by (metis comm_monoid_mult_class.mult.left_neutral real_mult_less_iff1)
  have "0 < b ∧ a < 1"
    by (metis a1 b0)
  hence "a * b < b"
    using f2 by metis
  hence "0 < a ∧ a * b < b"
    by (metis a0)
  hence "a * b < a + b"
    by (metis pos_add_strict)
  hence "a * b + 0 < a + b"
    by (metis comm_monoid_add_class.add.right_neutral)
  thus "0 < a + b - a * b"
    using f1 by metis
qed
```

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

Sledgehammer

- Isar Proof
- Compression

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"

proof -
  have "a * b < b"
    by (metis a1 b0 mult_strict_right_mono
        comm_semiring_1_class.normalizing_semiring_rules(11))
  hence "a * b < a + b"
    by (metis a0 pos_add_strict)
  thus "0 < a + b - a * b"
    by (metis add_less_imp_less_right diff_add_cancel
        comm_semiring_1_class.normalizing_semiring_rules(5))
qed
```

25 ms

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

Sledgehammer

- Isar Proof
- Compression
- Try0 & Fact Minimization

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"
```

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"

proof -
  have "a * b < b"
    using a1 b0 by simp
  hence "a * b < a + b"
    using a0 pos_add_strict by simp
  thus "0 < a + b - a * b"
    by simp
qed
```

5 ms

```
lemma
  fixes a :: real and b :: real
  assumes a0: "0<a" and a1: "a<1"
      and b0: "0<b" and b1: "b<1"
  shows "a+b - a*b > 0"

proof -
  have "a * b < b"
    using a1 b0 by simp
  hence "a * b < a + b"
    using a0 pos_add_strict by simp
  thus "0 < a + b - a * b"
    by simp
qed
```

5 ms

```
sledgehammer[isar_proofs, isar_compress=2]
```

Robust, Semi-Intelligible Isabelle Proofs from ATP Proofs

Steffen Smolka
Advisor: Jasmin Blanchette