Recursive Program Synthesis

Aws Albarghouthi (UToronto), Sumit Gulwani (MSR),
and Zachary Kincaid (UToronto)

CAV 2013
Saint Petersburg, Russia

Program Synthesis

Program Synthesis

Synthesis: Dreams $\Longrightarrow$ Programs

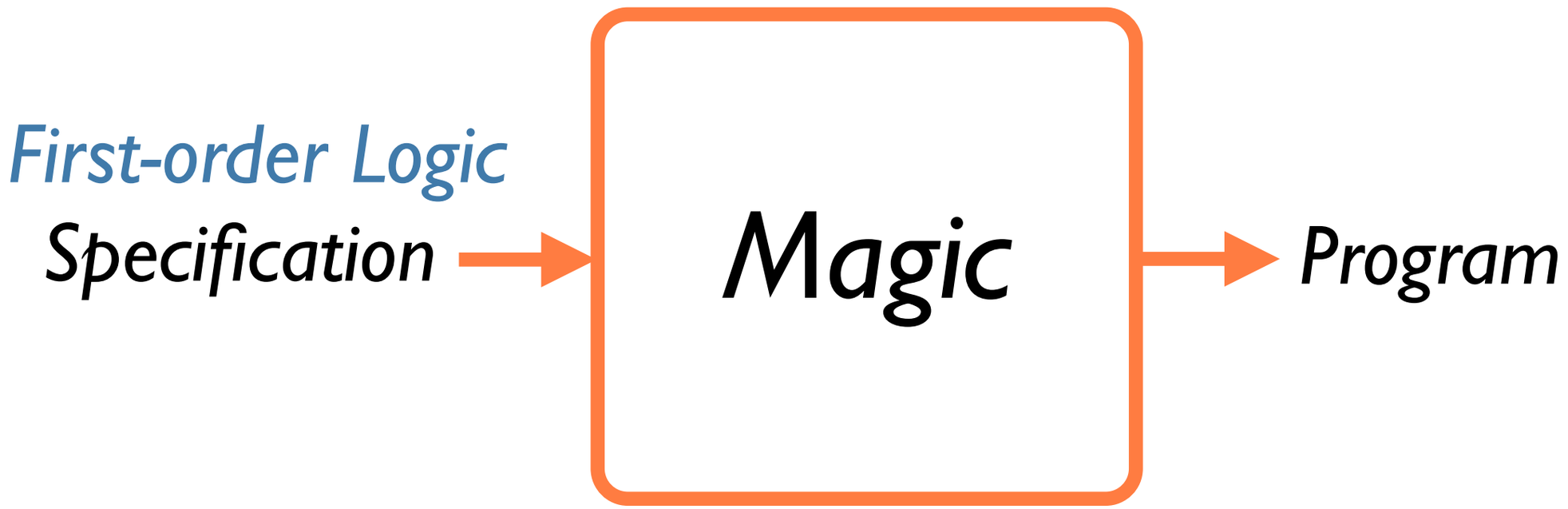
ZOHAR MANNA AND RICHARD WALDINGER

What are your Dreams?

Synthesis: Dreams $\Longrightarrow$ Programs

ZOHAR MANNA AND RICHARD WALDINGER

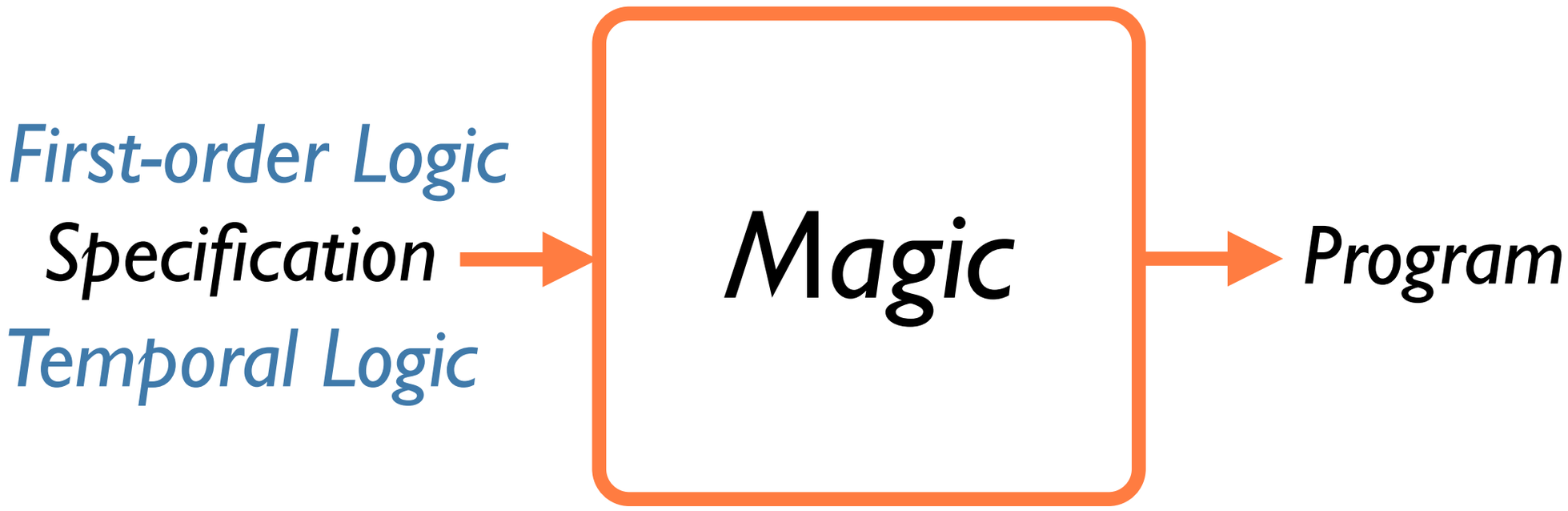
What are your Dreams?



Synthesis: Dreams $\Longrightarrow$ Programs

ZOHAR MANNA AND RICHARD WALDINGER

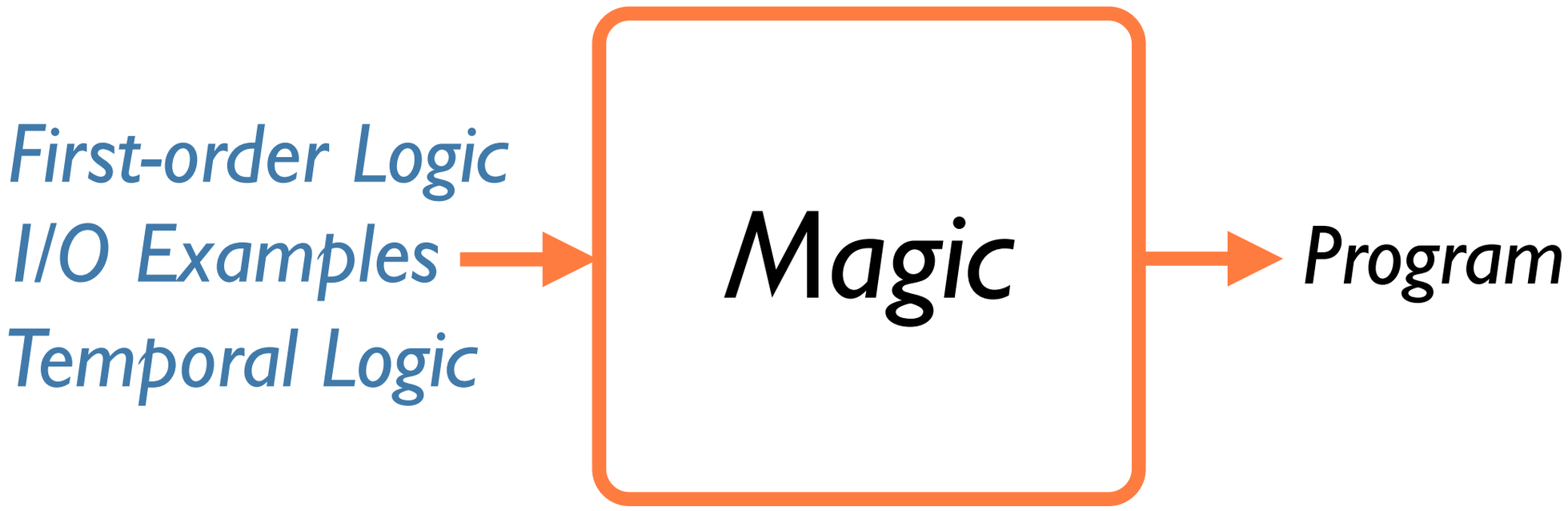
What are your Dreams?



Synthesis: Dreams $\Longrightarrow$ Programs

ZOHAR MANNA AND RICHARD WALDINGER

What are your Dreams?



294 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-5, NO. 4, JULY 1979

Synthesis: Dreams $\Longrightarrow$ Programs

ZOHAR MANNA AND RICHARD WALDINGER

Why Synthesis from I/O?

I/O examples are *easy to specify*

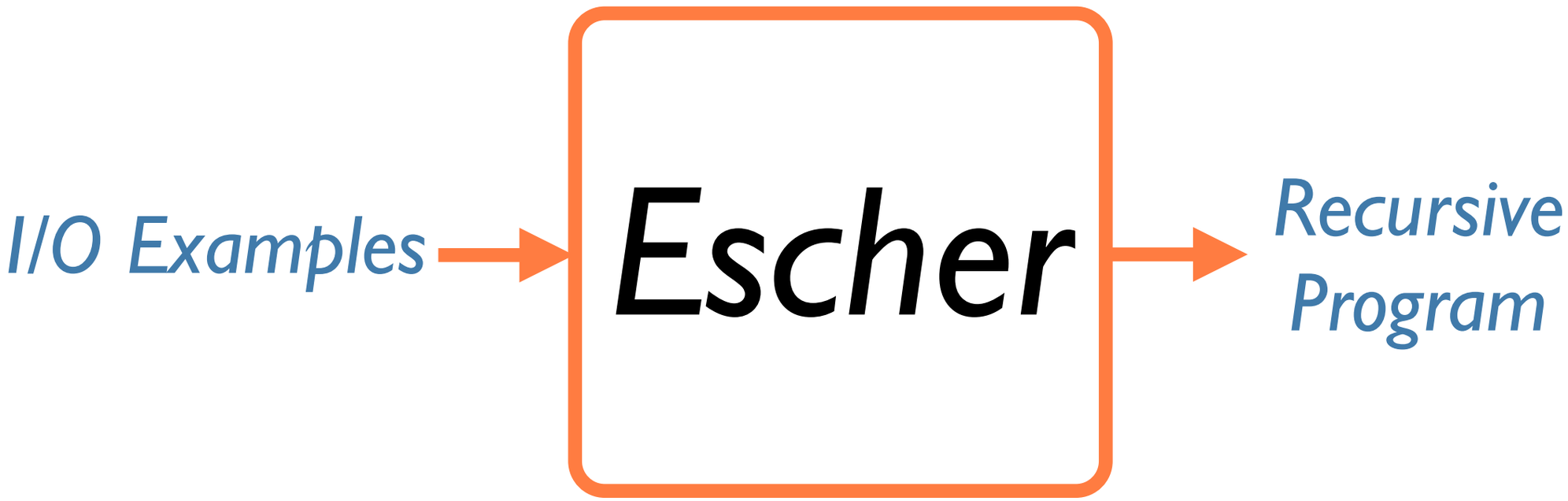
Why Synthesis from I/O?

I/O examples are *easy to specify*

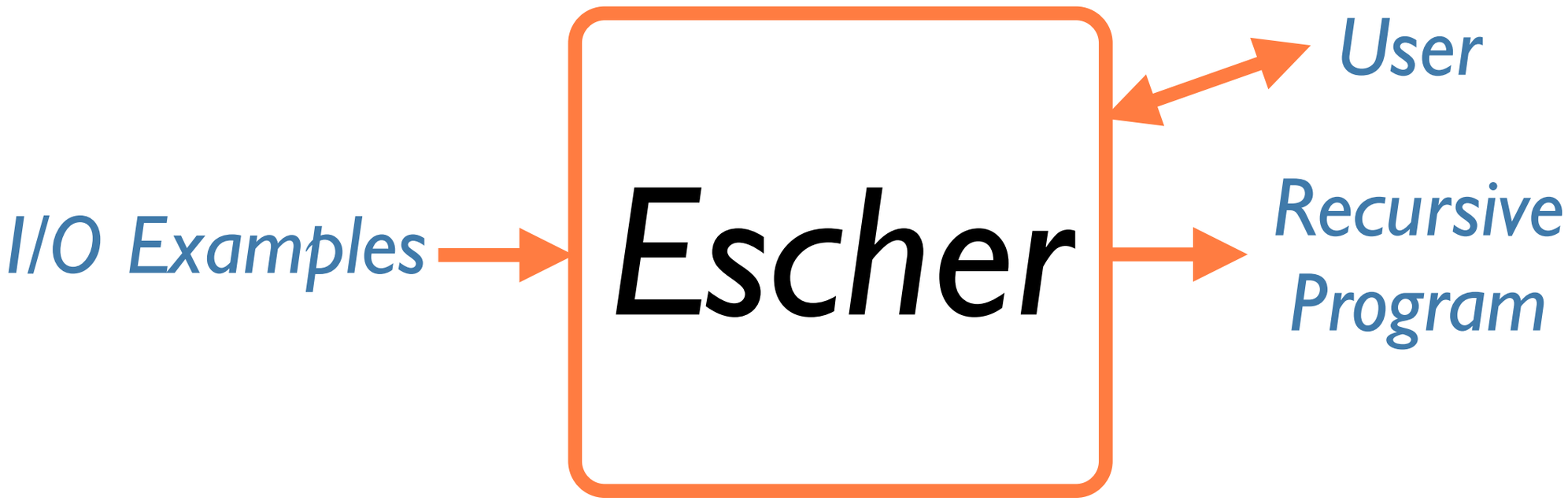
Non-expert users can specify I/O behaviour

- *See, e.g., FlashFill for Excel, Smartphone scripts, etc.*
- *Desired programs are usually simple*

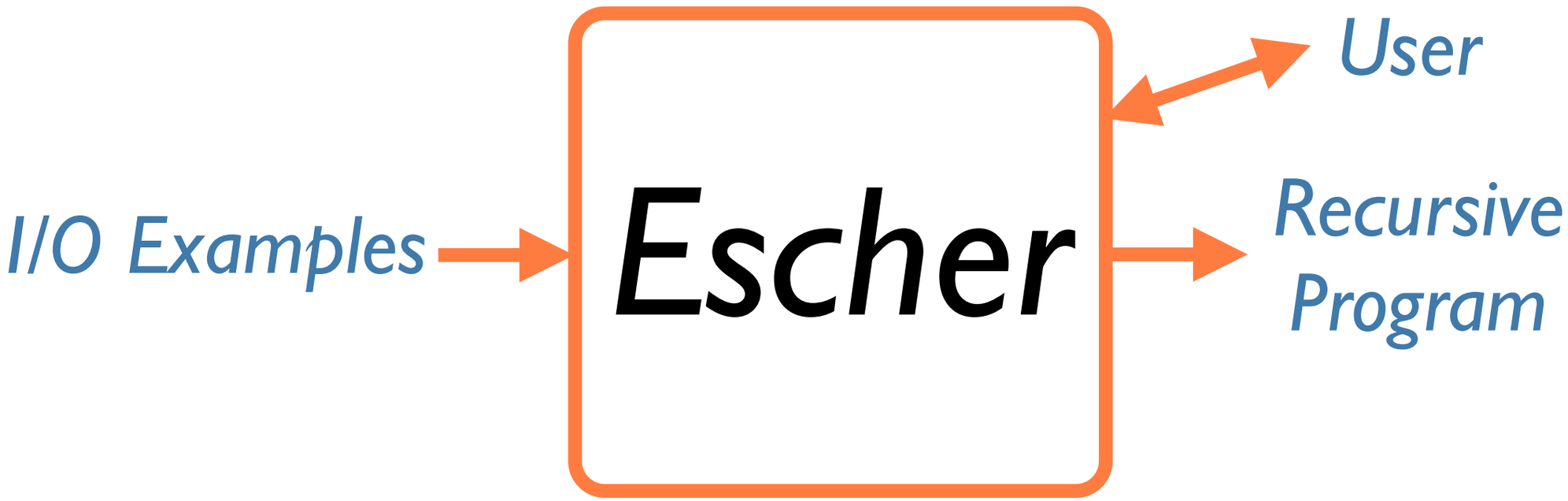
Contribution



Contribution



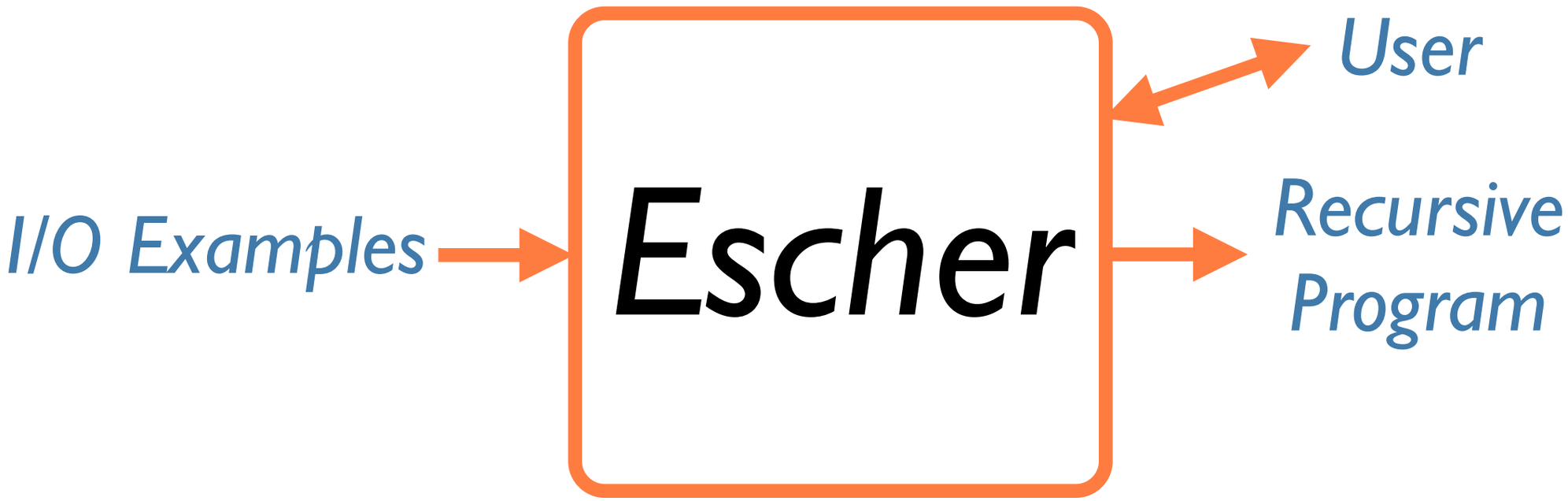
Contribution



Parameterized by a set of building blocks (operations)

integers, lists, trees, etc.

Contribution

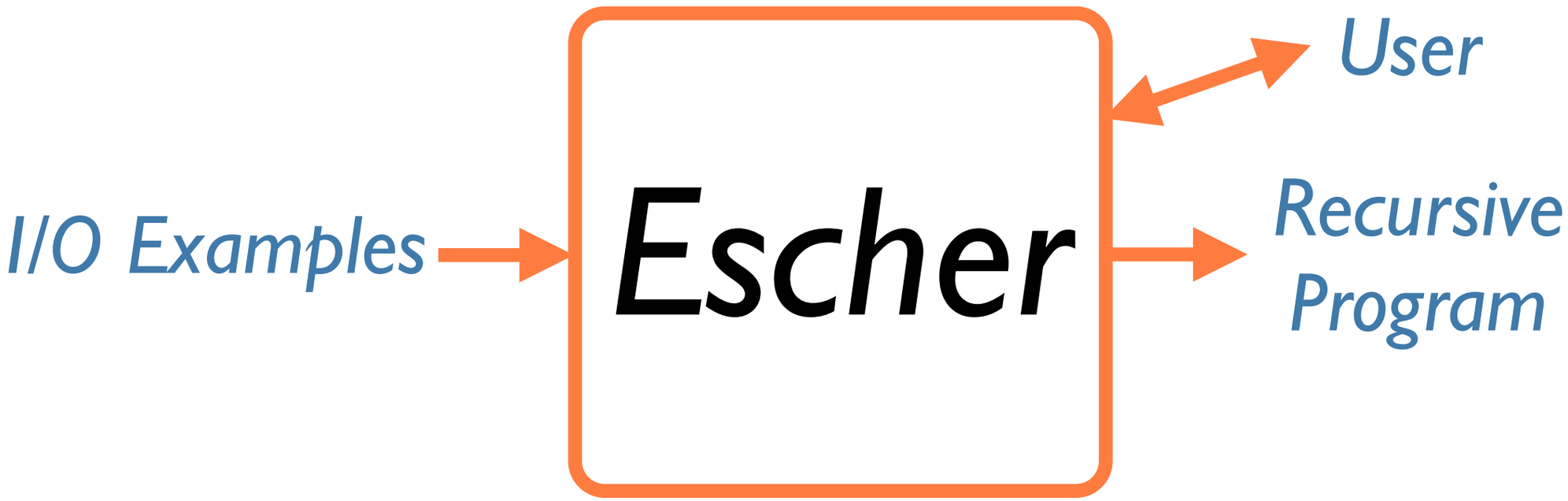


Parameterized by a set of building blocks (operations)

integers, lists, trees, etc.

Novel search-based synthesis technique

Contribution



Parameterized by a set of building blocks (operations)

integers, lists, trees, etc.

Novel search-based synthesis technique

No templates required

High Level View

Forward Search

High Level View

Forward Search

$$I_1 \cdots I_n$$

High Level View

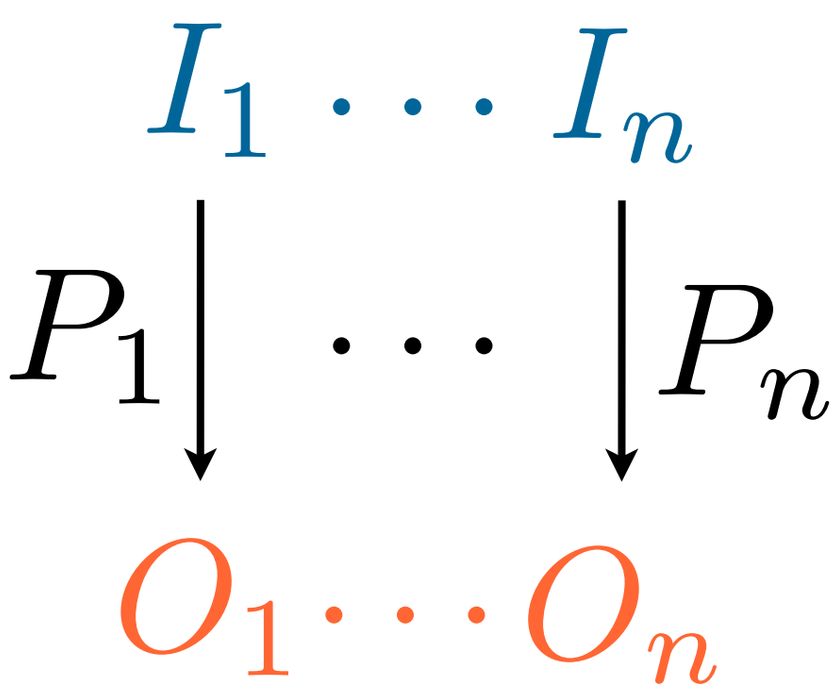
Forward Search

$$I_1 \cdots I_n$$

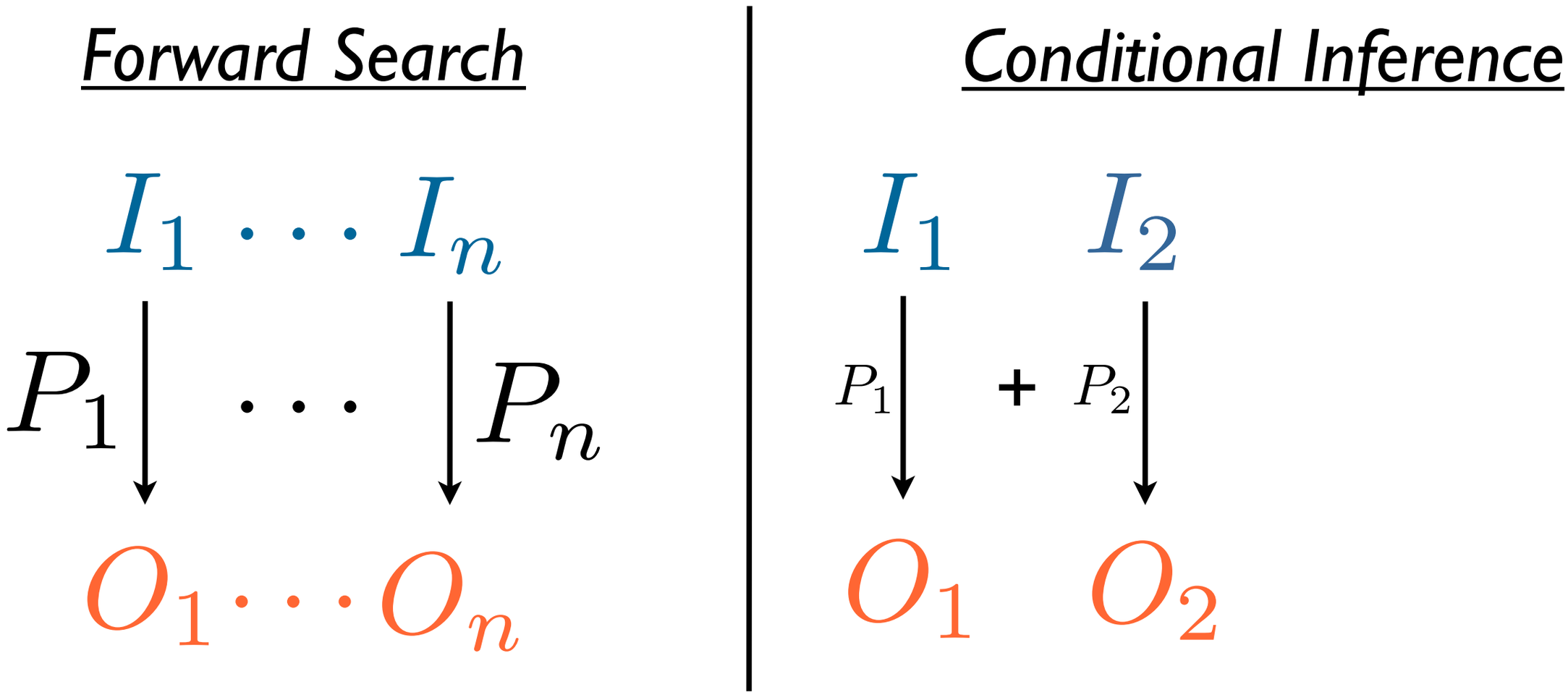
$$O_1 \cdots O_n$$

High Level View

Forward Search



High Level View



High Level View

Forward Search

$$I_1 \cdots I_n$$

$$P_1 \Big| \quad \cdots \quad \Big| P_n$$

$$O_1 \cdots O_n$$

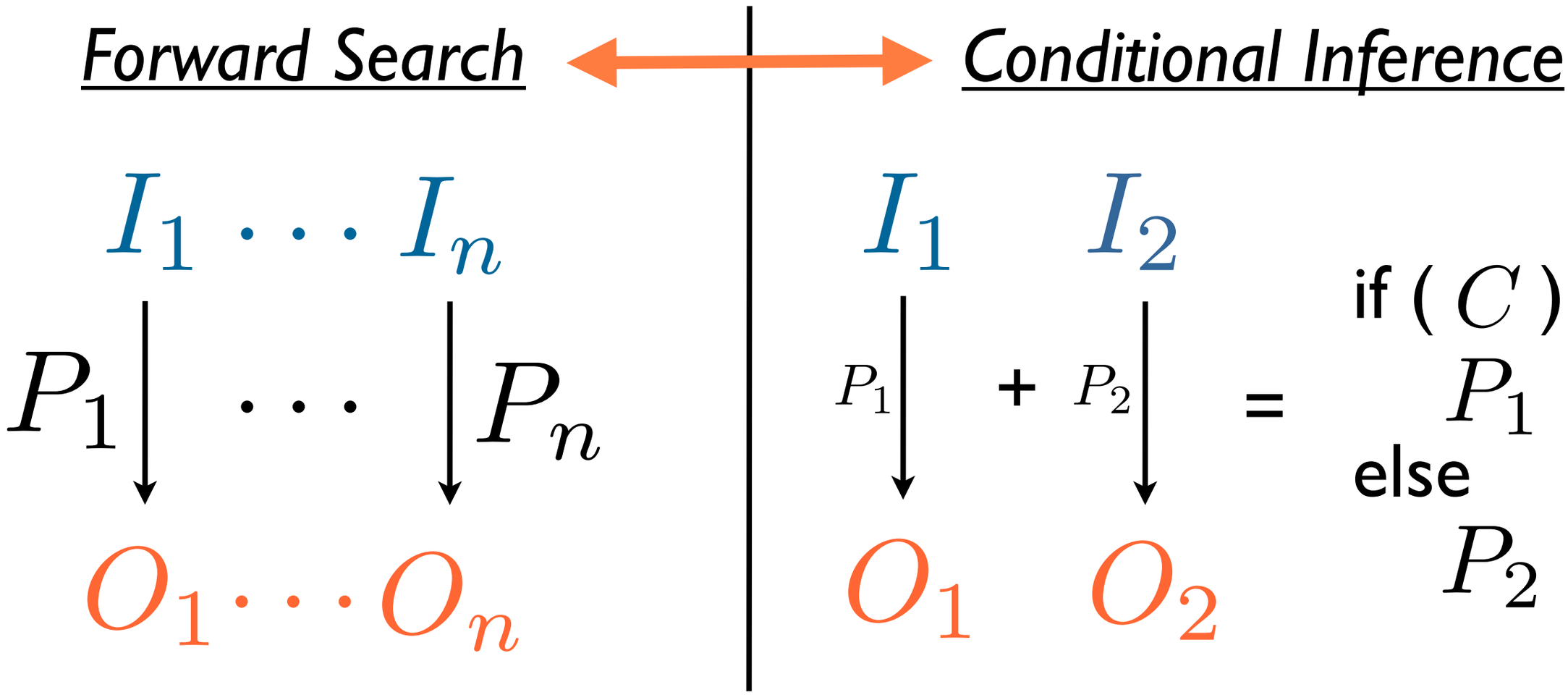
Conditional Inference

$$I_1 \quad I_2$$

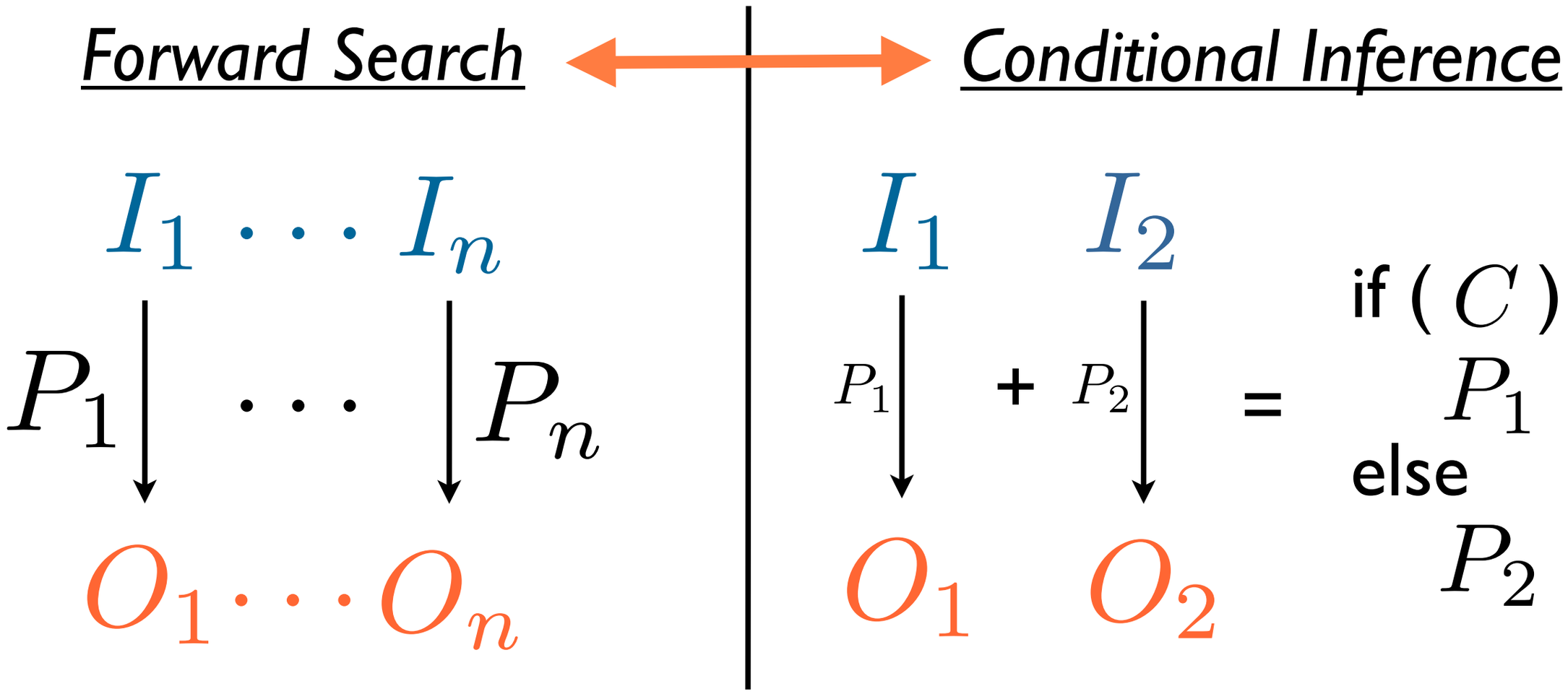
$$P_1 \Big| \quad + \quad P_2 \Big| \quad = \quad \begin{array}{l} \text{if } (C) \\ \quad P_1 \\ \text{else} \\ \quad P_2 \end{array}$$

$$O_1 \quad O_2$$

High Level View



High Level View



Recursive call synthesis

Reuse I/O as recursive call specification or query user

Example

Synthesize list length from examples:

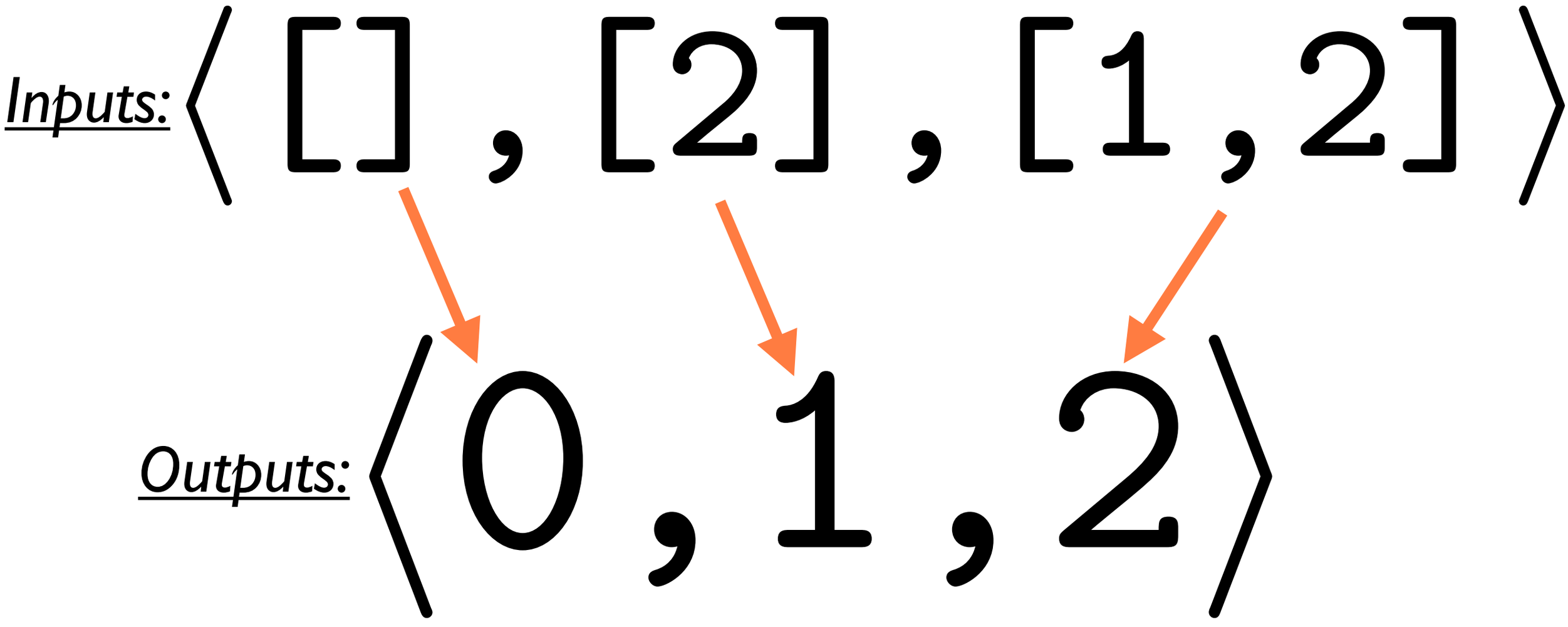
Example

Synthesize list length from examples:

Inputs: ⟨ [] , [2] , [1, 2] ⟩

Example

Synthesize list length from examples:



Ex: Forward Search

Components: inc, isEmpty, tail, zero

Ex: Forward Search

Components: inc, isEmpty, tail, zero

Programs of size 1:

Ex: Forward Search

Components: inc, isEmpty, tail, zero

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

Ex: Forward Search

Components: inc, isEmpty, tail, zero

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Ex: Forward Search

Components: inc, isEmpty, tail, zero

Programs of size 1:

$$\mathrm{P}_1 = \mathtt{i} \rightarrow \langle [], [2], [1,2] \rangle$$

$$\mathrm{P}_2 = \mathtt{zero} \rightarrow \langle 0, 0, 0 \rangle$$

matches output for first input

Recall outputs $\langle 0, 1, 2 \rangle$

Ex: Conditional Inference

$$\mathrm{P}_2 = \texttt{zero} \rightarrow \langle 0,0,0 \rangle$$

Outputs: $\langle 0,1,2 \rangle$

Ex: Conditional Inference

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle 0, 0, 0 \rangle$$

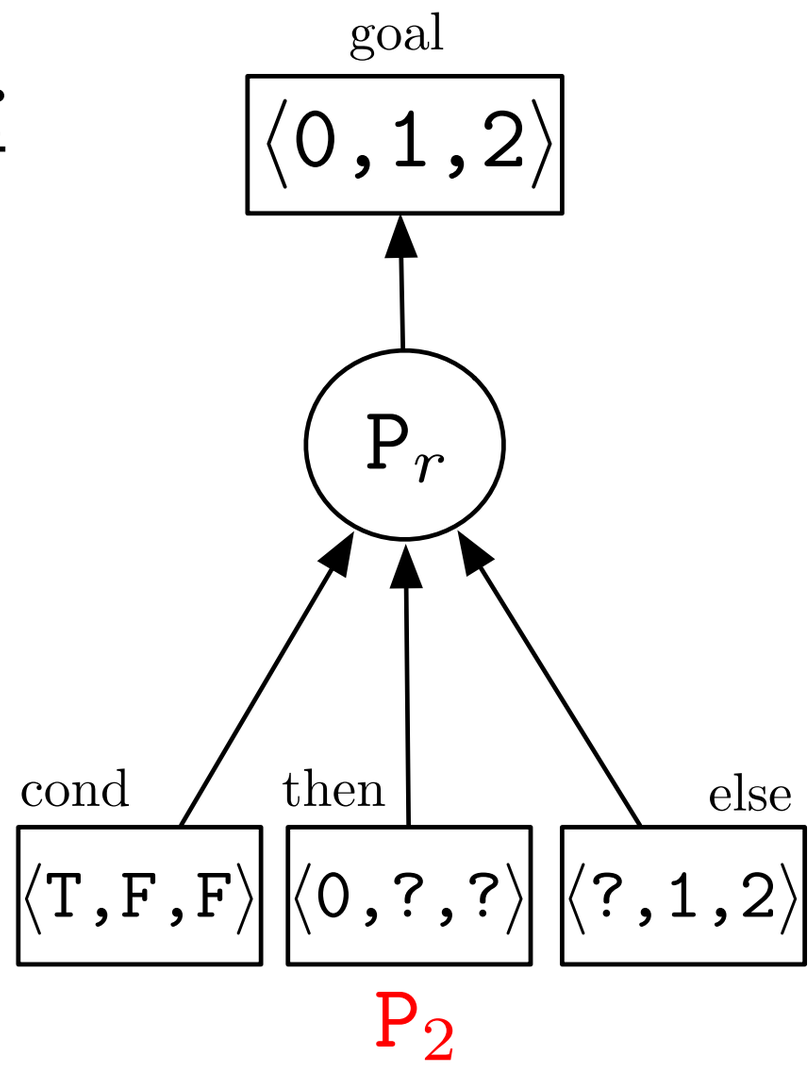
Outputs: $\langle 0, 1, 2 \rangle$

Ex: Conditional Inference

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Outputs: $\langle \mathtt{0}, \mathtt{1}, \mathtt{2} \rangle$

Goal Graph (GG):

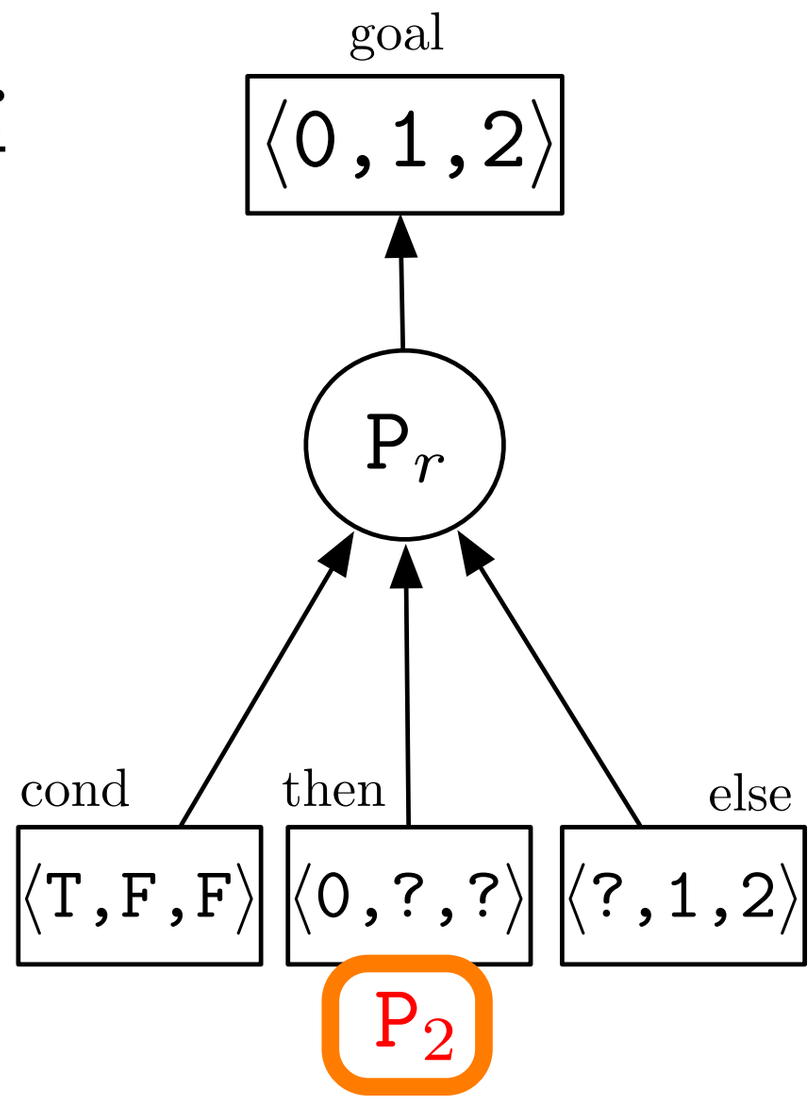


Ex: Conditional Inference

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Outputs: $\langle \mathtt{0}, \mathtt{1}, \mathtt{2} \rangle$

Goal Graph (GG):

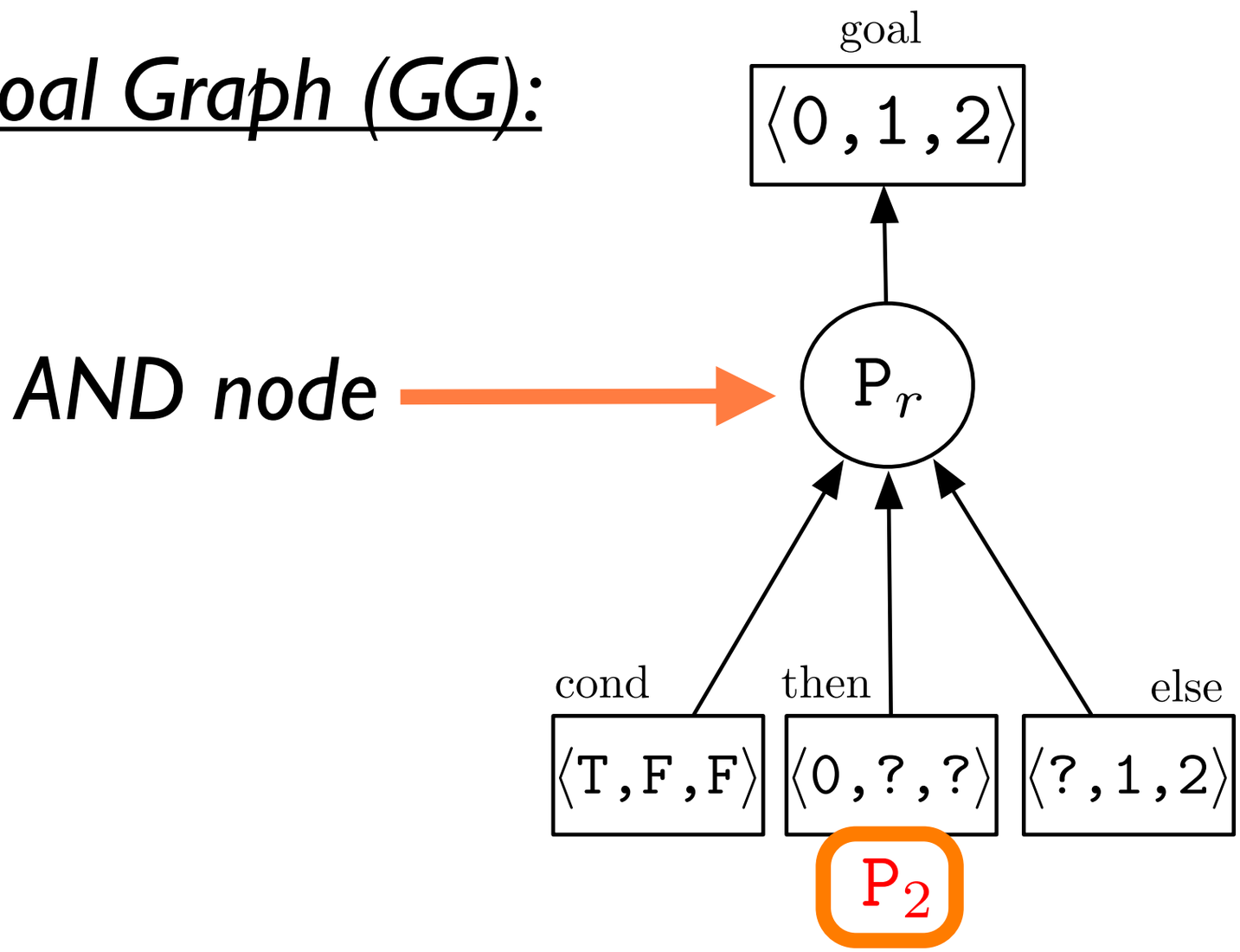


Ex: Conditional Inference

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Outputs: $\langle \mathtt{0}, \mathtt{1}, \mathtt{2} \rangle$

Goal Graph (GG):

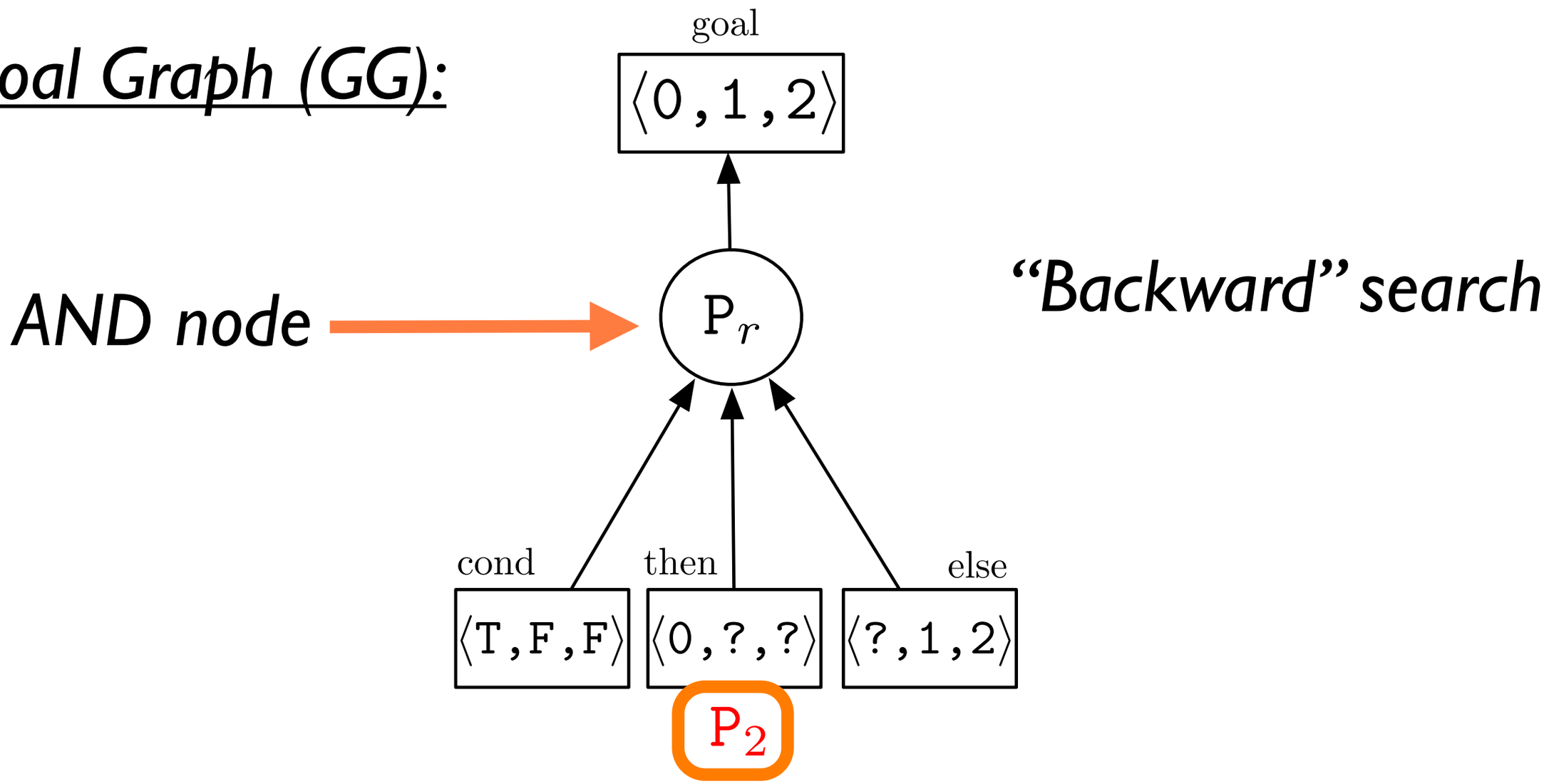


Ex: Conditional Inference

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Outputs: $\langle \mathtt{0}, \mathtt{1}, \mathtt{2} \rangle$

Goal Graph (GG):



AND node

"Backward" search

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Programs of size 2:

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, \mathtt{[]}, \mathtt{[2]} \rangle$$

$$\mathtt{P}_4 = \mathtt{inc}(\mathtt{P}_2) \rightarrow \langle \mathtt{1}, \mathtt{1}, \mathtt{1} \rangle$$

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T}, \mathtt{F}, \mathtt{F} \rangle$$

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle [], [2], [1,2] \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle 0,0,0 \rangle$$

Programs of size 2:

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathsf{err}, [], [2] \rangle$$

$$\mathtt{P}_4 = \mathtt{inc}(\mathtt{P}_2) \rightarrow \langle 1,1,1 \rangle$$

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T},\mathtt{F},\mathtt{F} \rangle$$

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Programs of size 2:

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, \mathtt{[]}, \mathtt{[2]} \rangle$$

$$\mathtt{P}_4 = \mathtt{inc}(\mathtt{P}_2) \rightarrow \langle \mathtt{1}, \mathtt{1}, \mathtt{1} \rangle$$

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T}, \mathtt{F}, \mathtt{F} \rangle$$

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0,0,0} \rangle$$

Programs of size 2:

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, \mathtt{[]}, \mathtt{[2]} \rangle$$

$$\mathtt{P}_4 = \mathtt{inc}(\mathtt{P}_2) \rightarrow \langle \mathtt{1,1,1} \rangle$$

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T,F,F} \rangle$$

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle [] , [2] , [1,2] \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle 0,0,0 \rangle$$

Programs of size 2:

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [] , [2] \rangle$$

$$\mathtt{P}_4 = \mathtt{inc}(\mathtt{P}_2) \rightarrow \langle 1,1,1 \rangle$$

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T},\mathtt{F},\mathtt{F} \rangle$$

Ex: Forward Search 2

Programs of size 1:

$$\mathtt{P}_1 = \mathtt{i} \rightarrow \langle \mathtt{[]}, \mathtt{[2]}, \mathtt{[1,2]} \rangle$$

$$\mathtt{P}_2 = \mathtt{zero} \rightarrow \langle \mathtt{0}, \mathtt{0}, \mathtt{0} \rangle$$

Programs of size 2:

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, \mathtt{[]}, \mathtt{[2]} \rangle$$

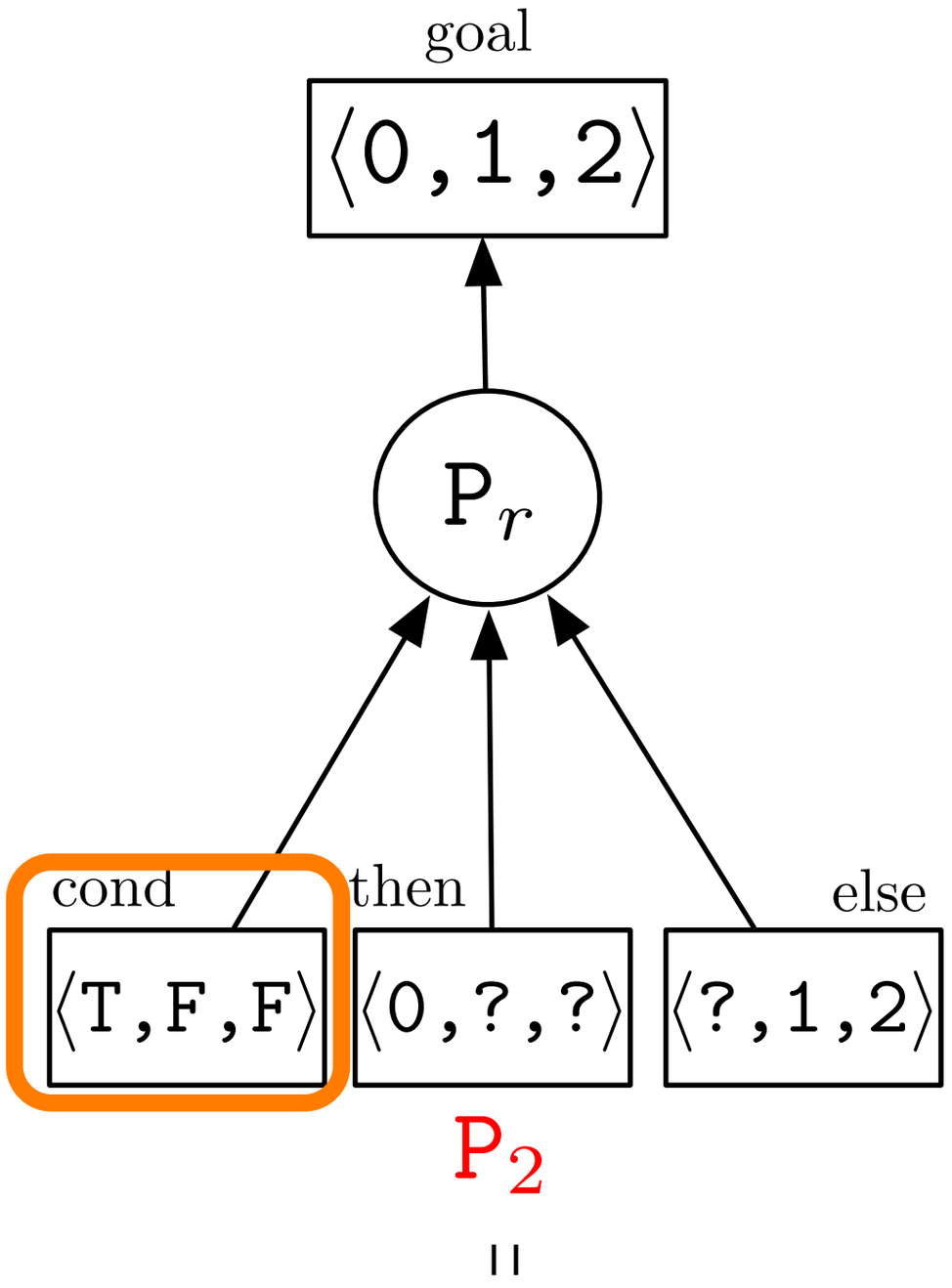
$$\mathtt{P}_4 = \mathtt{inc}(\mathtt{P}_2) \rightarrow \langle \mathtt{1}, \mathtt{1}, \mathtt{1} \rangle$$

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T}, \mathtt{F}, \mathtt{F} \rangle$$

Satisfies one of our goals

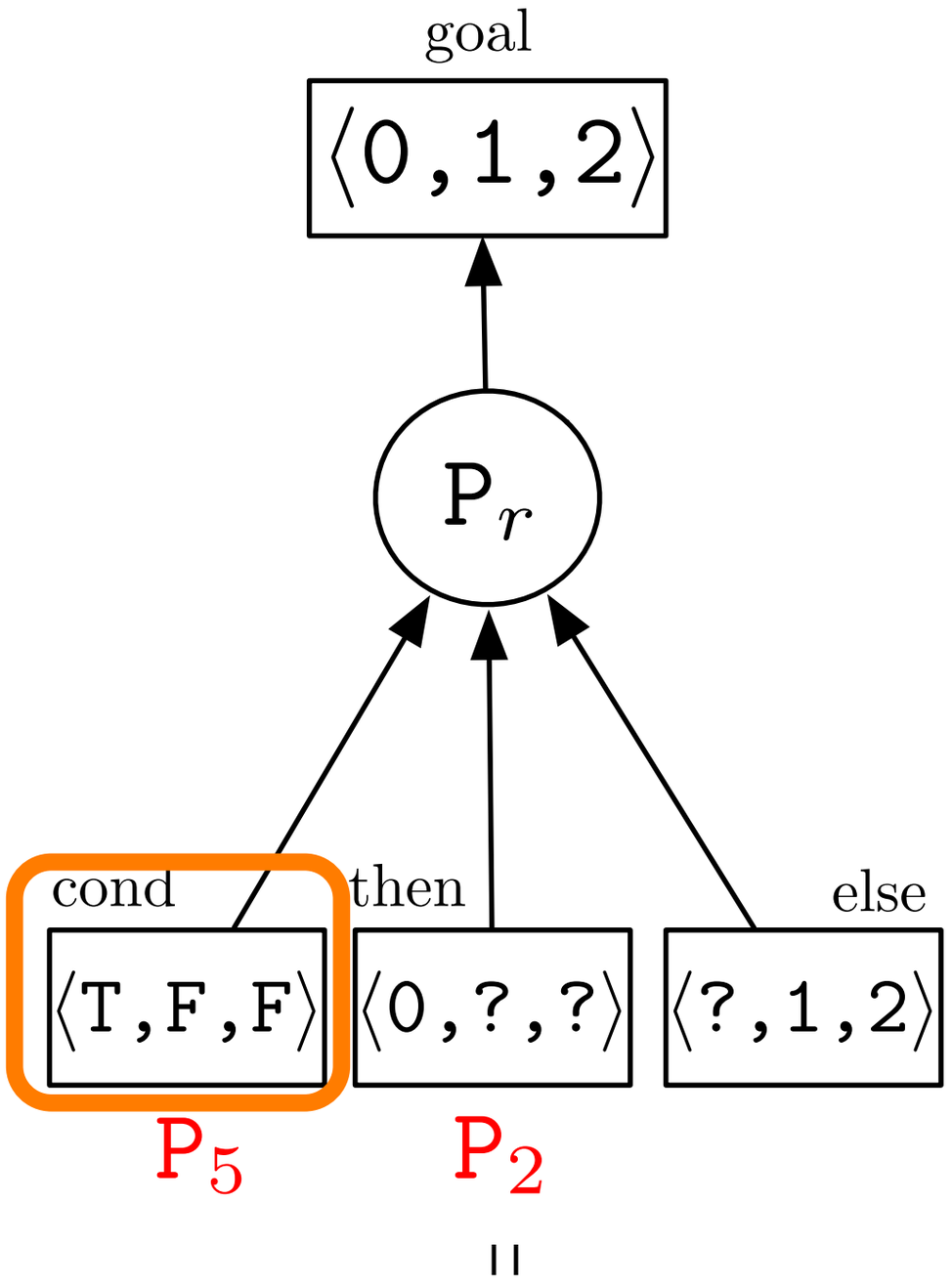
Ex: Conditional Inference 2

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T},\mathtt{F},\mathtt{F} \rangle$$



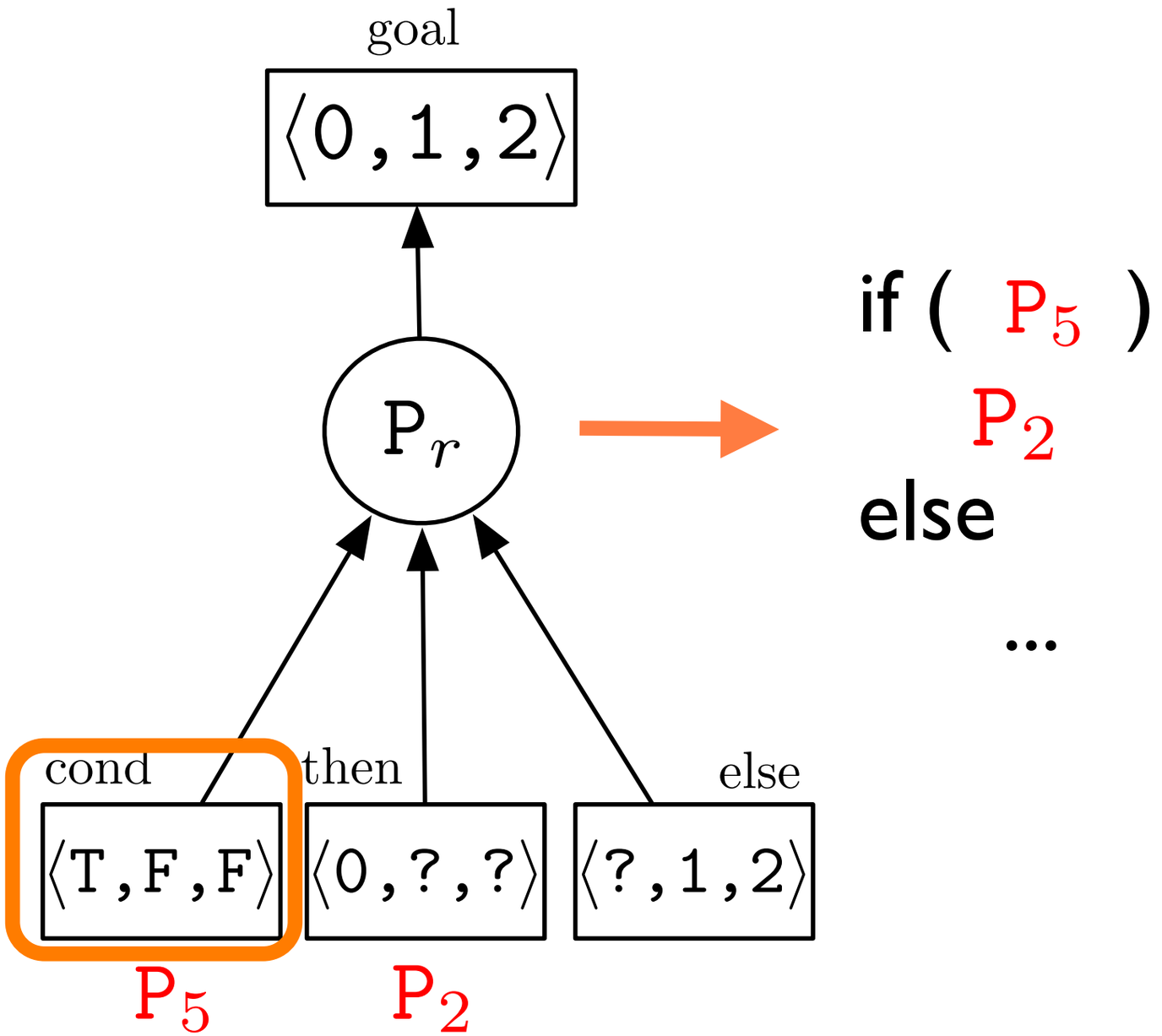
Ex: Conditional Inference 2

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T},\mathtt{F},\mathtt{F} \rangle$$



Ex: Conditional Inference 2

$$\mathtt{P}_5 = \mathtt{isEmpty}(\mathtt{P}_1) \rightarrow \langle \mathtt{T},\mathtt{F},\mathtt{F} \rangle$$



Ex: Forward Search 3

Programs of size 2

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [], [2] \rangle$$

...

Ex: Forward Search 3

Programs of size 2

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [], [2] \rangle$$

...

Programs of size 3

$$\mathtt{P}_6 = \mathtt{length}(\mathtt{P}_3) \rightarrow \langle \mathtt{err}, 0, 1 \rangle$$

Ex: Forward Search 3

Programs of size 2

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [], [2] \rangle$$

...

Programs of size 3

$$\mathtt{P}_6 = \mathtt{length}(\mathtt{P}_3) \rightarrow \langle \mathtt{err}, 0, 1 \rangle$$

Use I/O values to simulate recursive call

Ex: Forward Search 3

Programs of size 2

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [\,], [2] \rangle$$

...

Programs of size 3

$$\mathtt{P}_6 = \mathtt{length}(\mathtt{P}_3) \rightarrow \langle \mathtt{err}, 0, 1 \rangle$$

Use I/O values to simulate recursive call

Ex: Forward Search 3

Programs of size 2

$$\mathrm{P}_3 = \texttt{tail}(\mathrm{P}_1) \rightarrow \langle \texttt{err}, \texttt{[]}, \texttt{[2]} \rangle$$

...

Programs of size 3

$$\mathrm{P}_6 = \texttt{length}(\mathrm{P}_3) \rightarrow \langle \texttt{err}, \texttt{0}, \texttt{1} \rangle$$

Use I/O values to simulate recursive call

Ex: Forward Search 3

Programs of size 2

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [], [2] \rangle$$

...

Programs of size 3

$$\mathtt{P}_6 = \mathtt{length}(\mathtt{P}_3) \rightarrow \langle \mathtt{err}, 0, 1 \rangle$$

Use I/O values to simulate recursive call

Ex: Forward Search 3

Programs of size 2

$$\mathtt{P}_3 = \mathtt{tail}(\mathtt{P}_1) \rightarrow \langle \mathtt{err}, [], [2] \rangle$$

...

Programs of size 3

$$\mathtt{P}_6 = \mathtt{length}(\mathtt{P}_3) \rightarrow \langle \mathtt{err}, 0, 1 \rangle$$

Use I/O values to simulate recursive call

Only allow calls satisfying well-founded relation

Ex: Forward Search 3

Programs of size 2

$P_3 =$ ⟩

...

Programs o

P_6 = 0, 1⟩

But what if there isn't such an I/O pair?

Query the user for new examples!

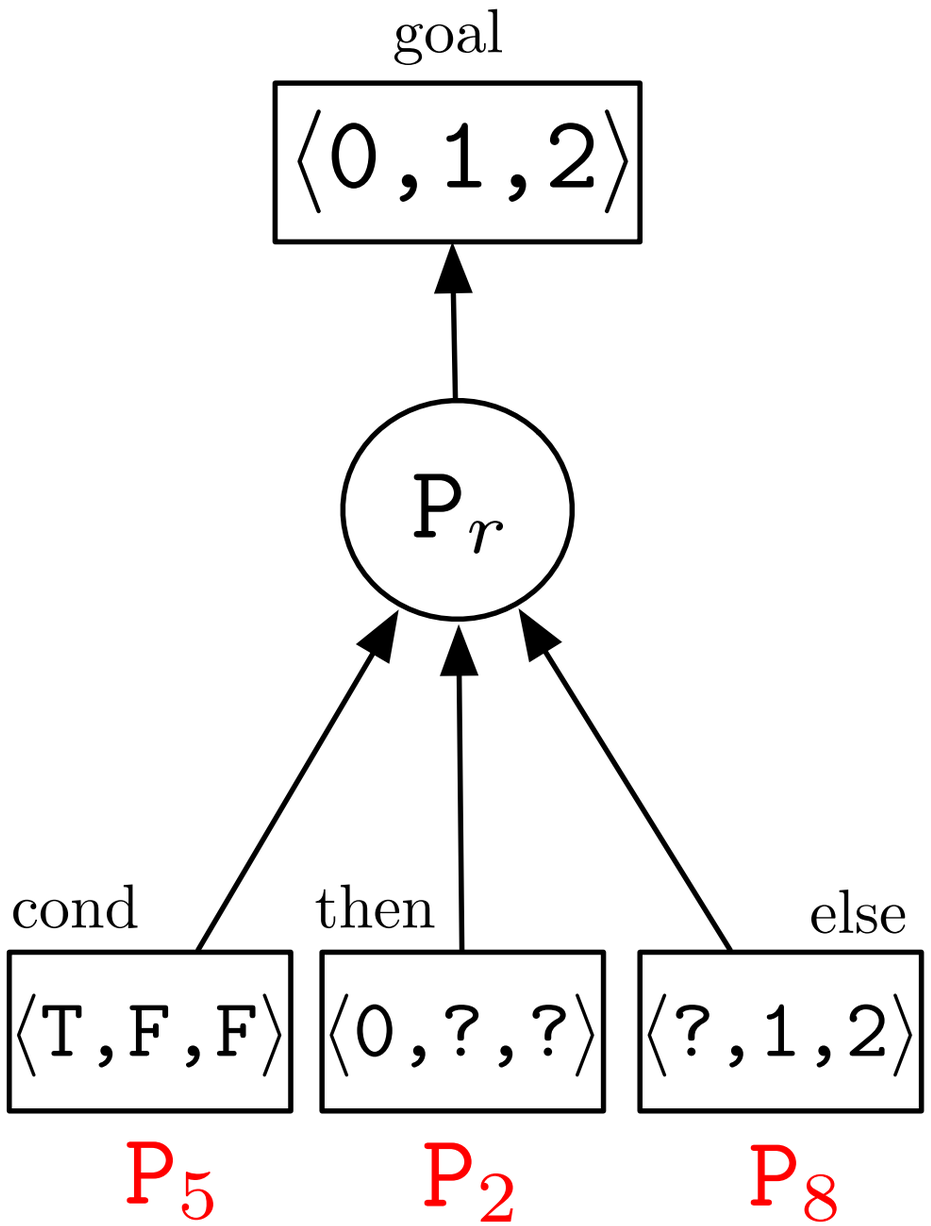
(too long for 15 min talk)

Use I/O

Only allow

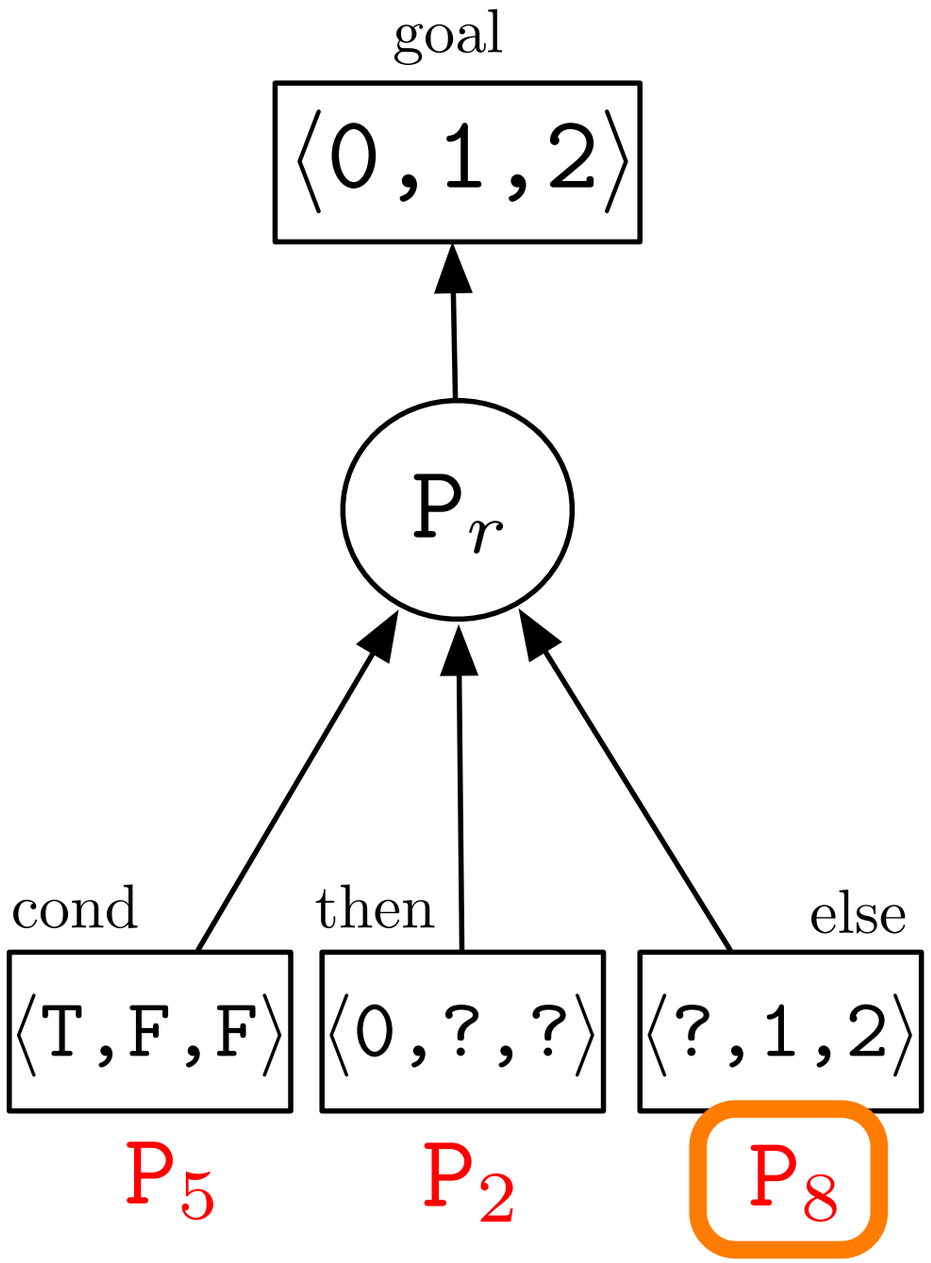
Ex: Conditional Inference

One iteration later:



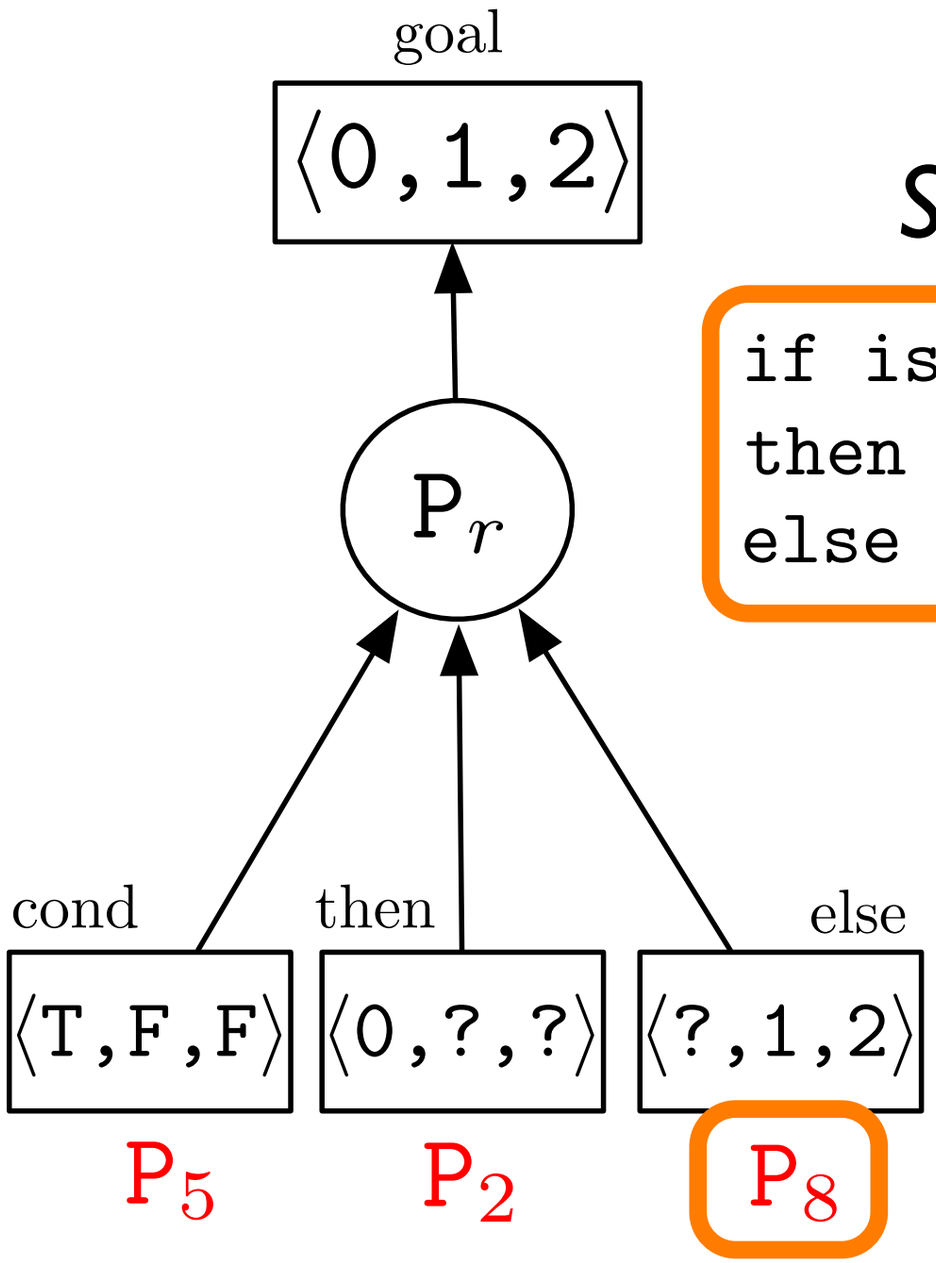
Ex: Conditional Inference

One iteration later:



Ex: Conditional Inference

One iteration later:



Synthesized program:

```
if isEmpty(i) P5
then 0 P2
else inc(length(tail(i))) P8
```

Escher: Recap

Forward Search

- *Apply component to all synthesized programs*
- *Heuristic-based search*

Escher: Recap

Forward Search

- *Apply component to all synthesized programs*
- *Heuristic-based search*

Conditional Inference

- *Uses goal graph to synthesize conditionals*
- *Conditionals synthesized on demand*

Escher: Recap

Forward Search

- *Apply component to all synthesized programs*
- *Heuristic-based search*

Conditional Inference

- *Uses goal graph to synthesize conditionals*
- *Conditionals synthesized on demand*

Recursive call synthesis

- *Use I/O specification*
- *Query user for more output values if needed*

Implementation

Prototype implementation of Escher

- *Pluggable components*

Experimented with:

- *integer, list, and tree manipulating programs*

Experiments

(tree manipulating programs)

Time in seconds

	Escher	w/o ObsEquiv	w/o GoalGraph	w/o OE+GG
collect_leaves	**0.04**	0.09	68.9	81.8
count_leaves	**0.06**	0.2	9.3	12.3
hbal_tree	**1.5**	MEM	TIME	MEM
nodes_at_level	**10.74**	MEM	TIME	MEM

Experiments

(#components sensitivity: mult*)*

Escher: 0.7s

with all components

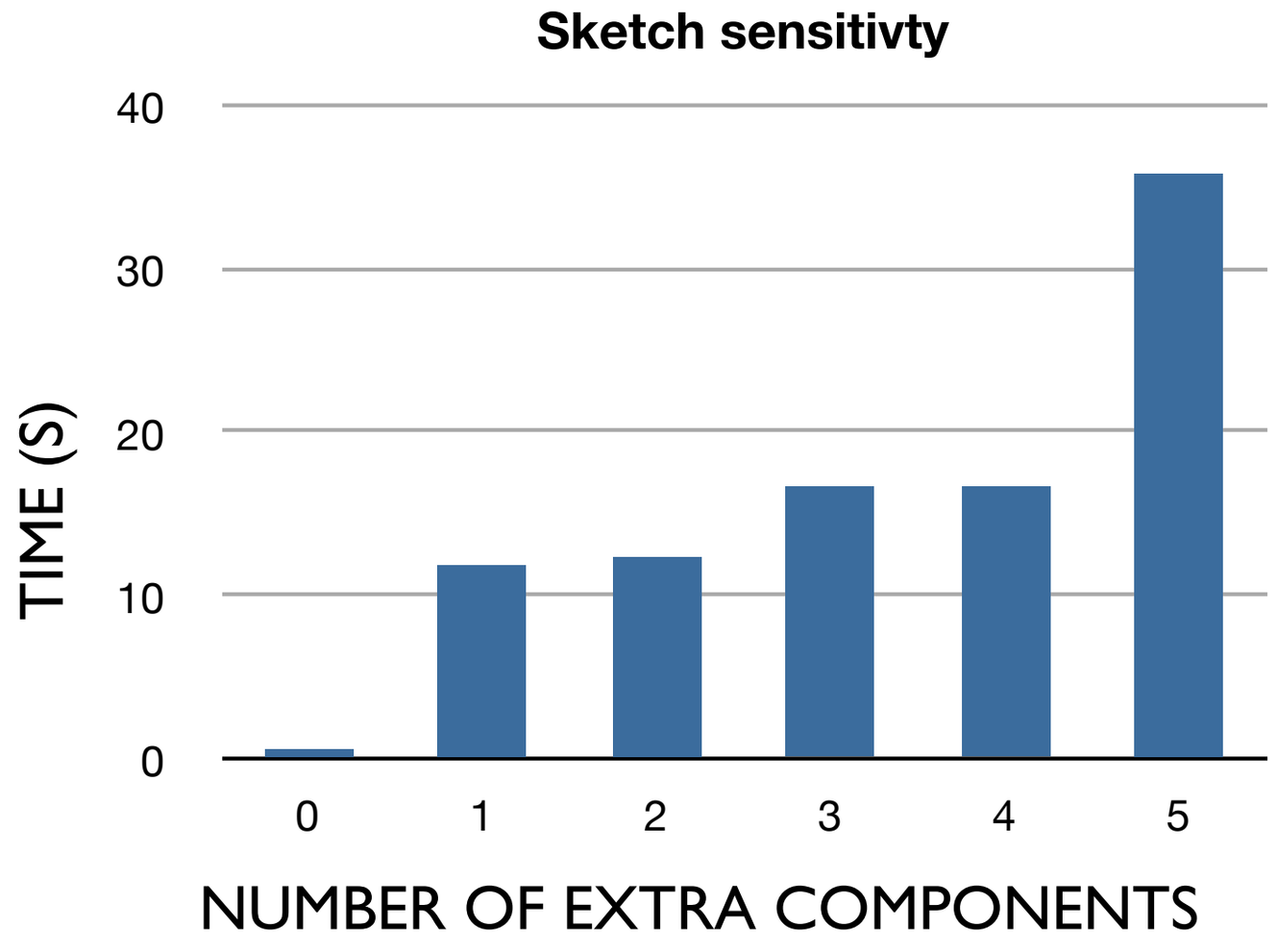
Experiments

(#components sensitivity: mult*)*

SAT-based synthesis [ASPLOS'06]

Escher: 0.7s

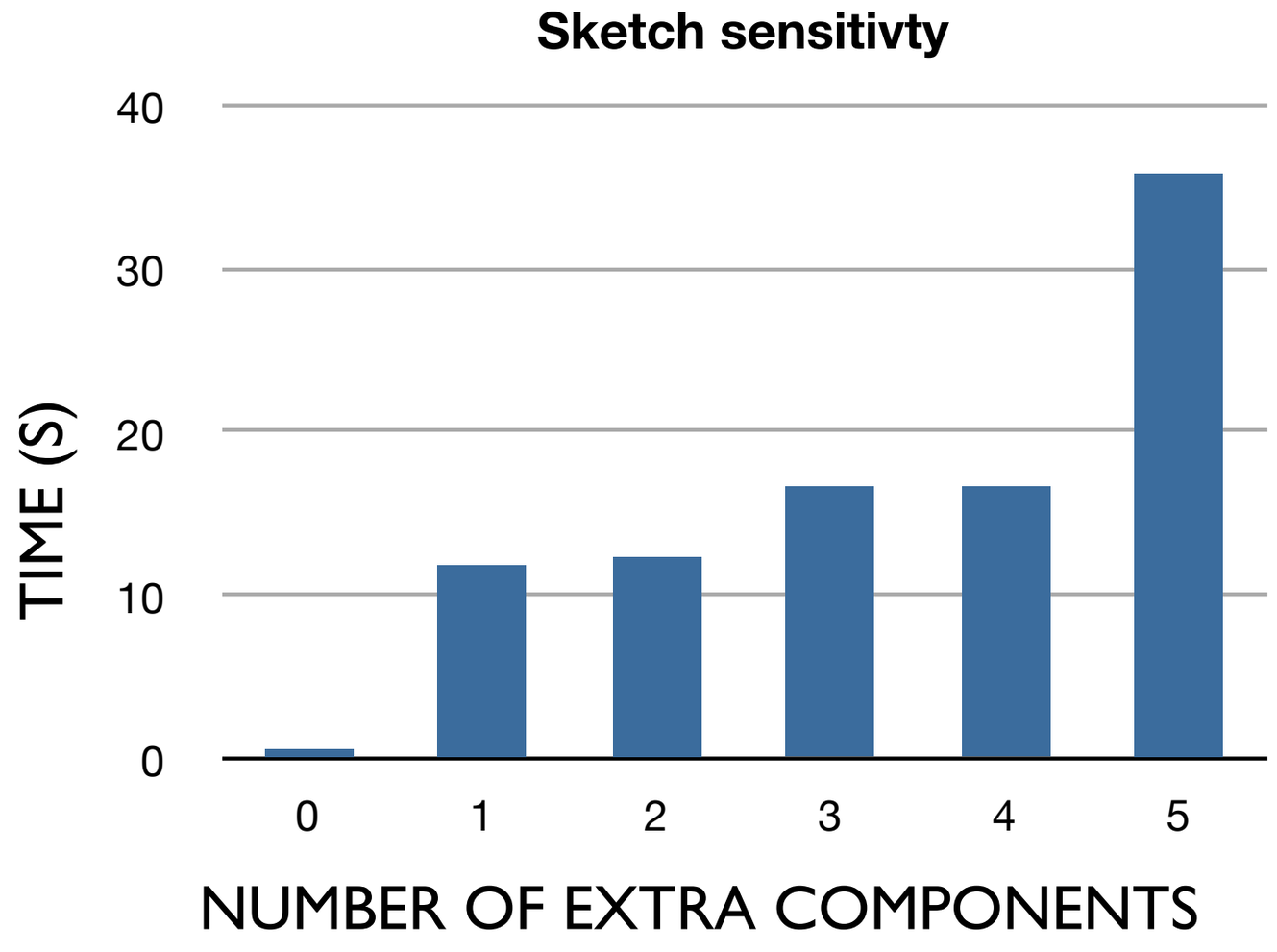
with all components



Experiments

(#components sensitivity: mult*)*

SAT-based synthesis [ASPLOS'06]

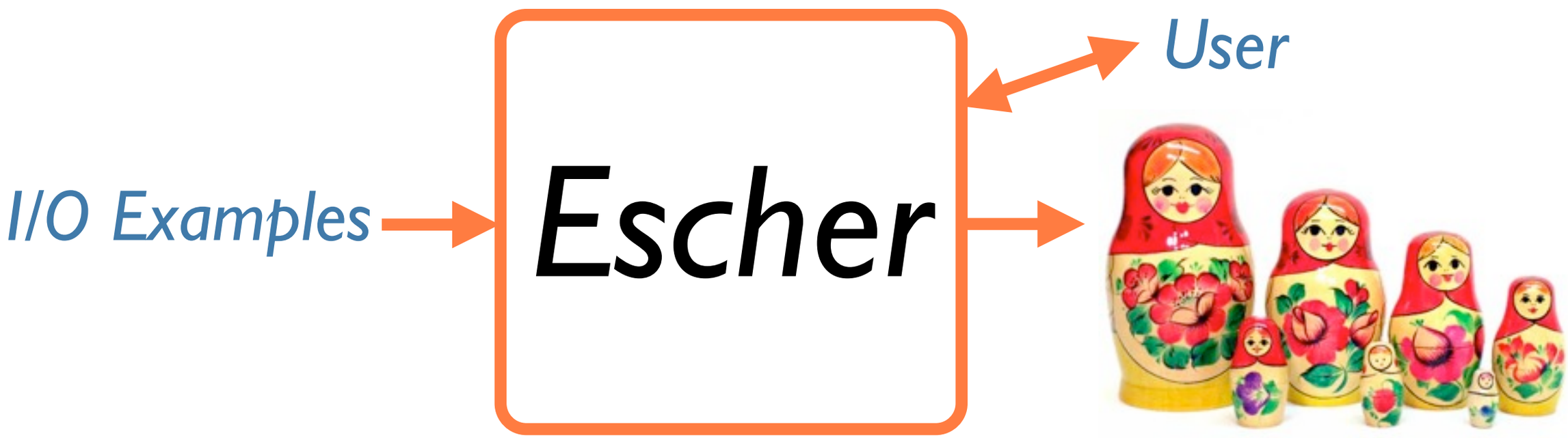


Escher: 0.7s

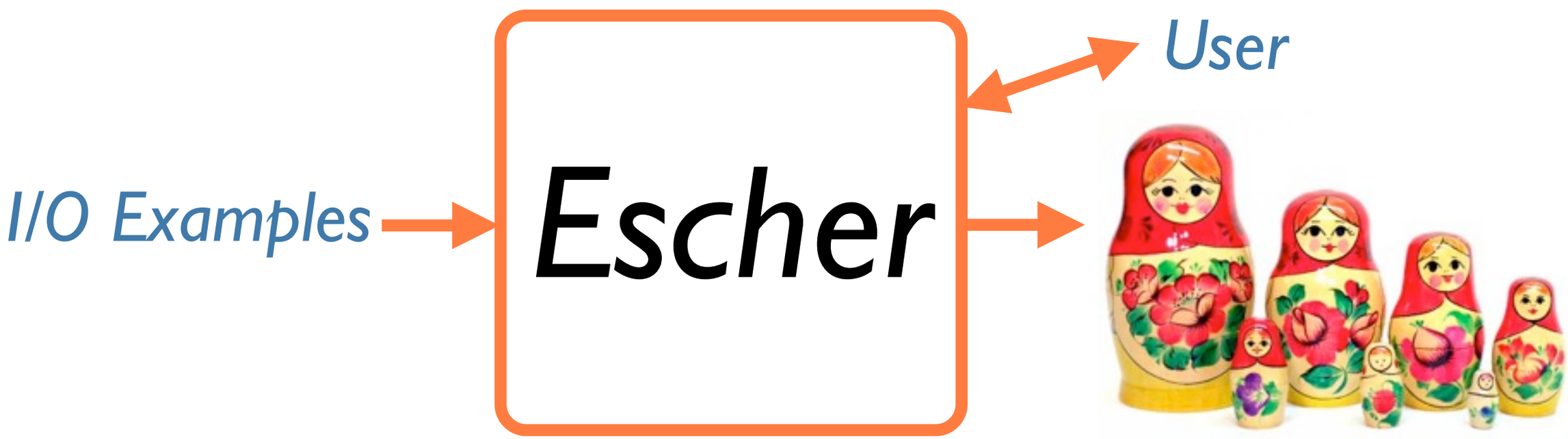
with all components

Had to supply Sketch with high-level conditional

Conclusion

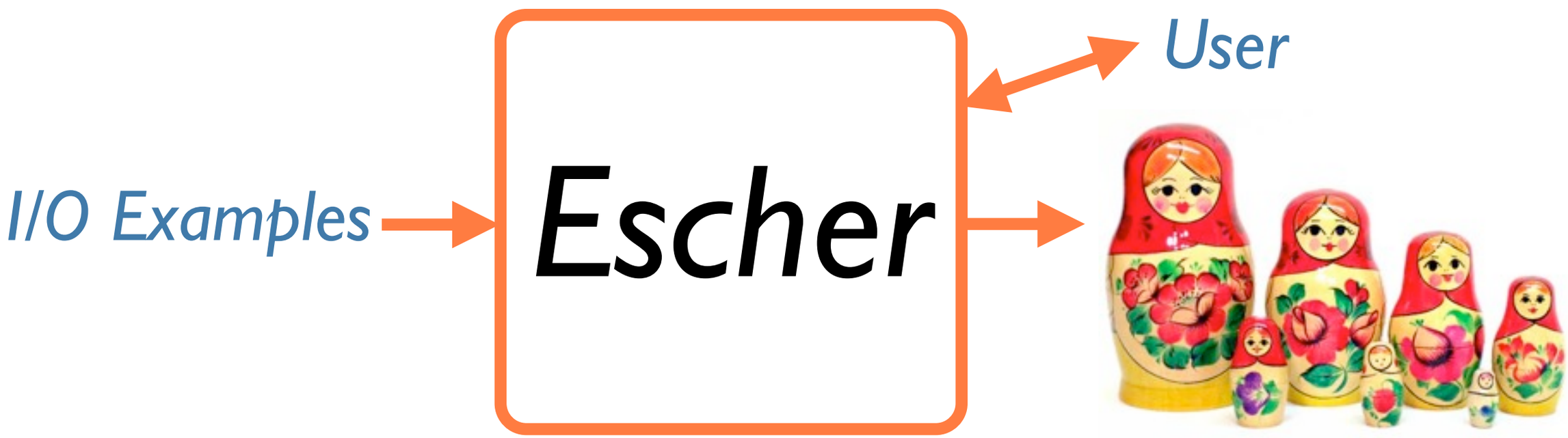


Conclusion



Interactive

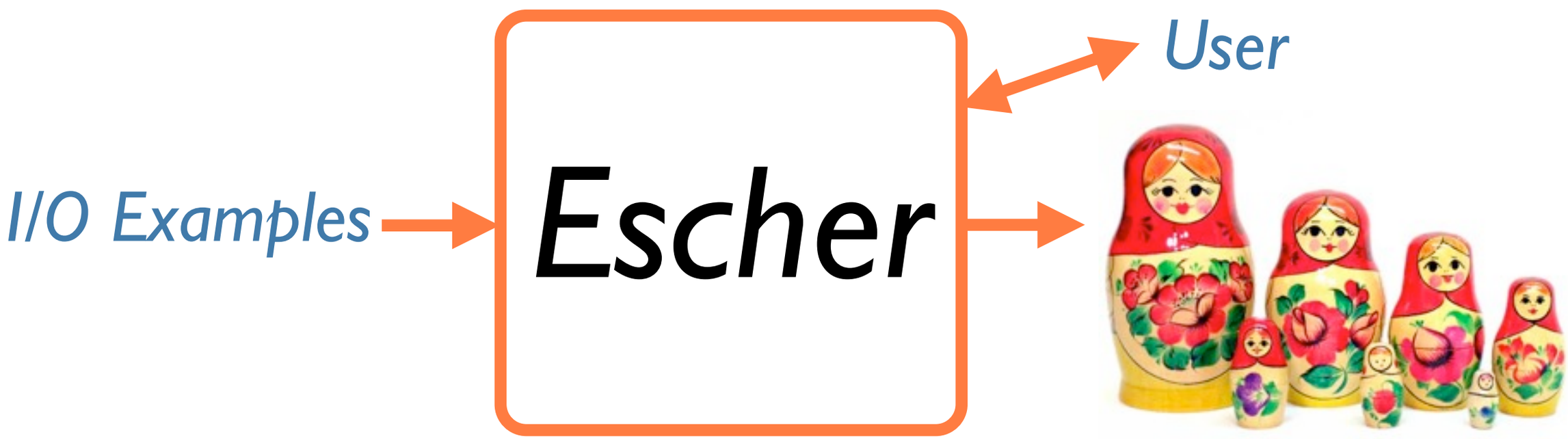
Conclusion



Interactive

Highly-customizable

Conclusion



Interactive

Highly-customizable

Novel synthesis techniques

Questions?

How can we synthesize loops?

Questions?

How can we synthesize loops?

Integration with SMT-based search

Questions?

How can we synthesize loops?

Integration with SMT-based search

Applications of Escher in eduction

- e.g., for interacting with students

Questions?

How can we synthesize loops?

Integration with SMT-based search

Applications of Escher in eduction

- e.g., for interacting with students

Theoretical under-pinnings

- e.g., see *Madhusudan [CSL'11]*

Thank You