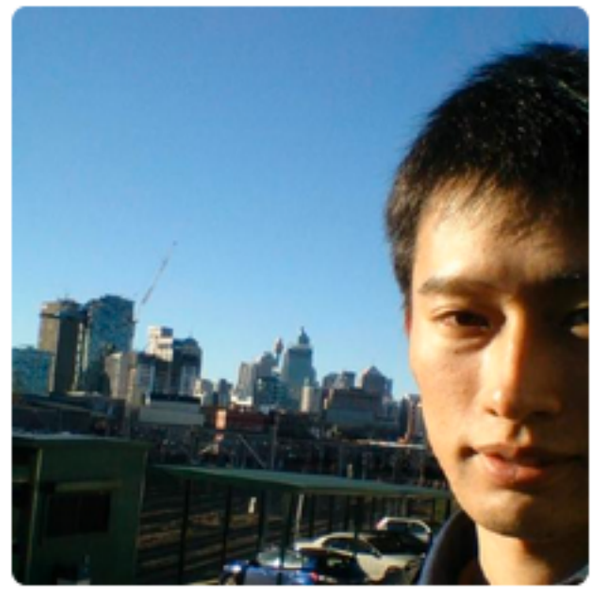
https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# AI for theorem proving in Isabelle/HOL

*This work was supported by the project AI&Reasoning (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000466).*

Yutaka Nagashima
University of Innsbruck
Czech Technical University



**Yutaka Ng**
yutakang

Block or report user

CVUT, CTU, CIIRC

## Mathematics

Number Theory

Analysis

Algebra

Geometry

Probability Theory

etc.

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic

## Mathematics

Number Theory

Analysis

Algebra

Geometry

Probability Theory

etc.

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

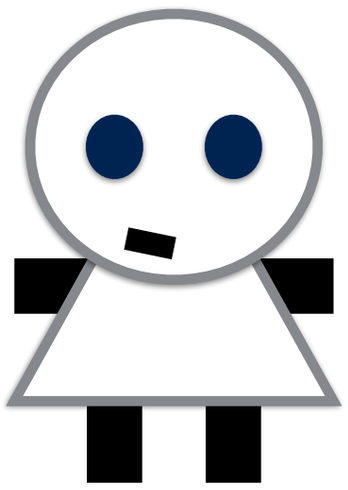
Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic



## Mathematics

Number Theory

Analysis

Algebra

Geometry

Probability Theory

etc.

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

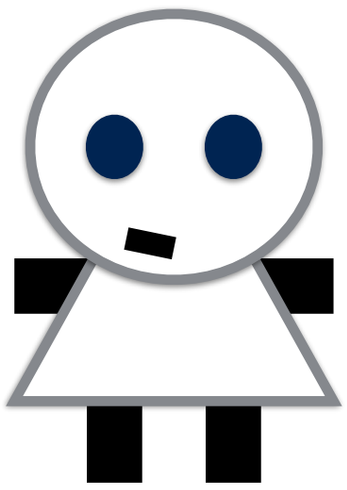
Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic



A tiny field inside Informatics. Who cares?

https://twitter.com/YutakangE

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic

https://twitter.com/YutakangE

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

Algorithms

Data Structures

Architecture

Software Engineering

### Formal Method

Computational Logic

## Mathematics The Language of Science.

Analysis Algebra Geometry Probability Theory

https://twitter.com/YutakangE

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic

## Mathematics The Language of Science.

Analysis Algebra Geometry Probability Theory

## Logic: the foundation of Mathematics.

https://twitter.com/YutakangE

Chemistry

Electronics

etc.

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic

## Mathematics The Language of Science.

Analysis Algebra Geometry Probability Theory

## Logic: the foundation of Mathematics.

https://twitter.com/YutakangE

Chemistry

Electronics

etc.

## Physics

Acoustics

Astrophysics

Electromagnetism

Molecular Physics

Quantum Physics

etc.

## Informatics

Language

Algorithms

Data Structures

Architecture

Software Engineering

**Formal Method**
Computational Logic

## Mathematics The Language of Science.

Analysis Algebra Geometry Probability Theory

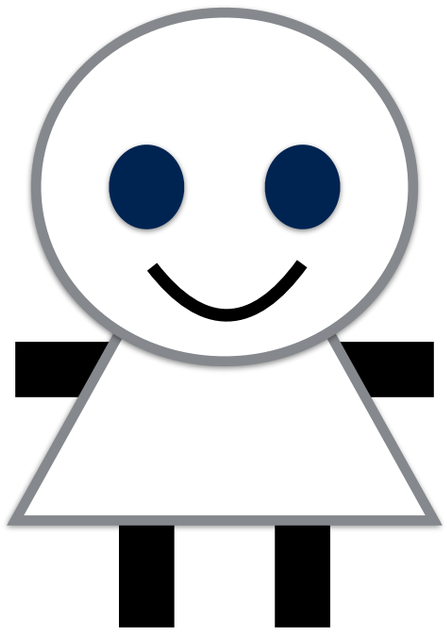
## Logic: the foundation of Mathematics.

Automate Logic using AI to Accelerate Science!

https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL



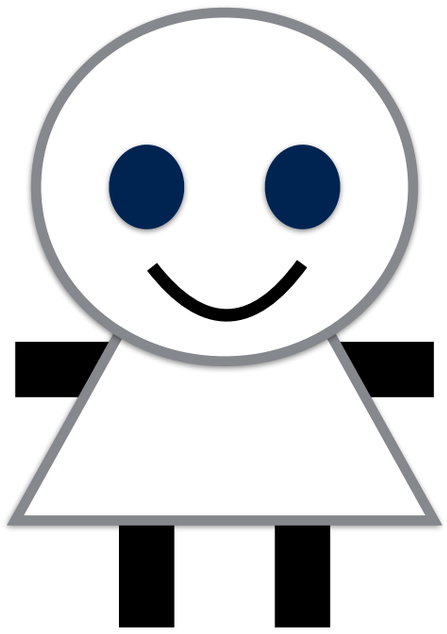
https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL

**proof goal** **context**

**tactic / proof method**



https://twitter.com/YutakangE

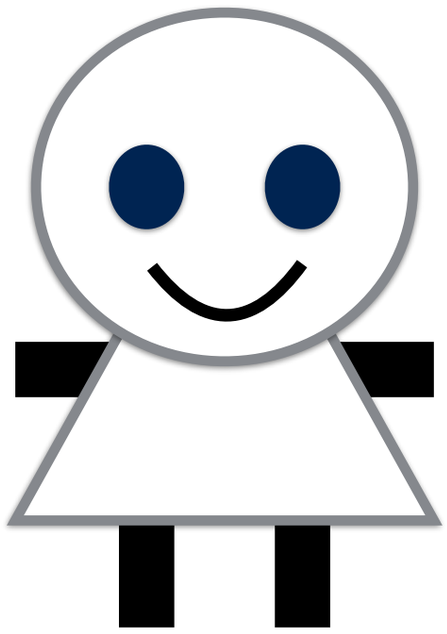
git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL

proof goal

context

tactic / proof method




error-message

subgoals

https://twitter.com/YutakangE

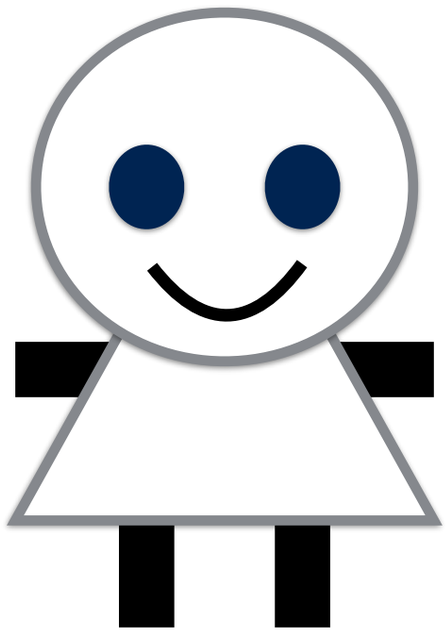
git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL

proof goal

context

tactic / proof method




error-message

subgoals

https://twitter.com/YutakangE

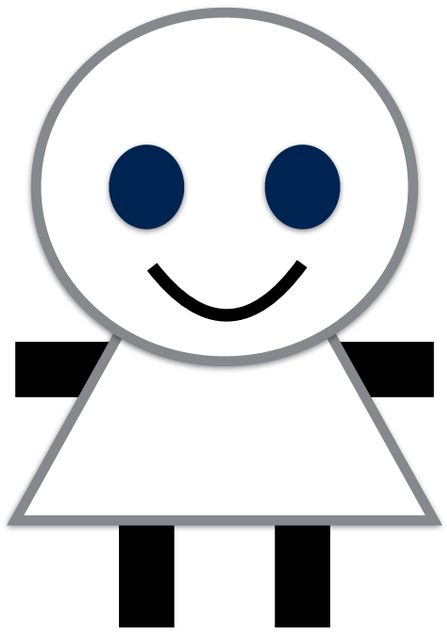
git clone https://github.com/data61/PSL

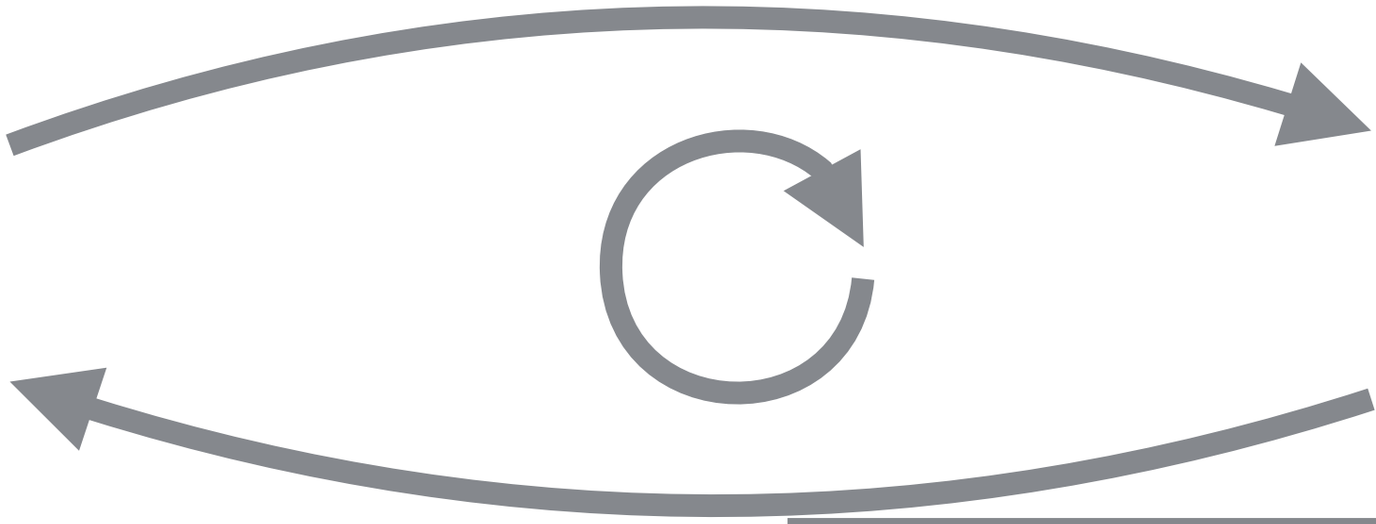
# Interactive theorem proving with Isabelle/HOL

proof goal

context

tactic / proof method






error-message

subgoals

no sub-goal!

https://twitter.com/YutakangE

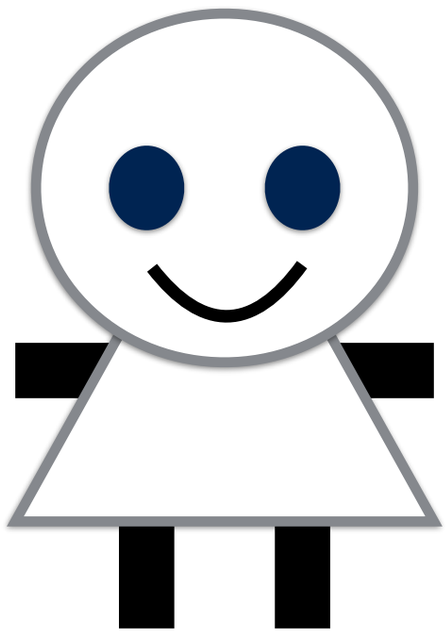
git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL

proof goal

context

tactic / proof method




error-message

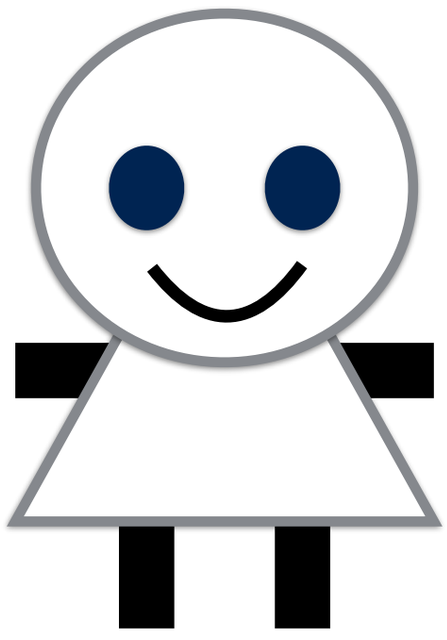
subgoals

no sub-goal!

https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

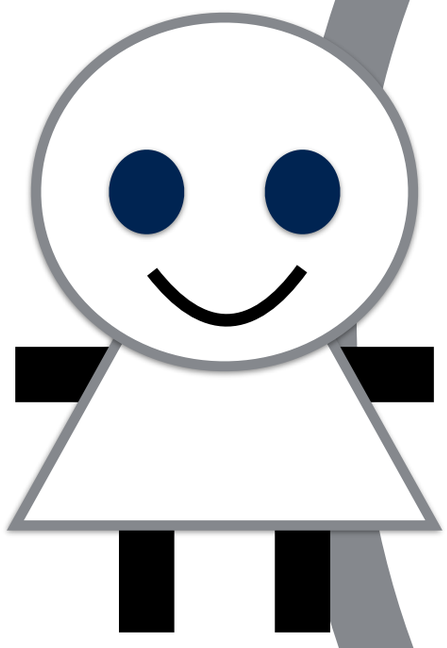
# Interactive theorem proving with Isabelle/HOL




https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL



proof goal

context

tactic / proof method


error-message

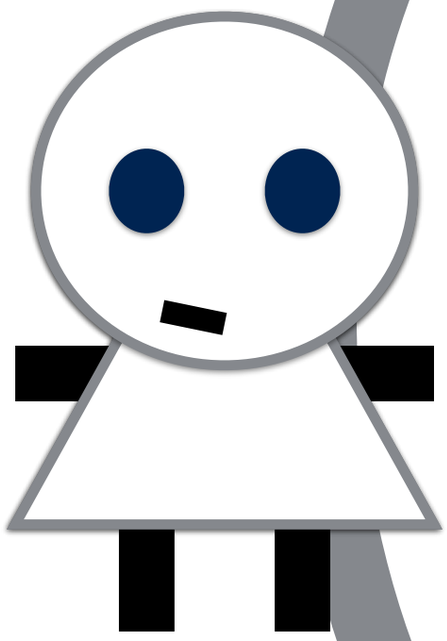
subgoals

no sub-goal!

https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL



proof goal

context

tactic / proof method


error-message

subgoals

no sub-goal!

https://twitter.com/YutakangE

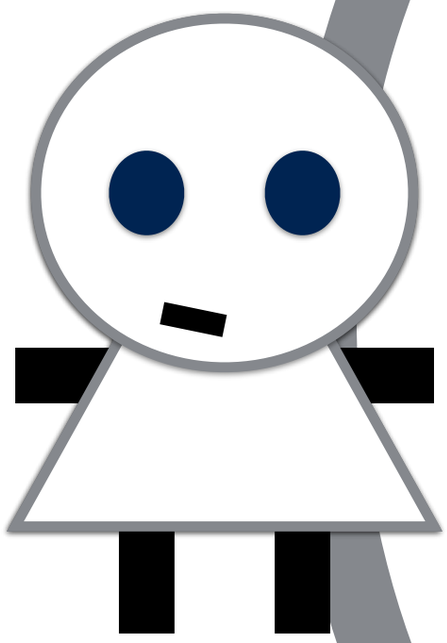
git clone https://github.com/data61/PSL

# Interactive theorem proving with Isabelle/HOL

proof goal

context

tactic / proof method



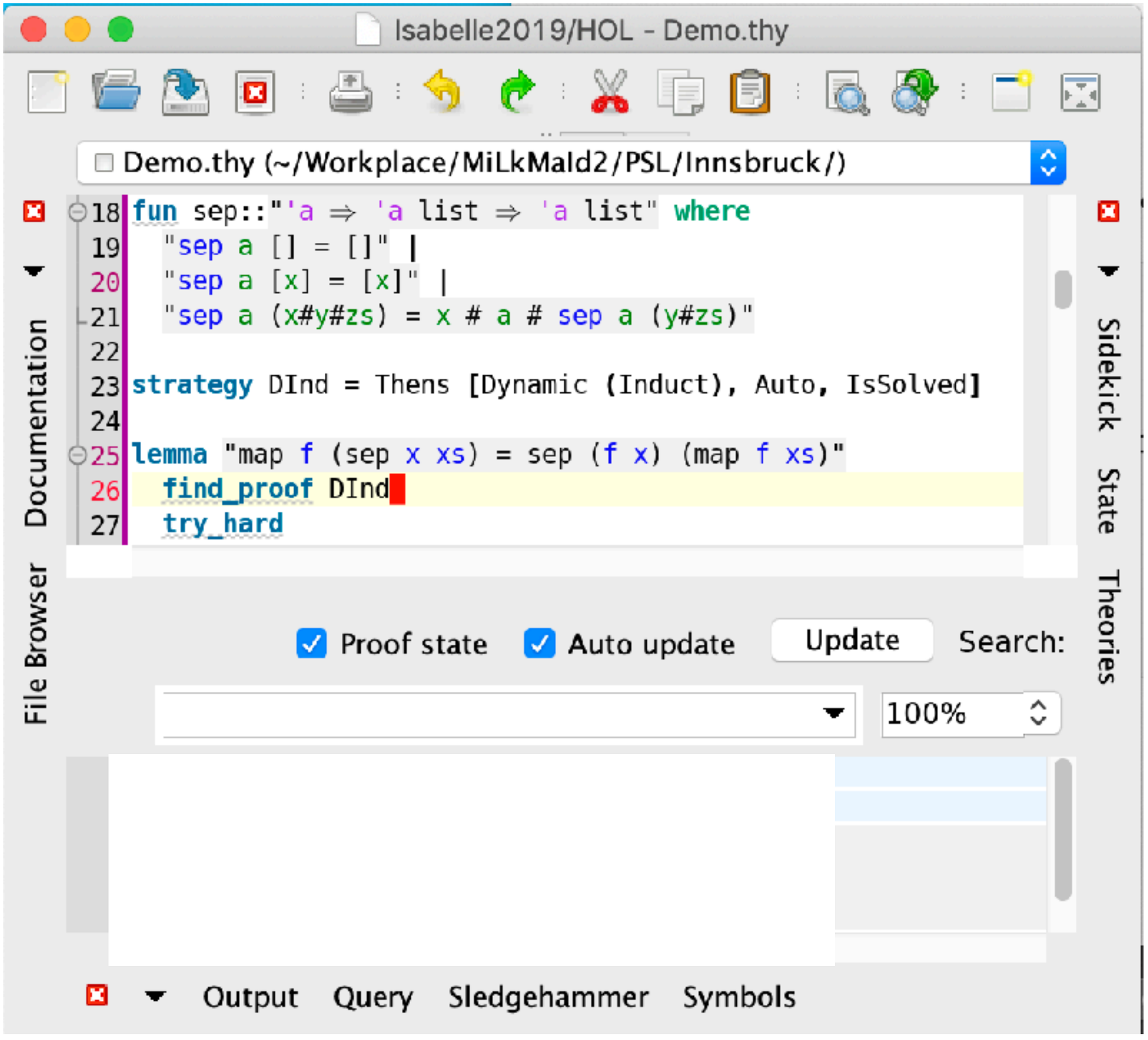

error-message

o-goal!

```
It's blatantly clear
You stupid machine, that what
I tell you is true
(Michael Norrish)
```

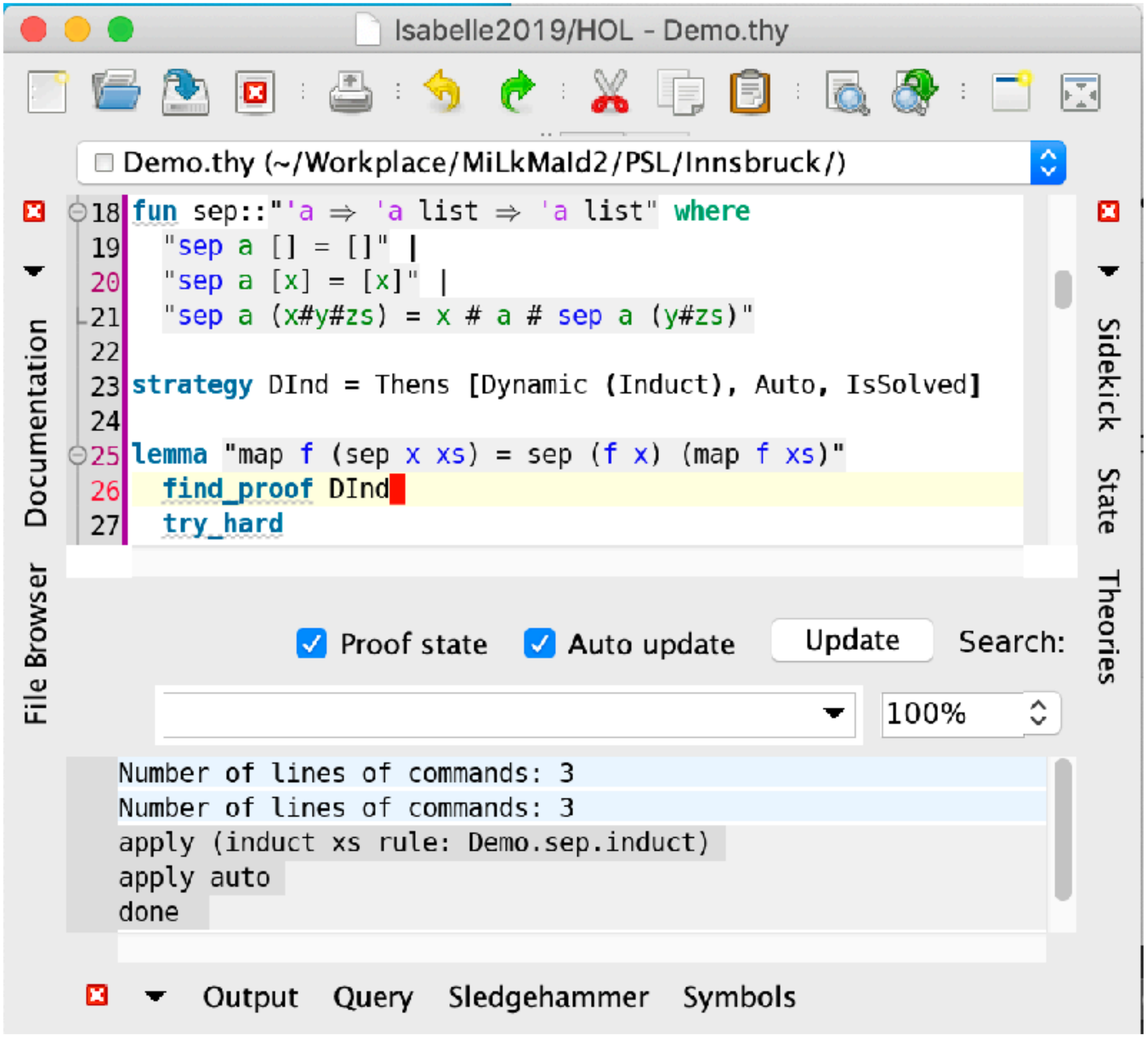
https://twitter.com/YutakangE

git clone https://github.com/data61/PSL



https://twitter.com/YutakangE

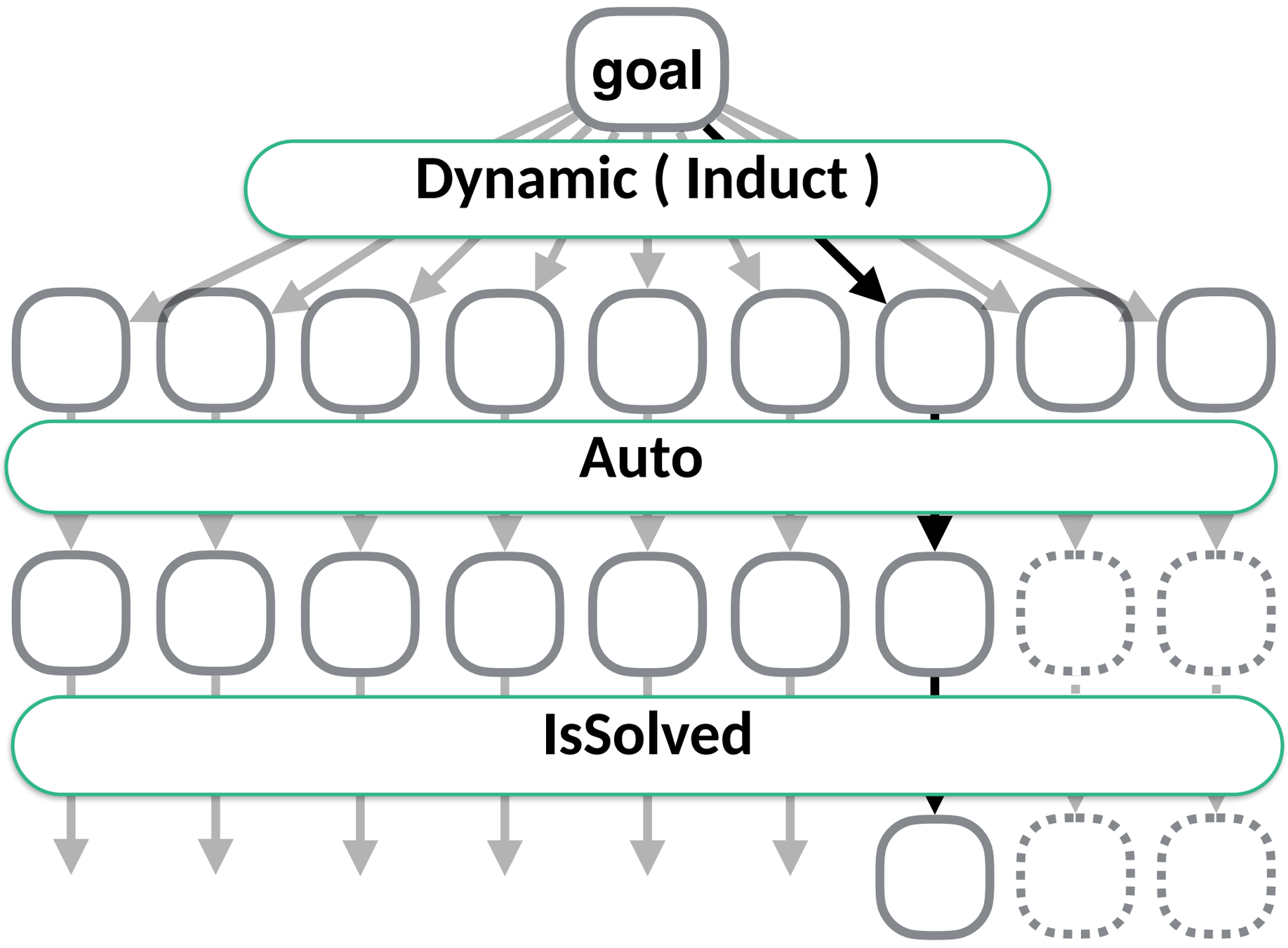
git clone https://github.com/data61/PSL



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

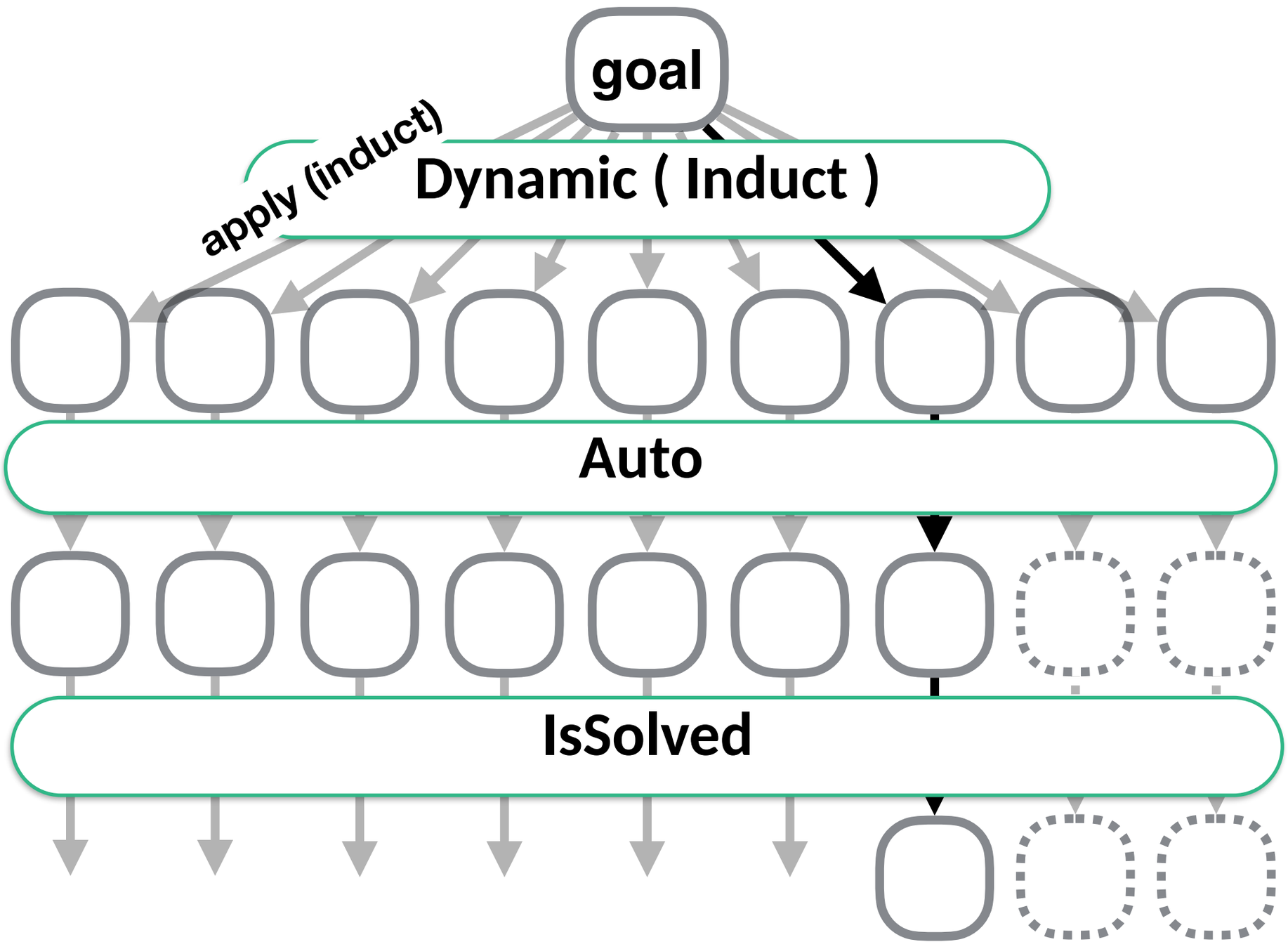
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

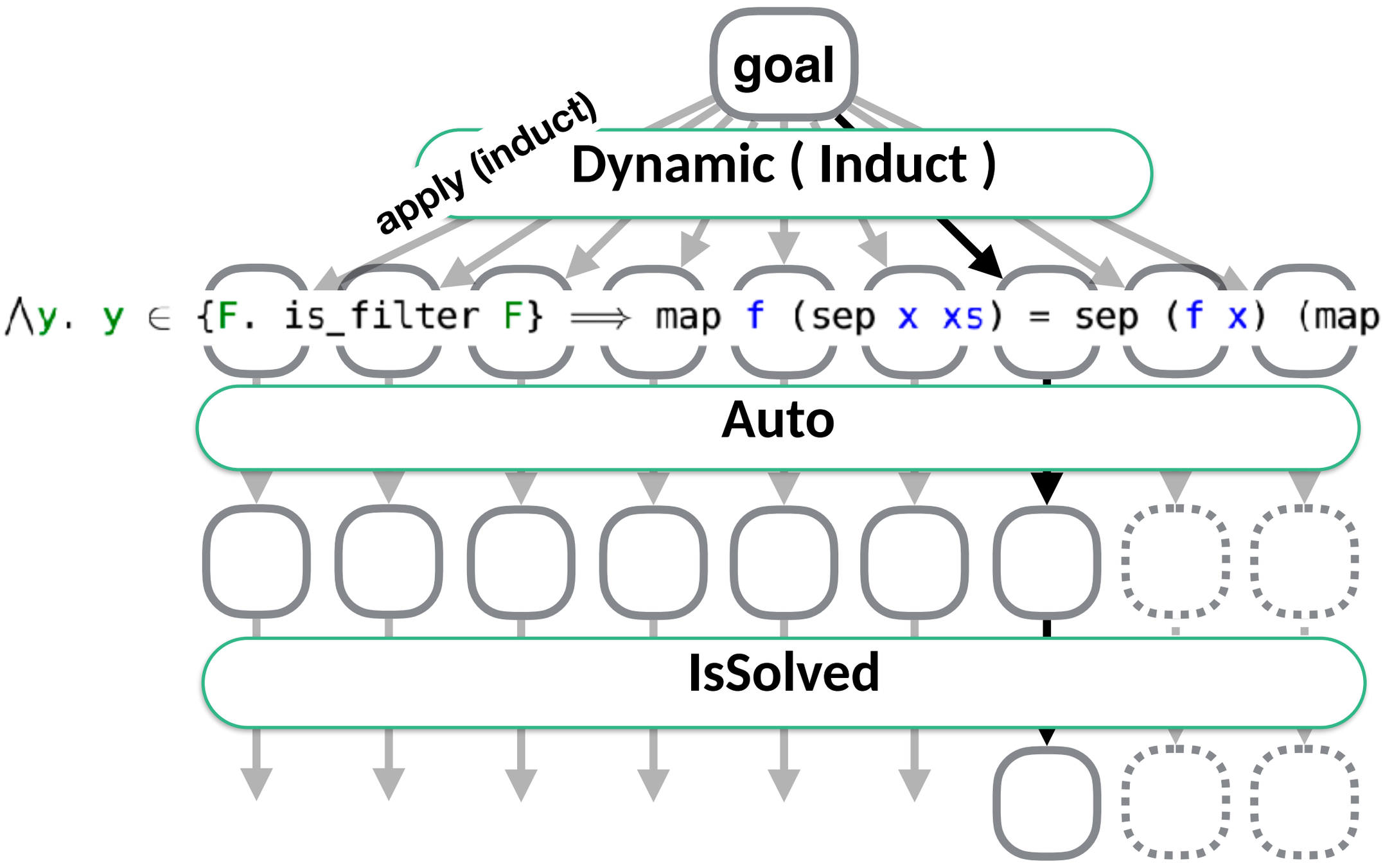
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

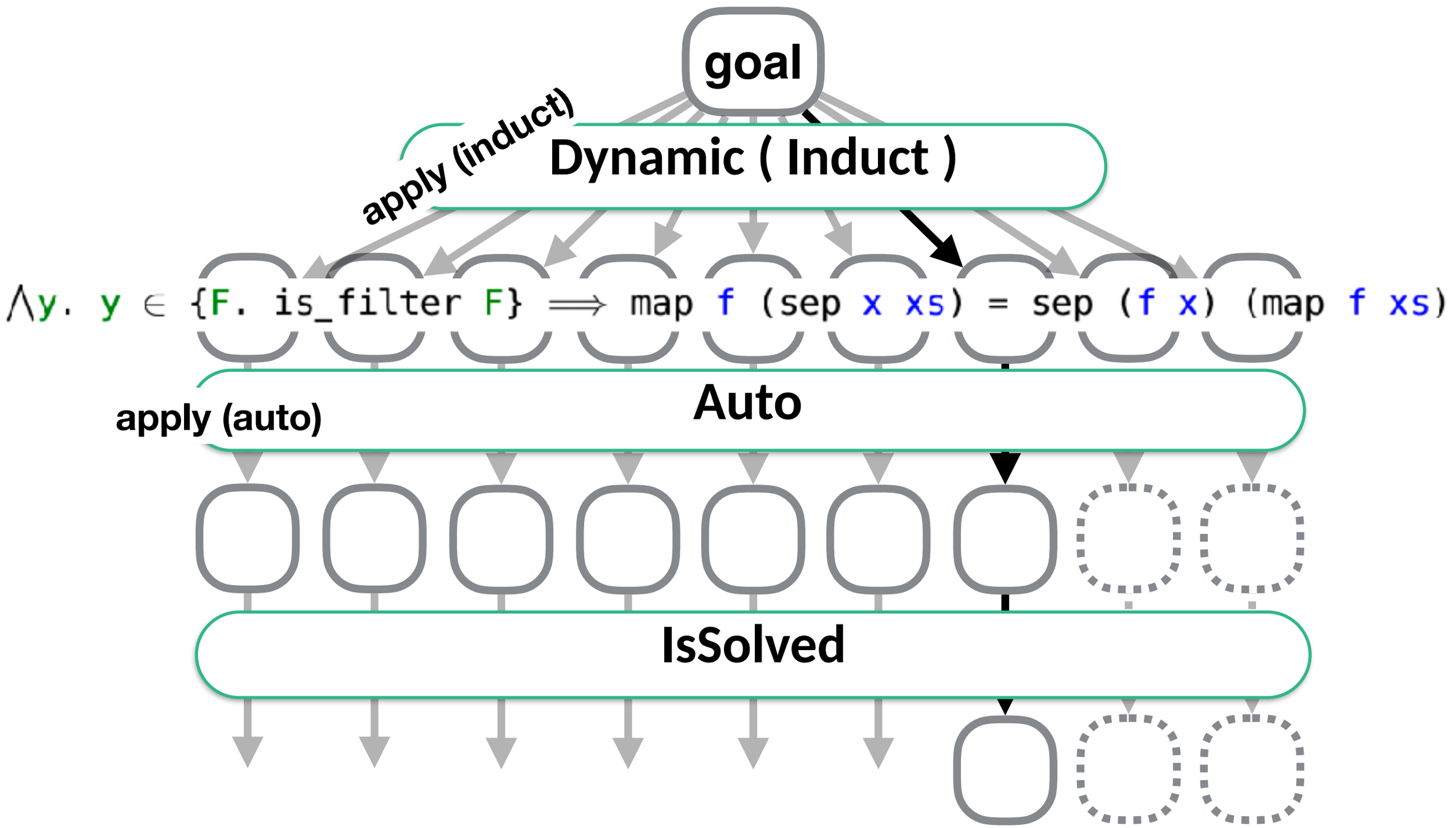
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"
find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

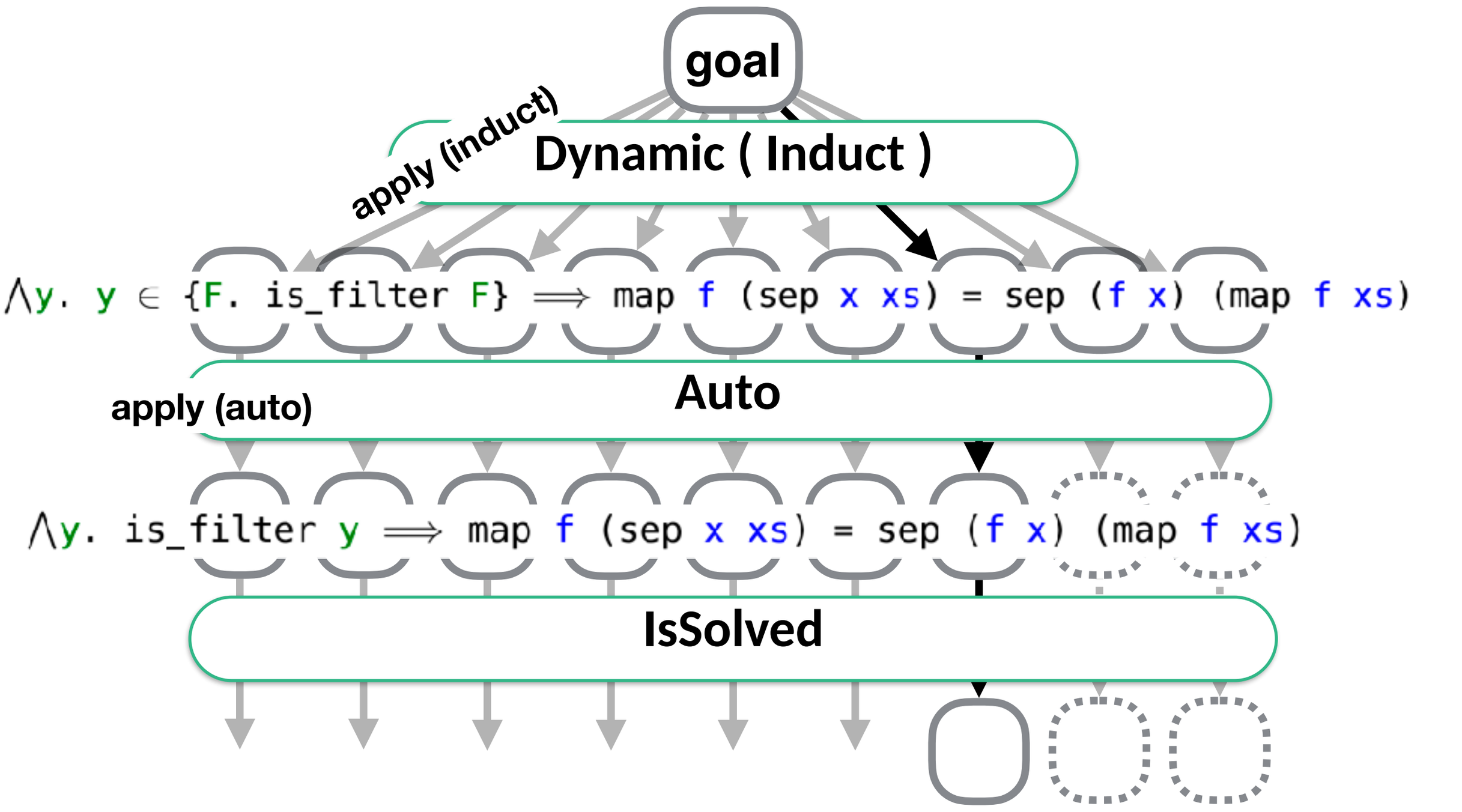
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

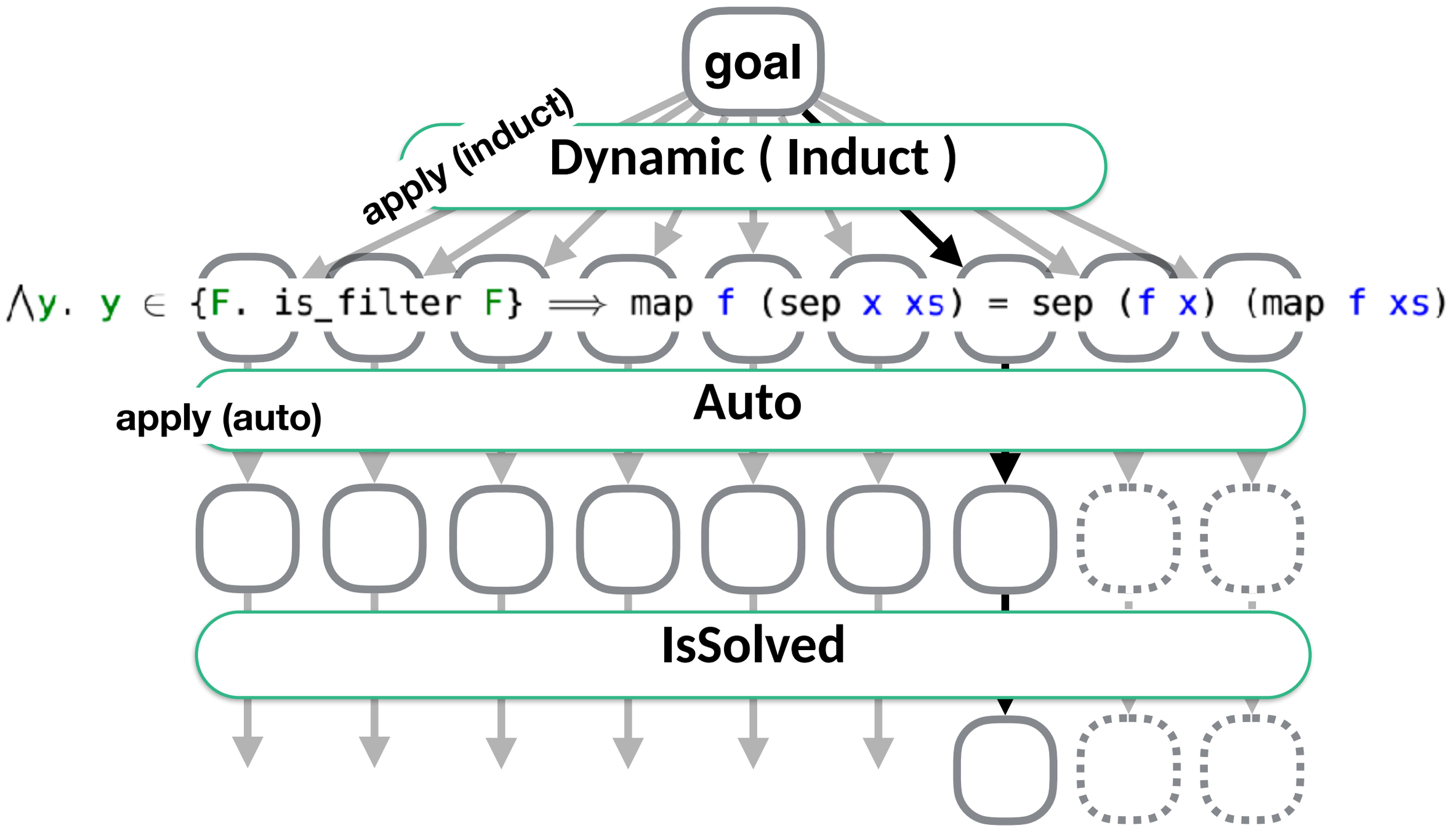
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"
find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

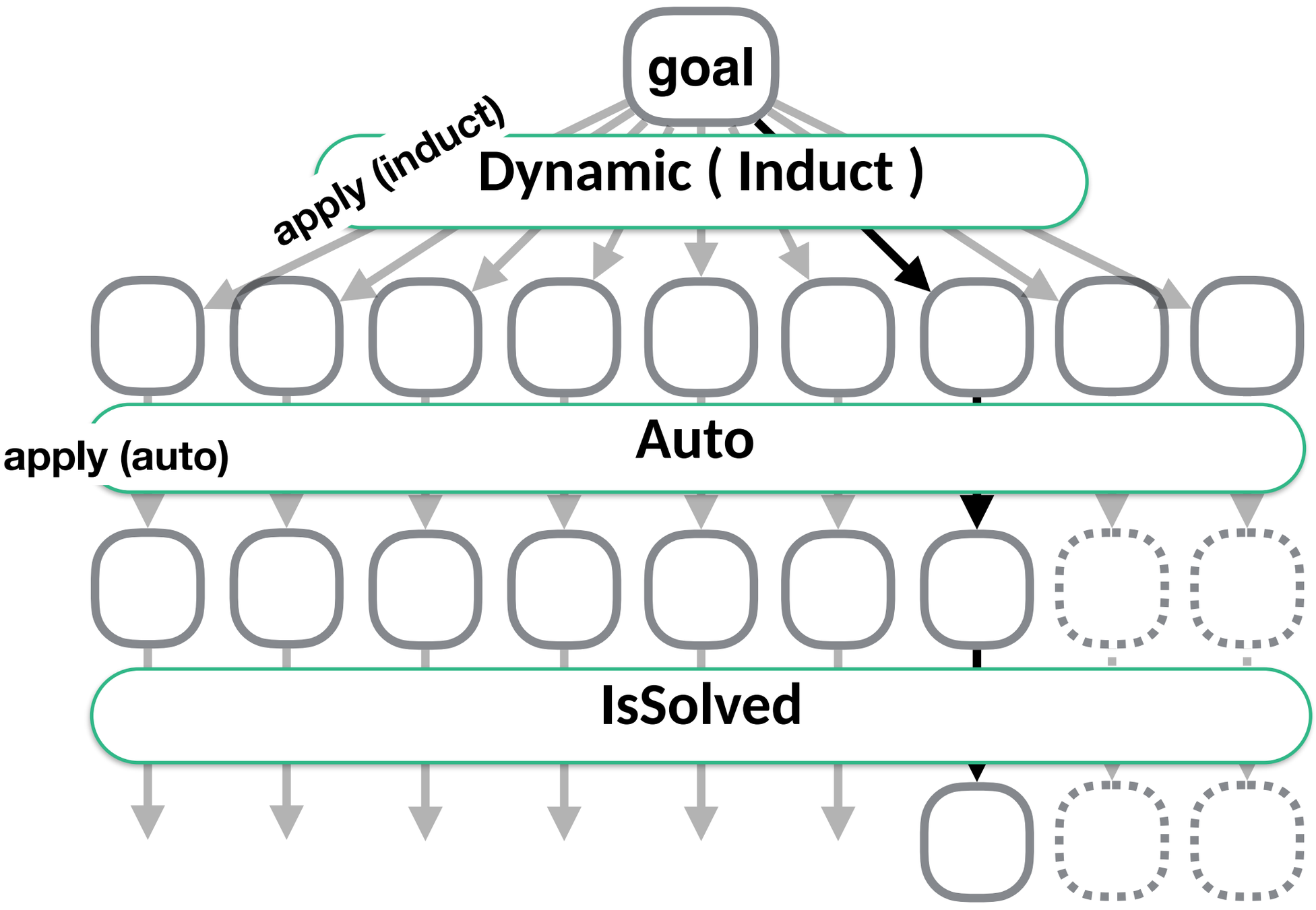
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

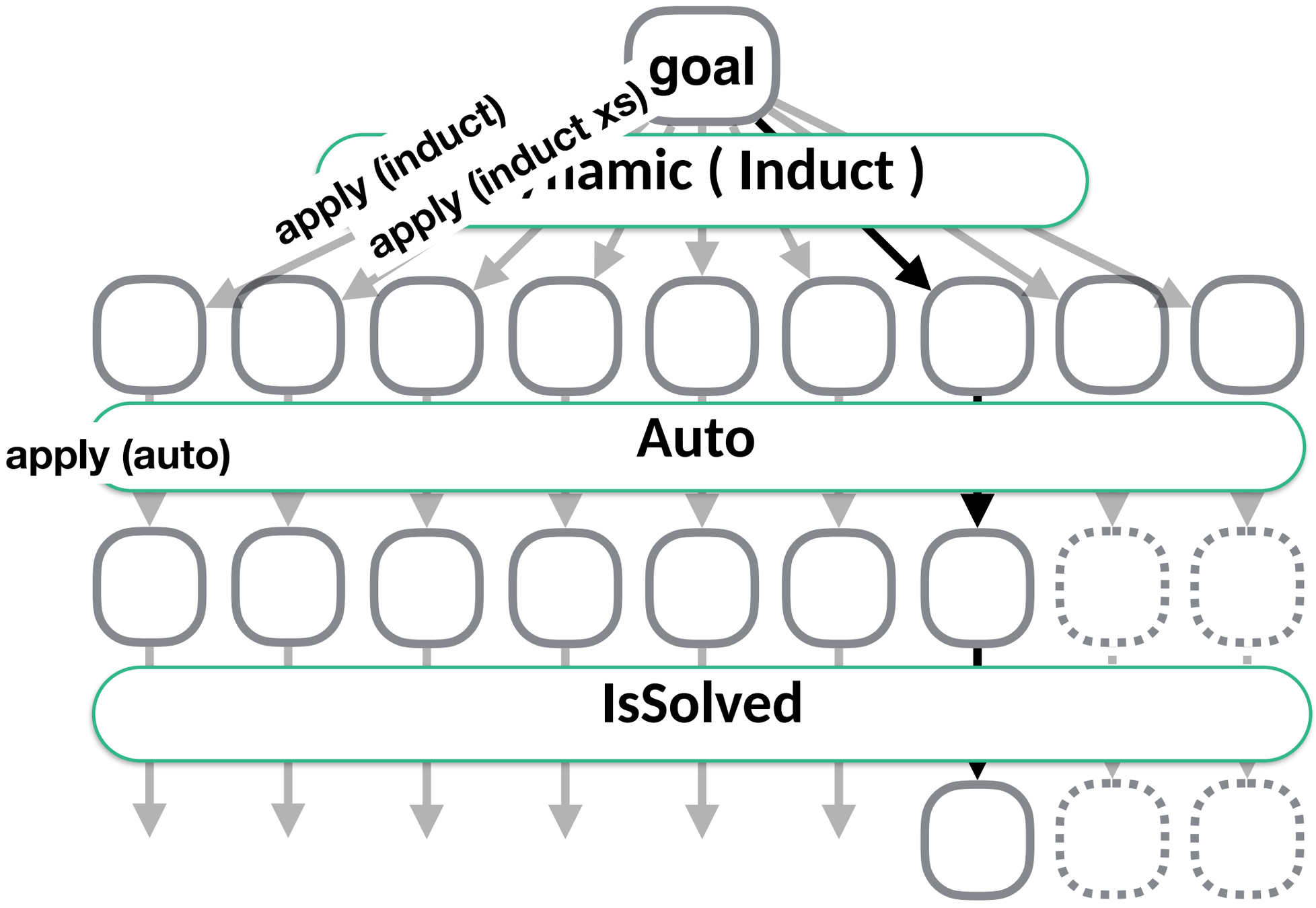
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

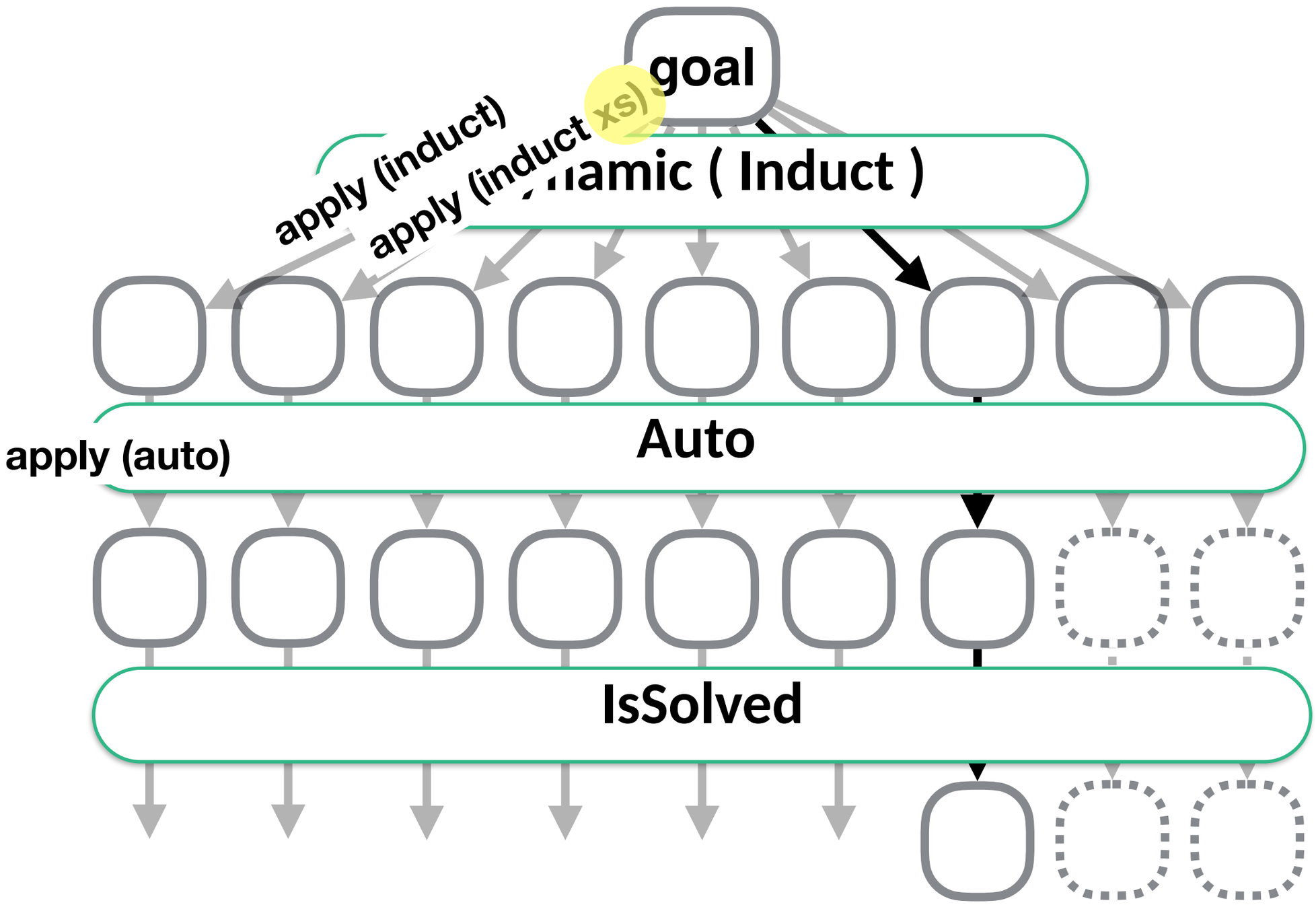
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

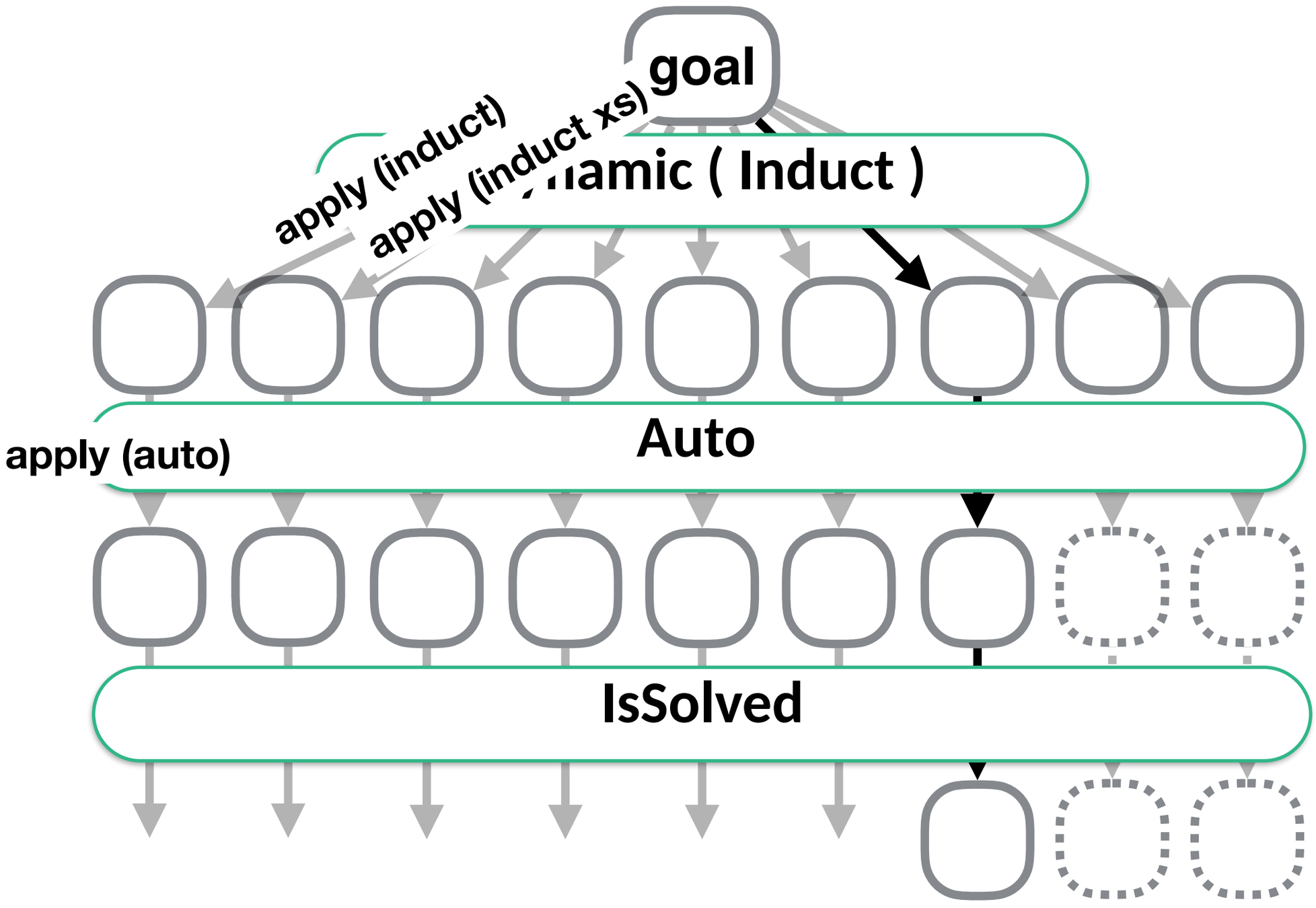
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"
find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

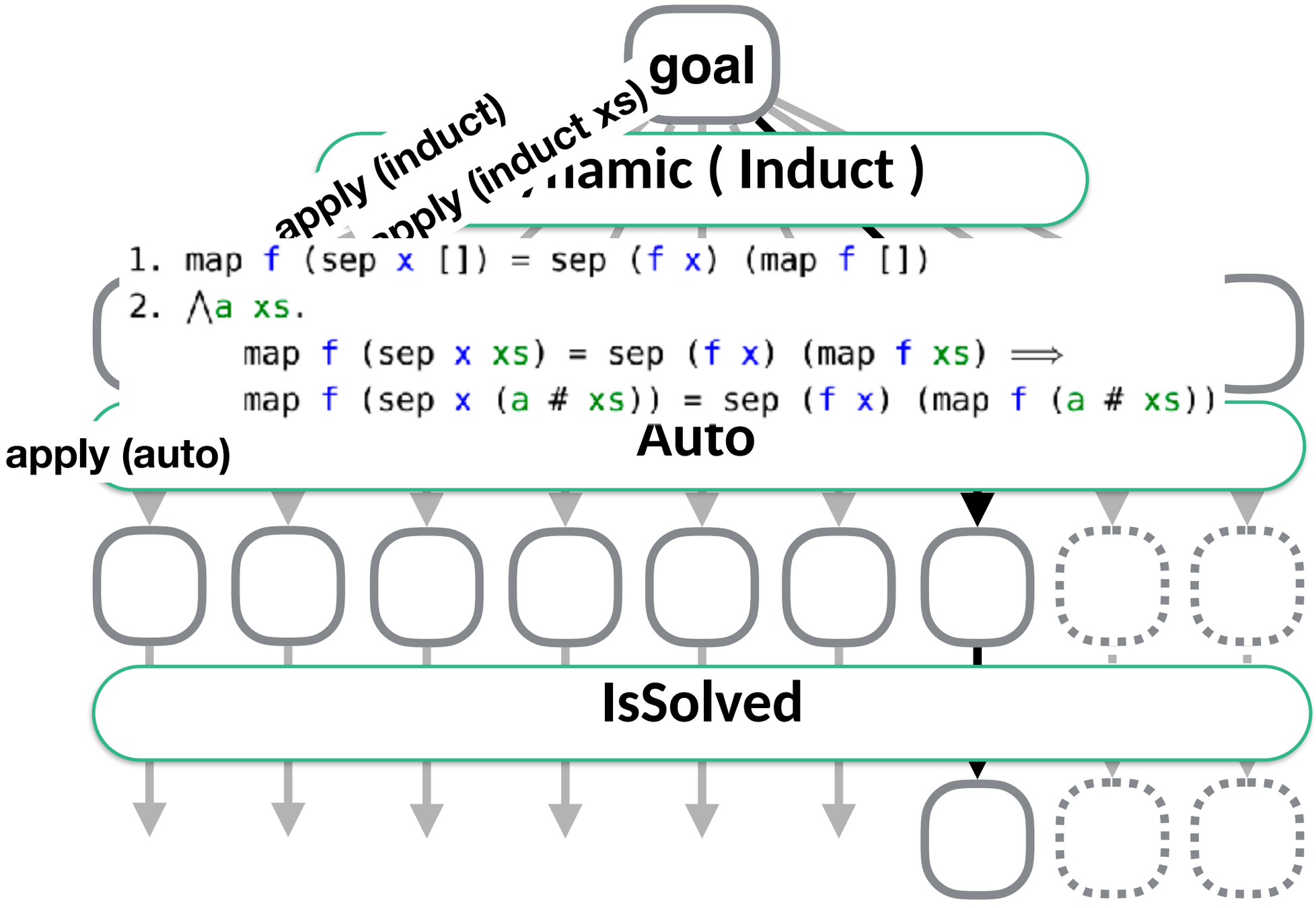
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

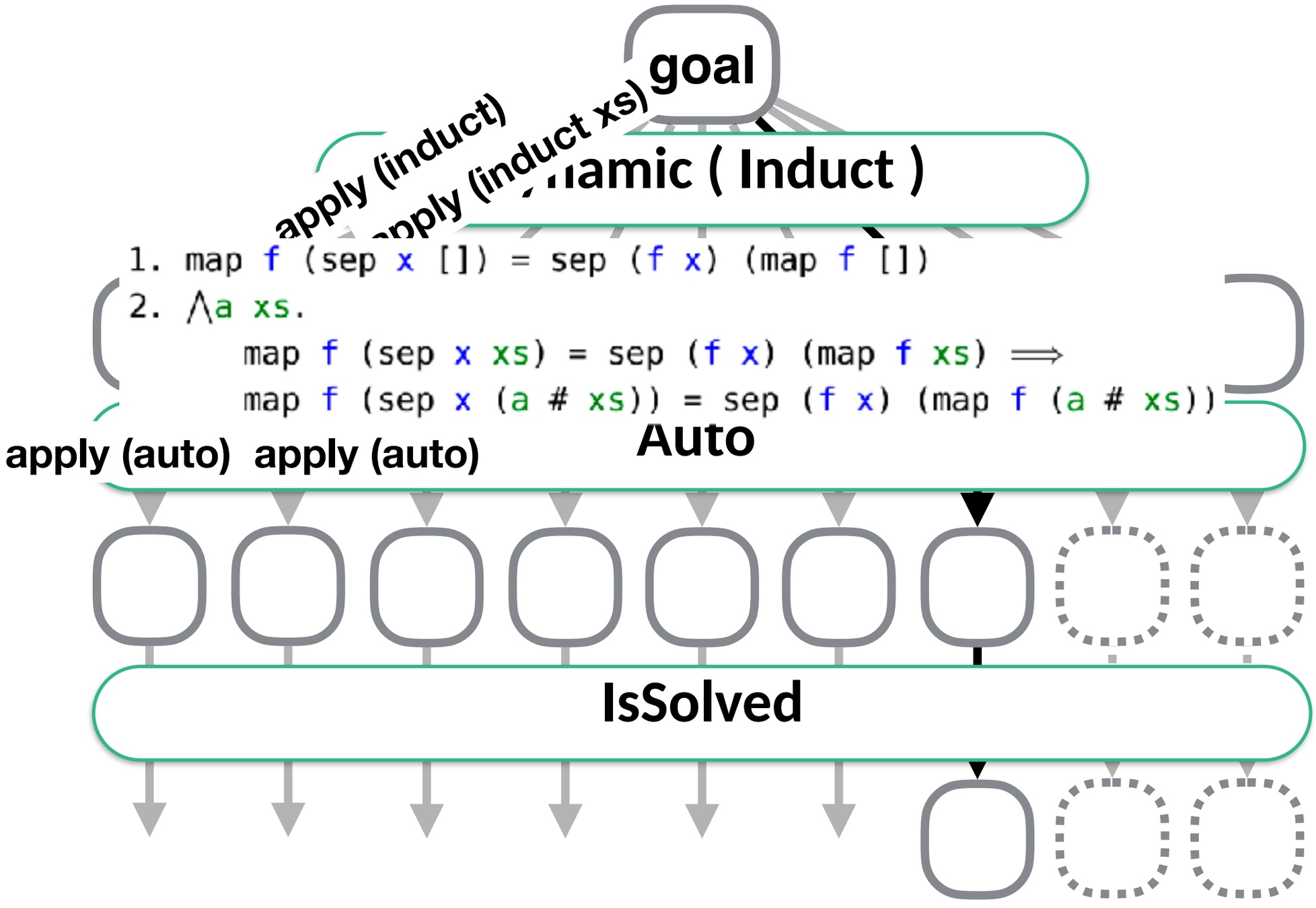
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

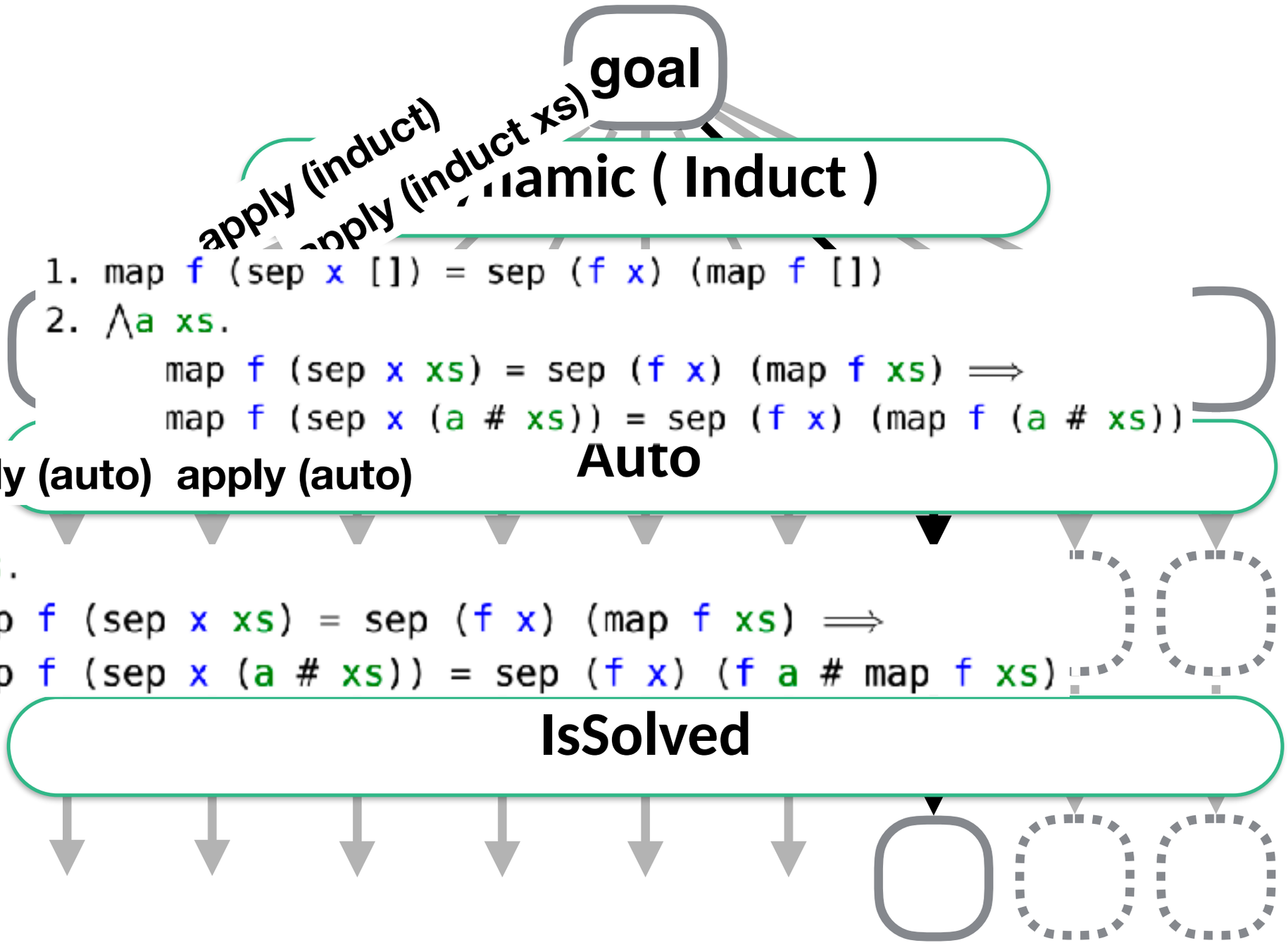
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

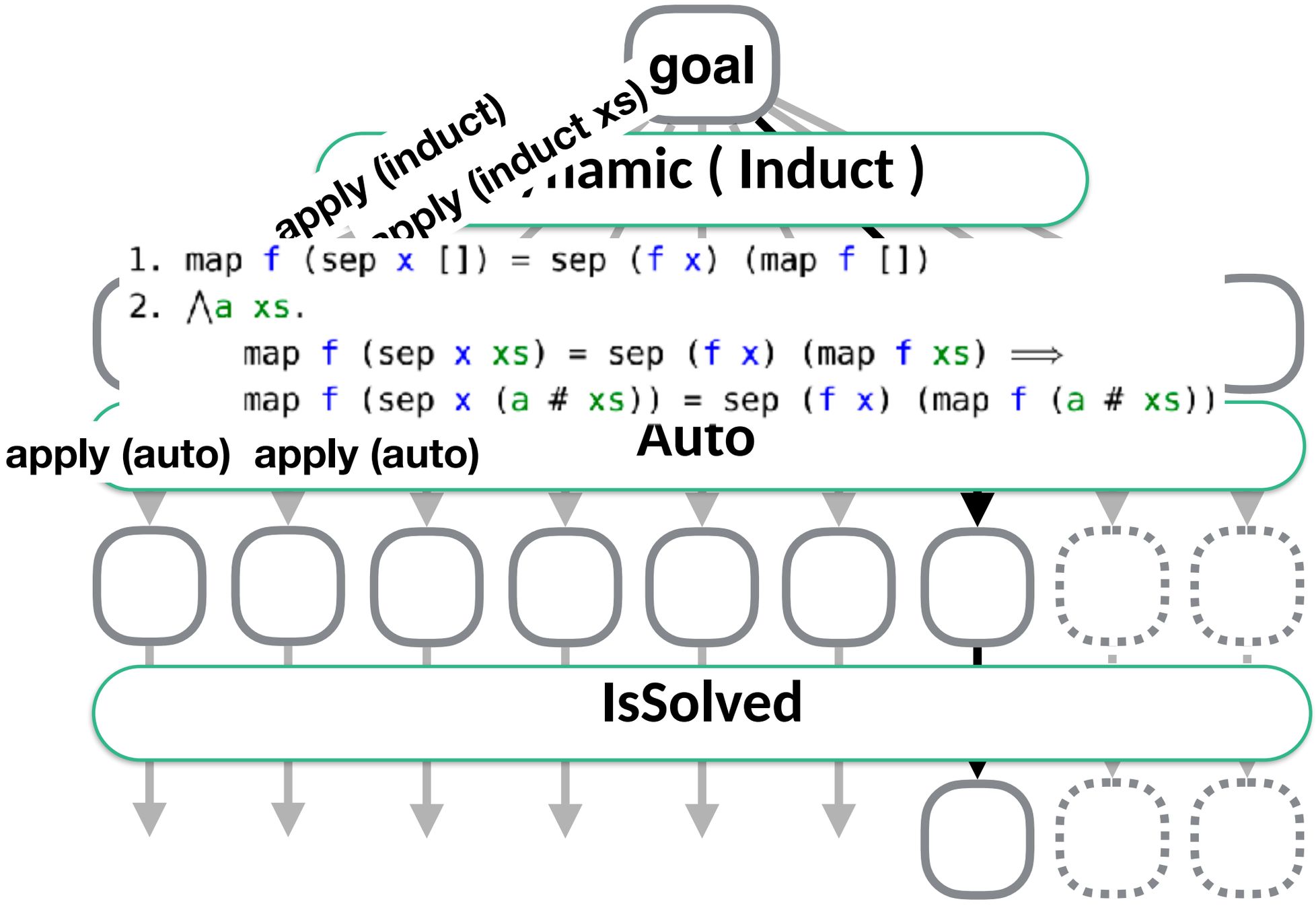
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"
find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

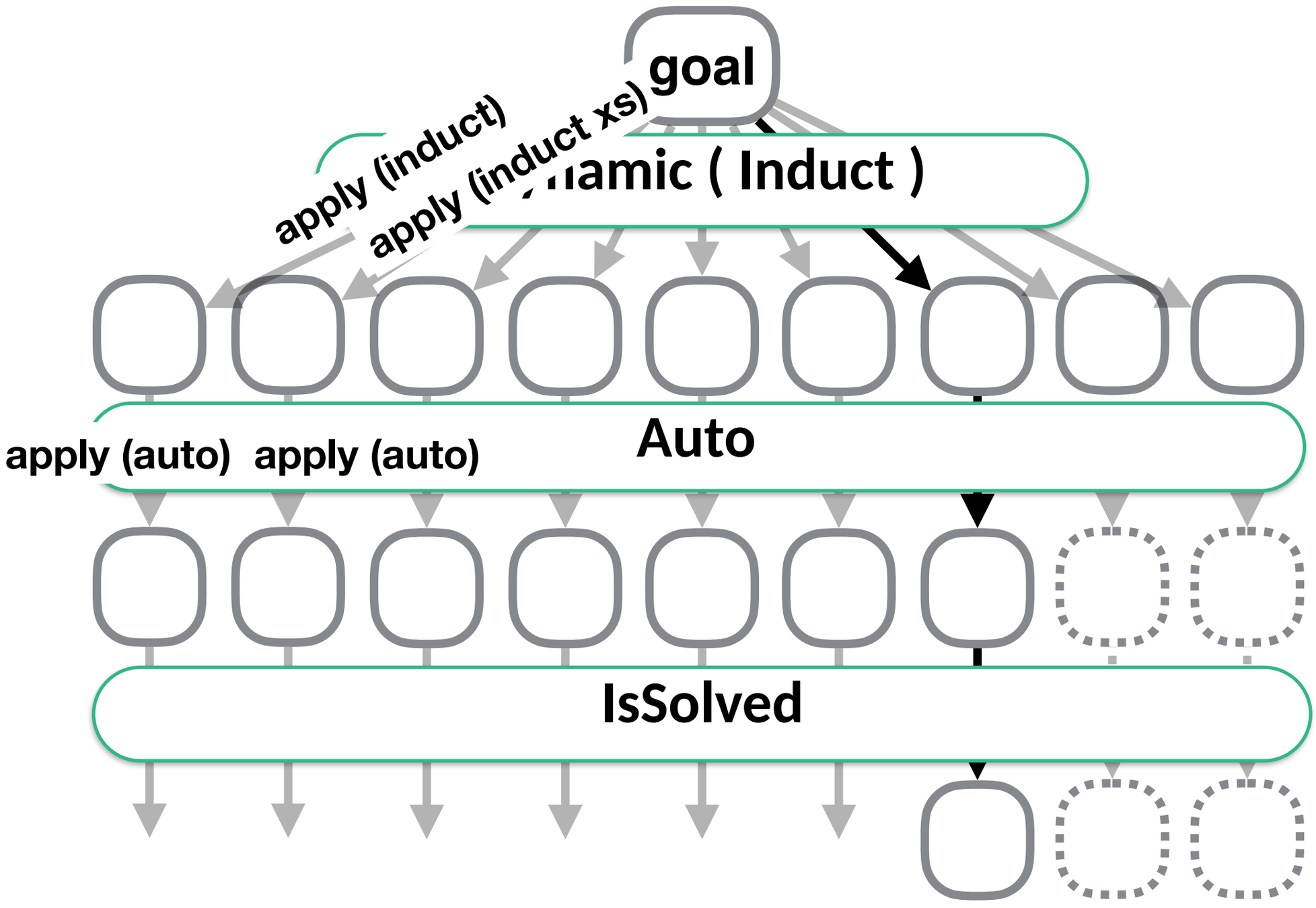
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

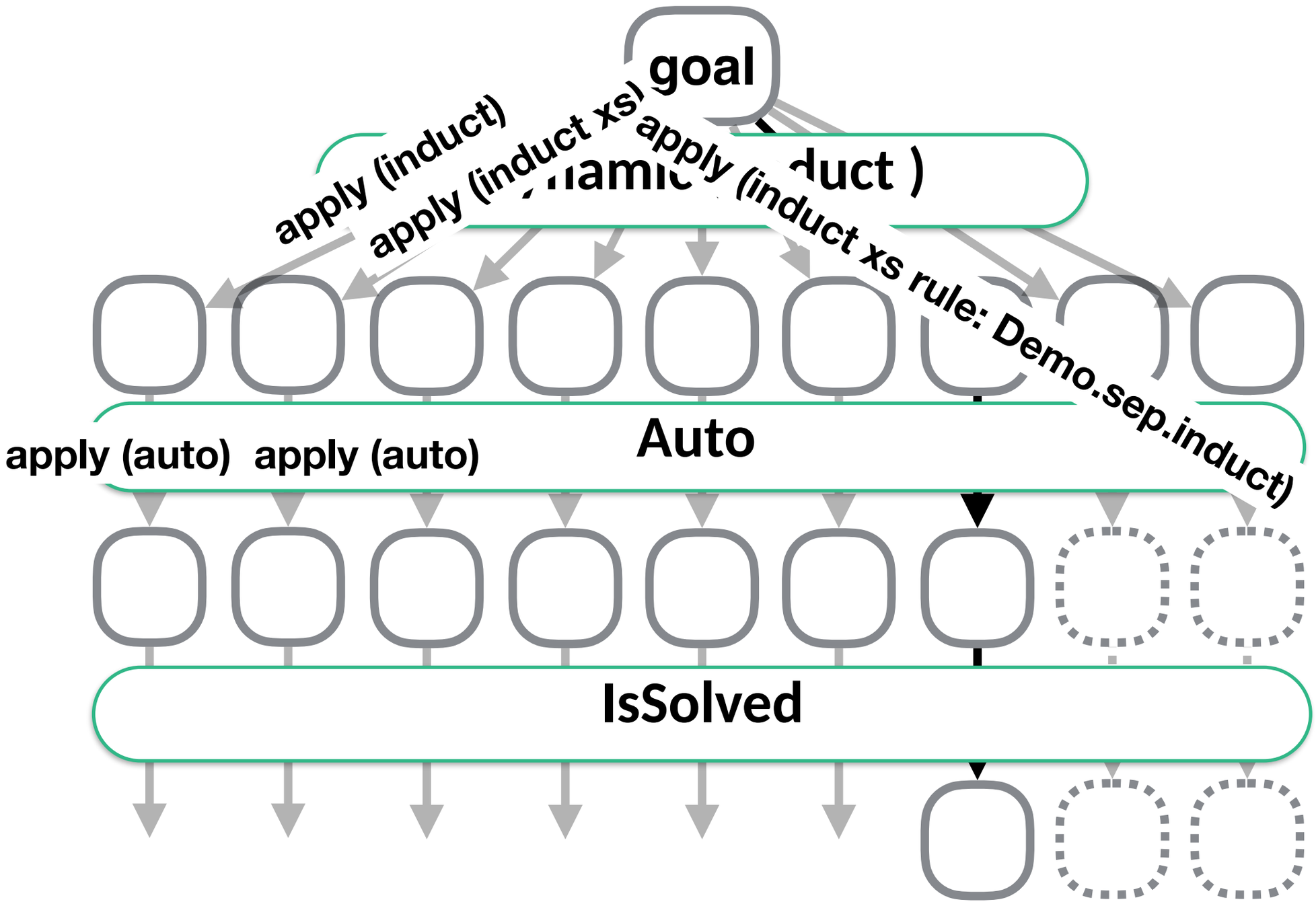
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

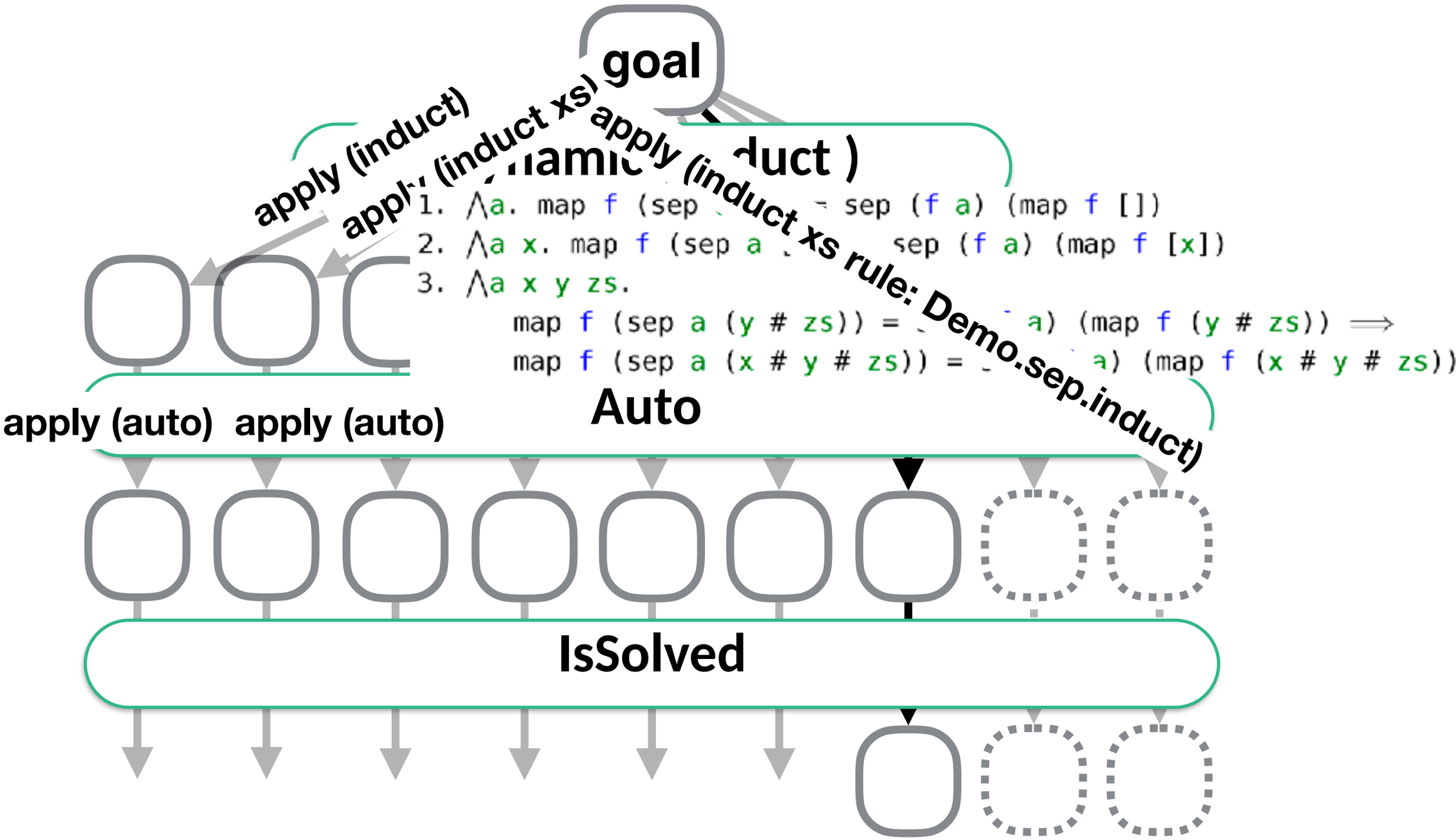
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

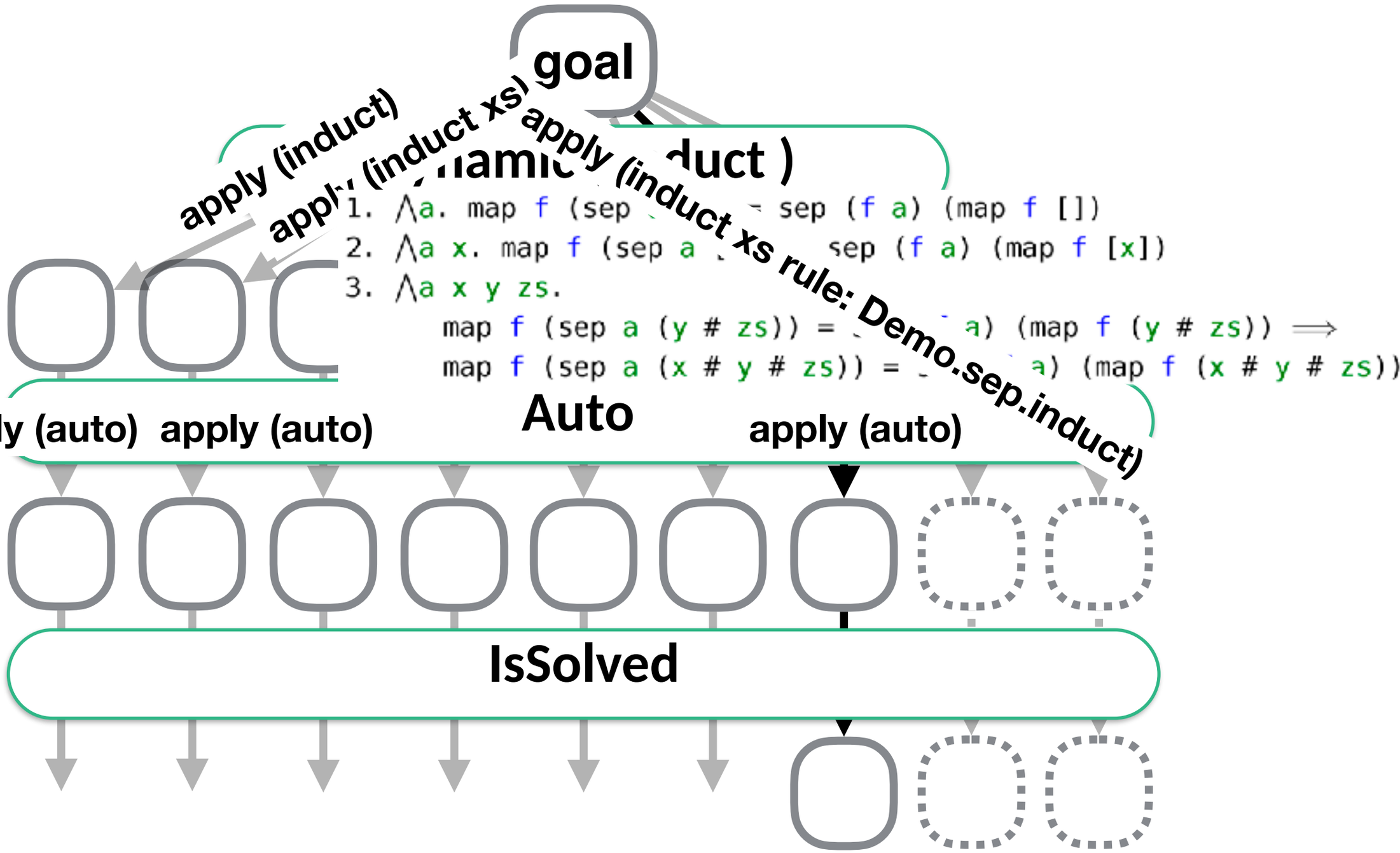
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

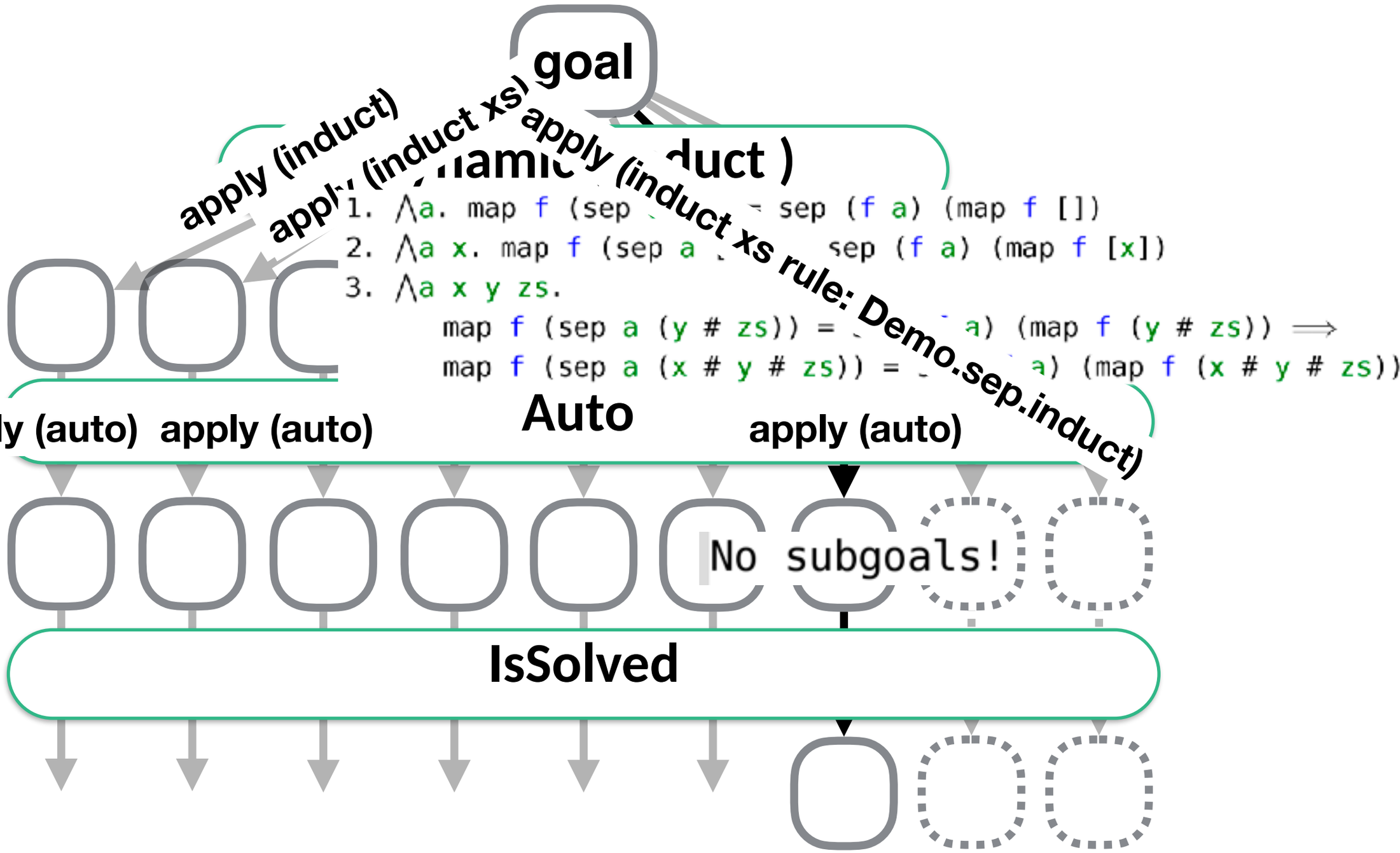
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

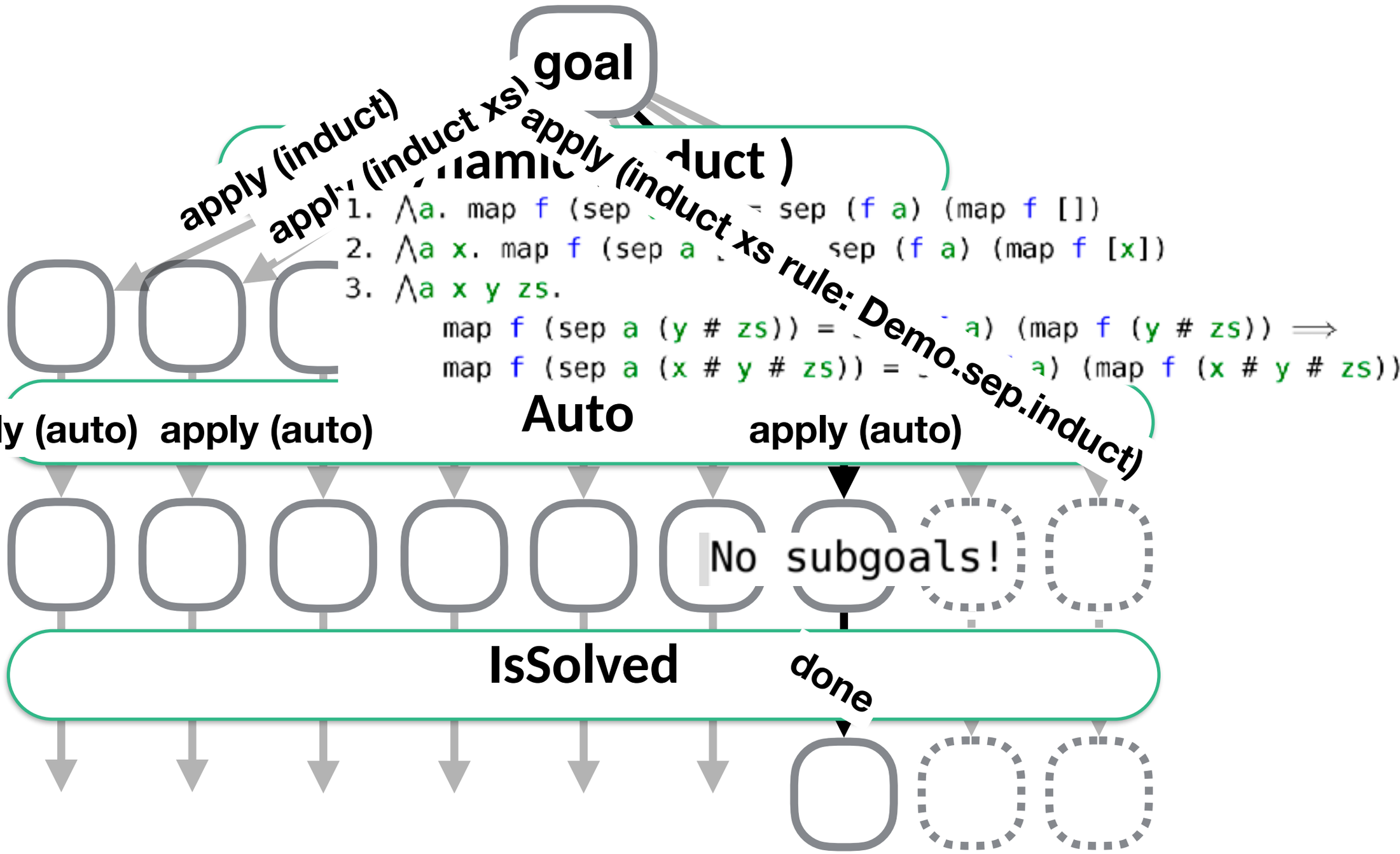
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

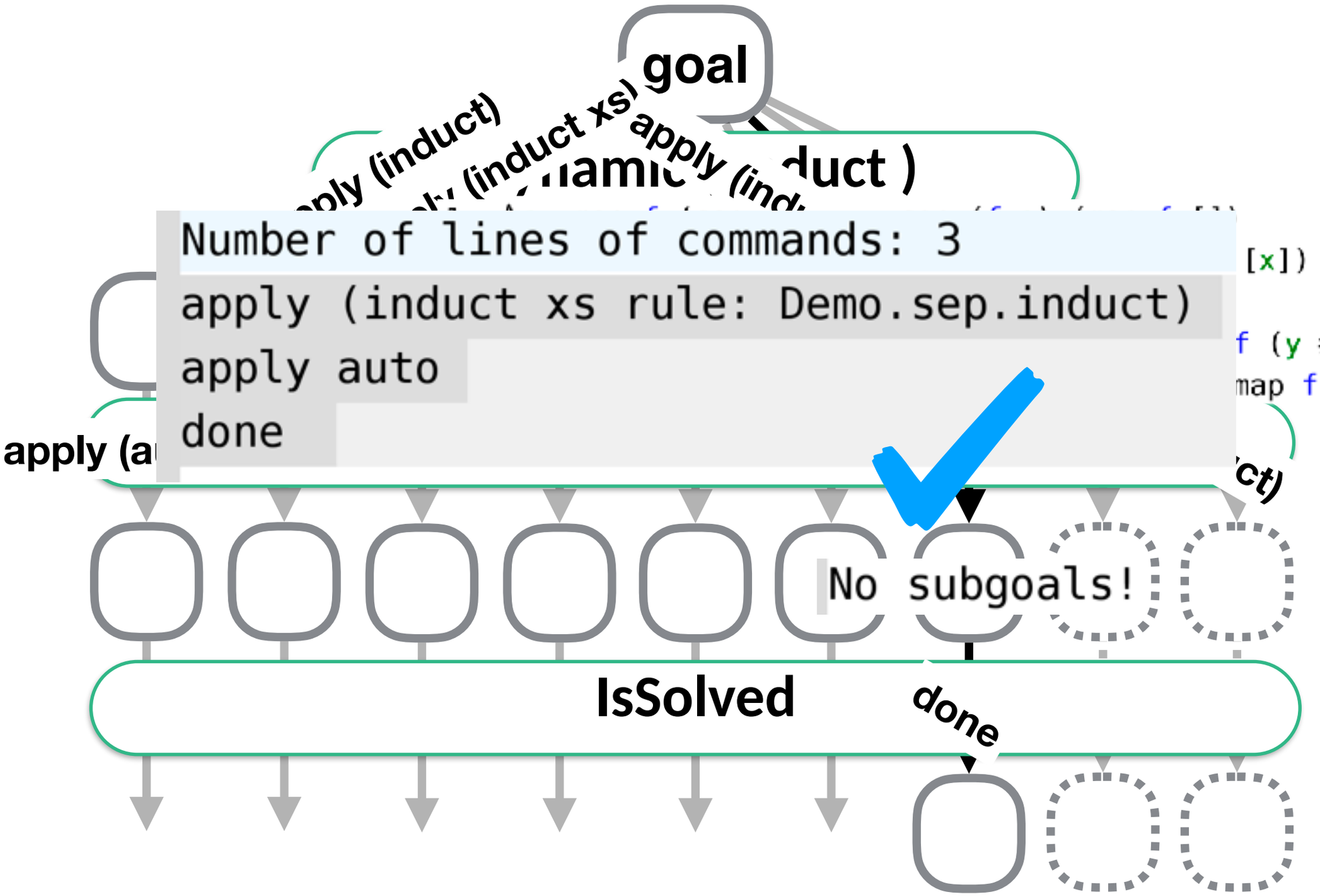
```
lemma "map f (sep x xs) = sep (f x) (map f xs)"

find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

```
lemma "map f (sep x xs) = sep (f x) (map f xs)"
find_proof DInd(*= Thens [Dynamic (Induct), Auto, IsSolved]*)
```



# Try_Hard: the default strategy

```
strategy Try_Hard =
Ors [Thens [Subgoal, Basic],
      Thens [DInductTac, Auto_Solve],
      Thens [DCaseTac, Auto_Solve],
      Thens [Subgoal, Advanced],
      Thens [DCaseTac, Solve_Many],
      Thens [DInductTac, Solve_Many] ]
```

```
strategy Basic =
  Ors [
      Auto_Solve,
      Blast_Solve,
      FF_Solve,
      Thens [IntroClasses, Auto_Solve],
      Thens [Transfer, Auto_Solve],
      Thens [Normalization, IsSolved],
      Thens [DInduct, Auto_Solve],
      Thens [Hammer, IsSolved],
      Thens [DCases, Auto_Solve],
      Thens [DCoinduction, Auto_Solve],
      Thens [Auto, RepeatN(Hammer), IsSolved],
      Thens [DAuto, IsSolved]]
```

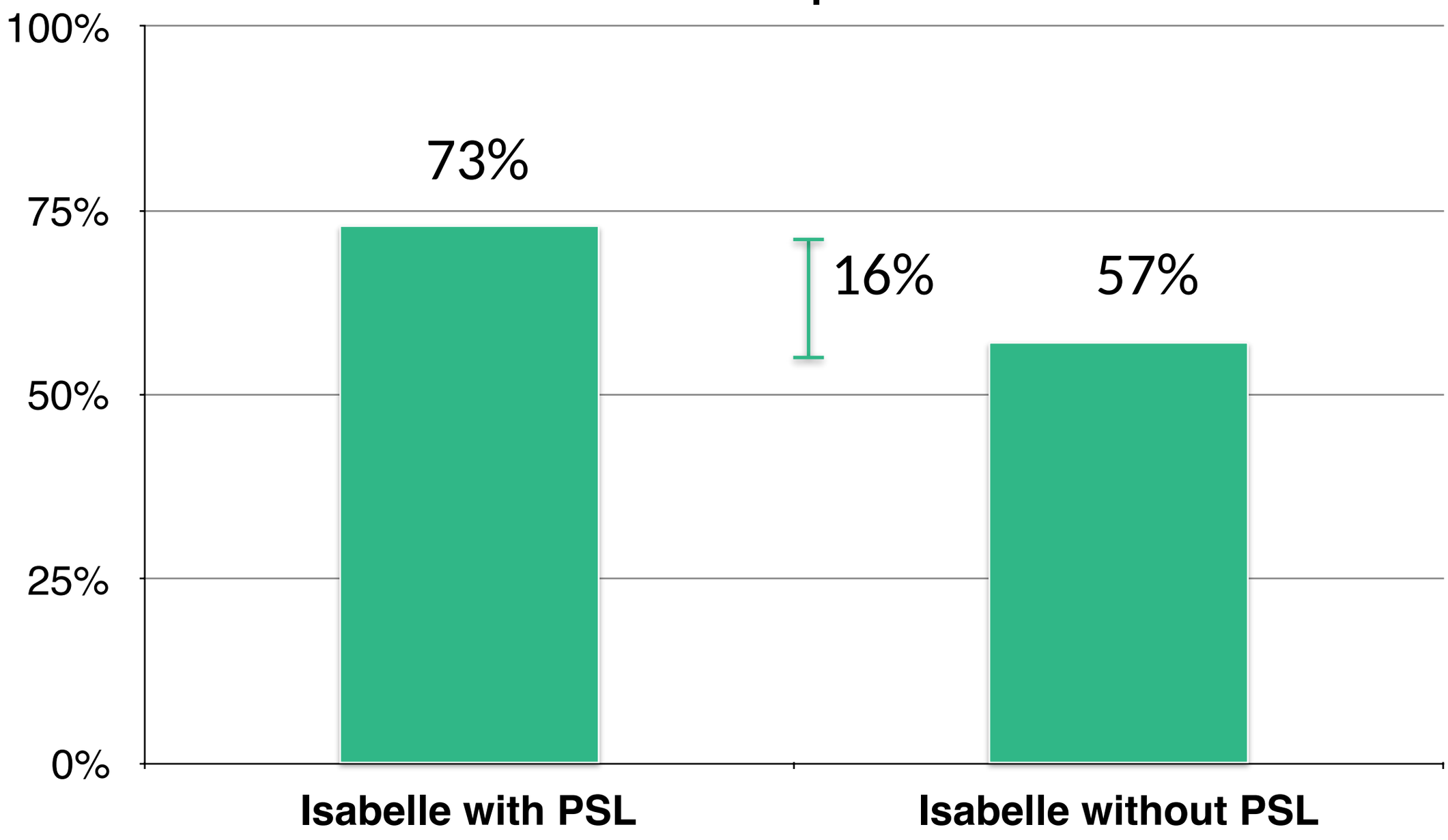
https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# Evaluation

but the search space explodes

PaMpeR: Proof Method Recommendation



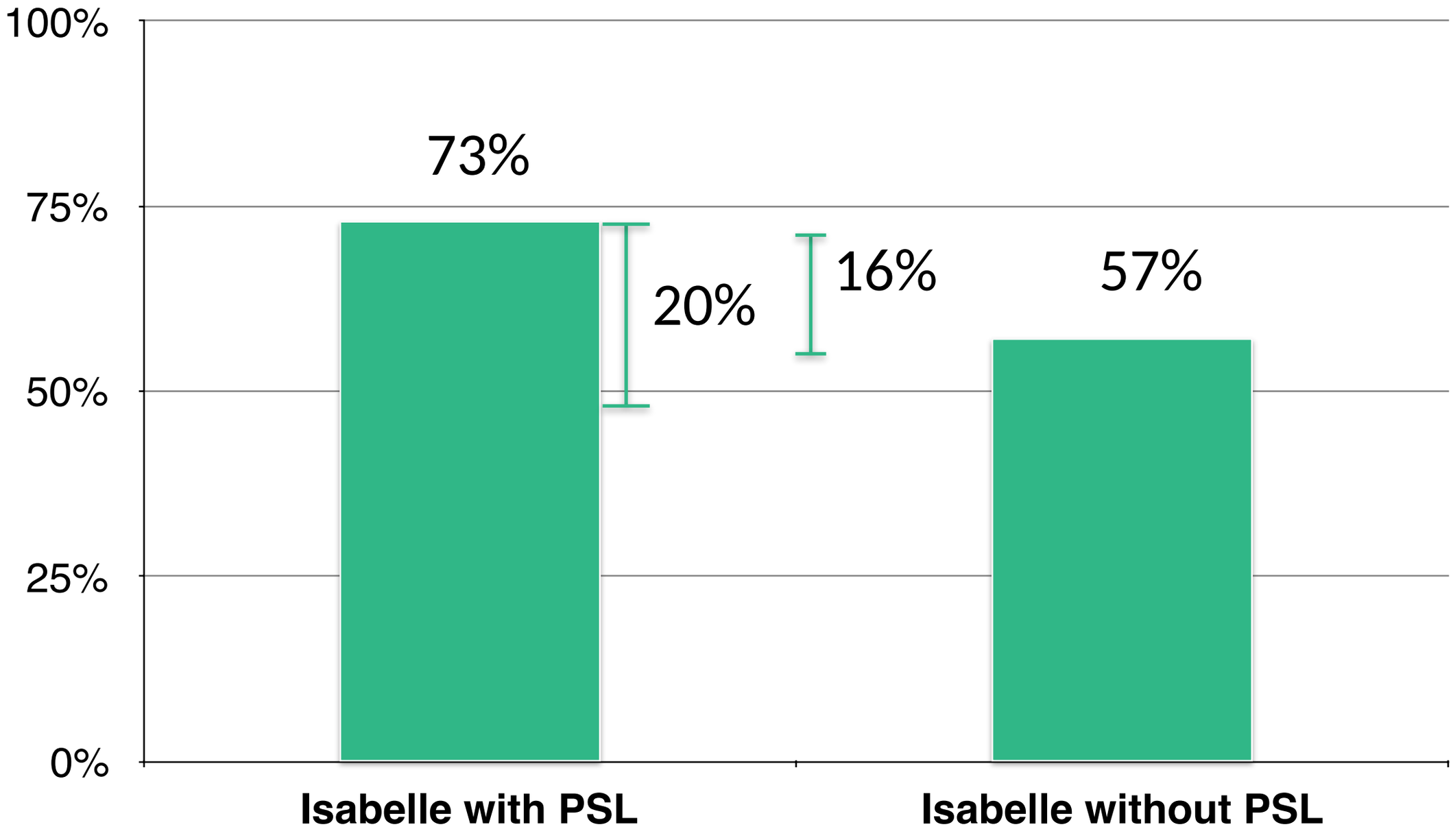
https://twitter.com/YutakangE

git clone https://github.com/data61/PSL

# Evaluation

but the search space explodes

PaMpeR: Proof Method Recommendation



preparation phase

# How does PaMpeR work?

recommendation phase

preparation phase

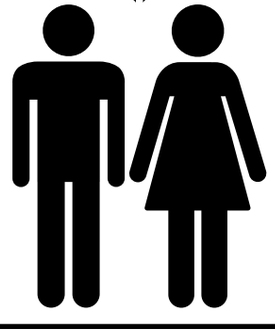
large proof corpora

AFP and standard library

# How does PaMpeR work?

recommendation phase

proof state



proof engineer

preparation phase

large proof corpora

AFP and standard library


# Statistics

## Archive of Formal Proofs (https://www.isa-afp.org)



## Statistics

Number of Articles: 468
Number of Authors: 313
Number of lemmas: ~128,900
Lines of Code: ~2,170,300

**Most used AFP articles:**

| | Name | Used by ? articles |
|---|---|---|
| 1. | Collections | 15 |
| 2. | List-Index | 14 |
| 3. | Coinductive | 12 |

preparation phase

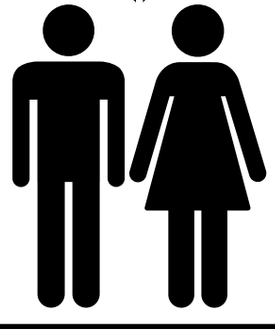
large proof corpora

AFP and standard library

# How does PaMpeR work?

recommendation phase

proof state



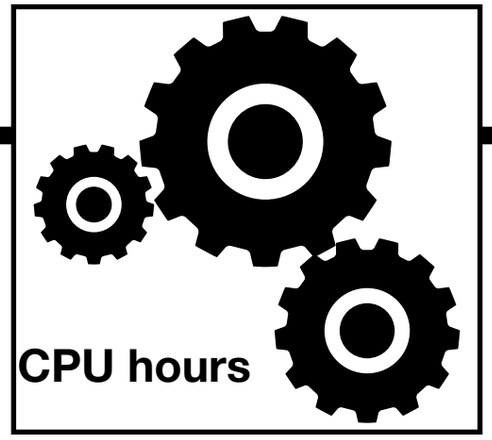
proof engineer

preparation phase

large proof corpora

AFP and standard library

full feature extractor



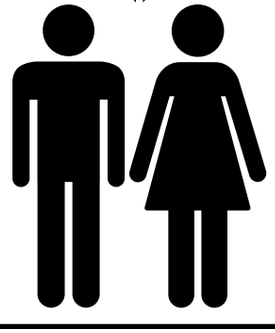
6021 CPU hours

108 assertions

# How does PaMpeR work?

recommendation phase

proof state



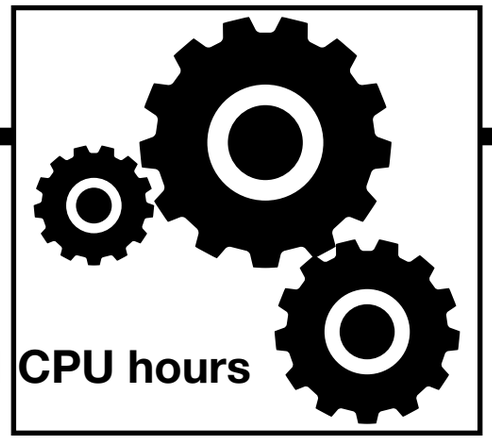
proof engineer

preparation phase

large proof corpora

AFP and standard library

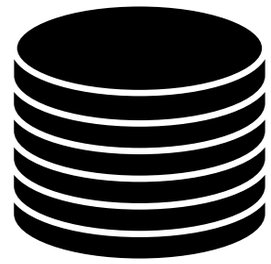
full feature extractor



6021 CPU hours

108 assertions

database ( 425334 data points )

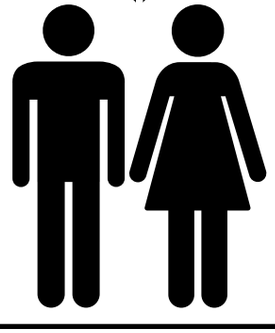


:: ( tactic_name, [ bool ] )

# How does PaMpeR work?

recommendation phase

proof state



proof engineer

preparation phase

large proof corpora

AFP and standard library

full feature extractor

6021 CPU hours

108 assertions

database ( 425334 data points )

:: ( tactic_name, [ bool ] )

preprocess

decision tree construction

# How does PaMpeR work?

recommendation phase

proof state

proof engineer

preparation phase

large proof corpora

AFP and standard library

full feature extractor

6021 CPU hours

108 assertions

database ( 425334 data points )

:: ( tactic_name, [ bool ] )

preprocess

decision tree construction

# How does PaMpeR work?

recommendation phase

proof state

proof engineer

fast feature extractor

preparation phase

large proof corpora

AFP and standard library

full feature extractor

6021 CPU hours

108 assertions

database ( 425334 data points )

:: ( tactic_name, [ bool ] )

preprocess

decision tree construction

# How does PaMpeR work?

recommendation phase

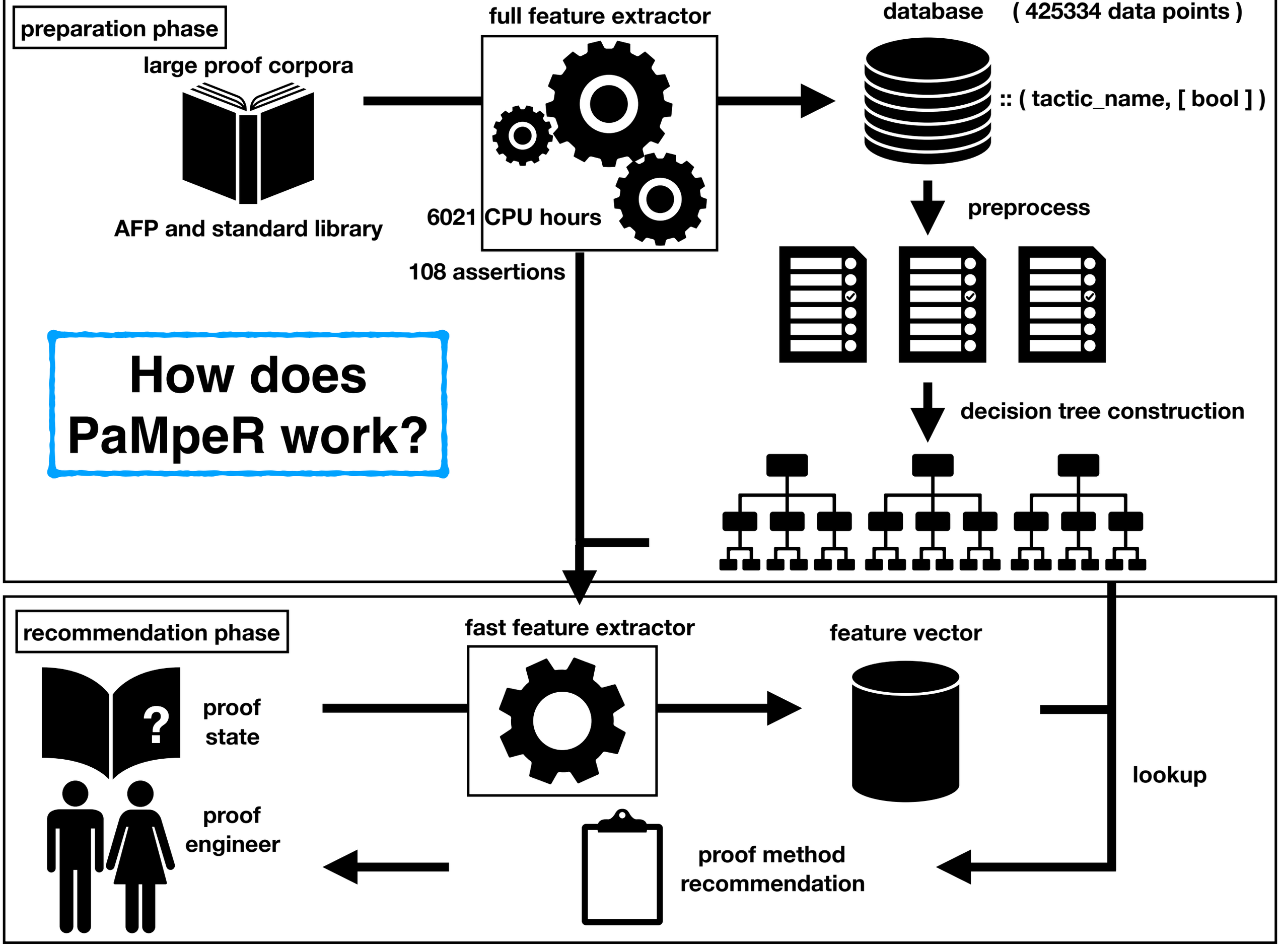
proof state

proof engineer

fast feature extractor

feature vector

# How does PaMpeR work?



https://twitter.com/YutakangE

# Summary

PSL can find how to apply induction for easy problems.
CADE2017 (https://link.springer.com/10.1007/978-3-319-63046-5_32)

PaMpeR recommends which proof methods to use.
ASE2018 (https://dx.doi.org/10.1145/3238147.3238210)