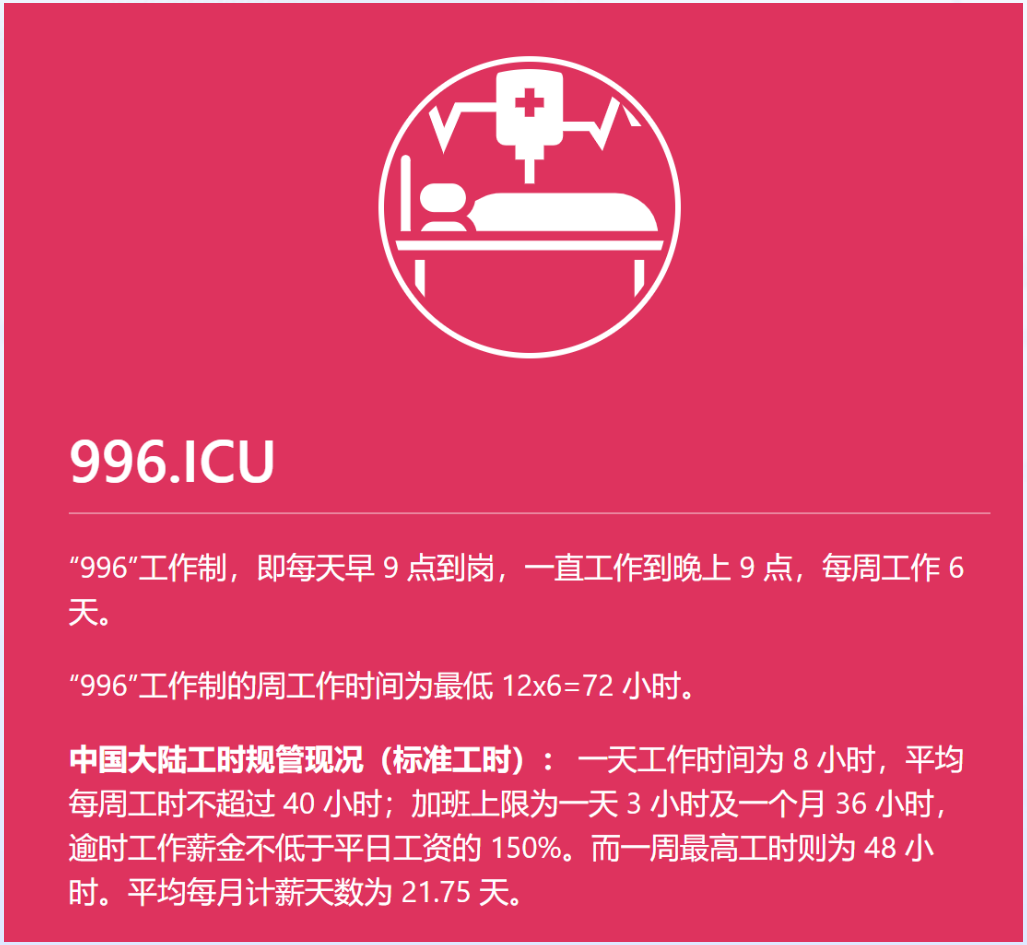
交互式程序综合中的问题选择

熊英飞
北京大学

现代程序员的工作时间



996.ICU

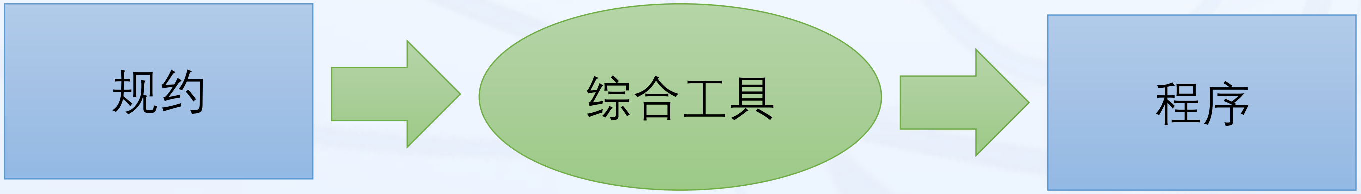
“996”工作制，即每天早 9 点到岗，一直工作到晚上 9 点，每周工作 6 天。

“996”工作制的周工作时间为最低 12x6=72 小时。

中国大陆工时规管现况（标准工时）： 一天工作时间为 8 小时，平均每周工时不超过 40 小时；加班上限为一天 3 小时及一个月 36 小时，逾时工作薪金不低于平日工资的 150%。而一周最高工时则为 48 小时。平均每月计薪天数为 21.75 天。

科学家试图拯救程序员

程序综合



规约

综合工具

程序

“**One of the most central problems in the theory of programming.**”

----Amir Pneuli

图灵奖获得者

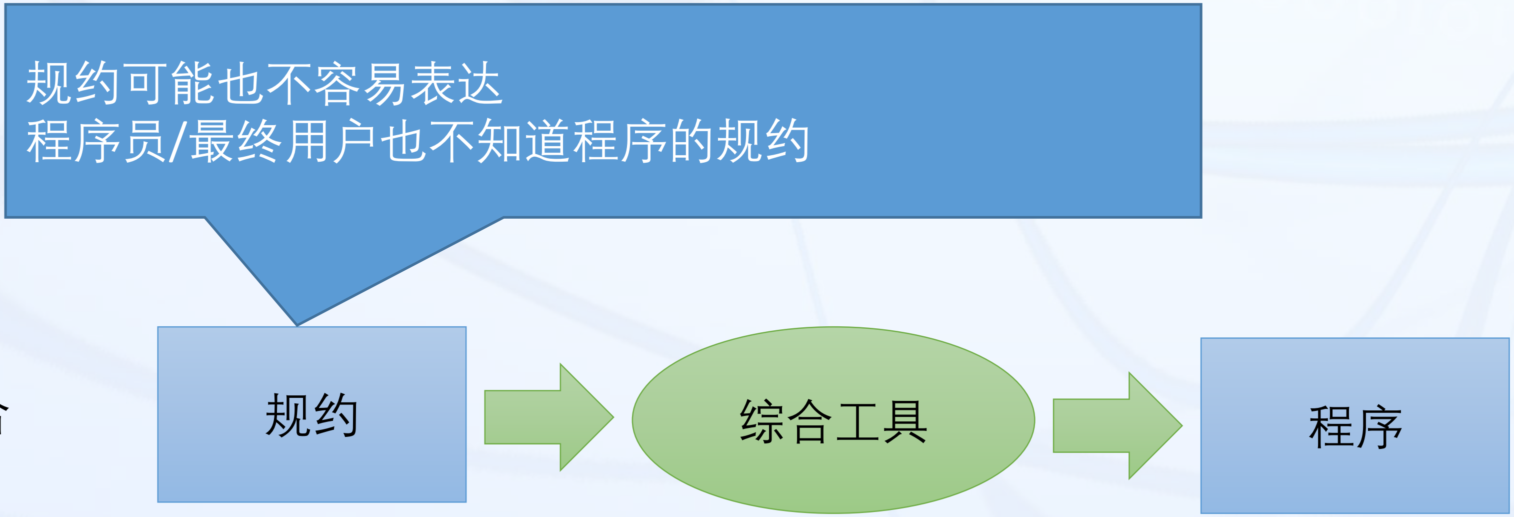
“**提升软件生产率的根本途径**”

----徐家福先生

中国软件先驱

科学家试图拯救程序员

程序综合



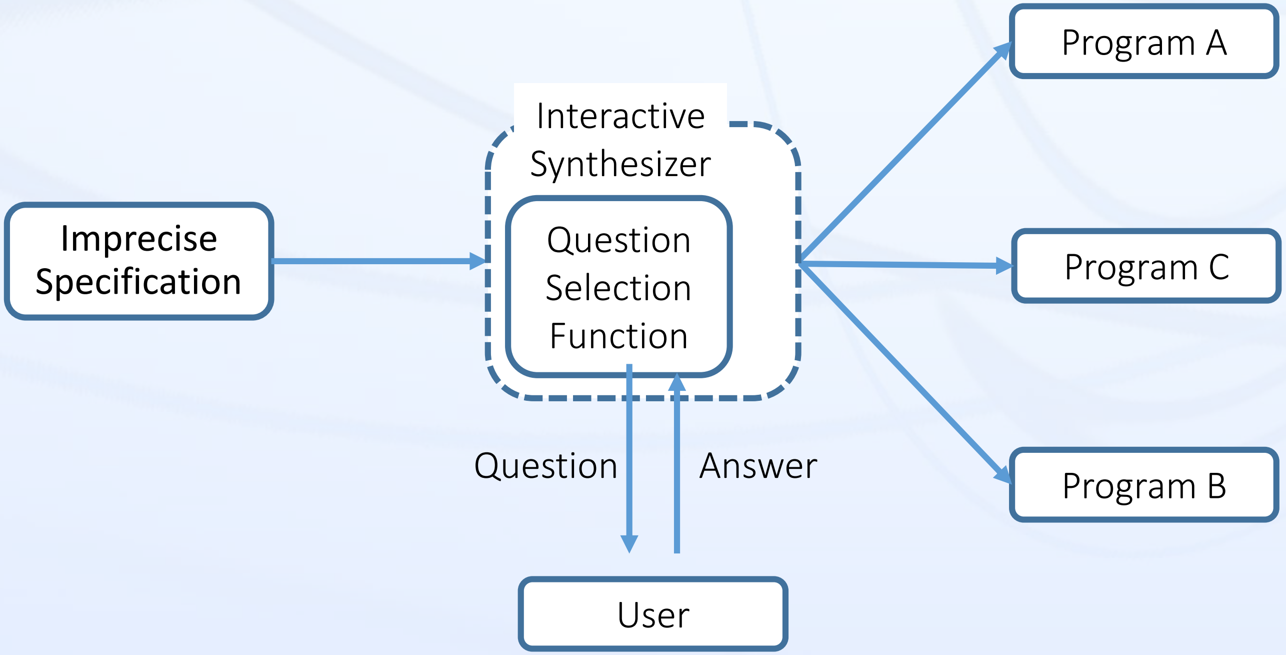
规约可能也不容易表达
程序员/最终用户也不知道程序的规约

“One of the most central problems in the theory of programming.”
----Amir Pneuli
图灵奖获得者

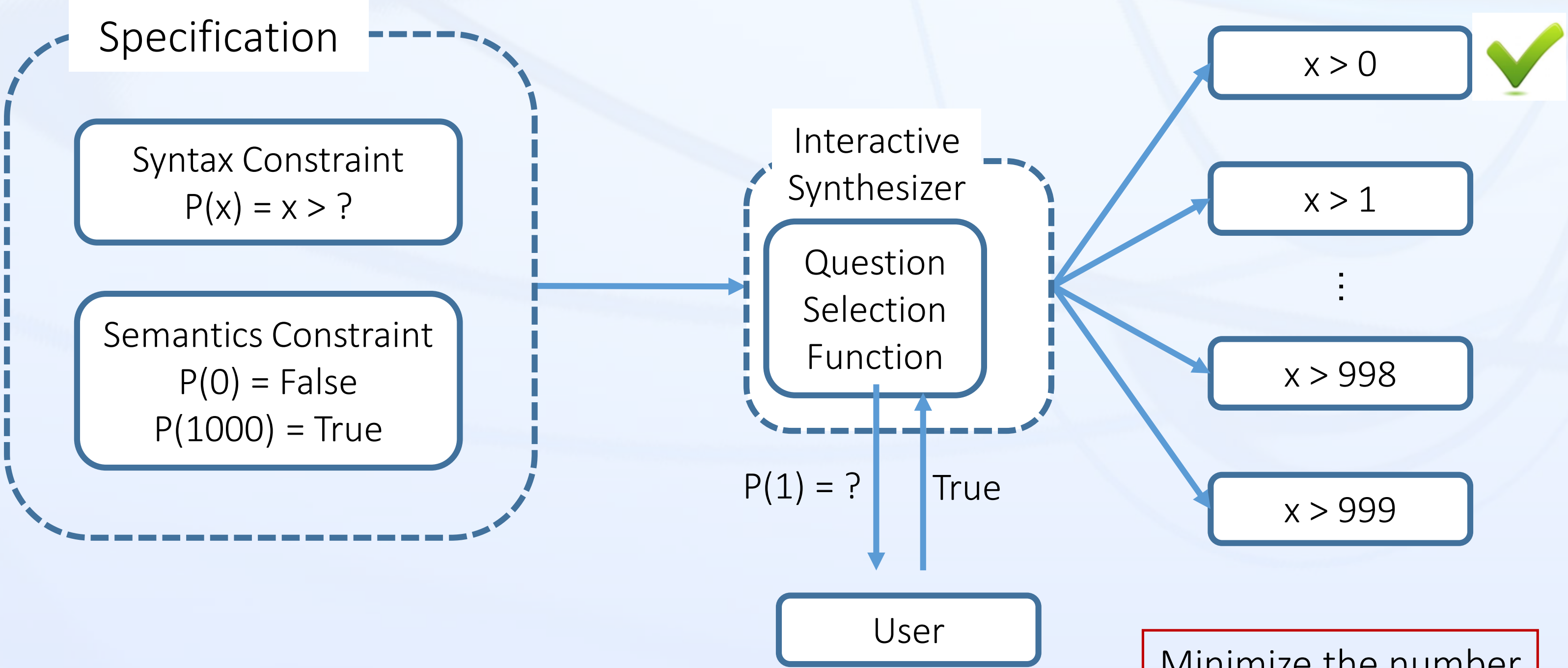
“提升软件生产率的根本途径”
----徐家福先生
中国软件先驱

能否通过向人类提问获取规约？

Interactive Program Synthesis



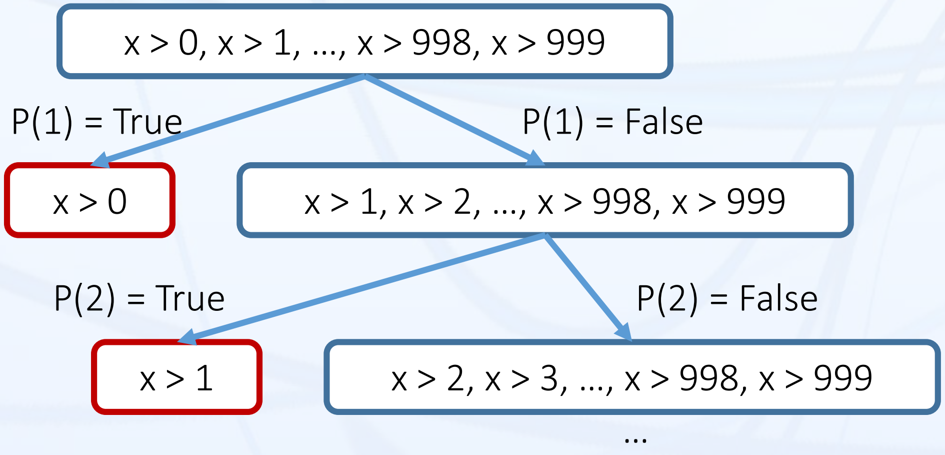
Example



Minimize the number of questions

What is a Good Question Selection Function?

- Strategy 1: Ask P(1) – P(999) in order
  - Minimal: 1 question
  - Maximal: 999 questions ✓



x > 0, x > 1, …, x > 998, x > 999

P(1) = True

x > 0

P(1) = False

x > 1, x > 2, …, x > 998, x > 999

P(2) = True

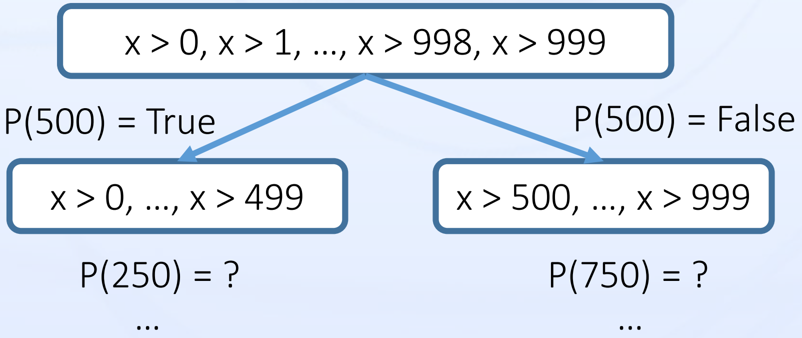
x > 1

P(2) = False

x > 2, x > 3, …, x > 998, x > 999

…

- Strategy 2: Binary Search
  - Minimal: 9 questions
  - Maximal: 10 questions ✓



x > 0, x > 1, …, x > 998, x > 999

P(500) = True

P(500) = False

x > 0, …, x > 499

x > 500, …, x > 999

P(250) = ?

…

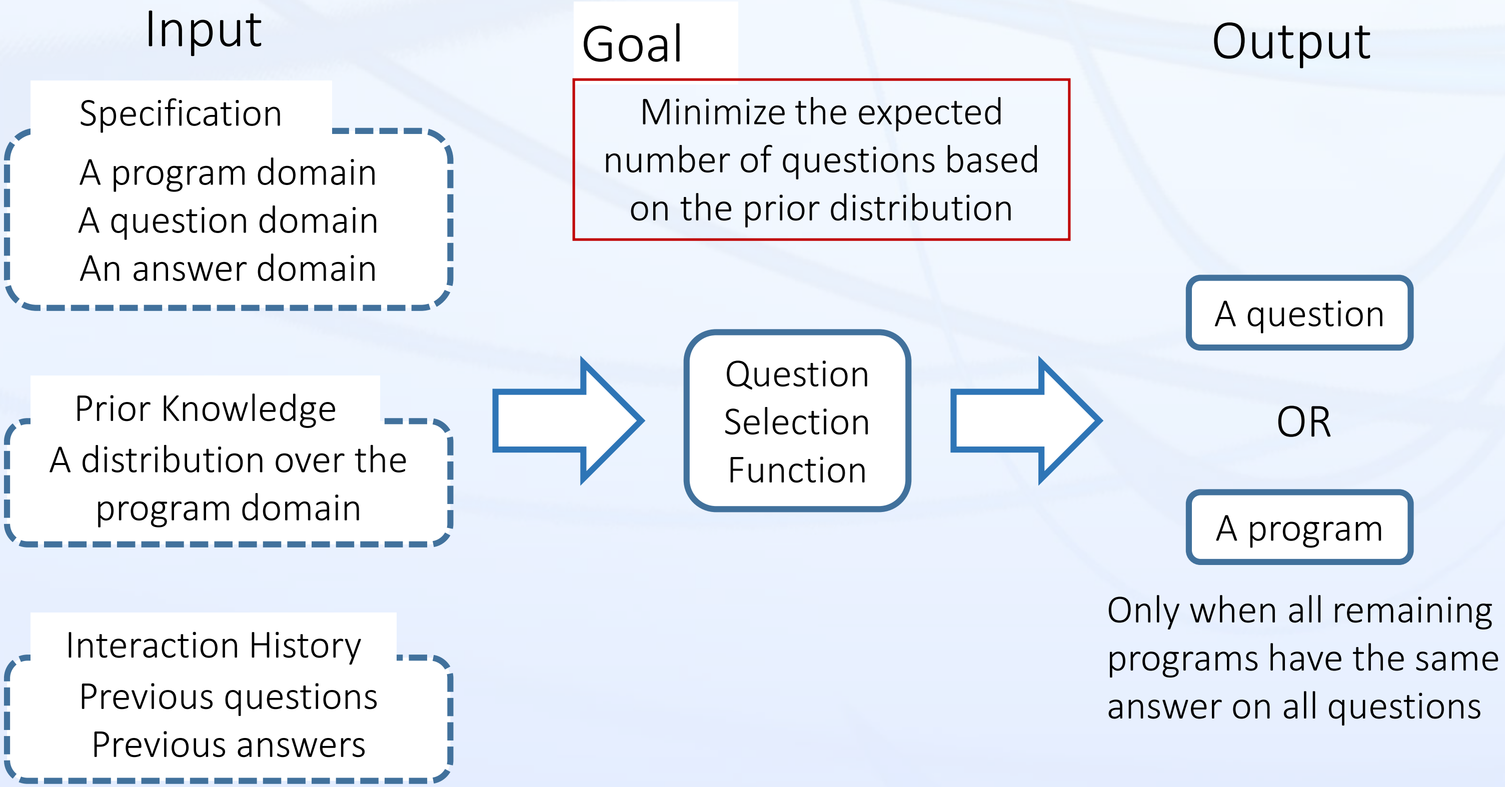
P(750) = ?

…

Prior Knowledge

All candidates have the same chance

Question Selection Problem



Input

Specification

A program domain
A question domain
An answer domain

Prior Knowledge

A distribution over the program domain

Interaction History

Previous questions
Previous answers

Goal

Minimize the expected number of questions based on the prior distribution

Question Selection Function

Output

A question

OR

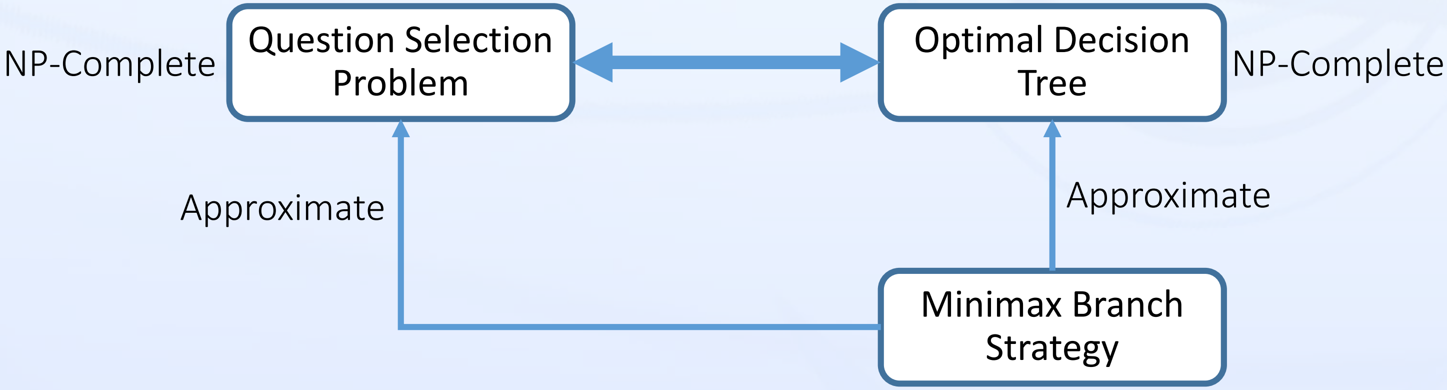
A program

Only when all remaining programs have the same answer on all questions

How Well Can We Do?

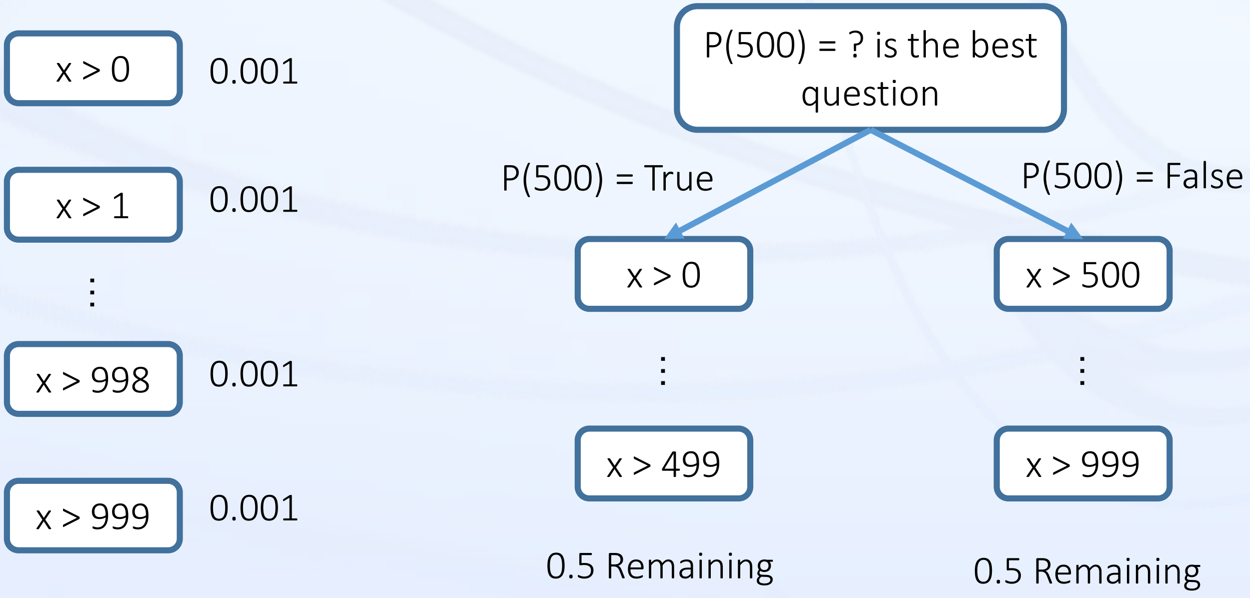
Can we implement an efficient question selection function that minimizes the expected number of questions?

- We cannot, unless P = NP.
- Fortunately, there is an effective approximation algorithm available.



Minimax Branch Strategy

- Minimize the probability of remaining programs in the worst case.



x > 0 0.001

x > 1 0.001

⋮

x > 998 0.001

x > 999 0.001

P(500) = ? is the best question

P(500) = True

P(500) = False

x > 0

⋮

x > 499

0.5 Remaining

x > 500

⋮

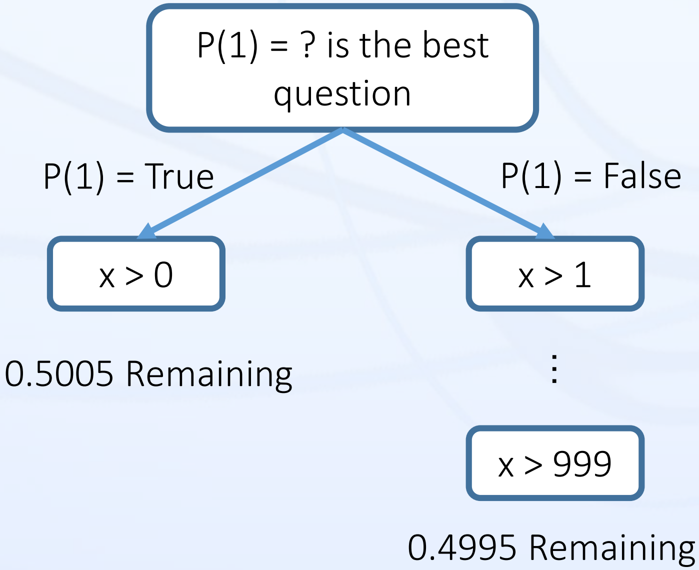
x > 999

0.5 Remaining

Minimax Branch Strategy

- Minimize the probability of remaining programs in the worst case.

x > 0	0.5005
x > 1	0.0005
⋮	
x > 998	0.0005
x > 999	0.0005



P(1) = ? is the best question

P(1) = True

P(1) = False

x > 0

0.5005 Remaining

x > 1

⋮

x > 999

0.4995 Remaining

Utilizing Minimax Branch Strategy

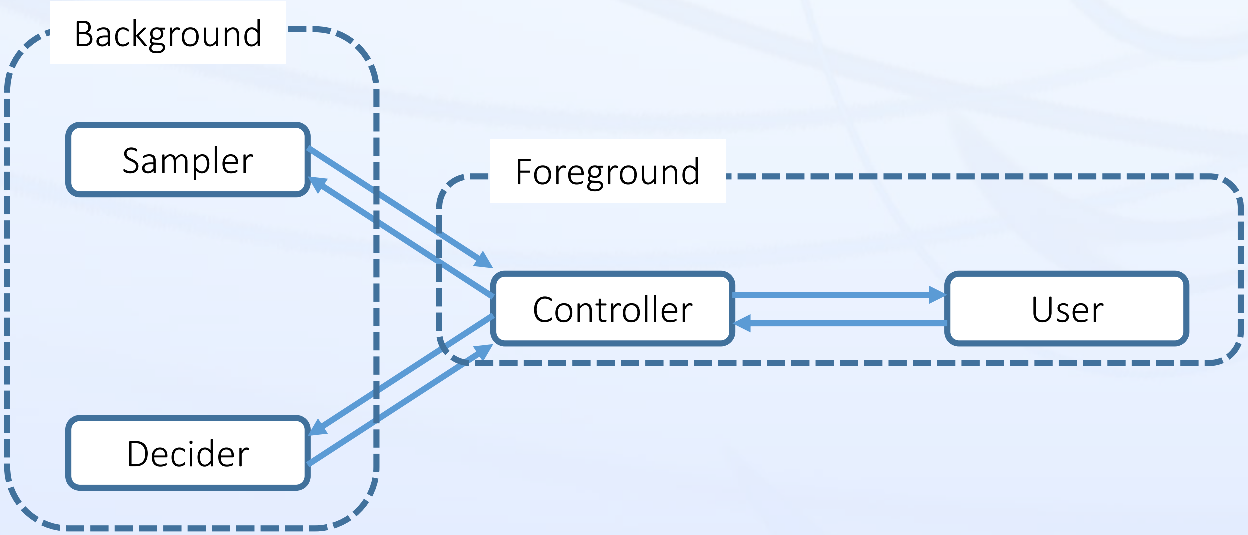
Key Challenges:

1. The program domain can be extremely large. It’s hard to find the best question quickly.
2. Interactive program synthesis has a high requirement on the number of questions. Even minimax branch strategy may use too many questions.

SampleSy & *EpsSy*

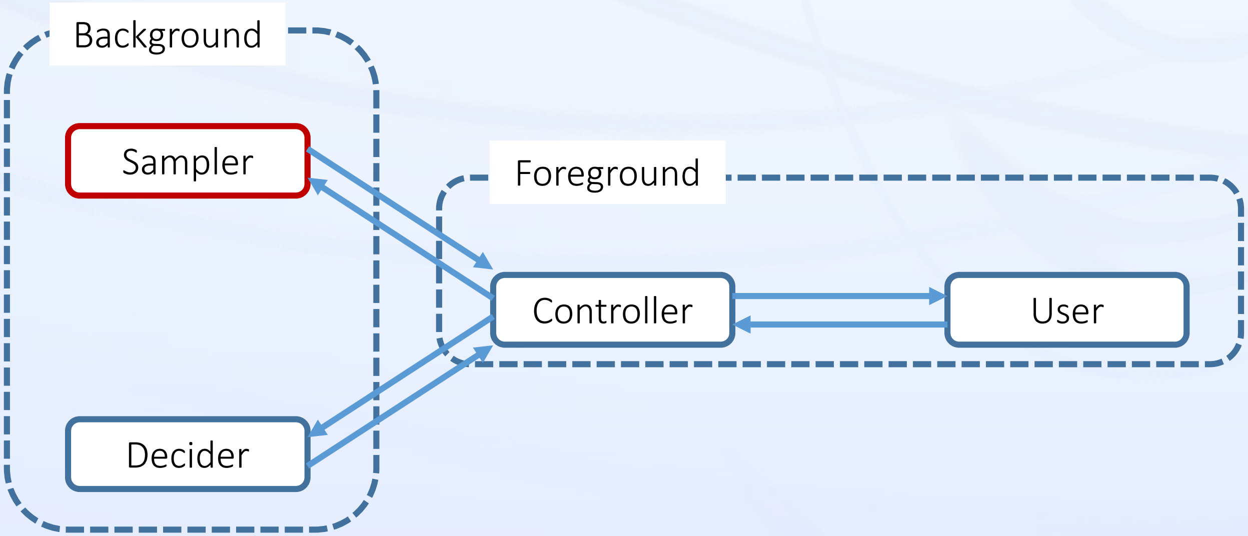
SampleSy

- **Key Idea.** Using samples to represent the whole program space.
- **Problem.** Sampling is time-consuming. The response time may be too long.
- **Parallelization.** Utilize the time when the user is answering the question.



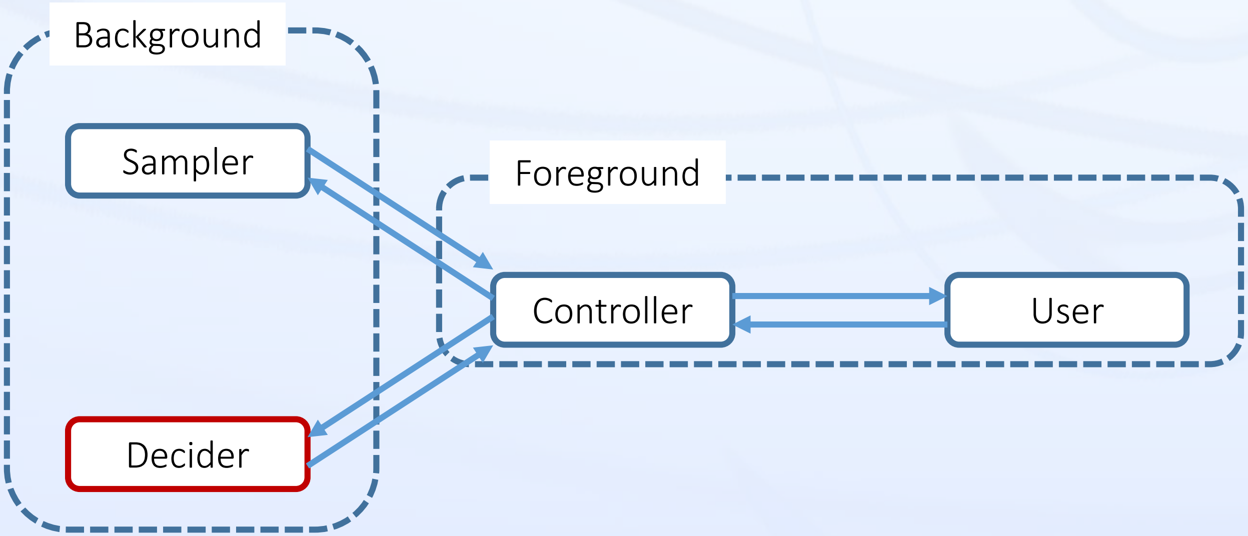
SampleSy

- **Key Idea.** Using samples to represent the whole program space.
- **Sampler.** A special synthesizer
  - Constantly sample from remaining programs according to the prior distribution



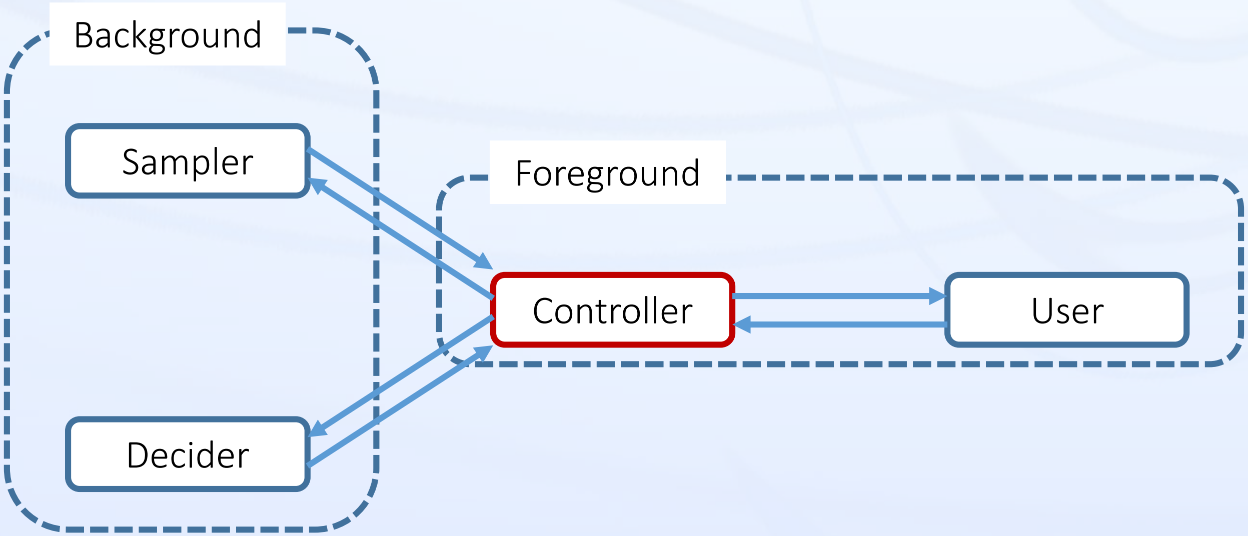
SampleSy

- **Key Idea.** Using samples to represent the whole program space.
- **Decider.** Whether there are two remaining programs differing on some question?
  - Encoding technique (e.g., component based synthesis) + SMT solver

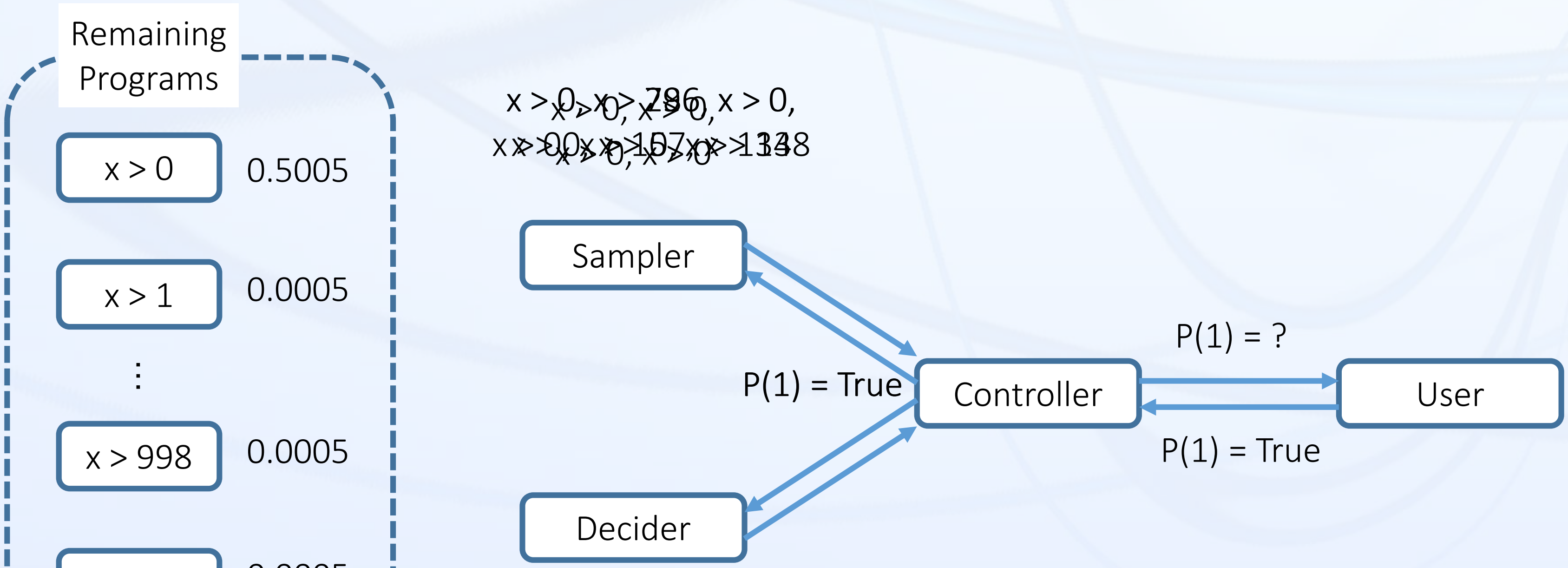


SampleSy

- **Key Idea.** Using samples to represent the whole program space.
- **Controller.** Use an SMT solver to find the best question on samples.
  - Select the question that excludes the most samples in the worst case.



Running Example: *SampleSy*



Remaining Programs

x > 0	0.5005
x > 1	0.0005
⋮	
x > 998	0.0005
x > 999	0.0005

x > 0, x > 296, x > 0,
x > 0, x > 107, x > 138

Sampler

Decider

Controller

User

P(1) = True

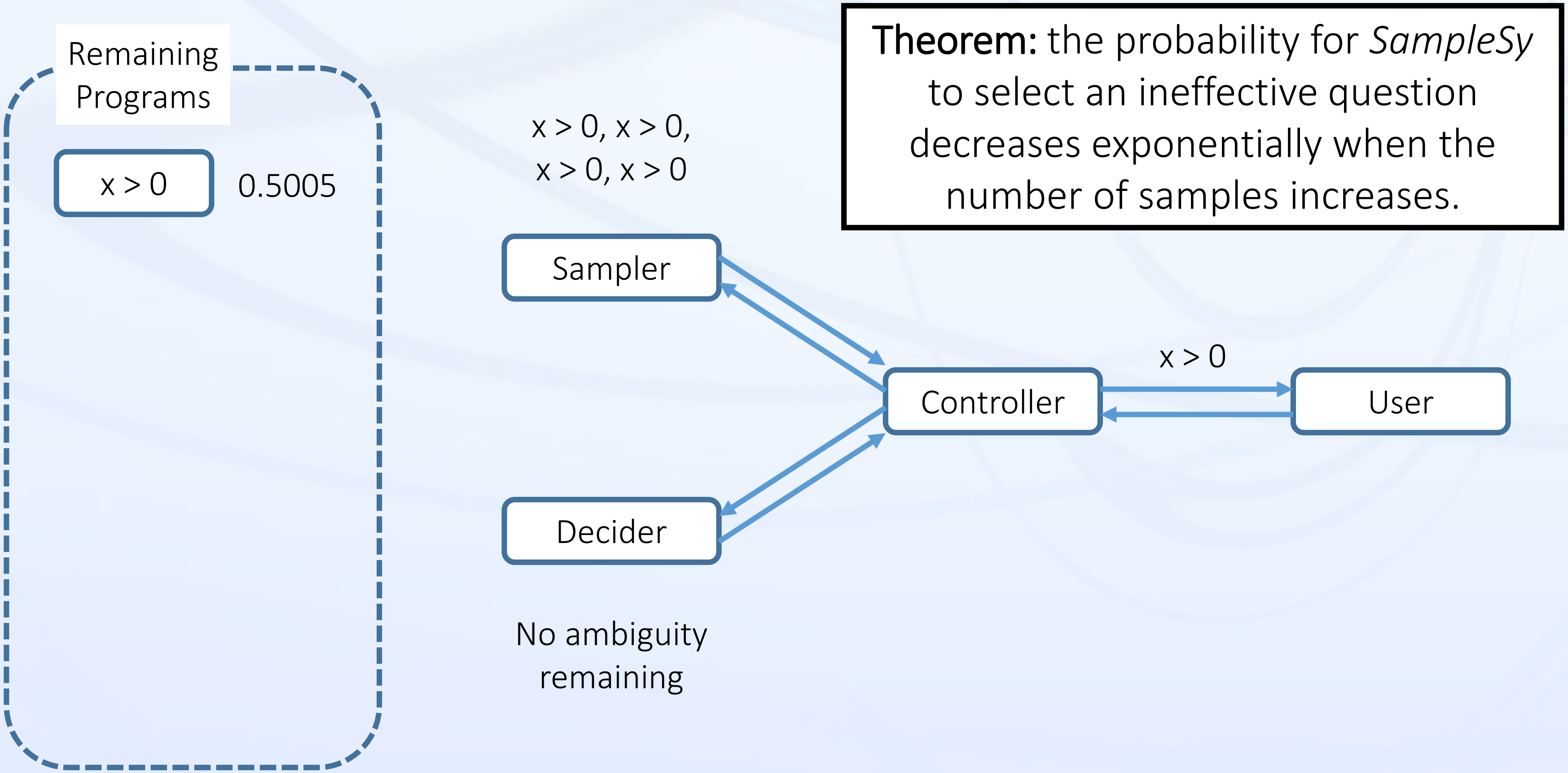
P(1) = ?

P(1) = True

There is ambiguity remaining

Running Example: *SampleSy*

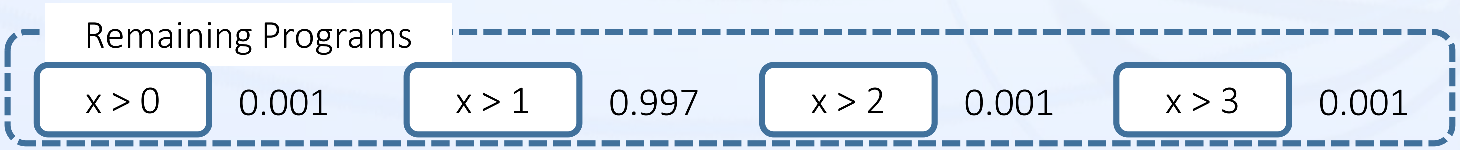
Theorem: the probability for *SampleSy* to select an ineffective question decreases exponentially when the number of samples increases.

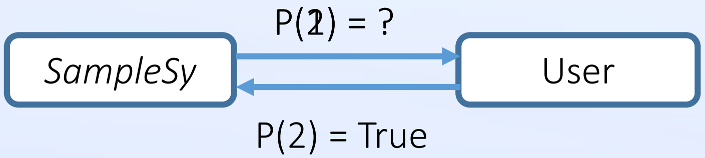


Involving Error Rate

Even minimax branch strategy may use too many questions.

- **Observation.** To ensure the correctness, *SampleSy* sometimes uses many questions to exclude programs that have extremely small probabilities.

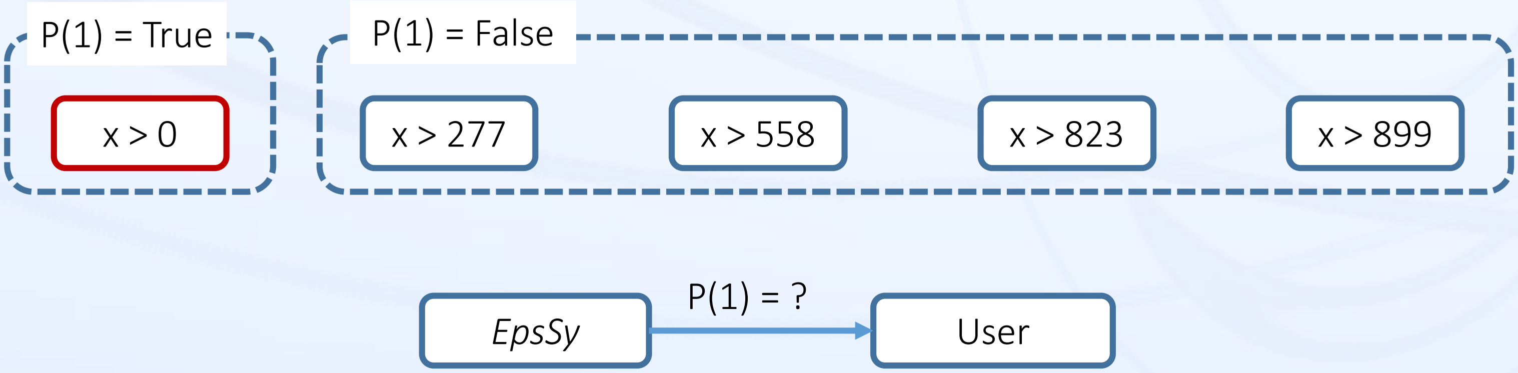




The error rate of directly returning x > 1 is only 0.3%

EpsSy

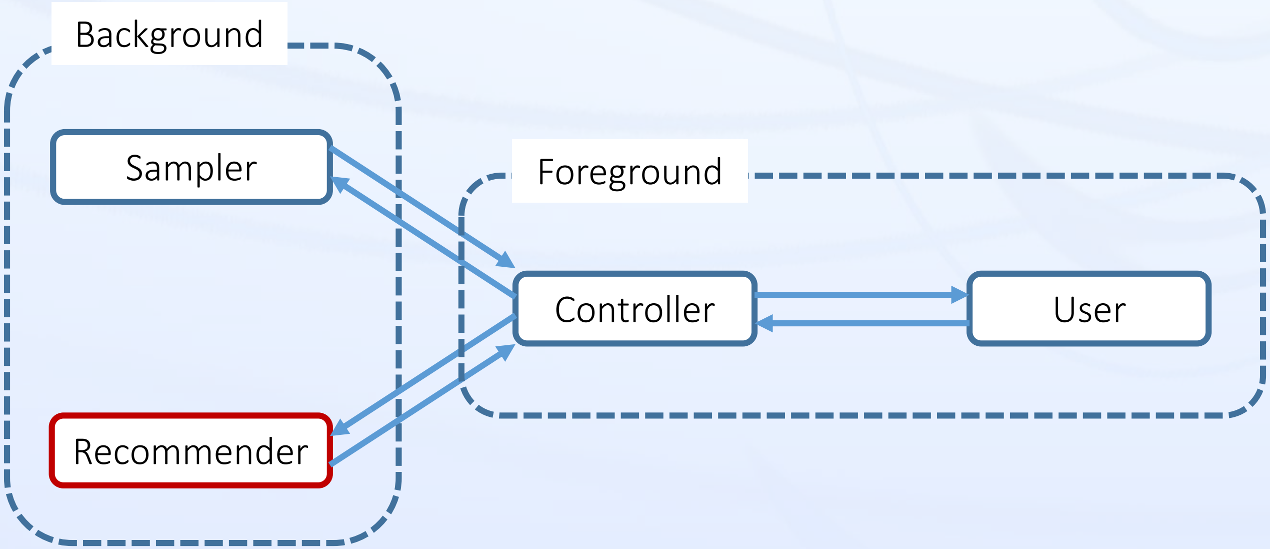
- *EpsSy* may return a wrong program, but its error rate is under control.
- **Key Idea.** Guess a program, and then verify it with questions.



If x > 0 is incorrect, it will be excluded in this round with a high probability. → If x > 0 survives, it will be more likely to be correct.

EpsSy

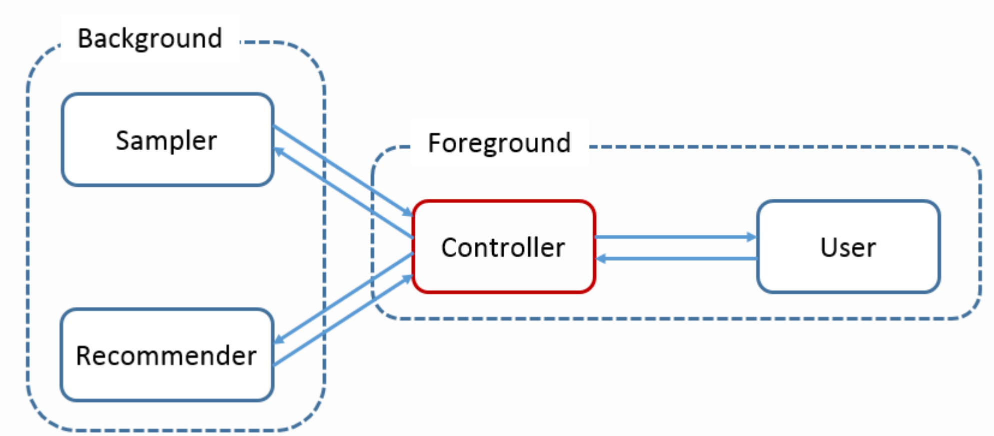
- **Key Idea.** Guess a program, and then verify it with questions.
- **Recommender.** Recommend a program from remaining programs.
  - Existing approaches, e.g., *Euphony*: Lee et al. (PLDI 18)



EpsSy

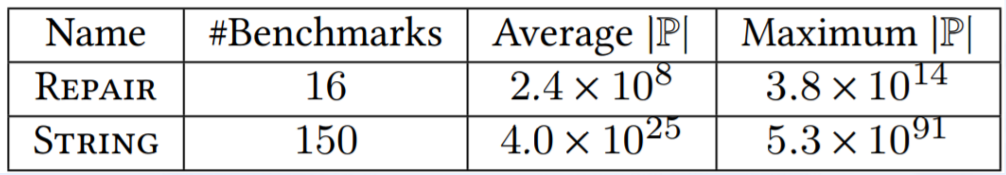
- **Controller.** Selecting a question which is both
  - **Challengeable.** The recommendation should perform differently from most samples.
  - **Effective.** Exclude many samples in the worst case.
- *EpsSy* will return the recommendation once the number of its surviving rounds reaches a threshold.

Theorem: *EpsSy* can reach an arbitrarily small error rate with a logarithmic level threshold.



Evaluation Setup

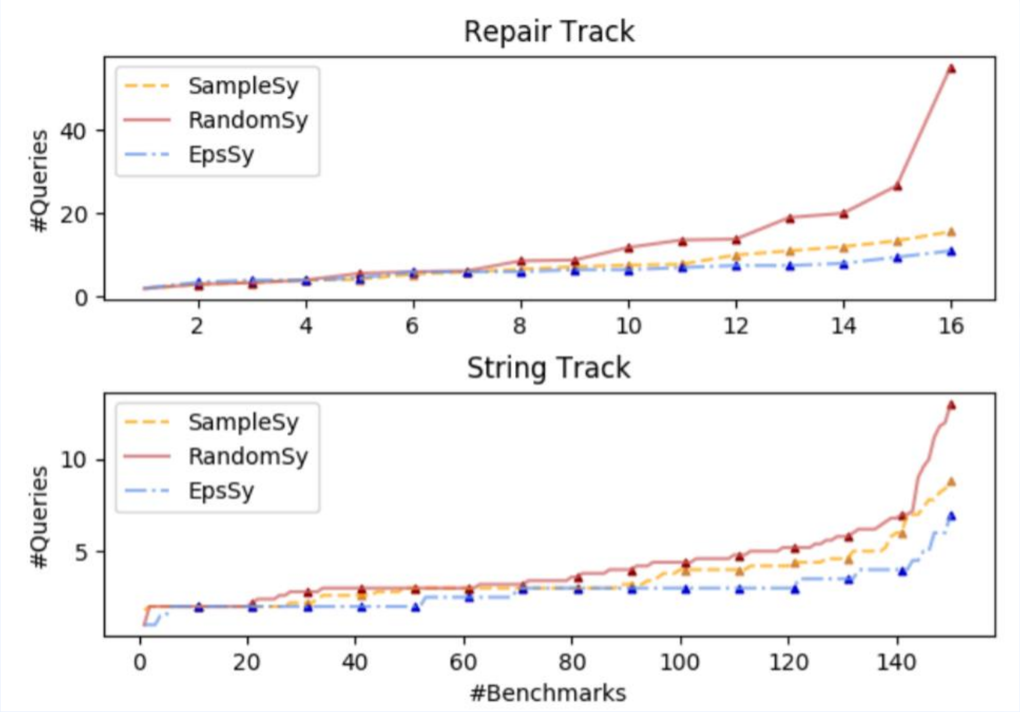
- Benchmarks:
  - □ Repair: constructed from the *Program Repair* track in SyGuS-Comp.
  - □ String: constructed from the string benchmarks collected by Lee et al. (PLDI 18), containing synthesis tasks from SyGuS-Comp and online forums.



Name	#Benchmarks	Average $\|\mathbb{P}\|$	Maximum $\|\mathbb{P}\|$
Repair	16	2.4×10^{8}	3.8×10^{14}
String	150	4.0×10^{25}	5.3×10^{91}

- Baseline:
  - □ *RandomSy*: randomly select a question that can distinguish remaining programs.

Evaluation Result



The overall error rate of *EpsSy* is only 0.60%.