Ph.D. Proposal

Software Side-Channel Analysis

Lucas Bang

Department of Computer Science, University of California, Santa Barbara

UCSB

INTRODUCTION

What is a side channel?

What is a side channel?

TIME

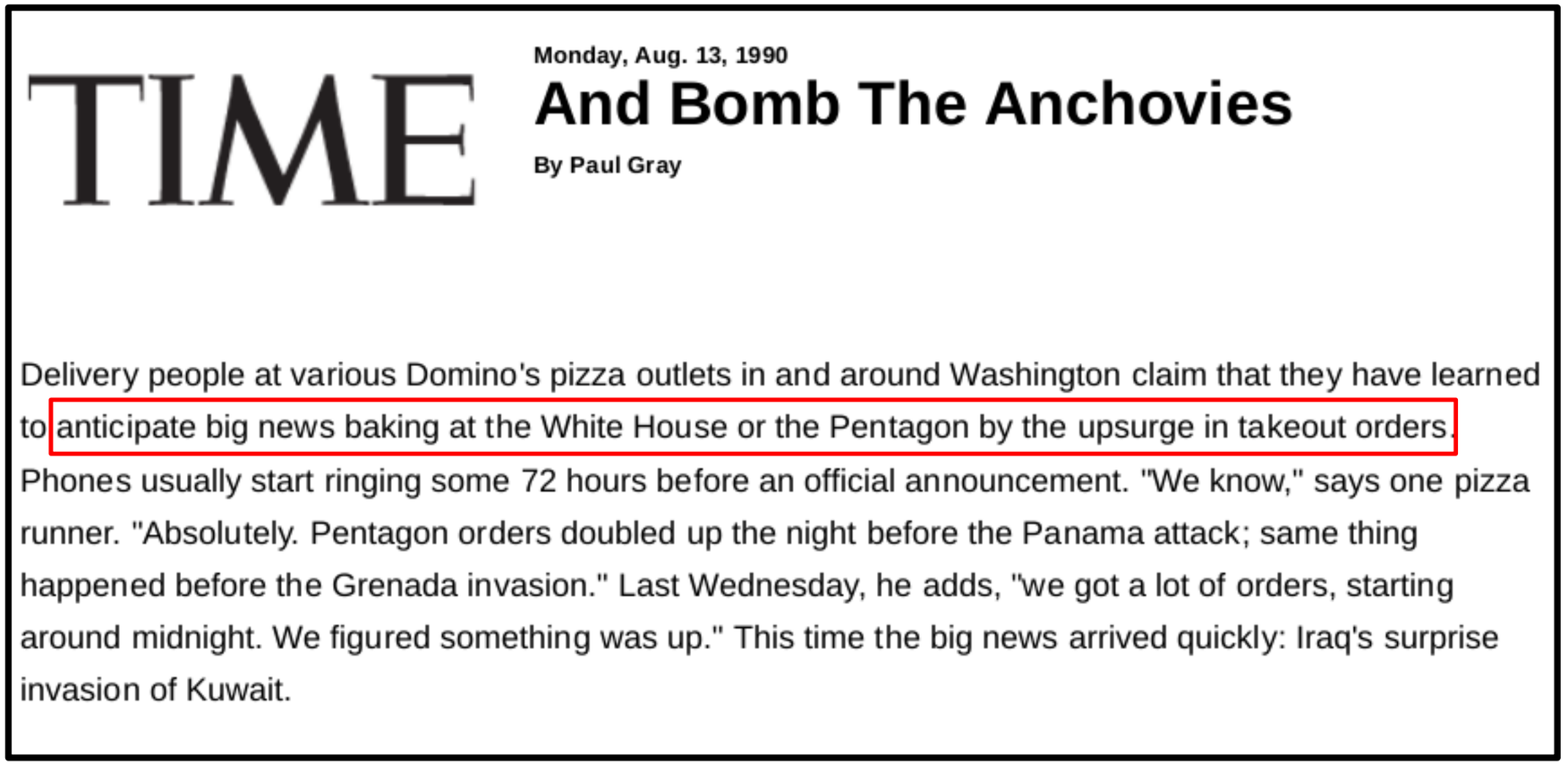
Monday, Aug. 13, 1990

And Bomb The Anchovies

By Paul Gray

Delivery people at various Domino's pizza outlets in and around Washington claim that they have learned to anticipate big news baking at the White House or the Pentagon by the upsurge in takeout orders. Phones usually start ringing some 72 hours before an official announcement. "We know," says one pizza runner. "Absolutely. Pentagon orders doubled up the night before the Panama attack; same thing happened before the Grenada invasion." Last Wednesday, he adds, "we got a lot of orders, starting around midnight. We figured something was up." This time the big news arrived quickly: Iraq's surprise invasion of Kuwait.

What is a side channel?



TIME

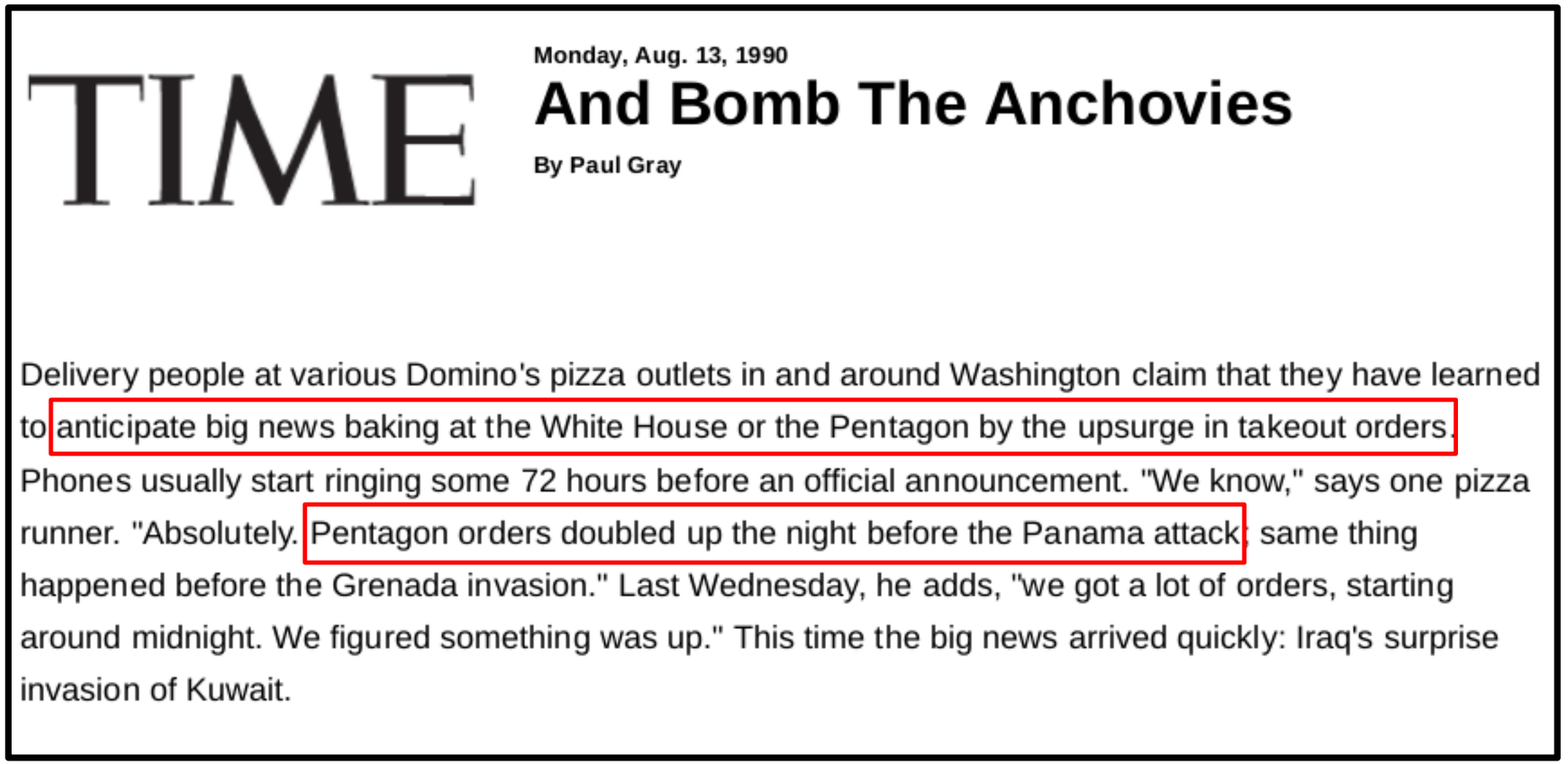
Monday, Aug. 13, 1990

And Bomb The Anchovies

By Paul Gray

Delivery people at various Domino's pizza outlets in and around Washington claim that they have learned to anticipate big news baking at the White House or the Pentagon by the upsurge in takeout orders. Phones usually start ringing some 72 hours before an official announcement. "We know," says one pizza runner. "Absolutely. Pentagon orders doubled up the night before the Panama attack; same thing happened before the Grenada invasion." Last Wednesday, he adds, "we got a lot of orders, starting around midnight. We figured something was up." This time the big news arrived quickly: Iraq's surprise invasion of Kuwait.

What is a side channel?



TIME

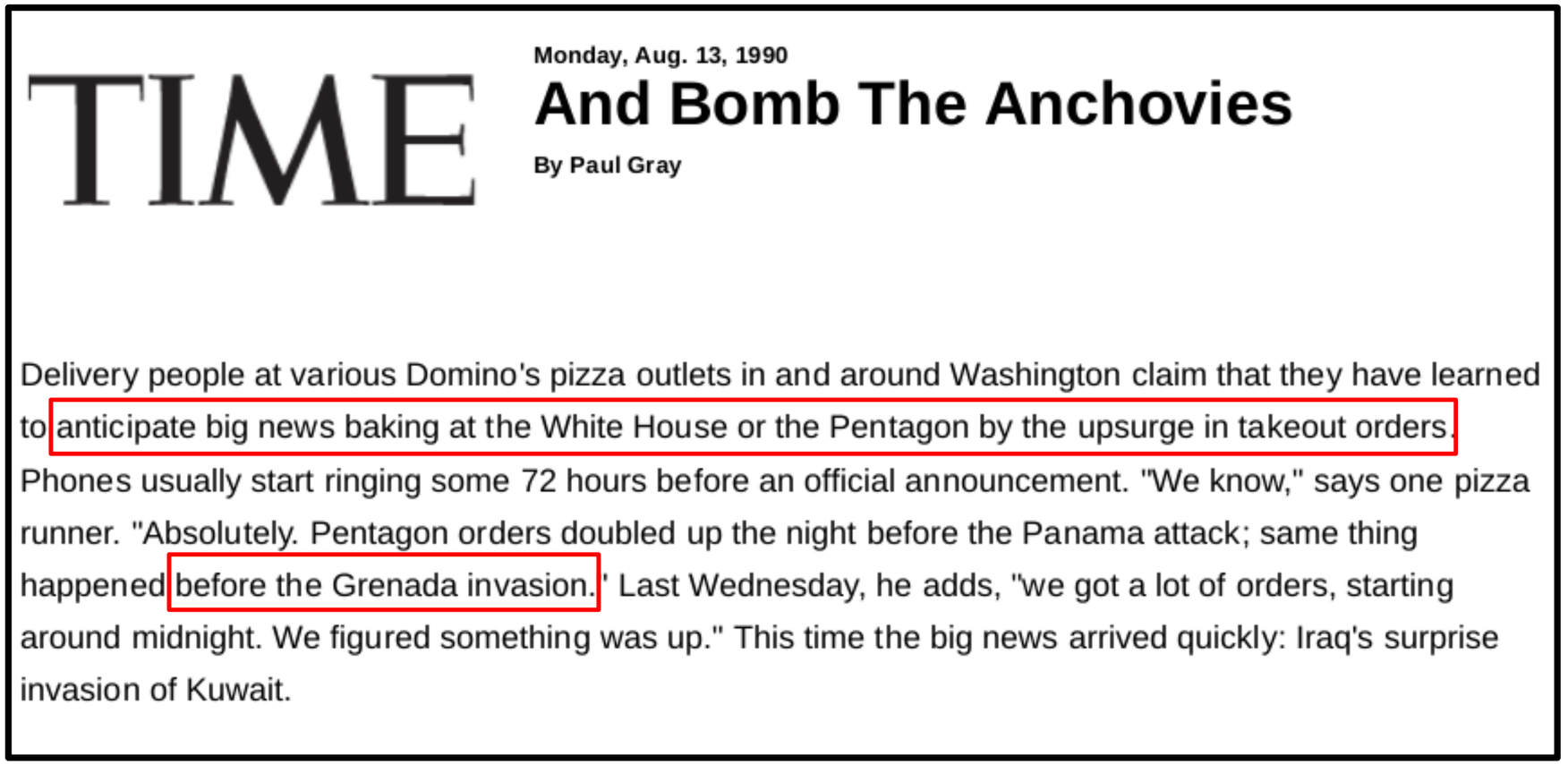
Monday, Aug. 13, 1990

And Bomb The Anchovies

By Paul Gray

Delivery people at various Domino's pizza outlets in and around Washington claim that they have learned to anticipate big news baking at the White House or the Pentagon by the upsurge in takeout orders. Phones usually start ringing some 72 hours before an official announcement. "We know," says one pizza runner. "Absolutely. Pentagon orders doubled up the night before the Panama attack; same thing happened before the Grenada invasion." Last Wednesday, he adds, "we got a lot of orders, starting around midnight. We figured something was up." This time the big news arrived quickly: Iraq's surprise invasion of Kuwait.

What is a side channel?



TIME

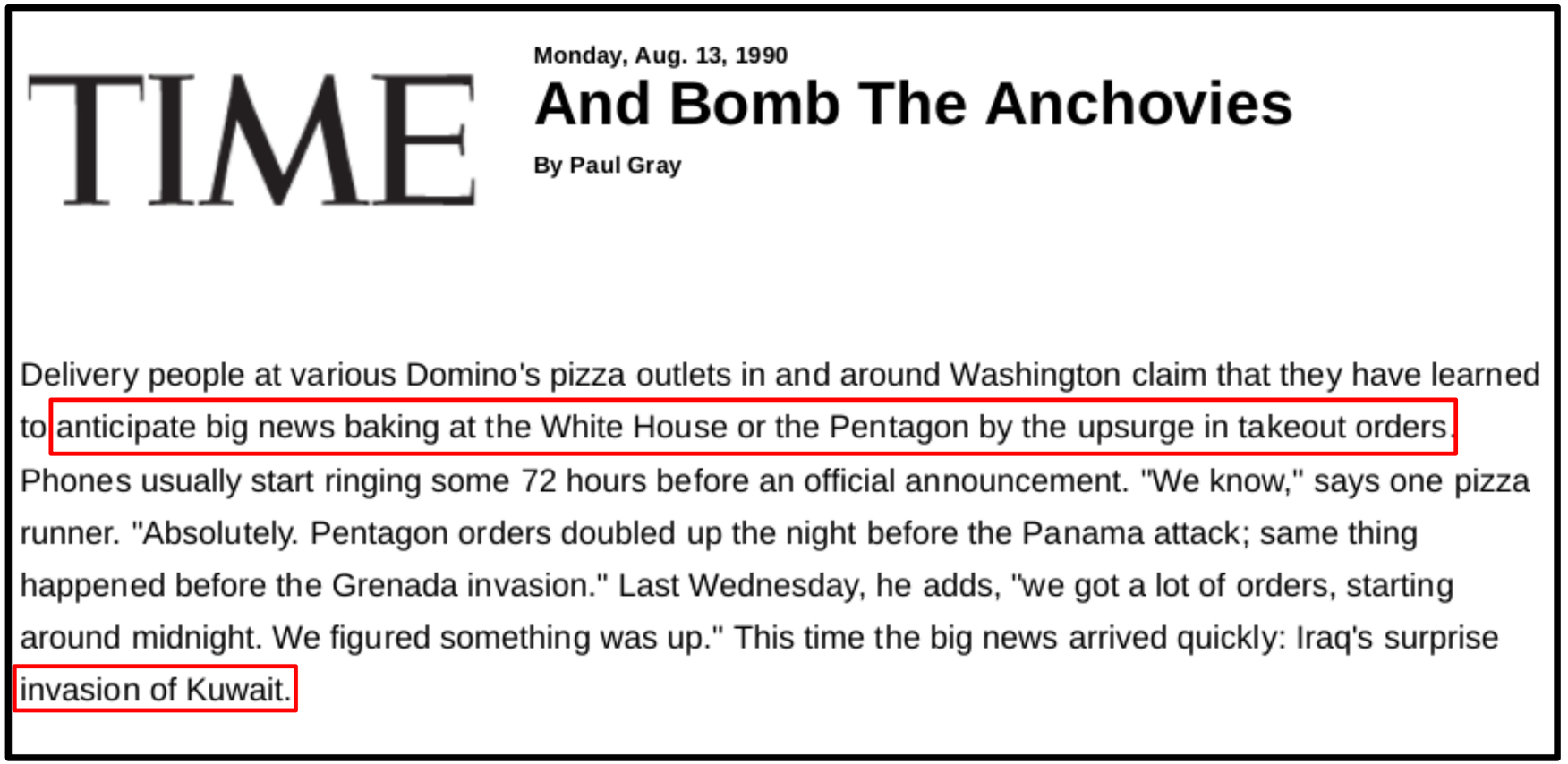
Monday, Aug. 13, 1990

And Bomb The Anchovies

By Paul Gray

Delivery people at various Domino's pizza outlets in and around Washington claim that they have learned to anticipate big news baking at the White House or the Pentagon by the upsurge in takeout orders. Phones usually start ringing some 72 hours before an official announcement. "We know," says one pizza runner. "Absolutely. Pentagon orders doubled up the night before the Panama attack; same thing happened before the Grenada invasion." Last Wednesday, he adds, "we got a lot of orders, starting around midnight. We figured something was up." This time the big news arrived quickly: Iraq's surprise invasion of Kuwait.

What is a side channel?



TIME

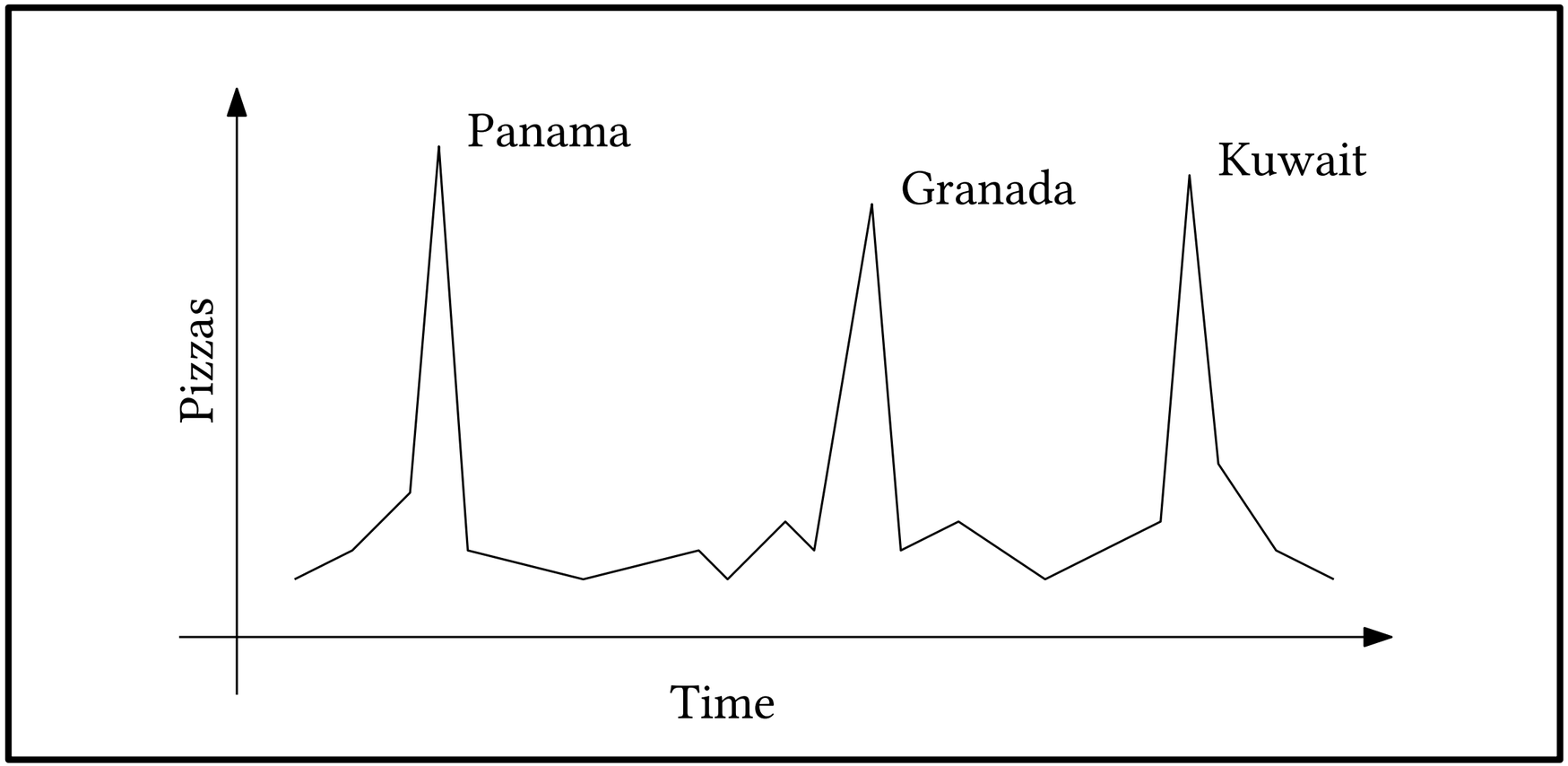
Monday, Aug. 13, 1990

And Bomb The Anchovies

By Paul Gray

Delivery people at various Domino's pizza outlets in and around Washington claim that they have learned to anticipate big news baking at the White House or the Pentagon by the upsurge in takeout orders. Phones usually start ringing some 72 hours before an official announcement. "We know," says one pizza runner. "Absolutely. Pentagon orders doubled up the night before the Panama attack; same thing happened before the Grenada invasion." Last Wednesday, he adds, "we got a lot of orders, starting around midnight. We figured something was up." This time the big news arrived quickly: Iraq's surprise invasion of Kuwait.

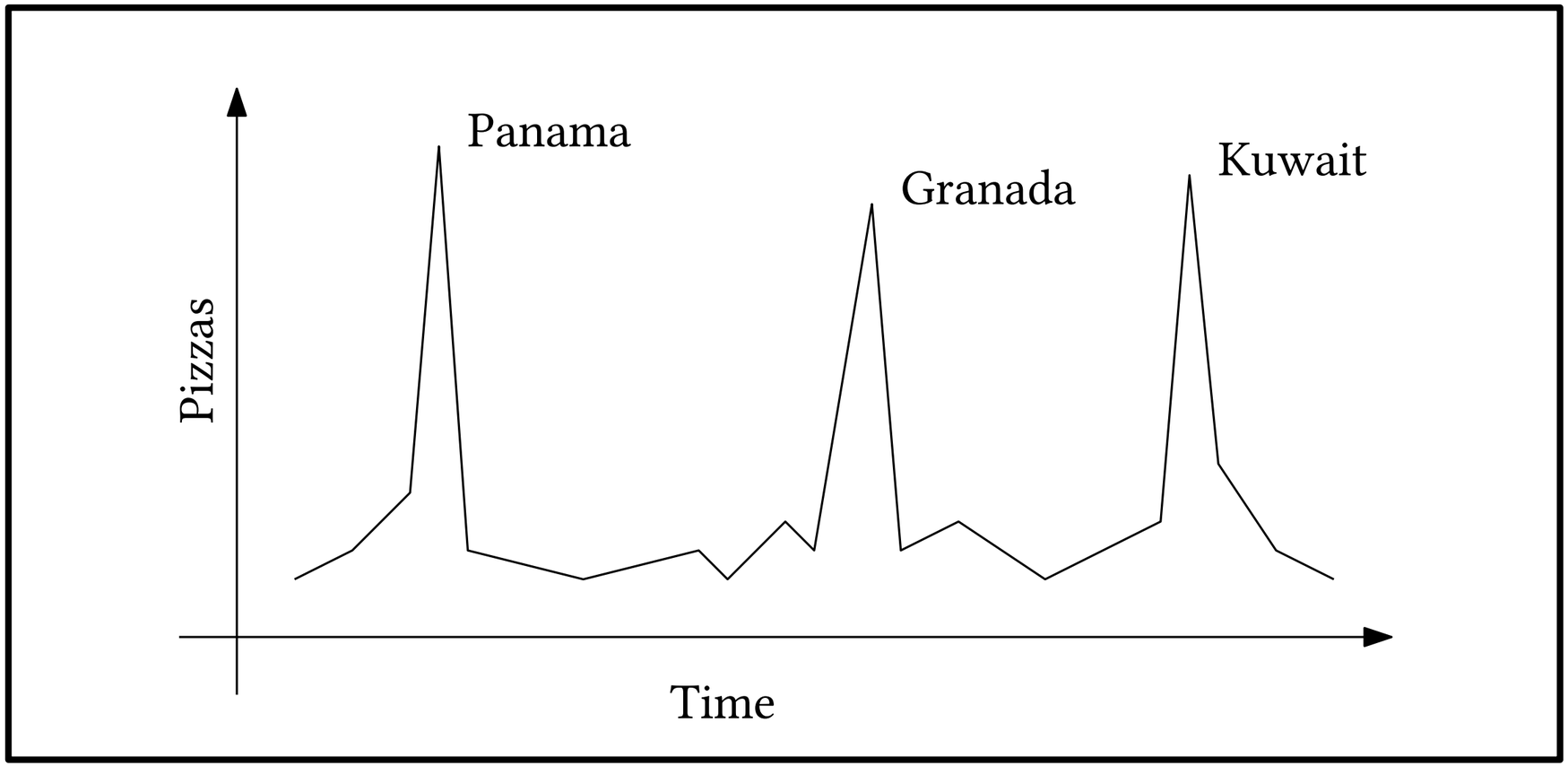
What is a side channel?



Side channel: learn secrets through indirect observation.

secret information correlates with observation ⟹ reveal secrets

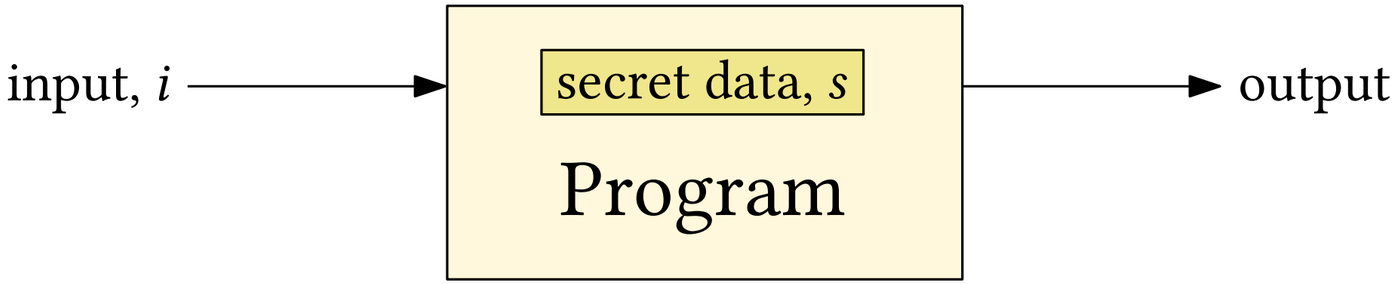
What is a side channel?



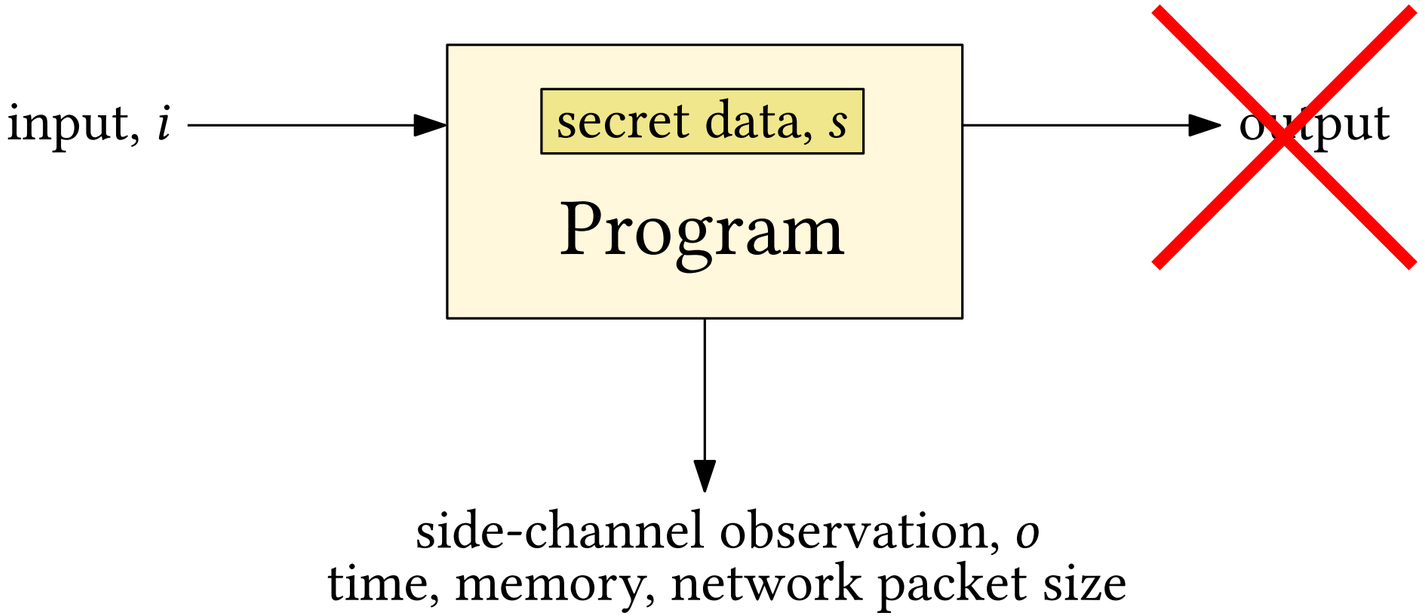
Side channel: learn secrets through indirect observation.

secret information correlates with observation ⟹ reveal secrets

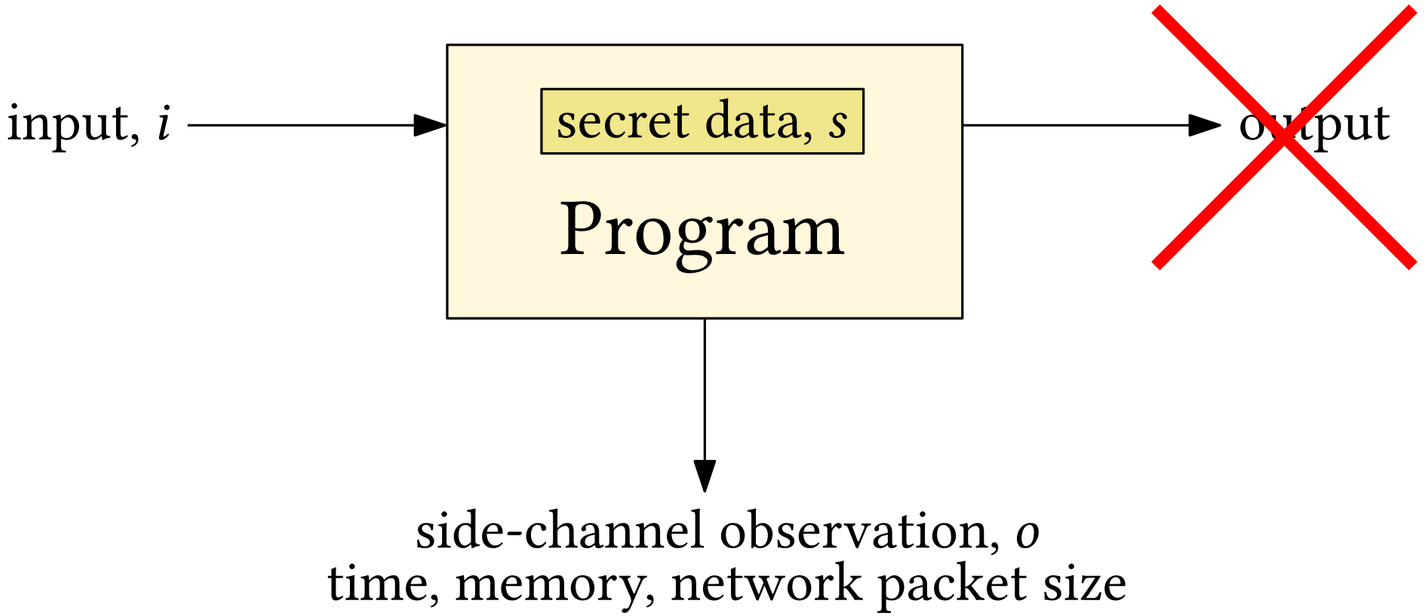
What is a software side channel?



What is a software side channel?



What is a software side channel?

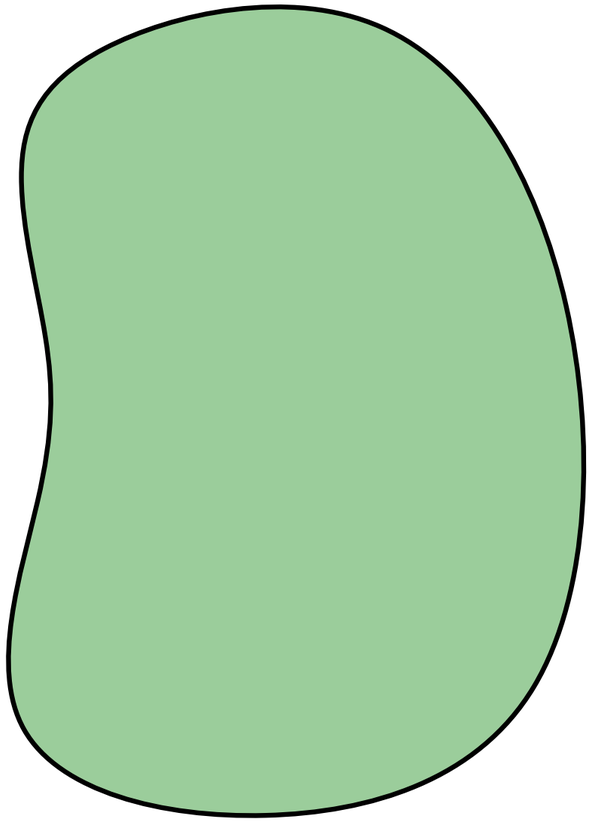


correlation between (i, o) and $s \Rightarrow$ vulnerability

Side Channels and Searching

$i_0 \in I$

secret $s \in S$

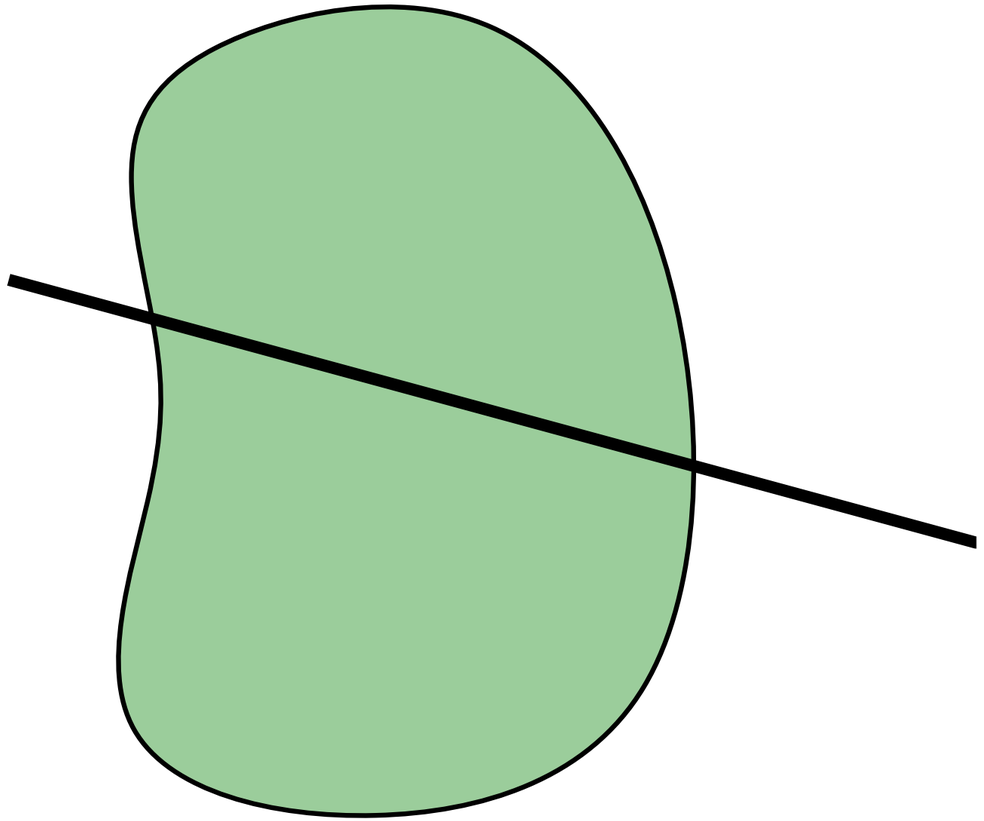


Side Channels and Searching

$i_0 \in I$

$P(i_0, s)$

secret $s \in S$

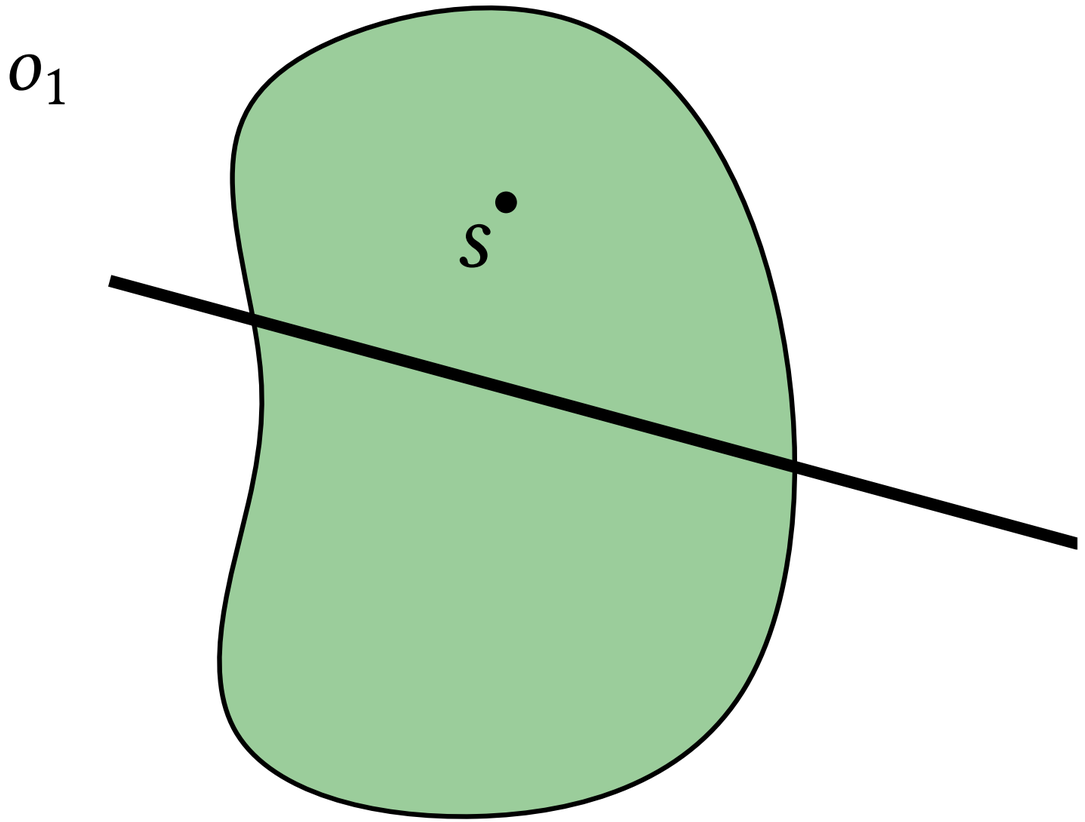


Side Channels and Searching

$i_0 \in I$

$P(i_0, s)$

secret $s \in S$

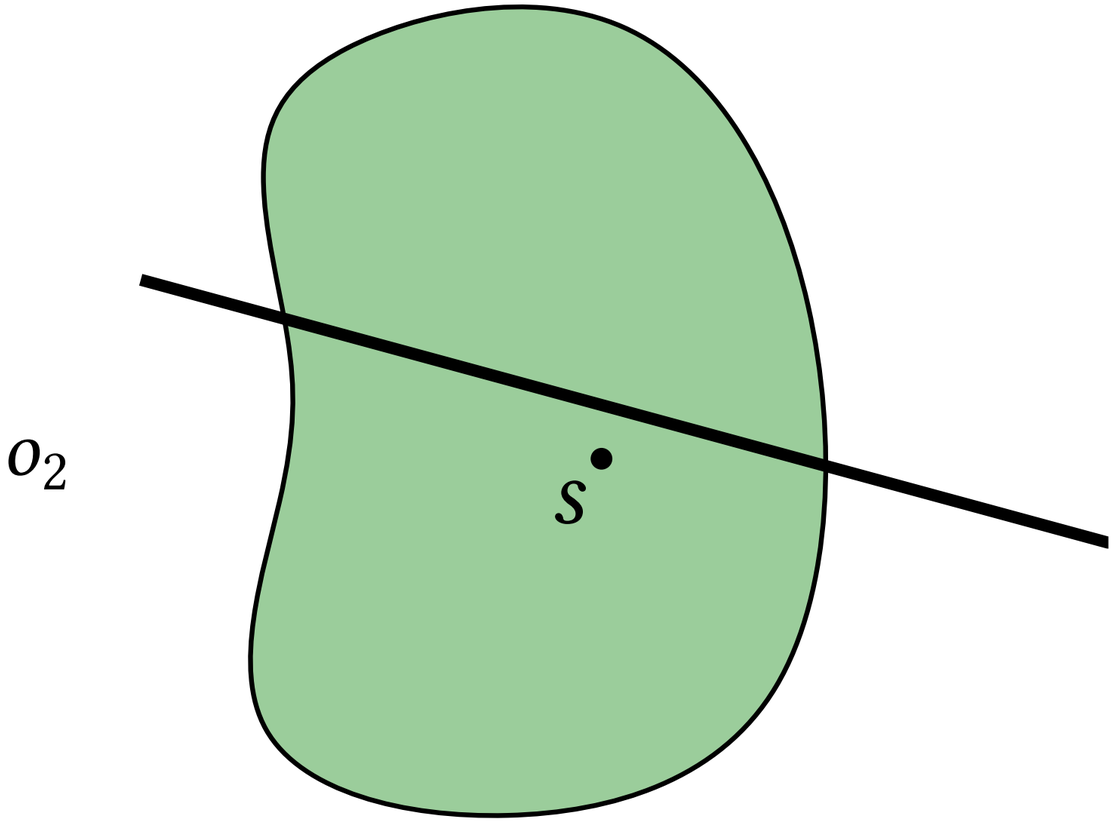


Side Channels and Searching

$i_0 \in I$

$P(i_0, s)$

secret $s \in S$

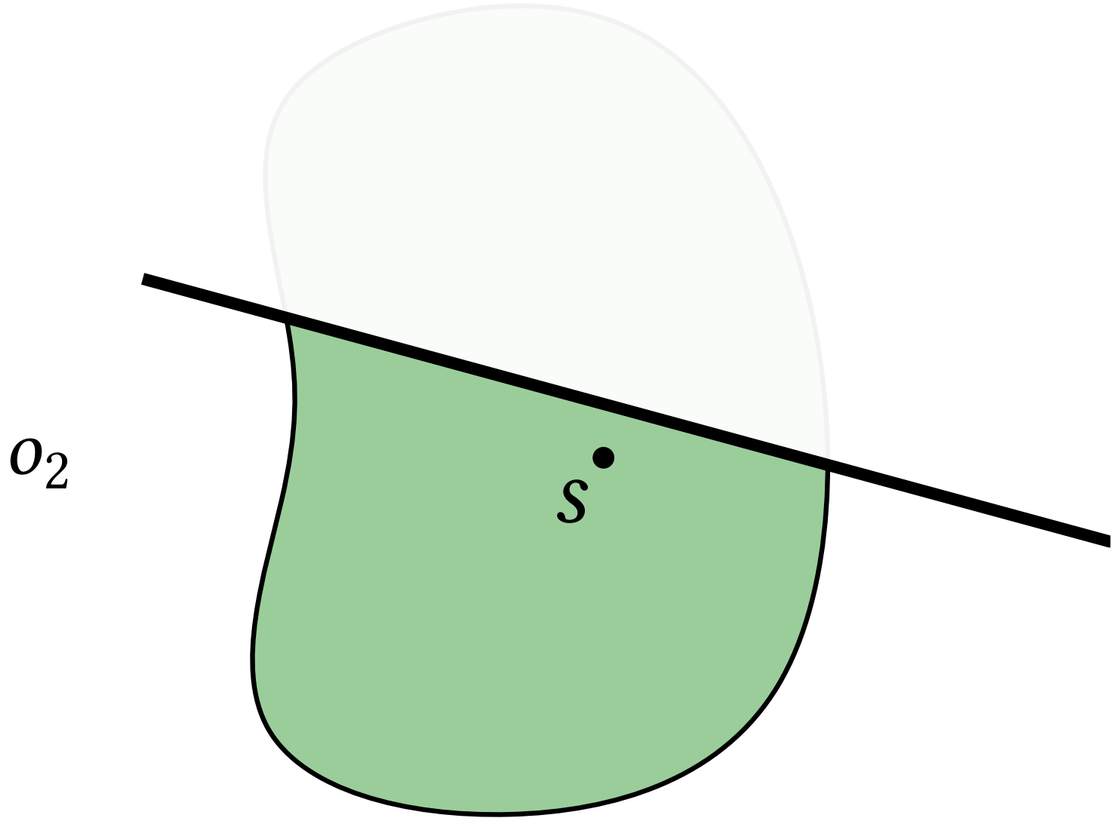


Side Channels and Searching

$i_0 \in I$

$P(i_0, s)$

secret $s \in S$

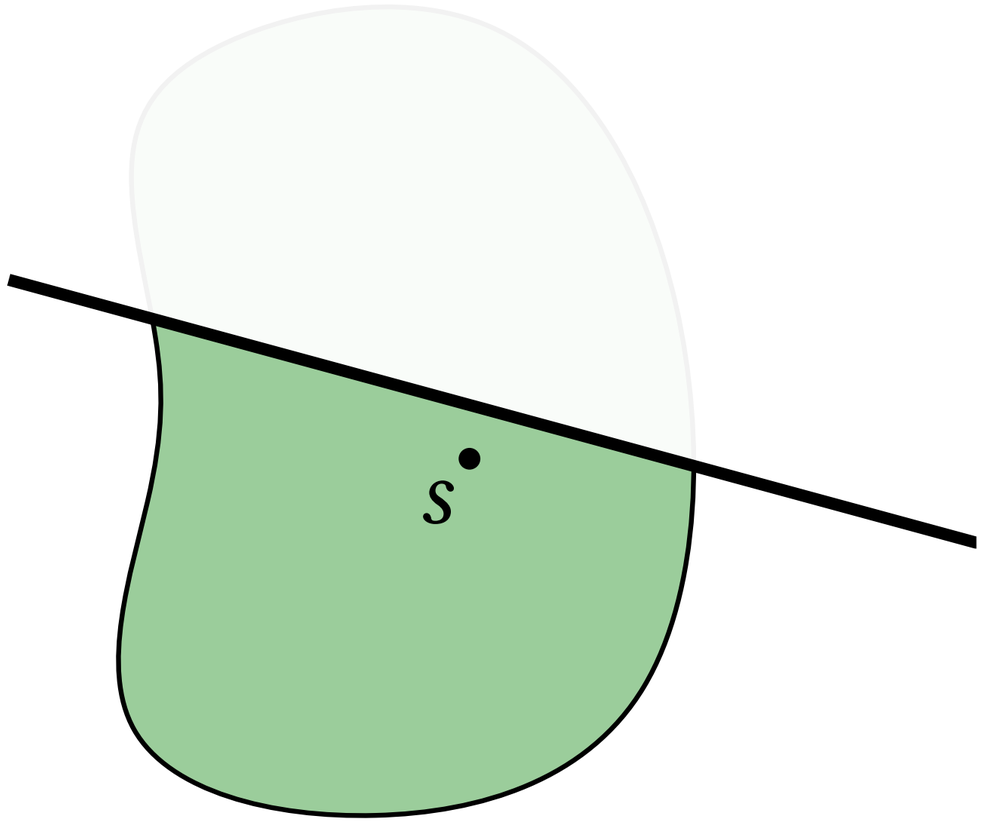


Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

secret $s \in S$

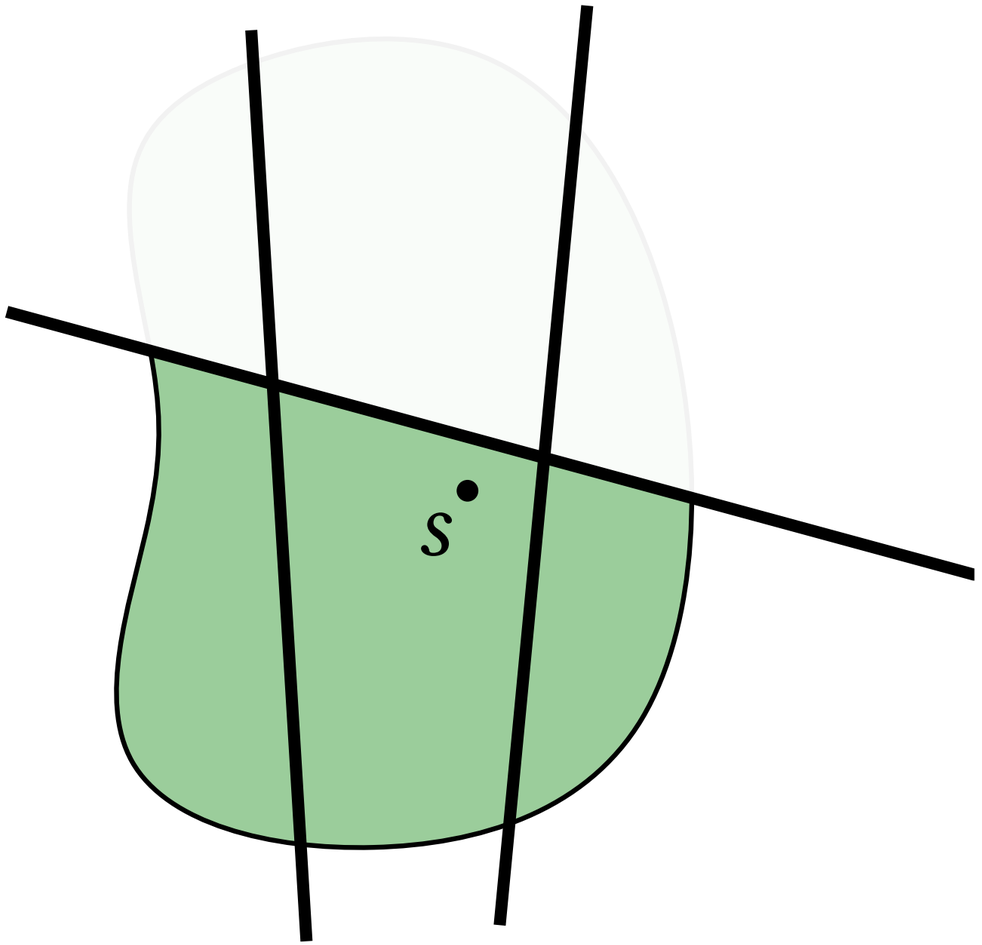


Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

secret $s \in S$

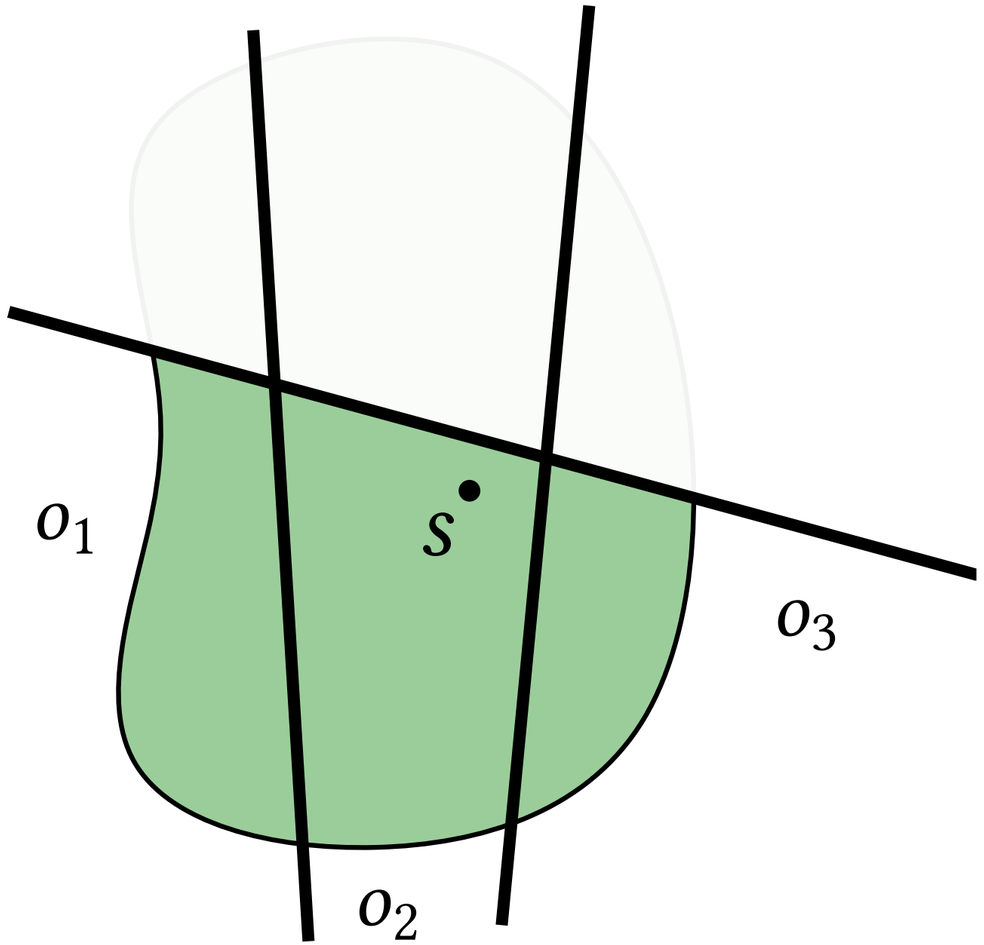


Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

secret $s \in S$

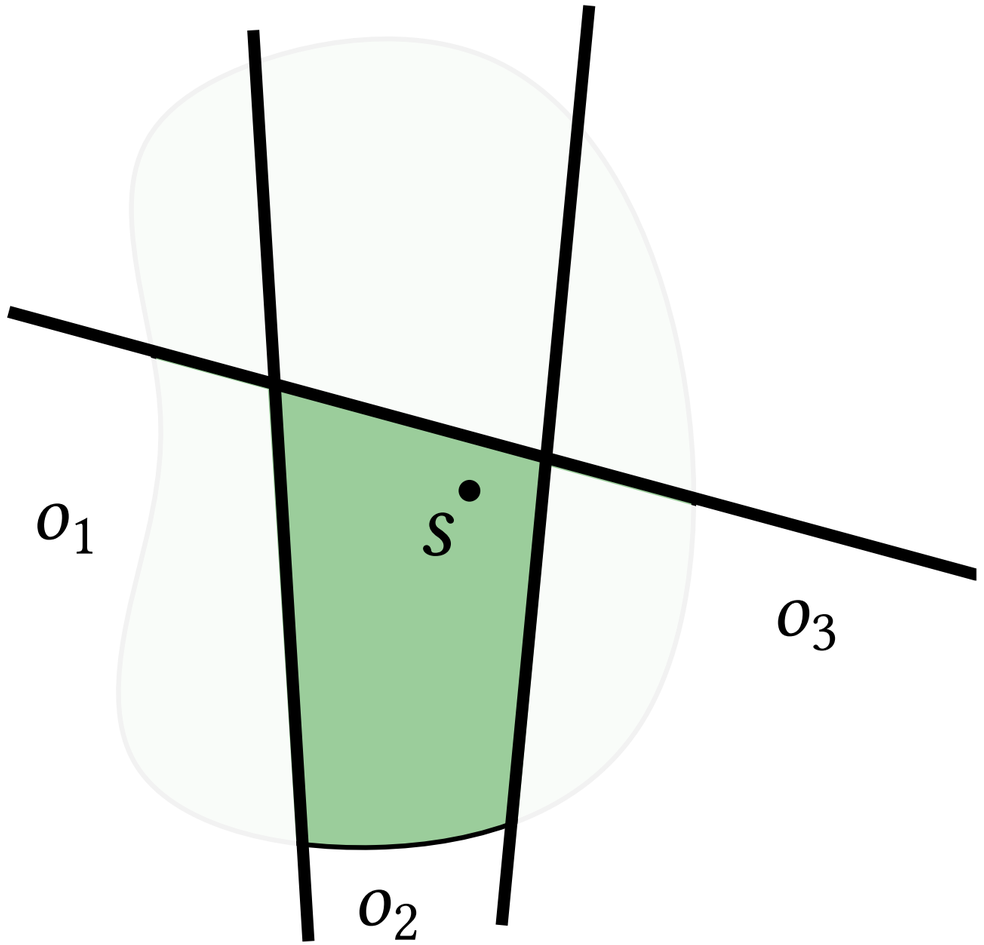


Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

secret $s \in S$

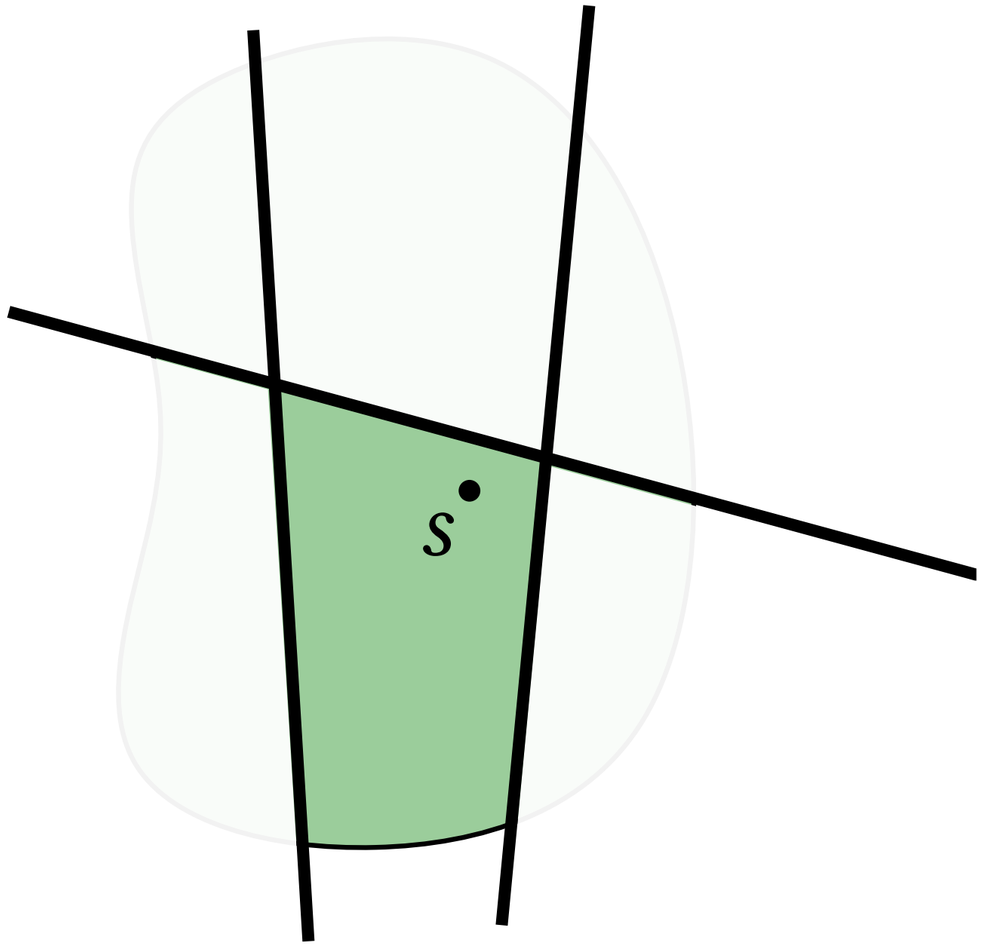


Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

secret $s \in S$



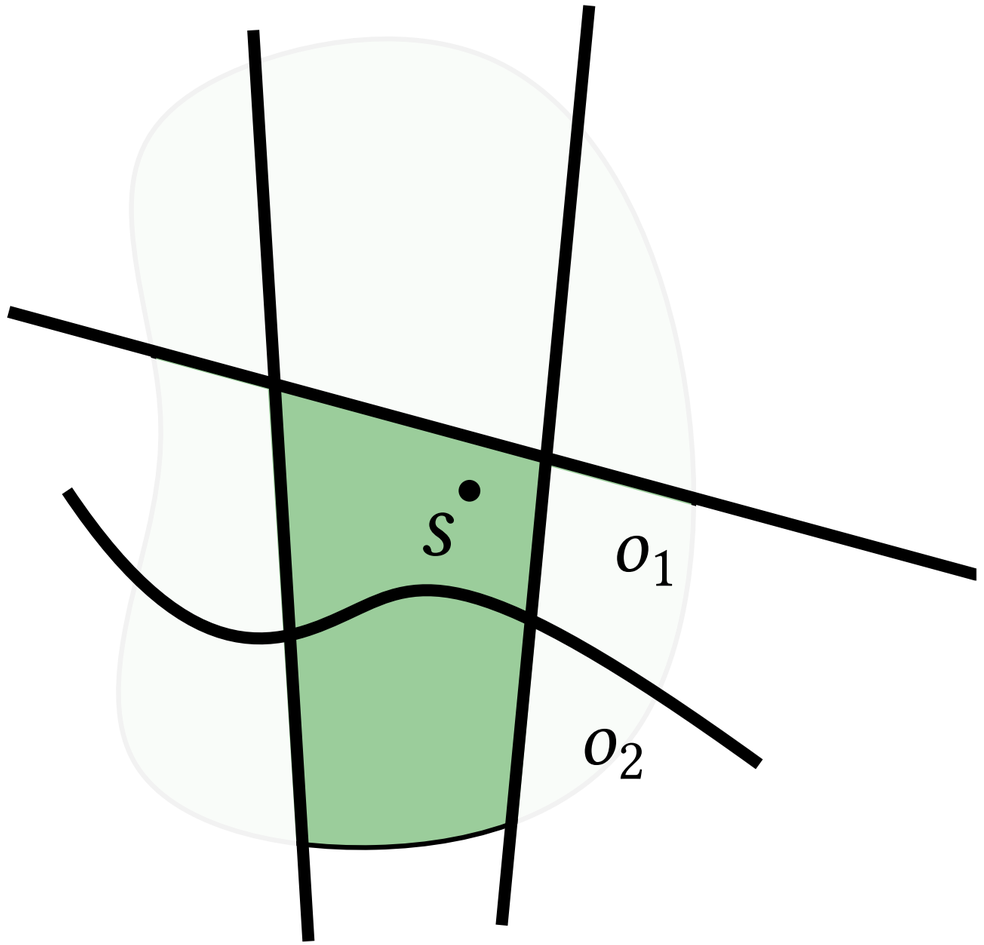
Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

$i_2 \in I$
$P(i_2, s)$

secret $s \in S$



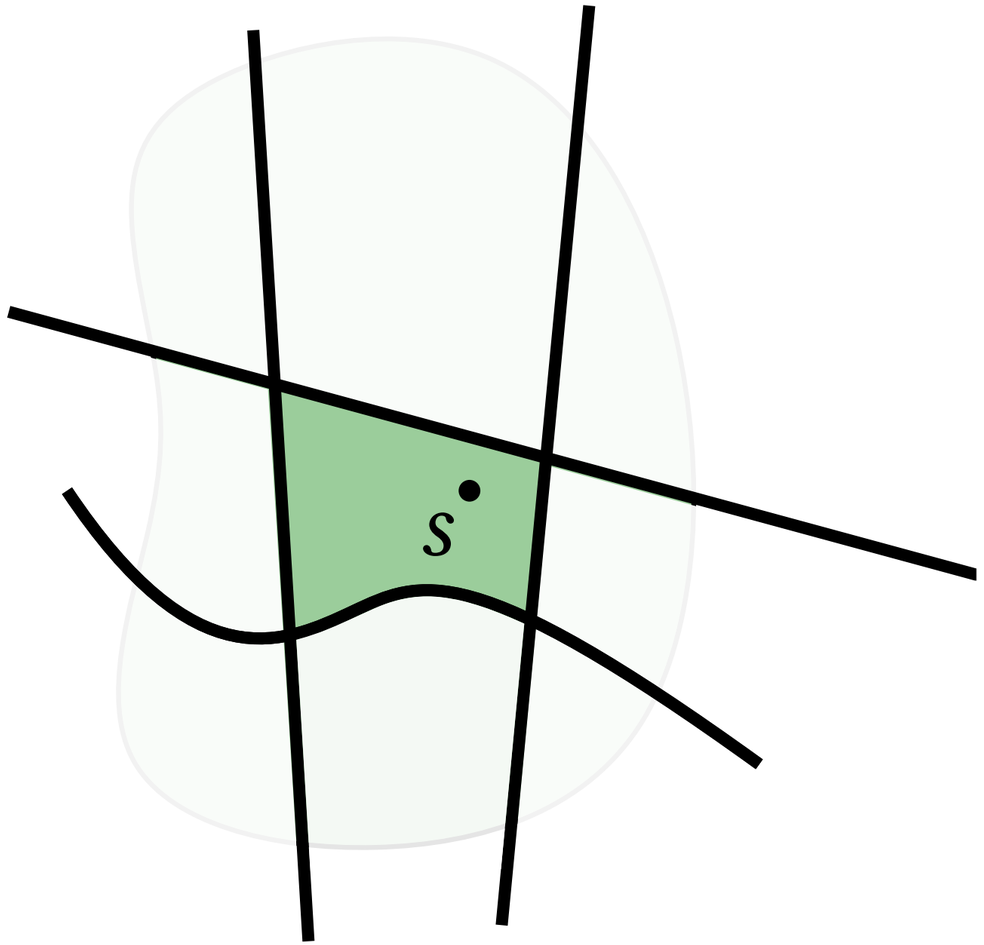
Side Channels and Searching

$i_0 \in I$
$P(i_0, s)$

$i_1 \in I$
$P(i_1, s)$

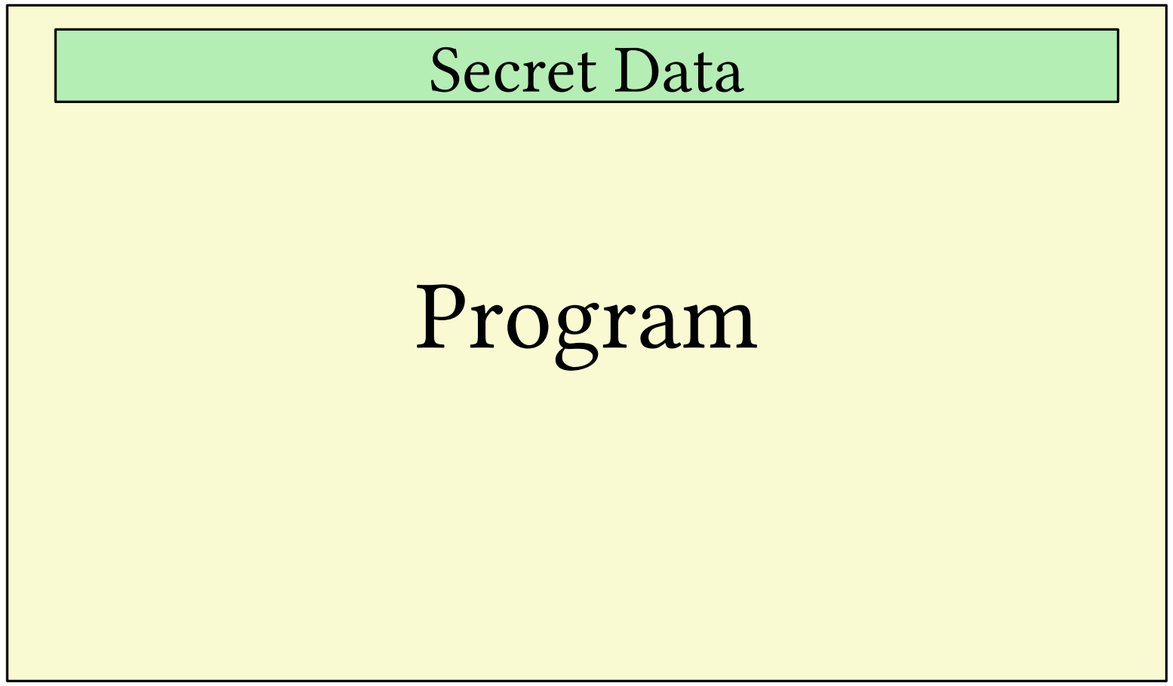
$i_2 \in I$
$P(i_2, s)$

secret $s \in S$

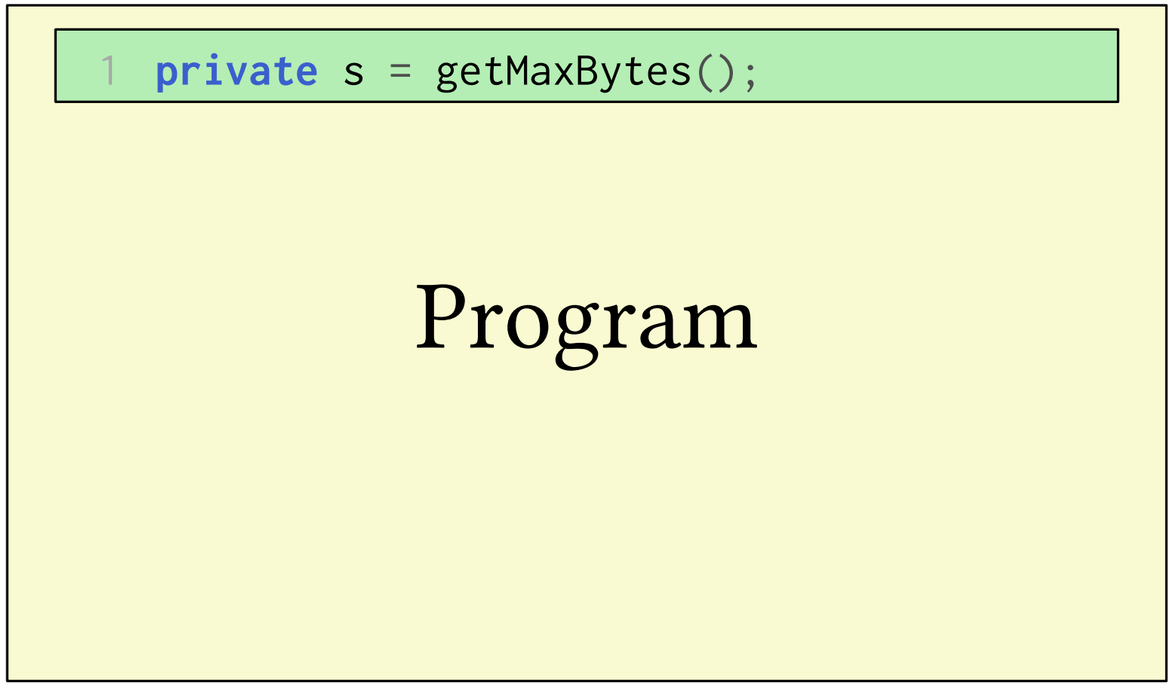


What is a software side channel?

What is a software side channel?



What is a software side channel?



What is a software side channel?

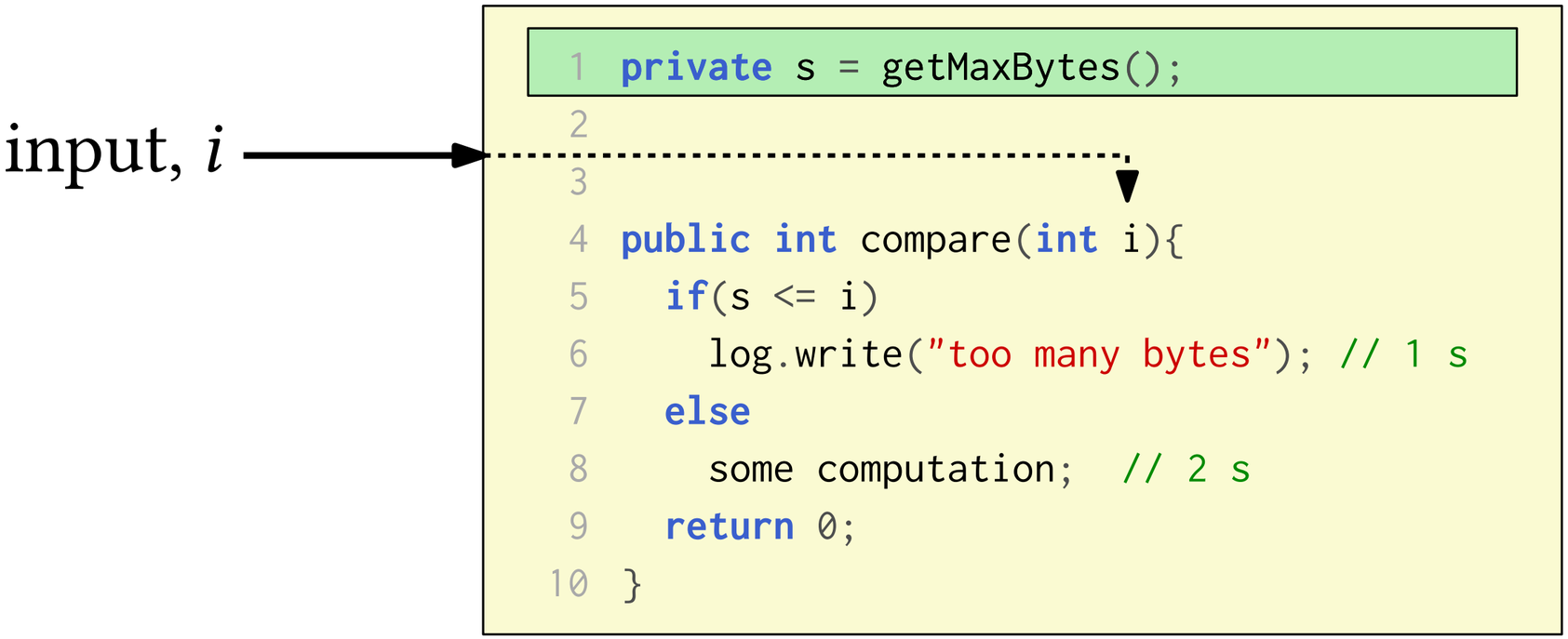
```
1  private s = getMaxBytes();
2
3
4  public int compare(int i){
5    if(s <= i)
6      log.write("too many bytes"); // 1 s
7    else
8      some computation;  // 2 s
9    return 0;
10 }
```

What is a software side channel?

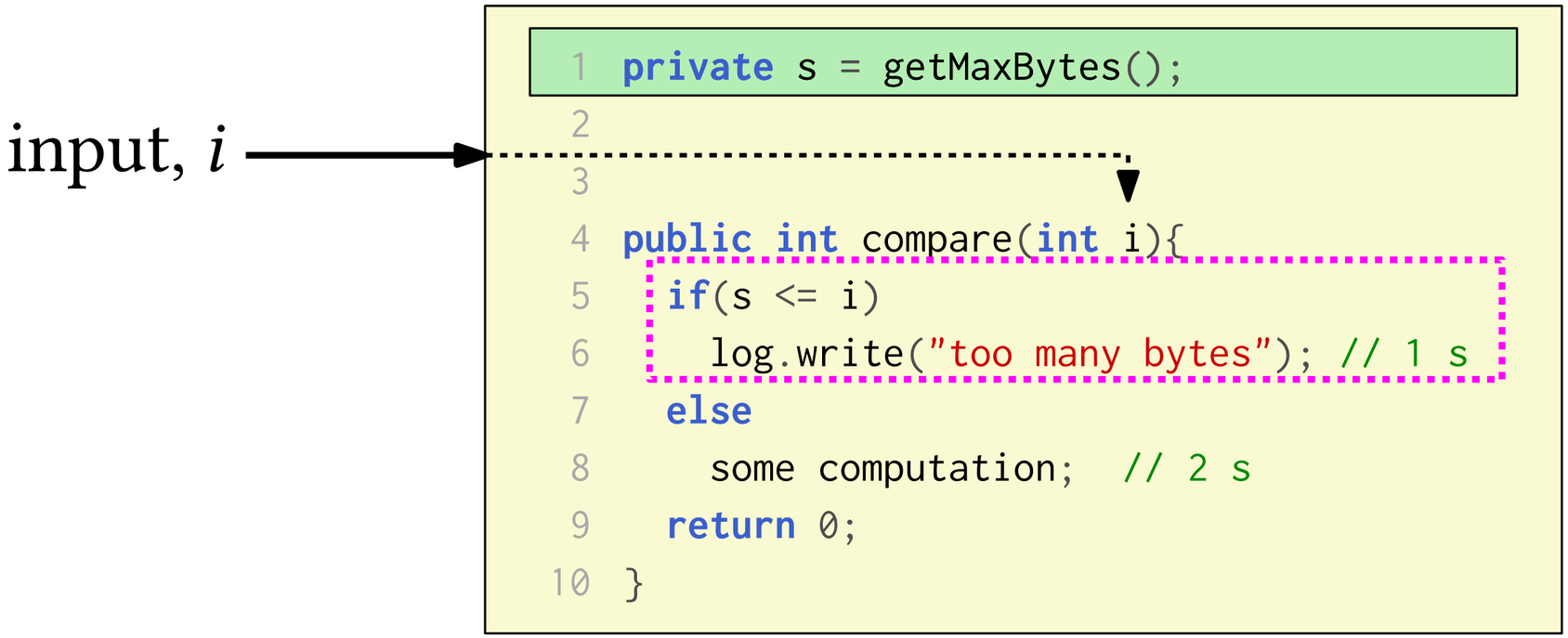
input, i →

```
1  private s = getMaxBytes();
2
3
4  public int compare(int i){
5    if(s <= i)
6      log.write("too many bytes"); // 1 s
7    else
8      some computation;  // 2 s
9    return 0;
10 }
```

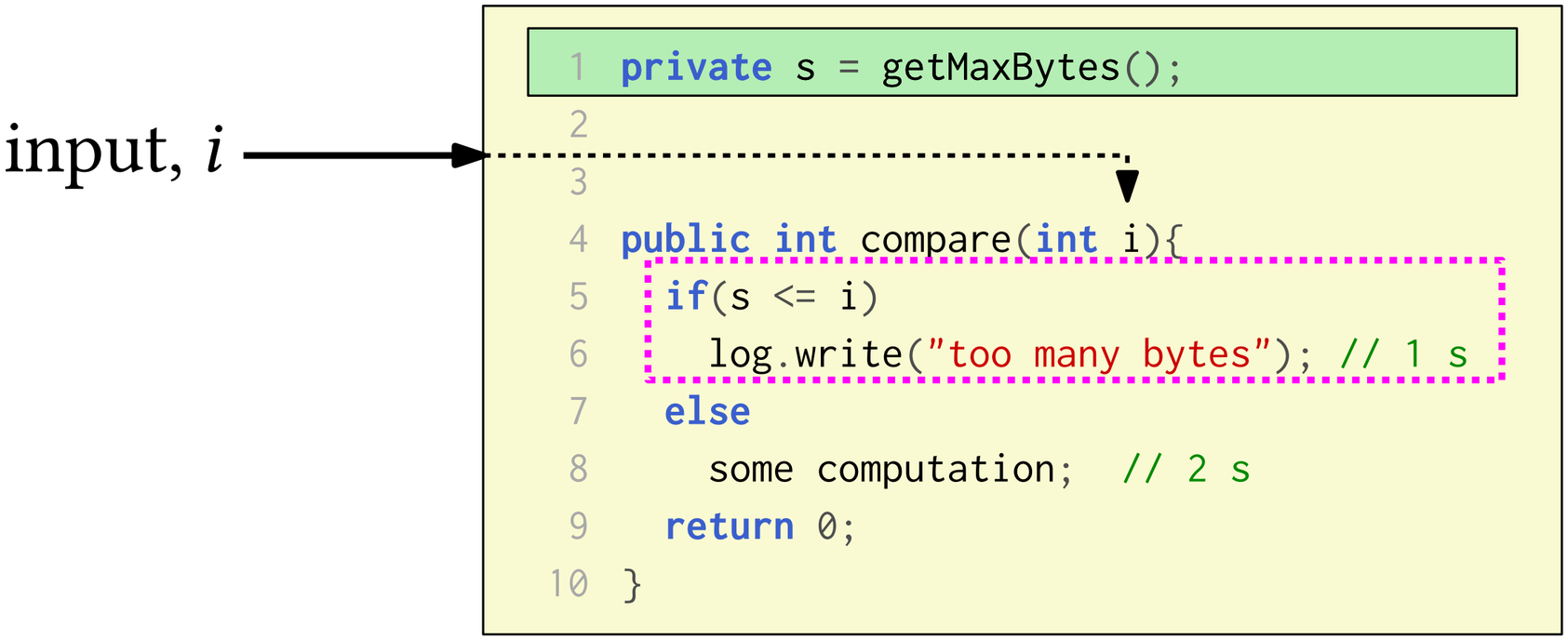
What is a software side channel?



What is a software side channel?

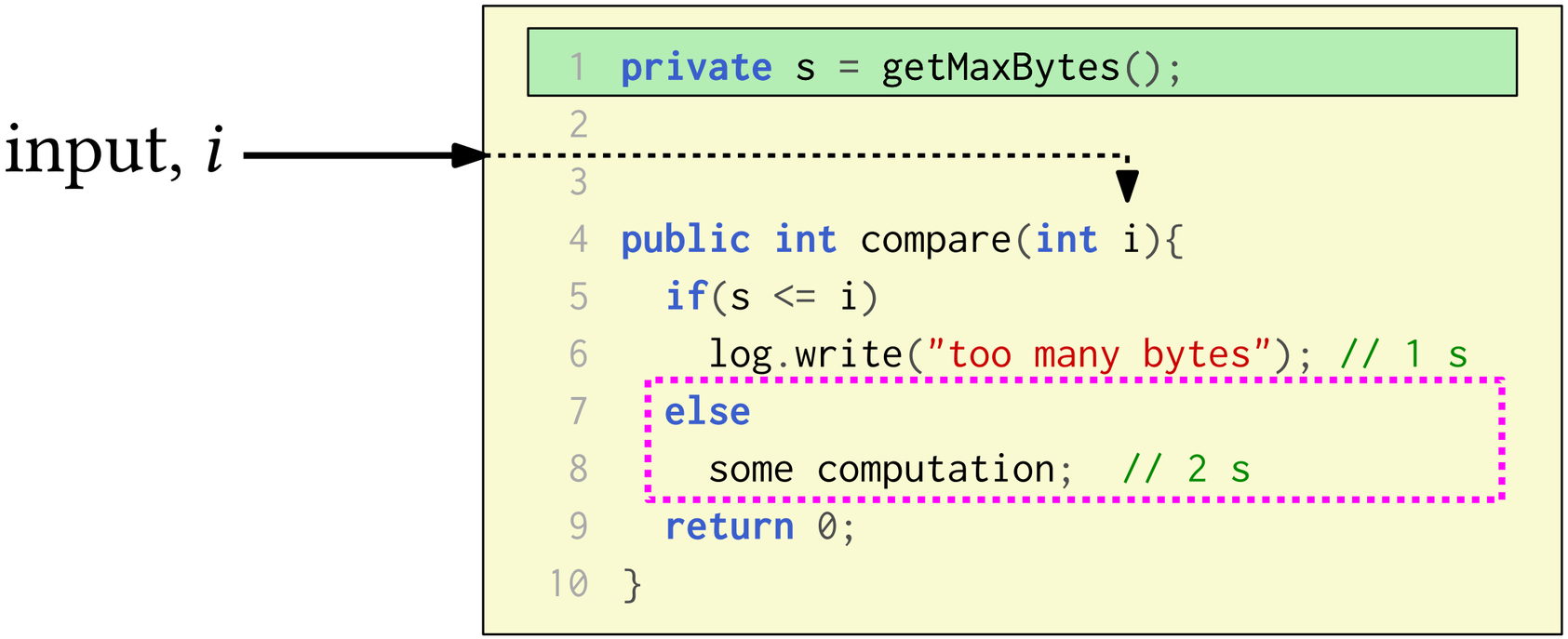


What is a software side channel?



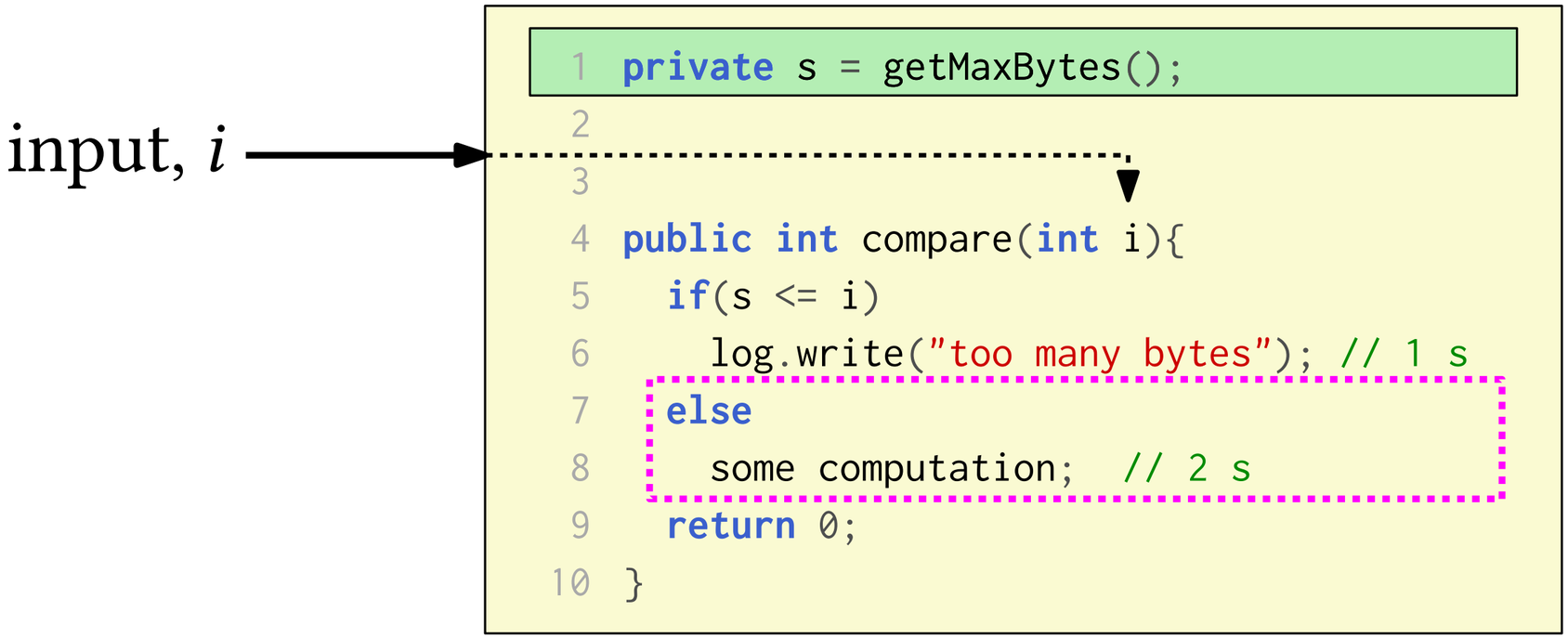
$$s \leq i \implies o = 1$$

What is a software side channel?



$$s \leq i \implies o = 1$$

What is a software side channel?

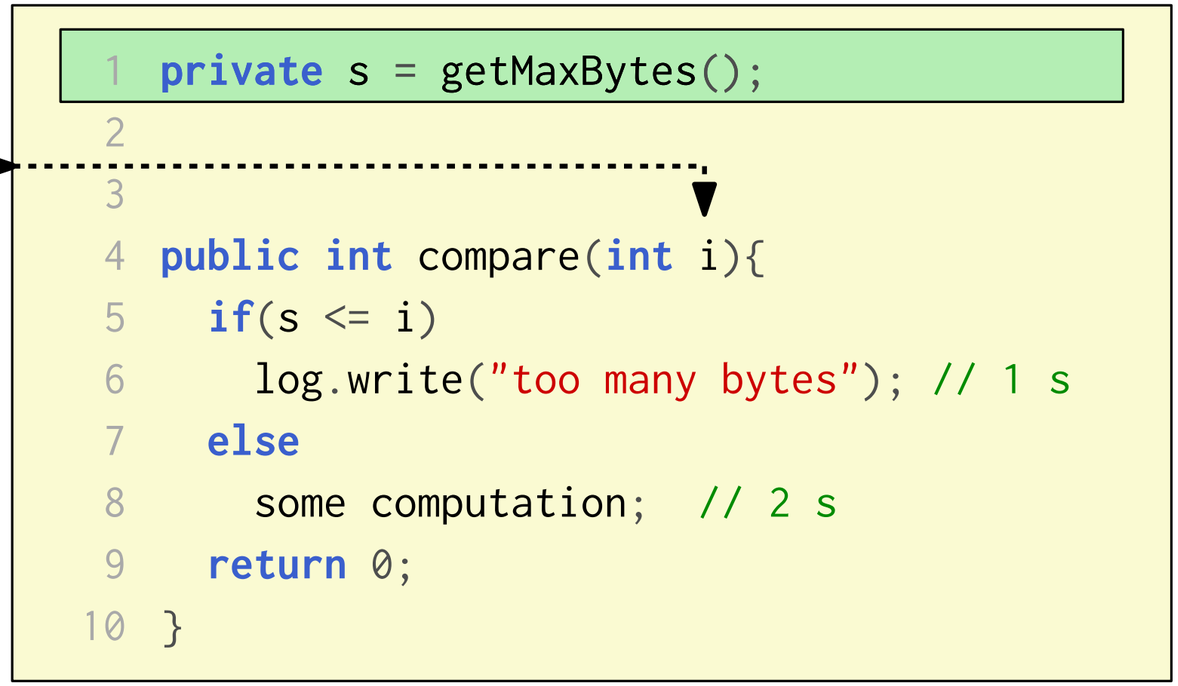


$$s \leq i \implies o = 1$$

$$s > i \implies o = 2$$

What is a software side channel?

input, i

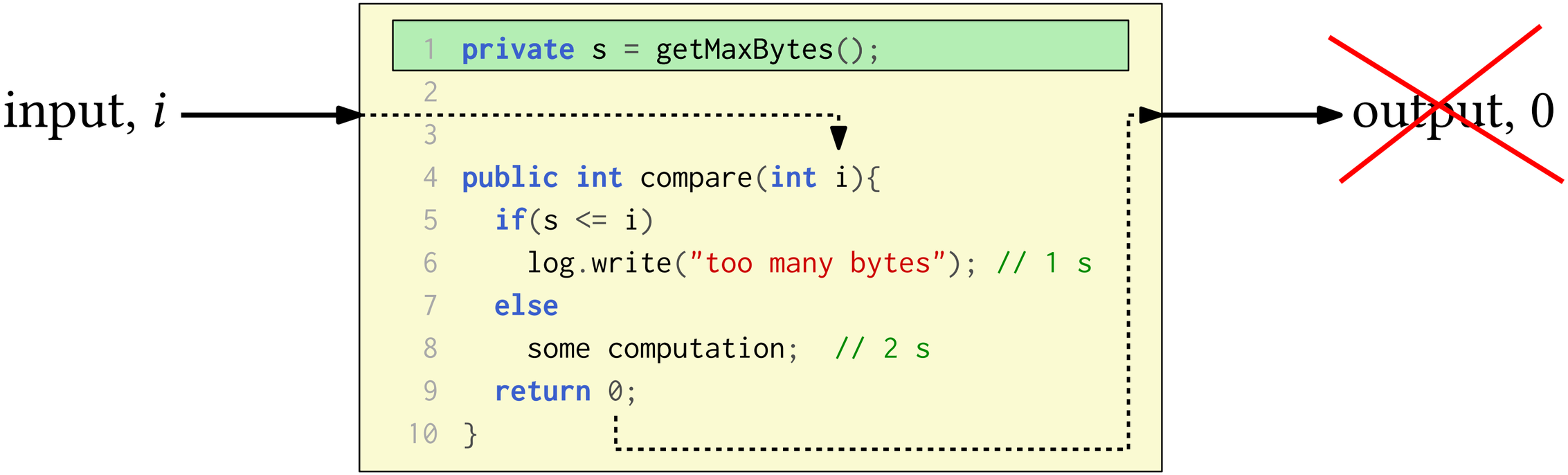


```
1  private s = getMaxBytes();
2
3
4  public int compare(int i){
5    if(s <= i)
6      log.write("too many bytes"); // 1 s
7    else
8      some computation;  // 2 s
9    return 0;
10 }
```

$$s \leq i \implies o = 1$$

$$s > i \implies o = 2$$

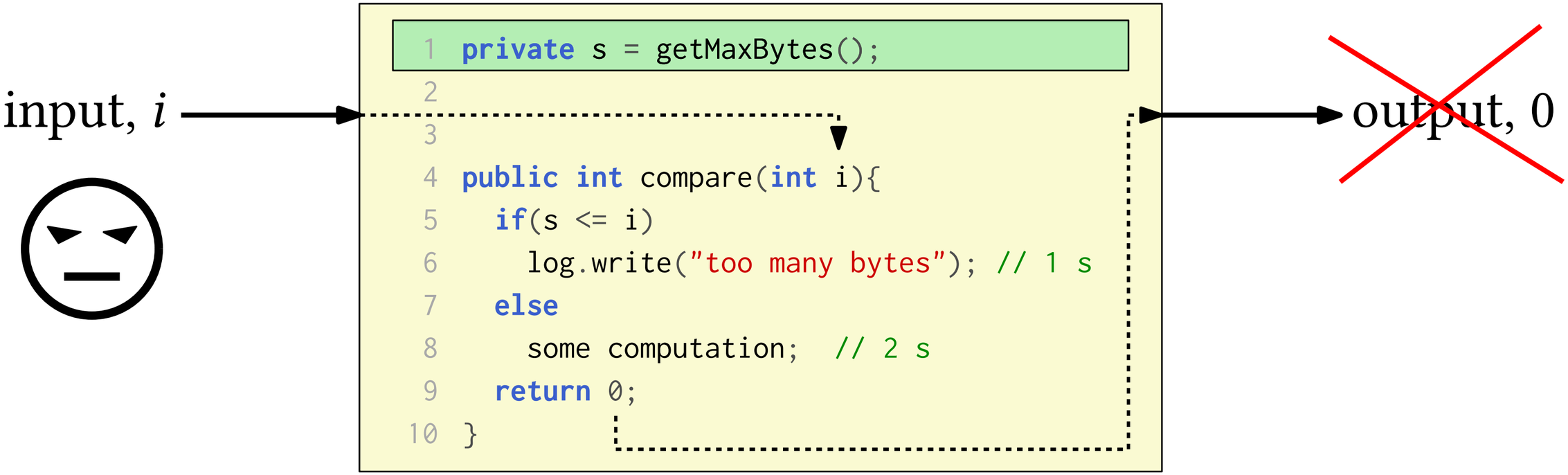
What is a software side channel?



$$s \leq i \implies o = 1$$

$$s > i \implies o = 2$$

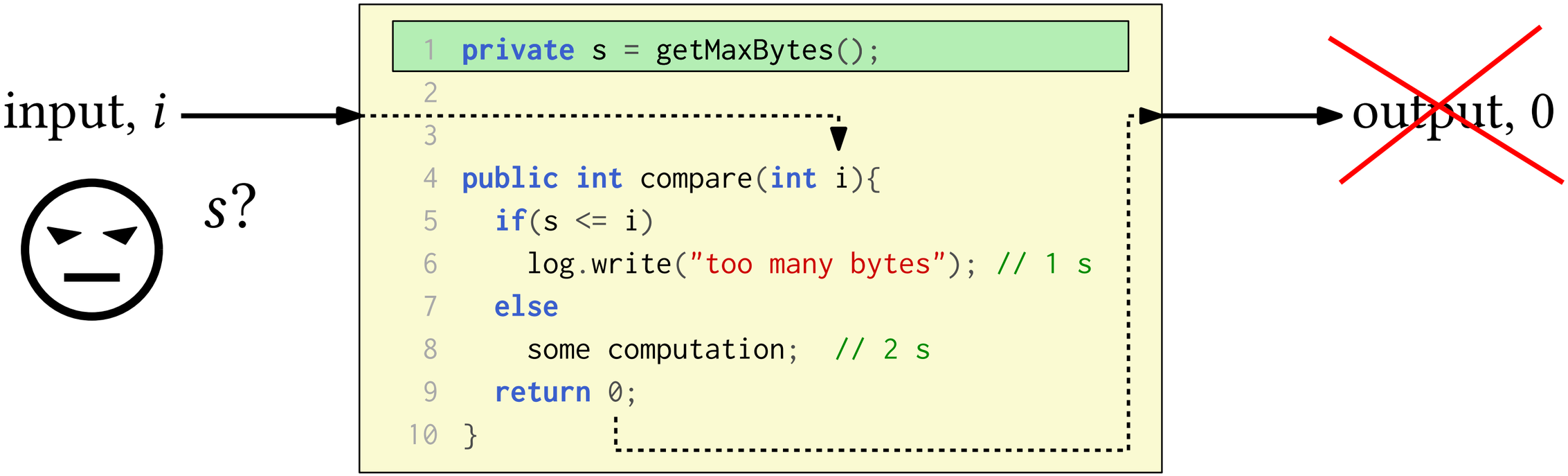
What is a software side channel?



$$s \leq i \implies o = 1$$

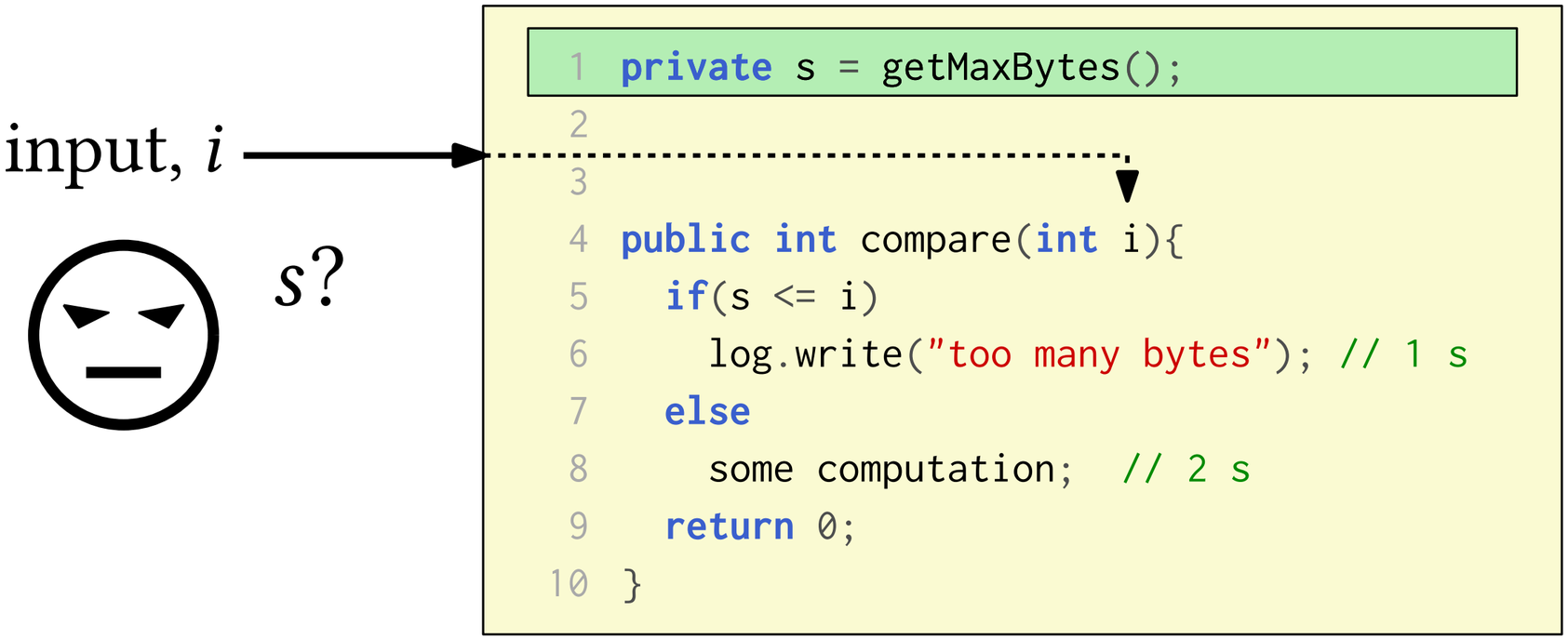
$$s > i \implies o = 2$$

What is a software side channel?



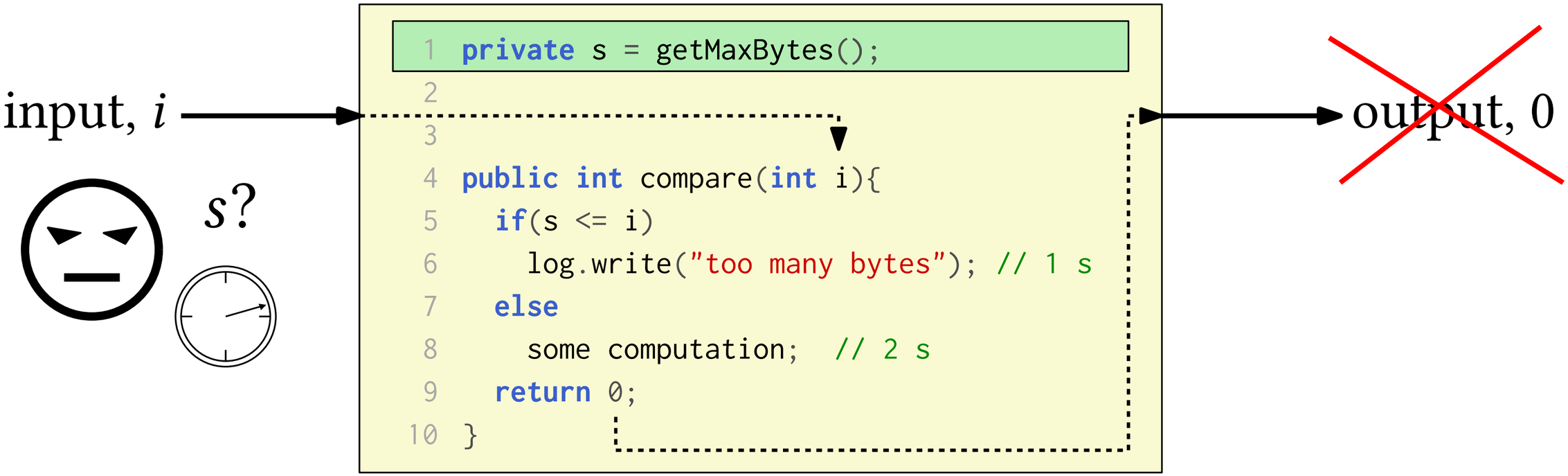
$$s \leq i \implies o = 1$$
$$s > i \implies o = 2$$

What is a software side channel?



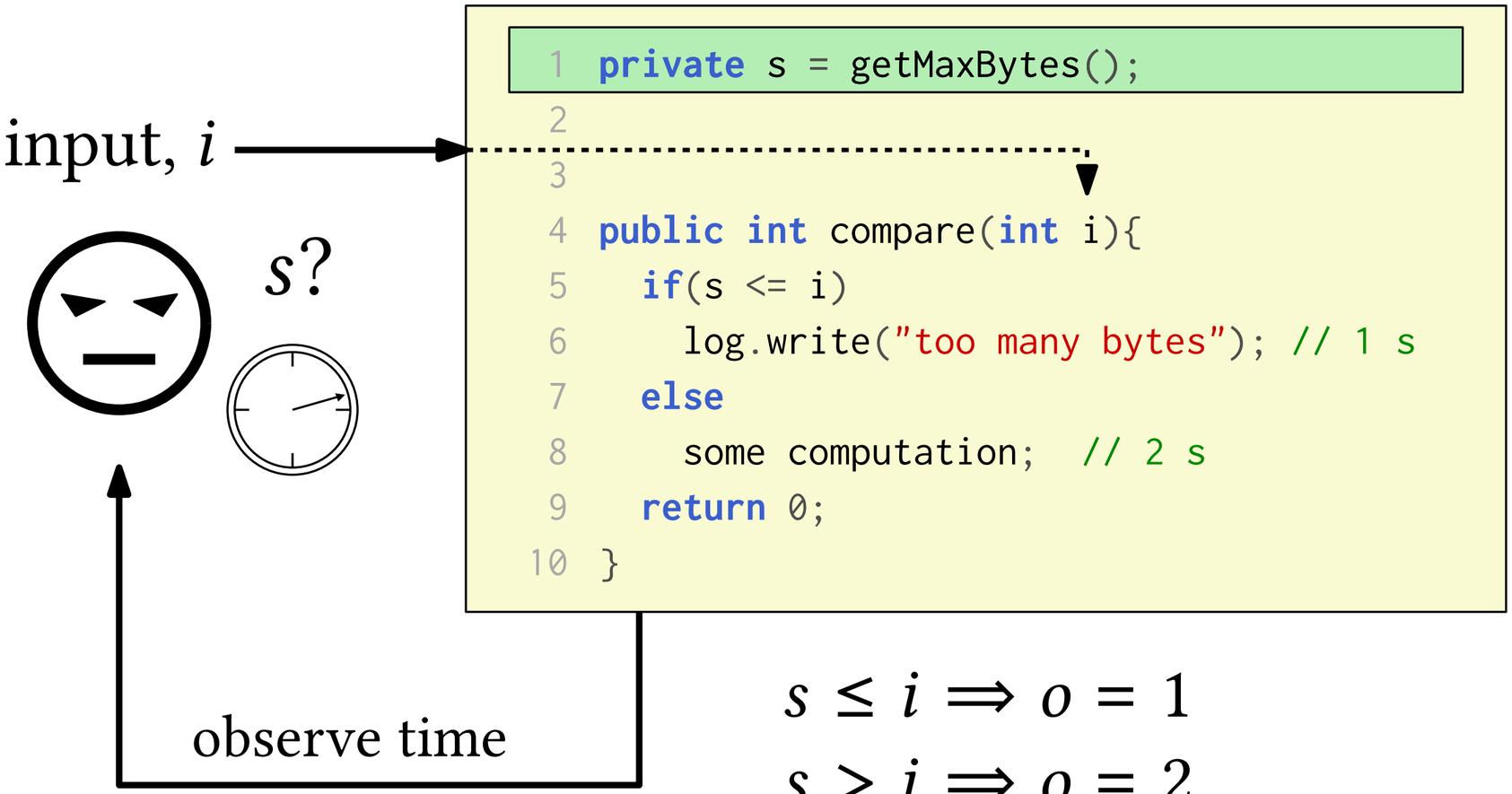
$$s \leq i \implies o = 1$$
$$s > i \implies o = 2$$

What is a software side channel?



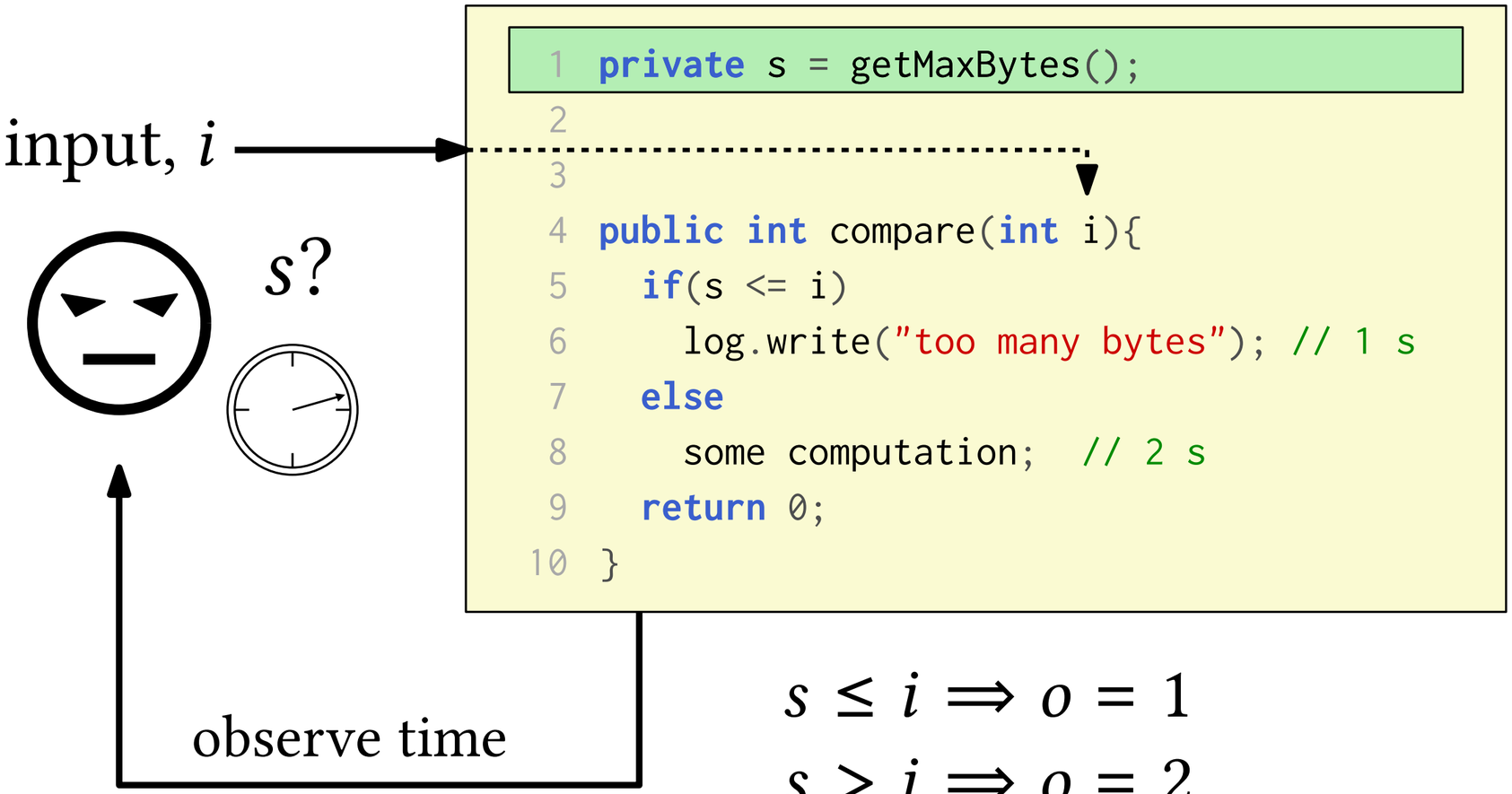
$$s \leq i \implies o = 1$$
$$s > i \implies o = 2$$

What is a software side channel?



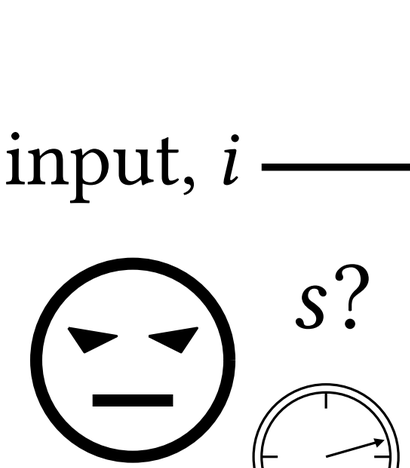
$$s \leq i \implies o = 1$$
$$s > i \implies o = 2$$

What is a software side channel?



$$s \leq i \implies o = 1$$
$$s > i \implies o = 2$$

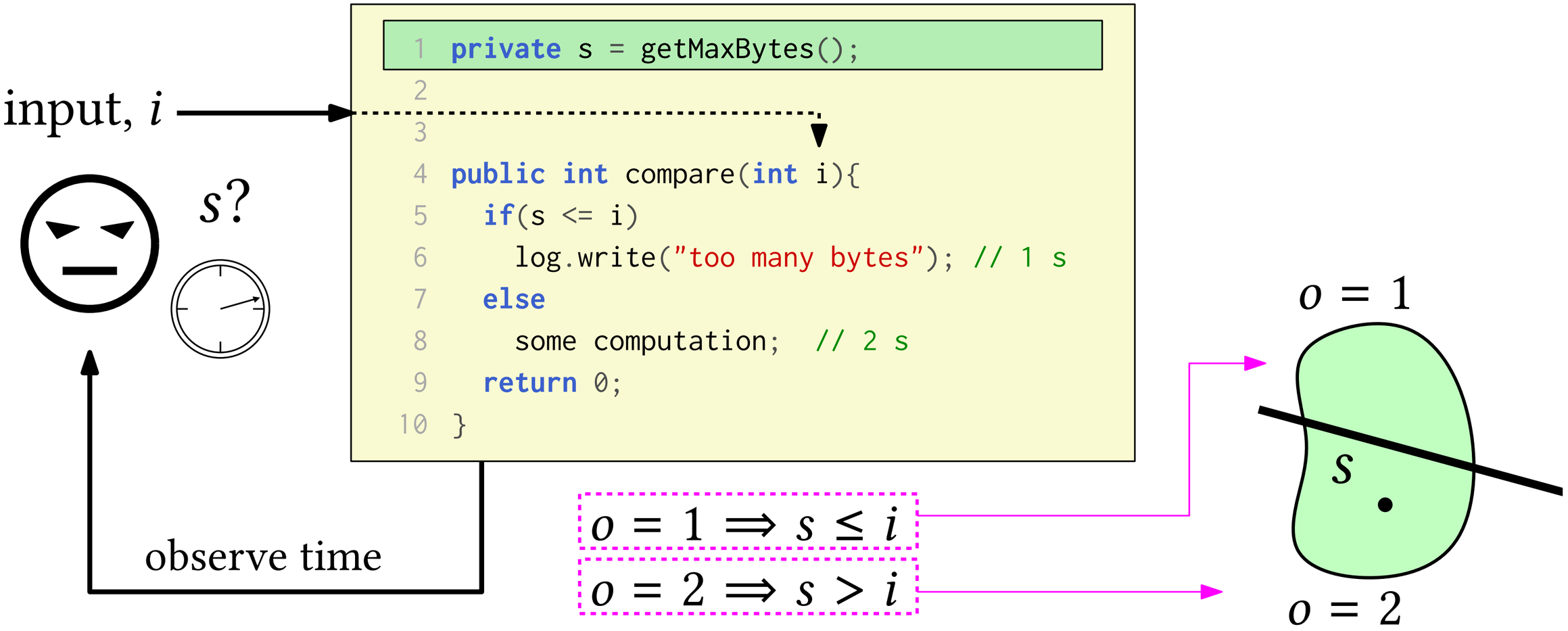
What is a software side channel?



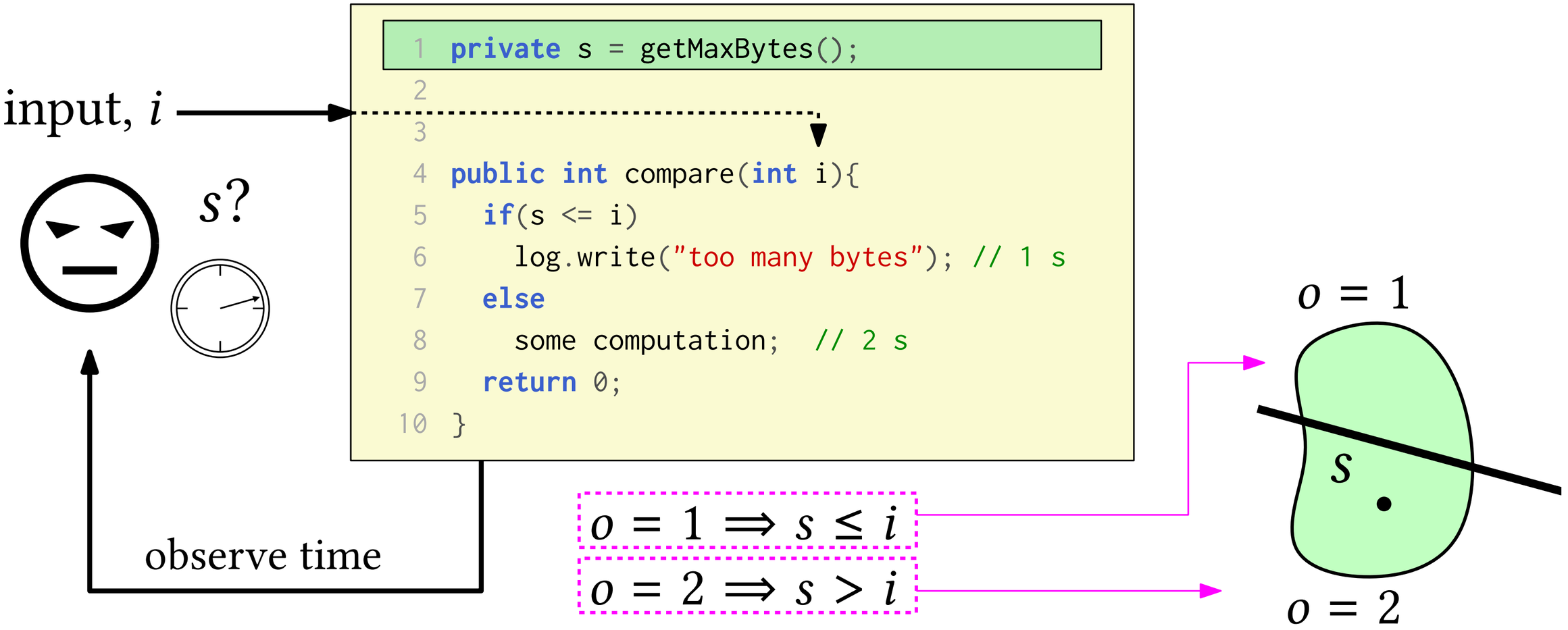

```
1  private s = getMaxBytes();
2
3
4  public int compare(int i){
5    if(s <= i)
6      log.write("too many bytes"); // 1 s
7    else
8      some computation;  // 2 s
9    return 0;
10 }
```

$$o = 1 \Rightarrow s \leq i$$
$$o = 2 \Rightarrow s > i$$

What is a software side channel?



What is a software side channel?



Side channel: (o, i) correlates with $s \Rightarrow$ reveal secret information

Goal:

Goal:

Given a program, P,

determine if P is vulnerable to side channel attacks

Goal:

Given a program, P,

determine if P is vulnerable to side channel attacks

How?

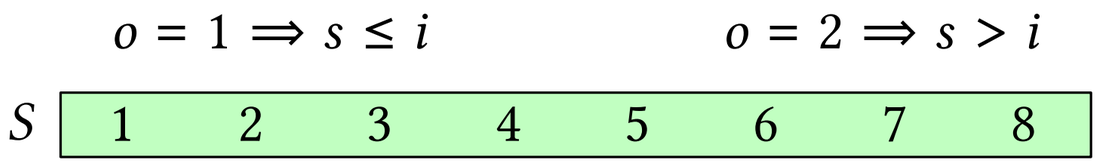
Synthesize an attack!

Adaptive Attack Trees

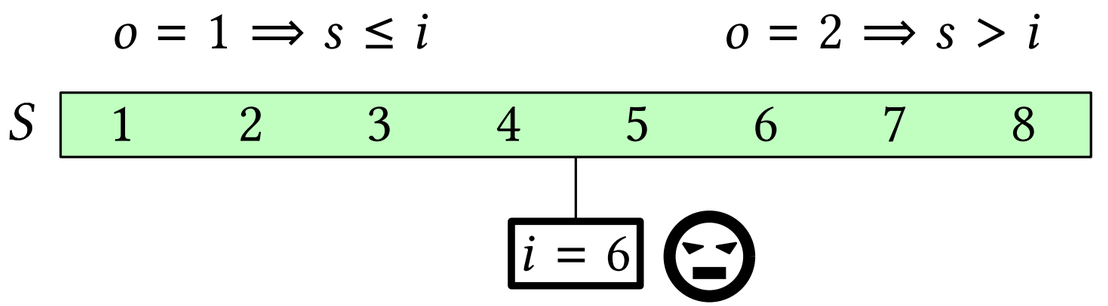
Adaptive Attack Trees

$o = 1 \Rightarrow s \leq i$ $o = 2 \Rightarrow s > i$

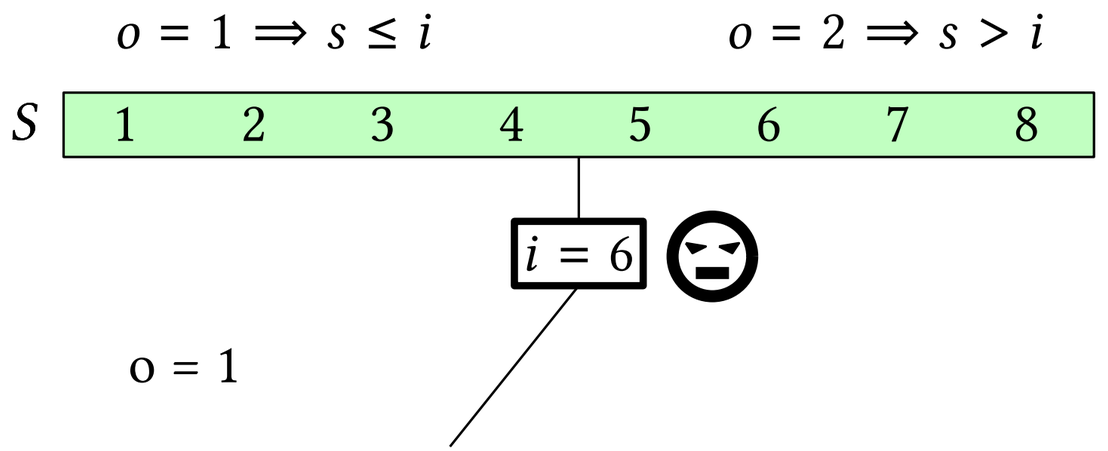
Adaptive Attack Trees



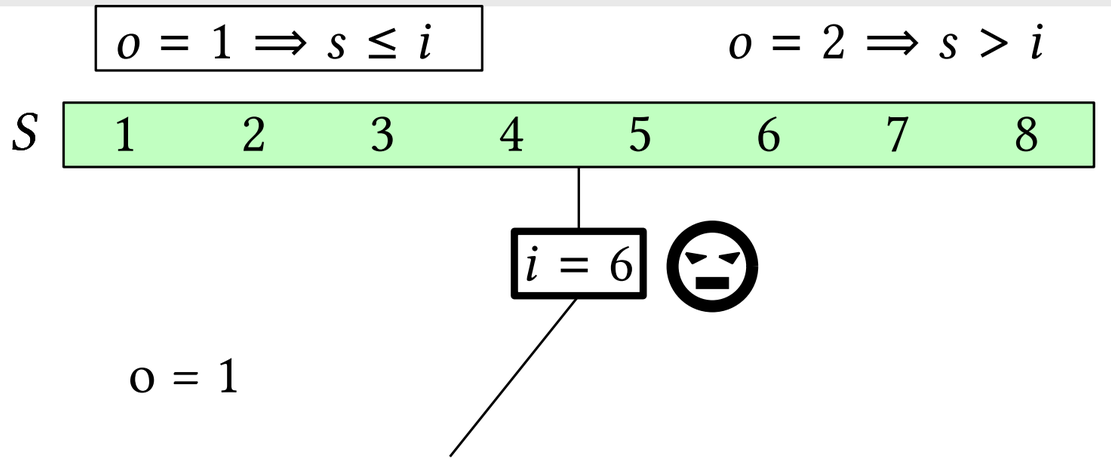
Adaptive Attack Trees



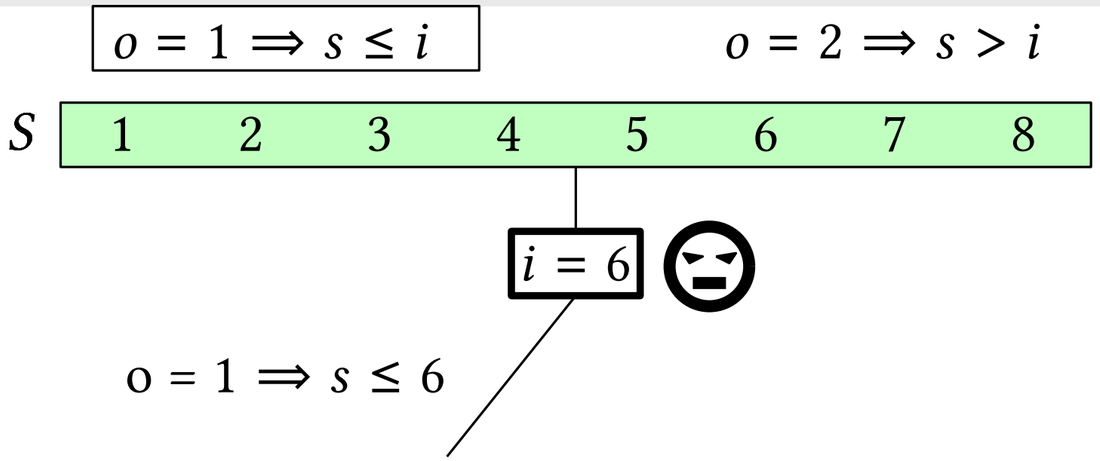
Adaptive Attack Trees



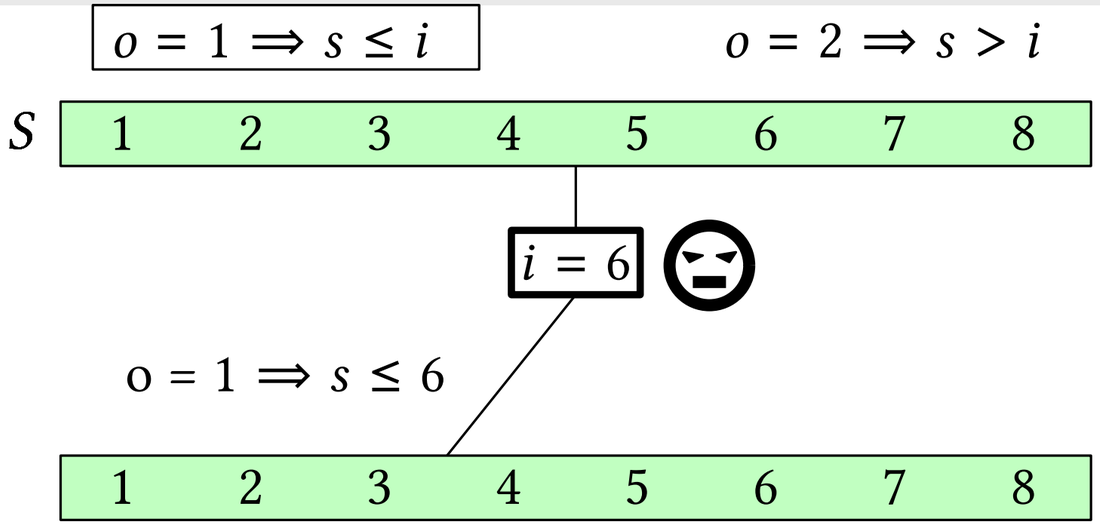
Adaptive Attack Trees



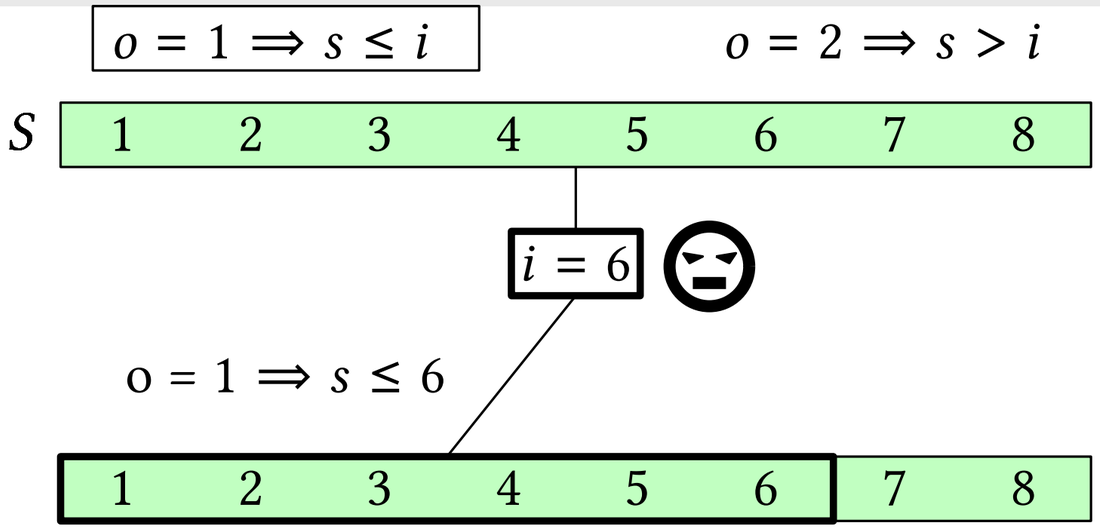
Adaptive Attack Trees



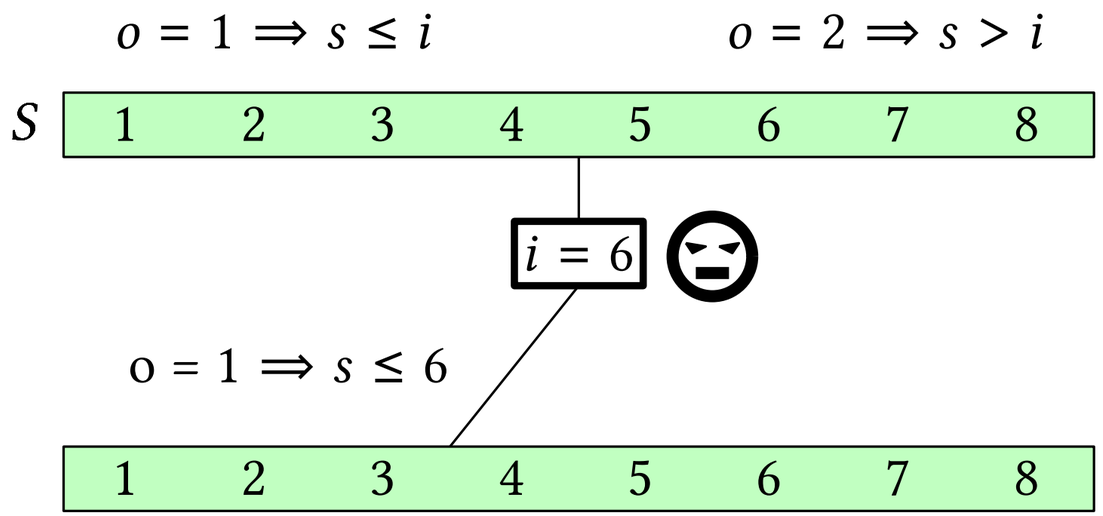
Adaptive Attack Trees



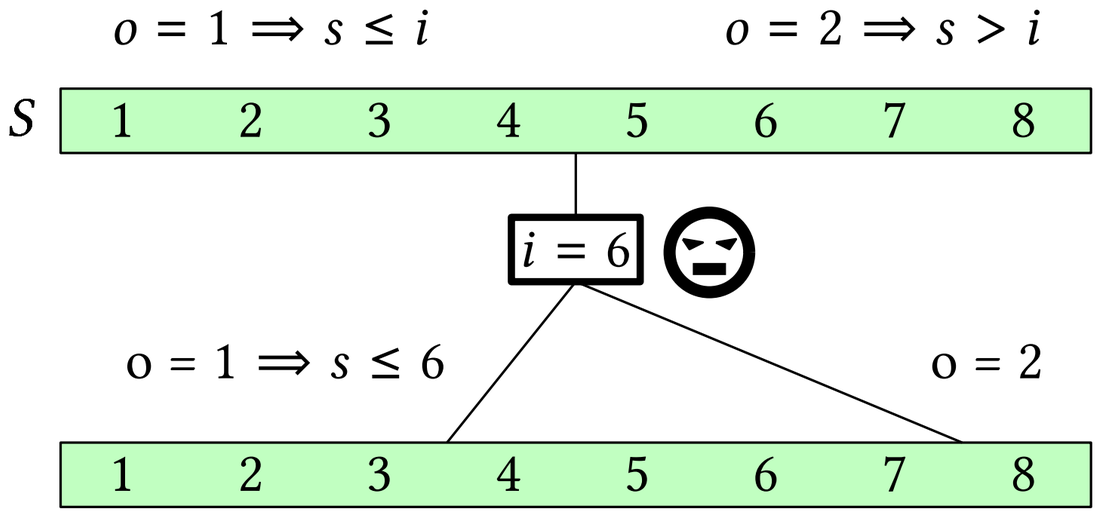
Adaptive Attack Trees



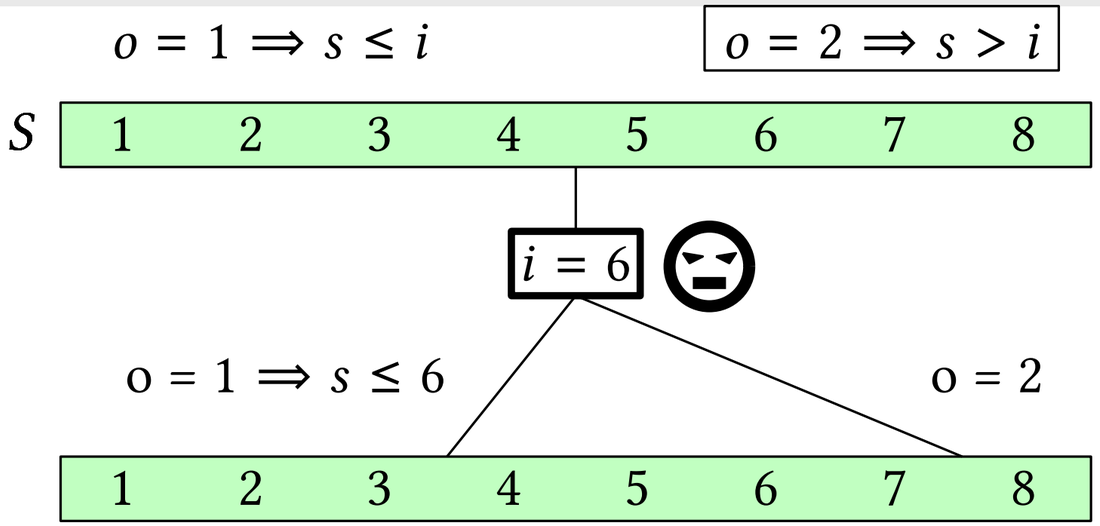
Adaptive Attack Trees



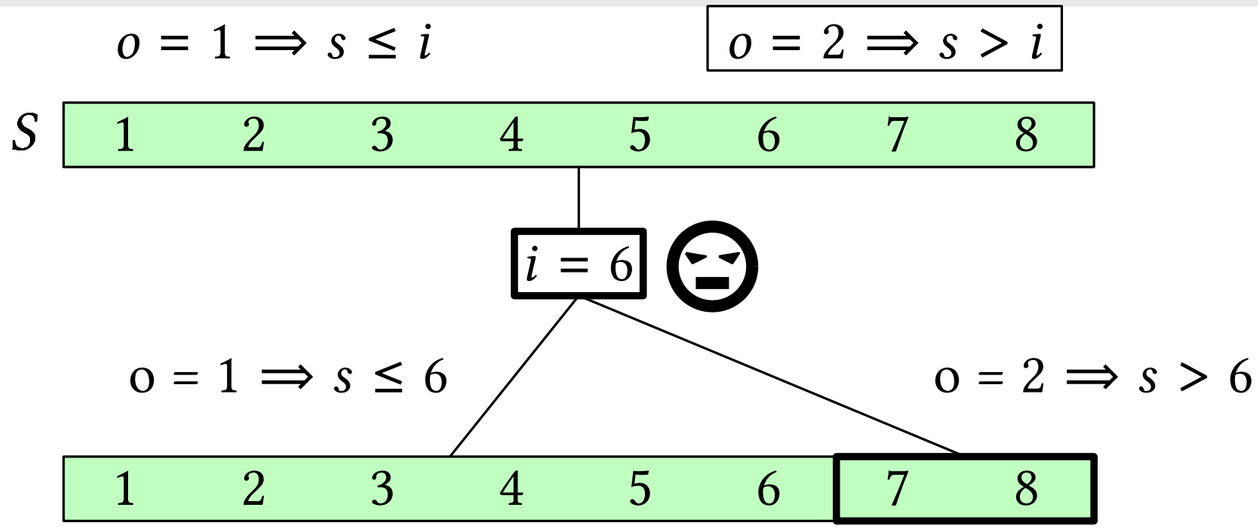
Adaptive Attack Trees



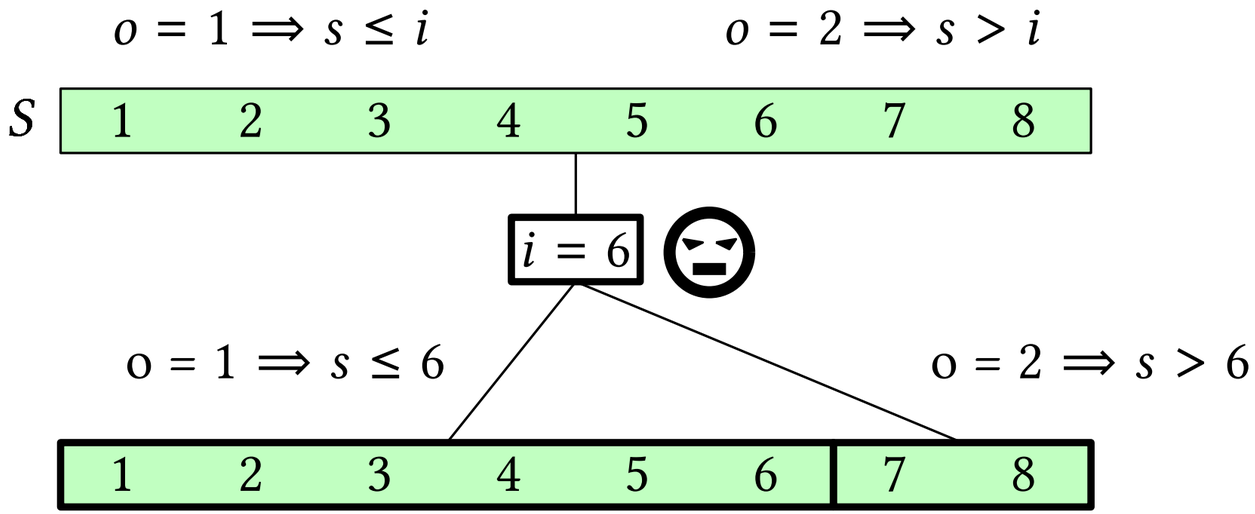
Adaptive Attack Trees



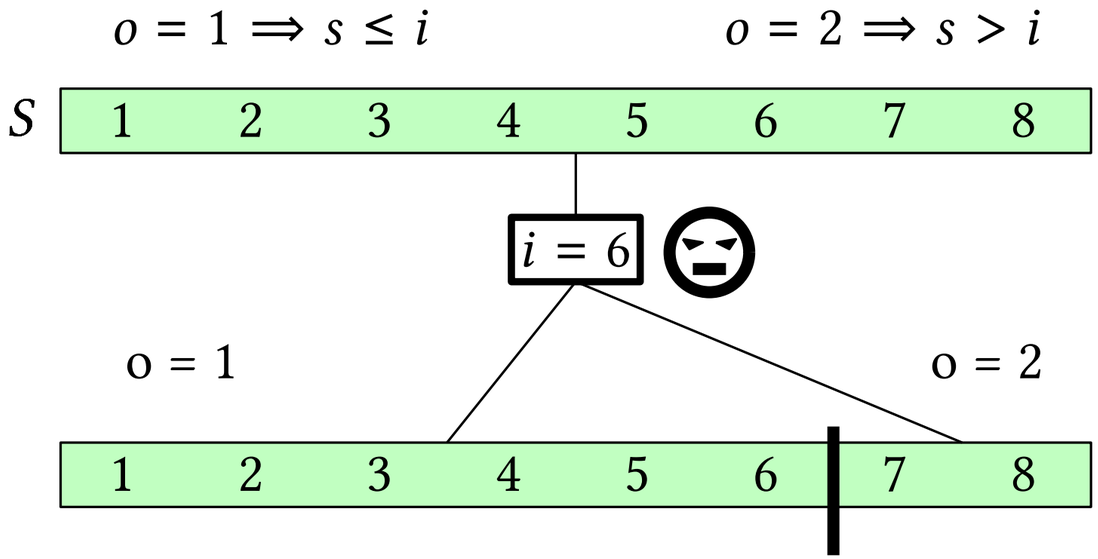
Adaptive Attack Trees



Adaptive Attack Trees

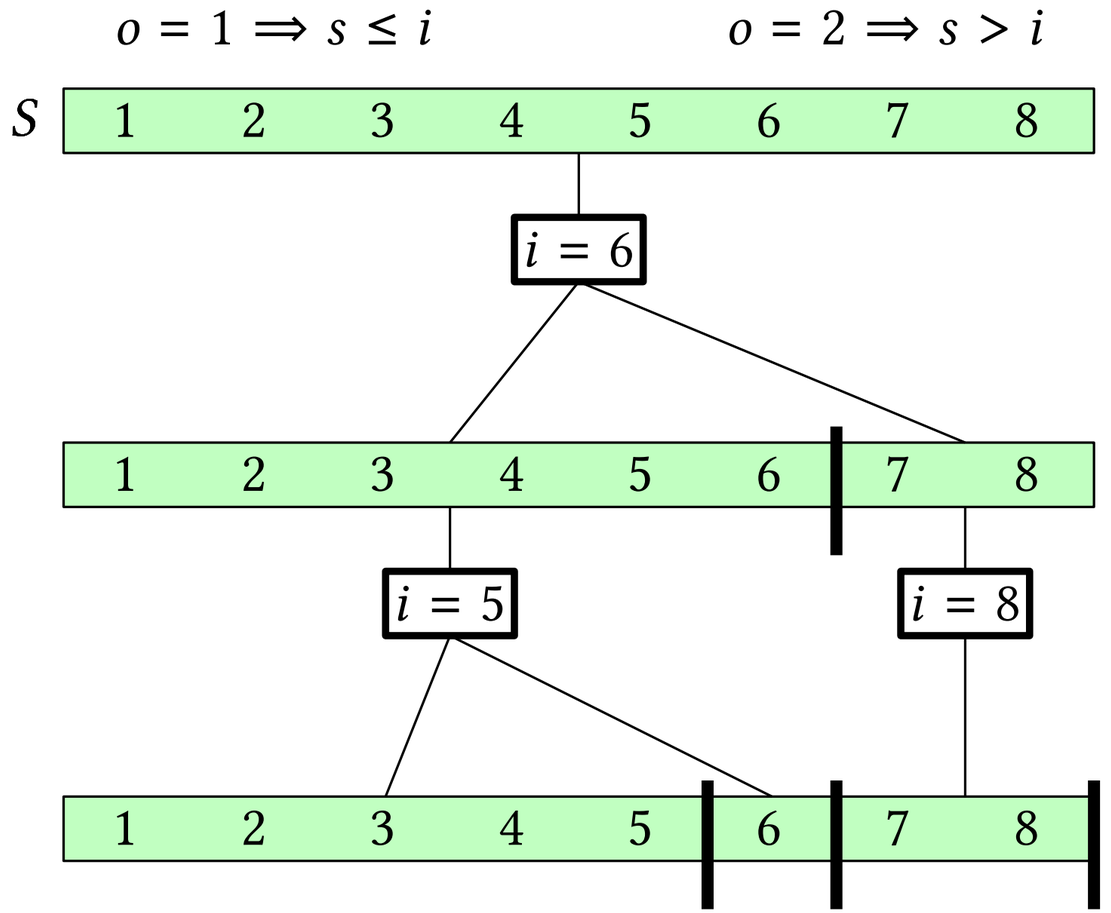


Adaptive Attack Trees

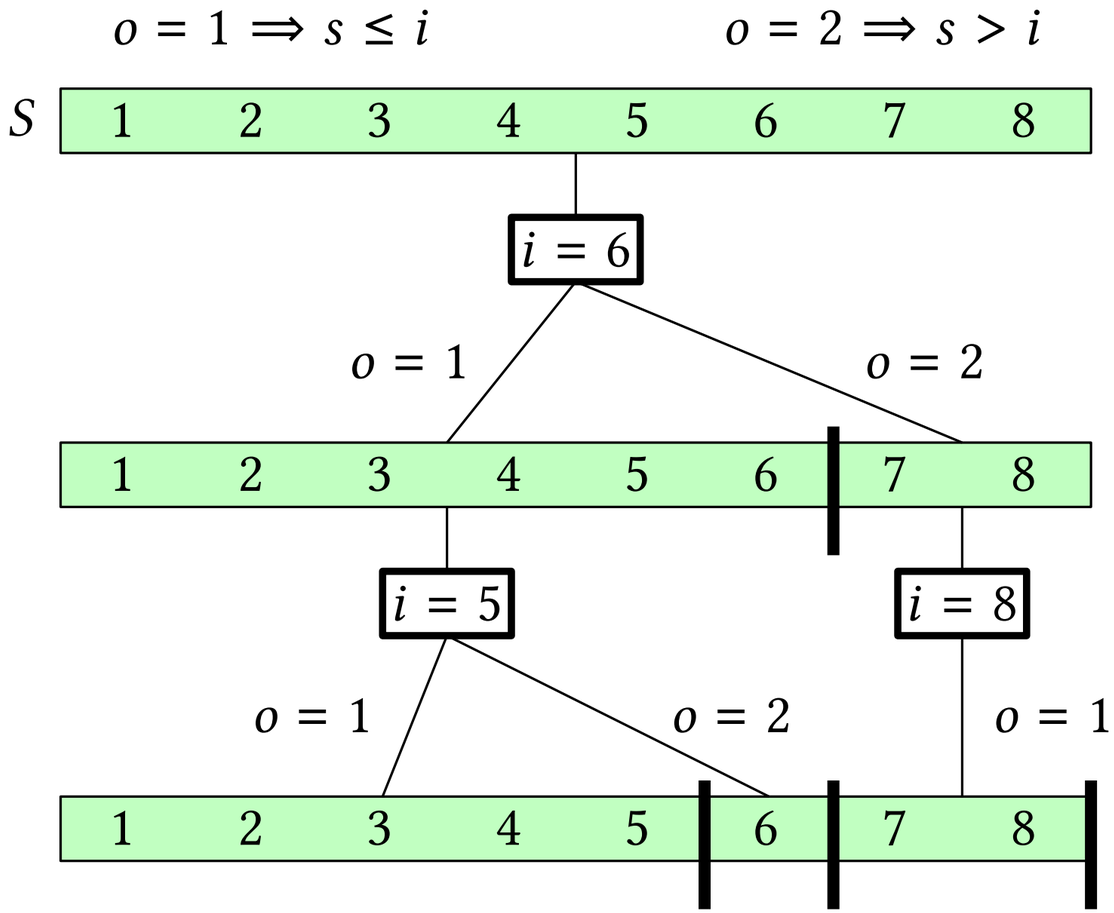


attacker's (i, o) **partitions** S domain

Adaptive Attack Trees

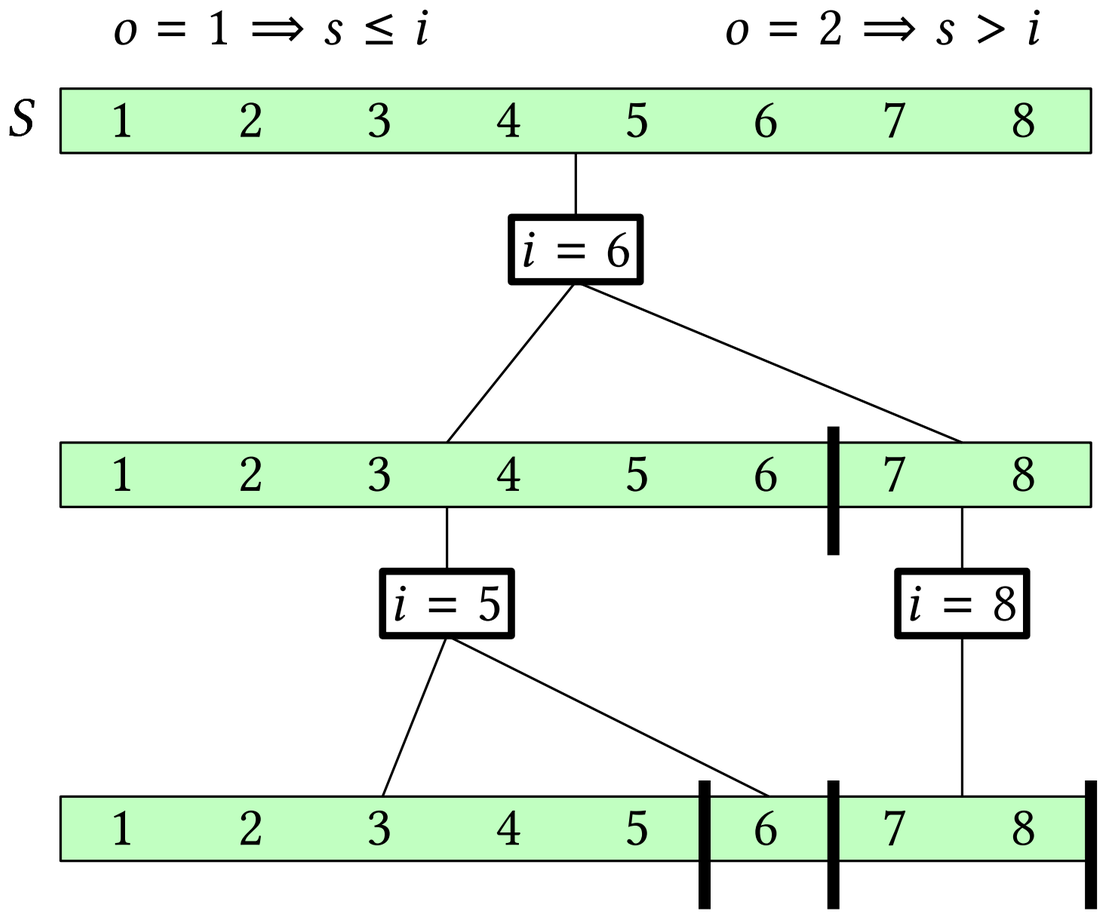


Adaptive Attack Trees

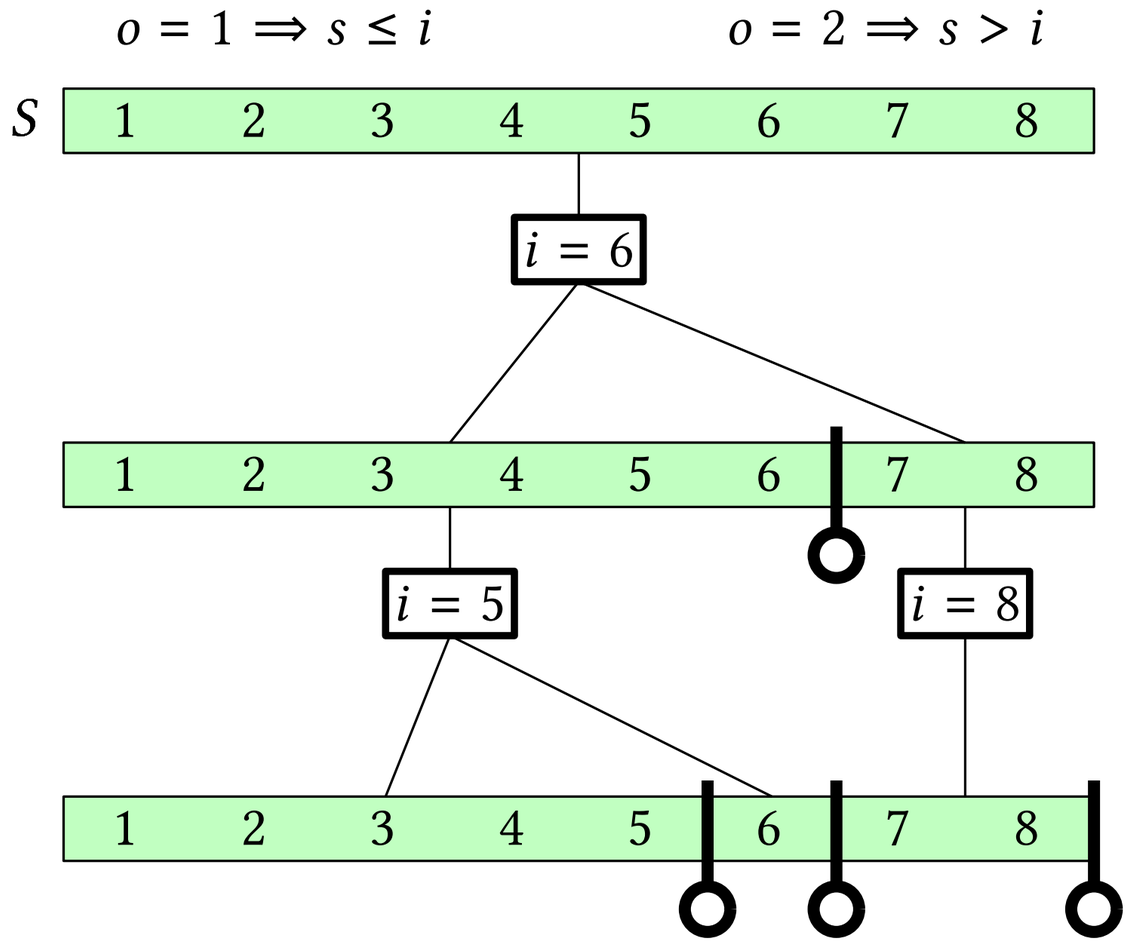


attacker's (i, o) sequences **partition** the S domain

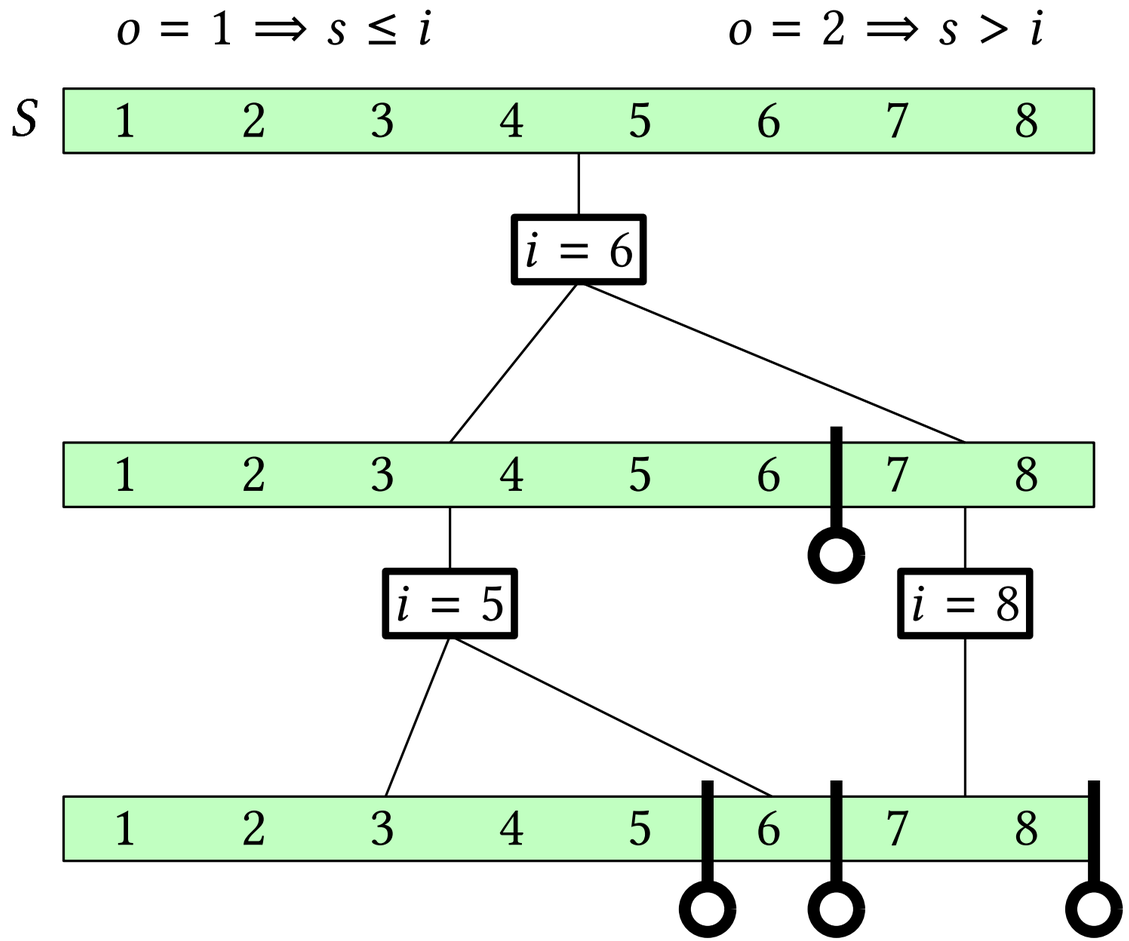
Adaptive Attack Trees



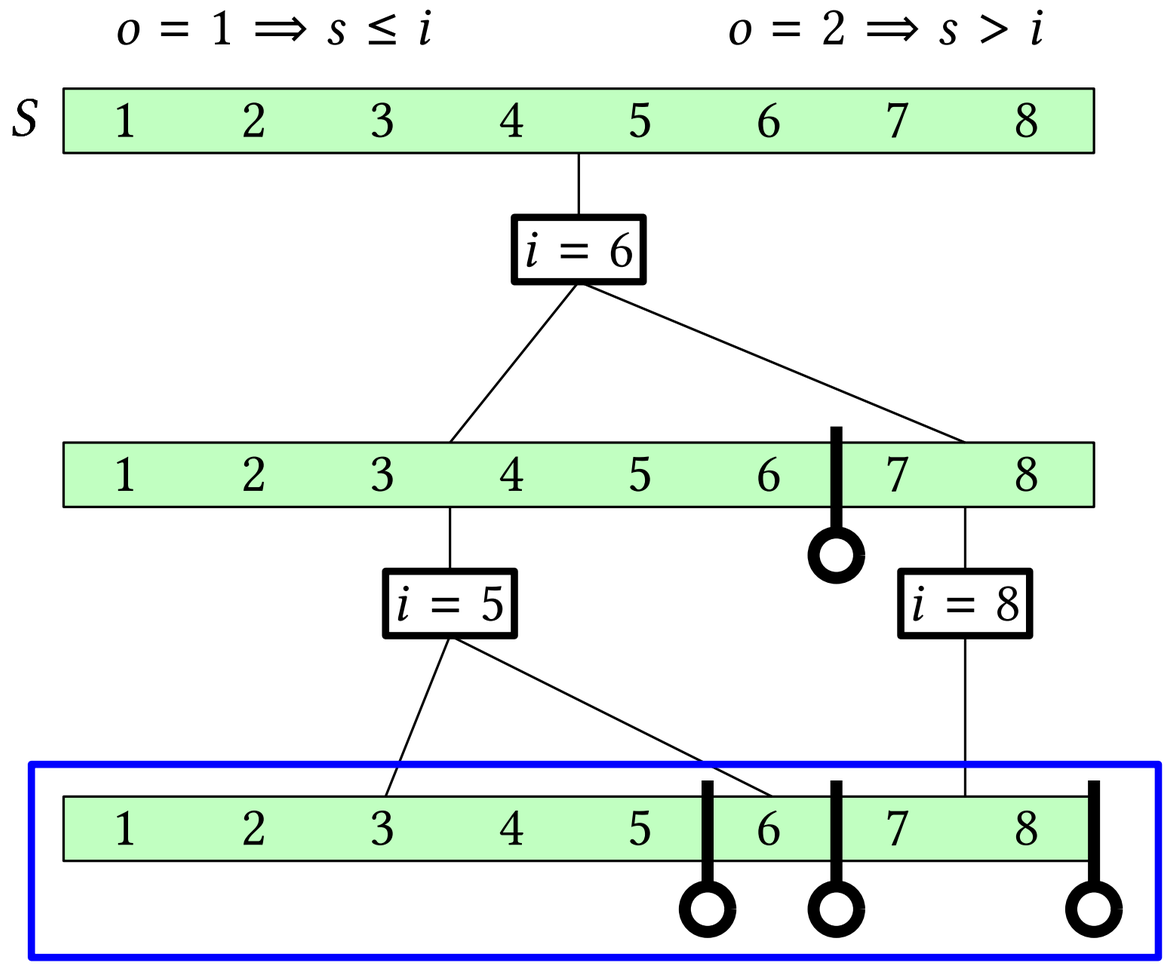
Adaptive Attack Trees



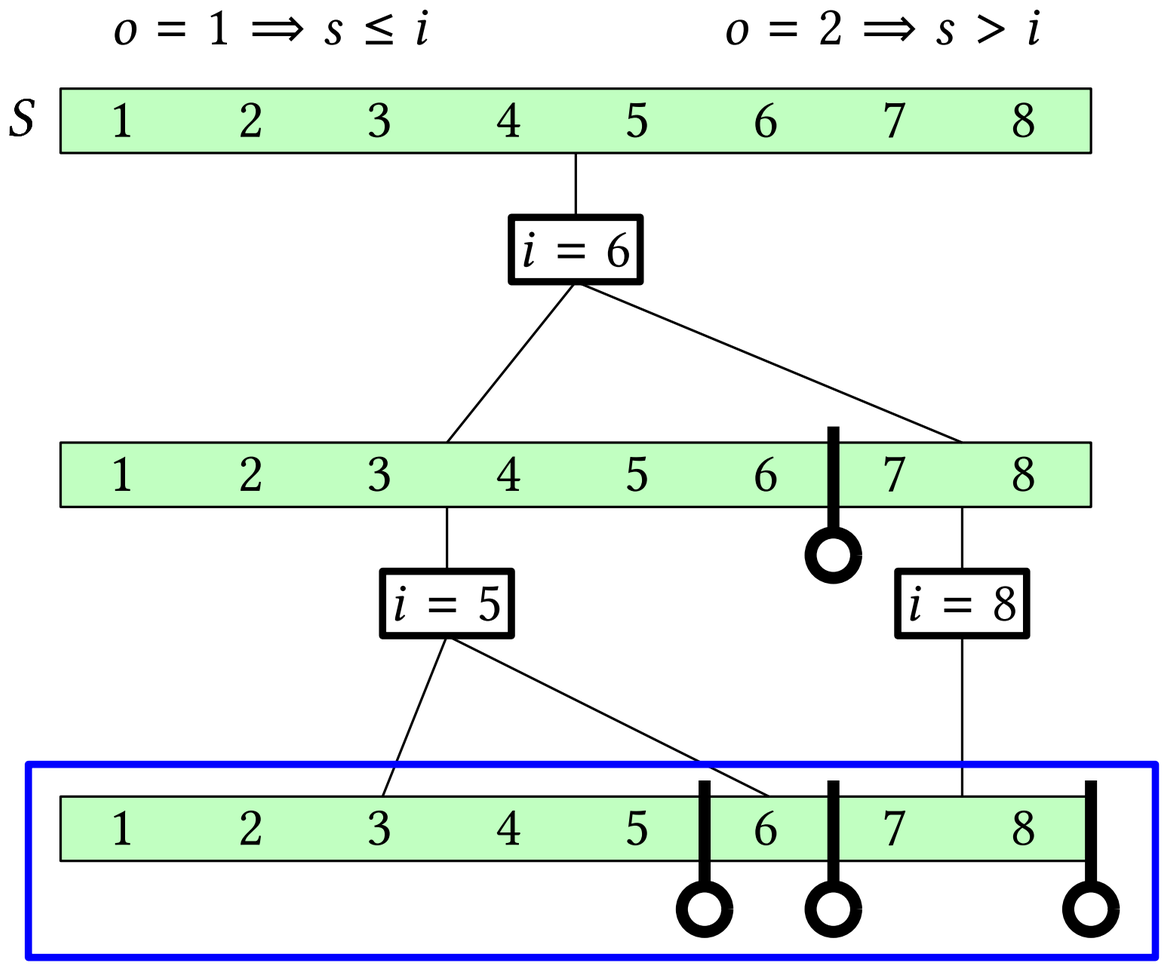
Adaptive Attack Trees



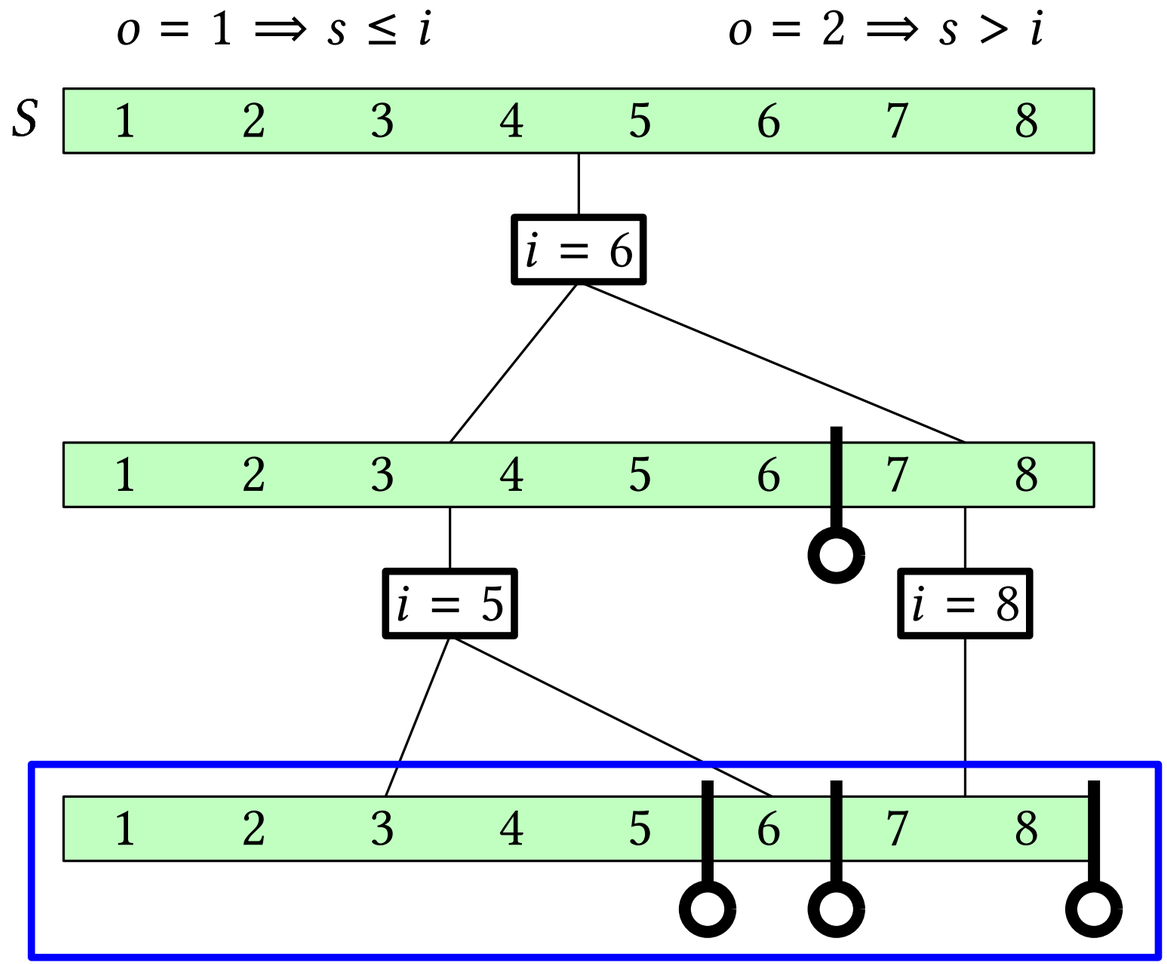
Adaptive Attack Trees



Adaptive Attack Trees



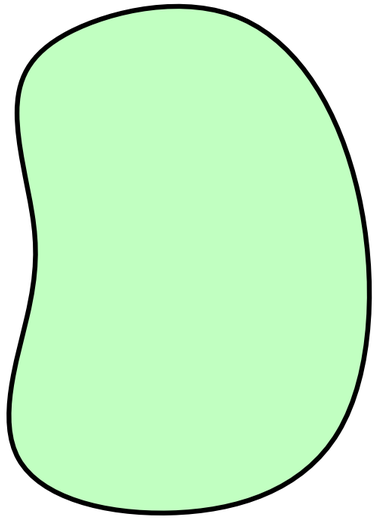
Adaptive Attack Trees



How to choose the best partition?

Entropy: Side Channels and Searching

$i_0 \in I$

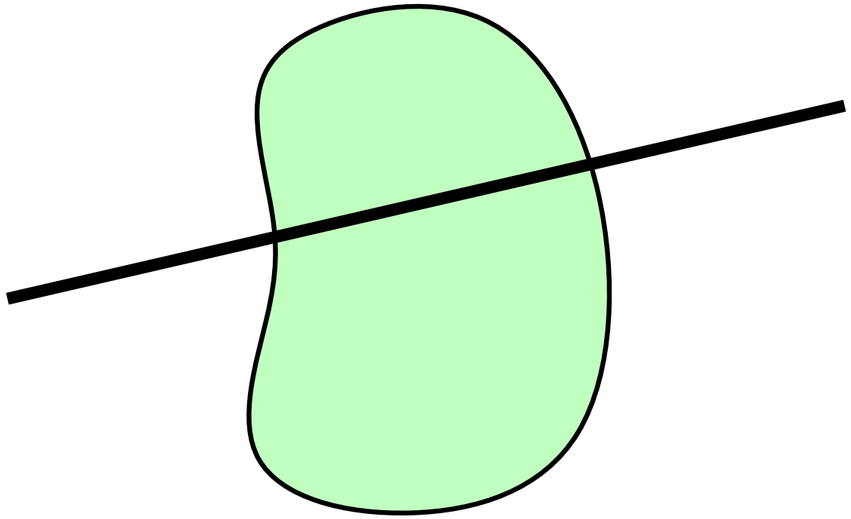


secret $s \in S$

Entropy: Side Channels and Searching

$i_0 \in I$

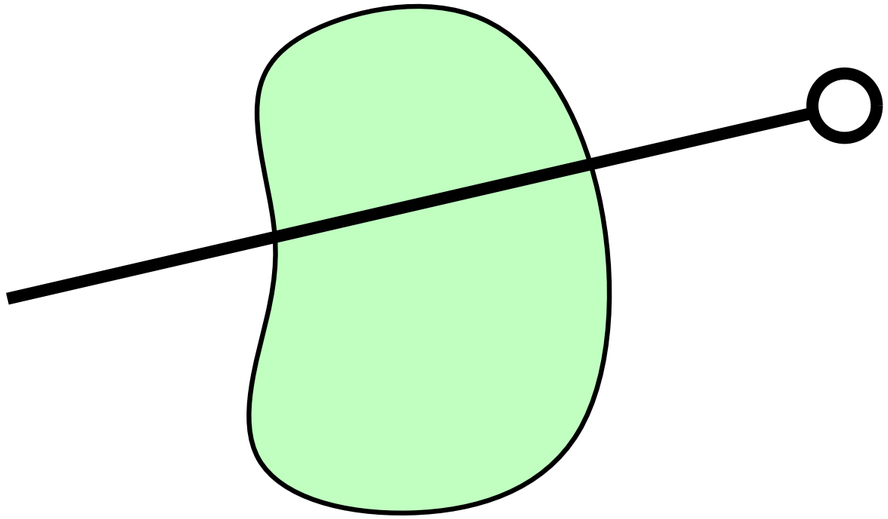
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

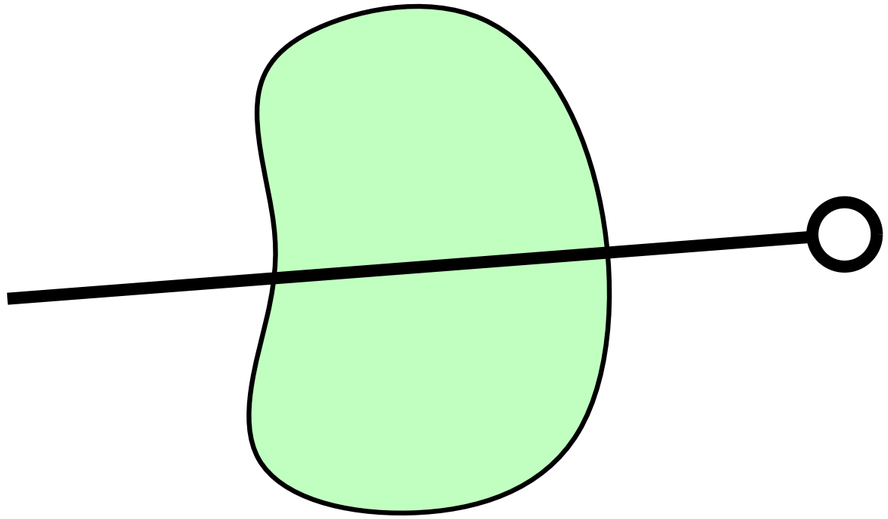
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

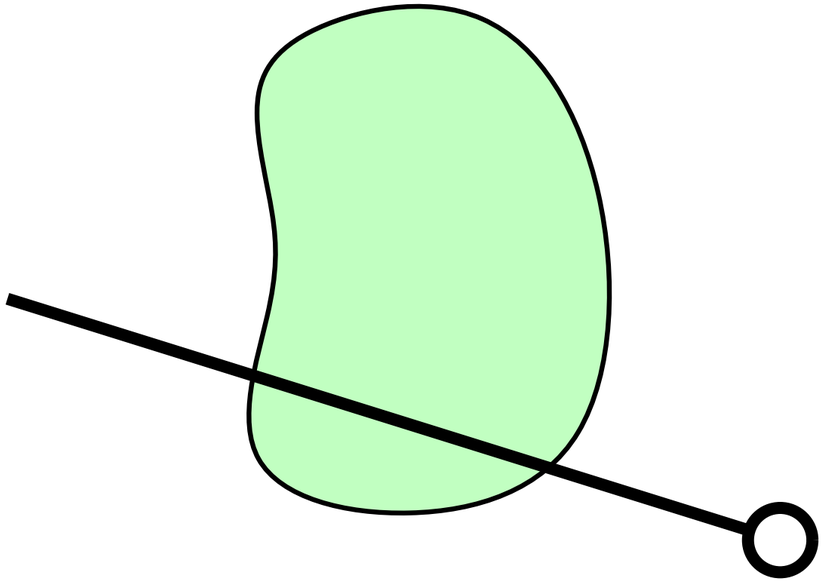
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

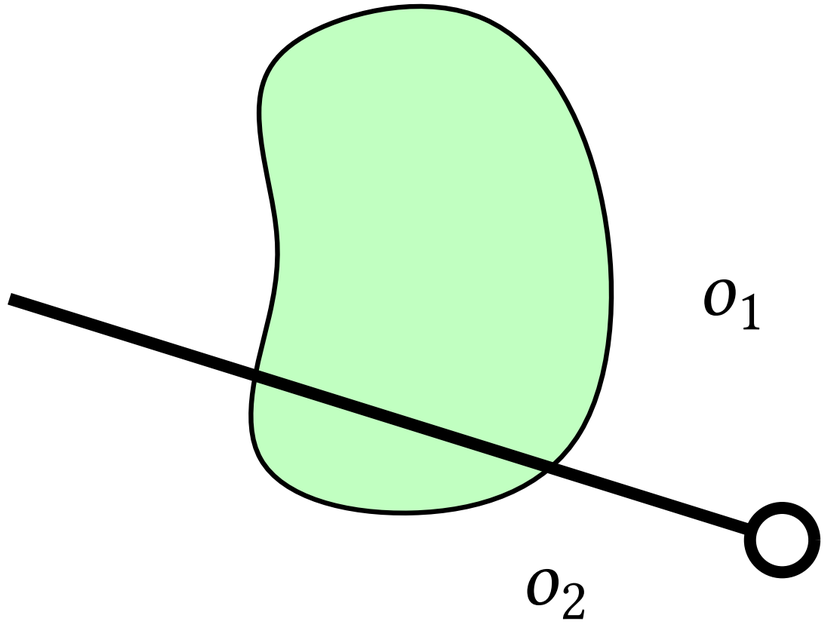
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

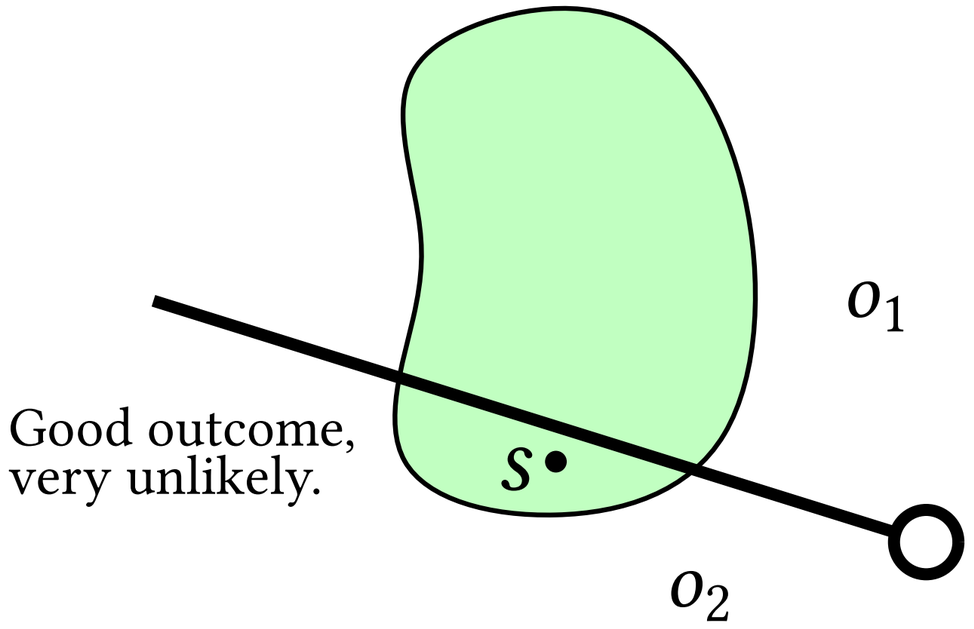
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

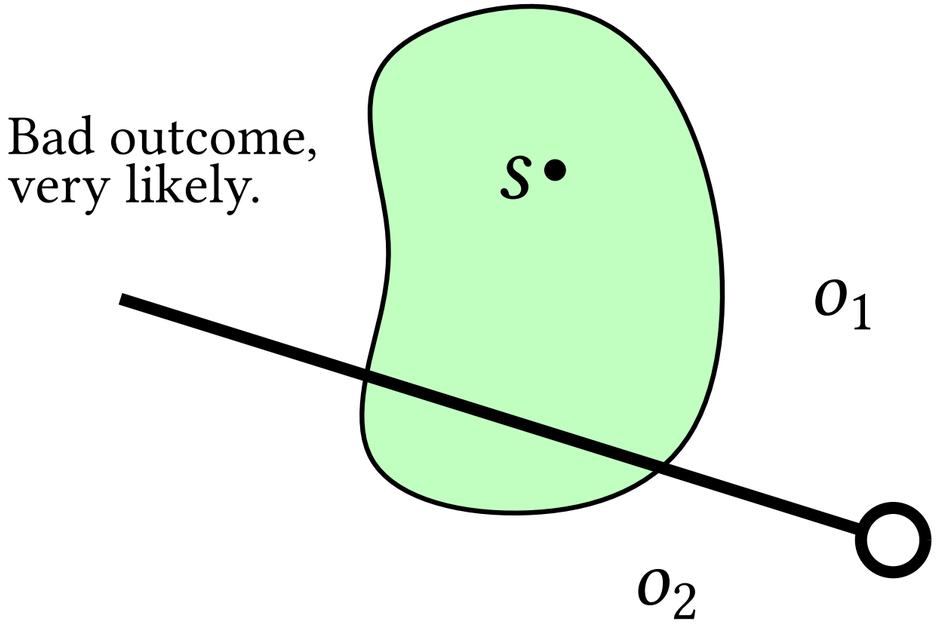
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

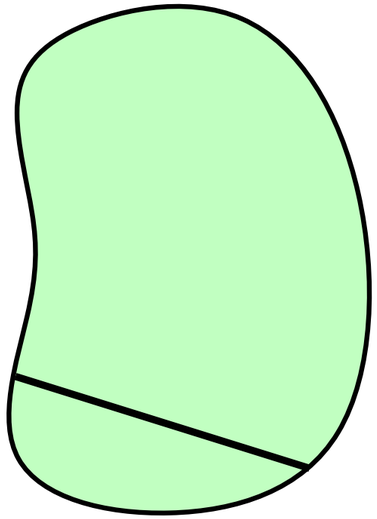
secret $s \in S$



Entropy: Side Channels and Searching

$i_0 \in I$

secret $s \in S$

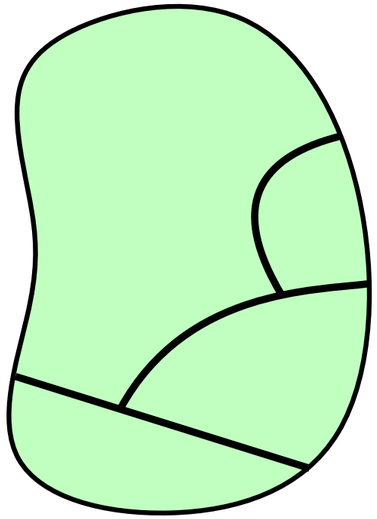


Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



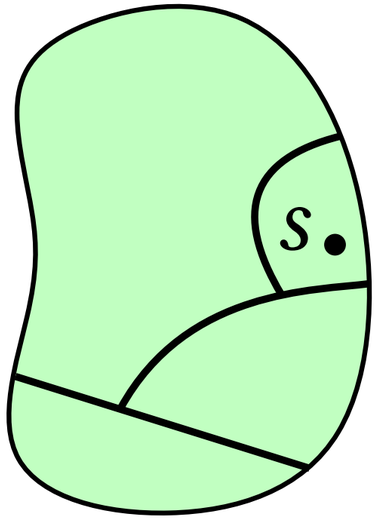
secret $s \in S$

Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



secret $s \in S$

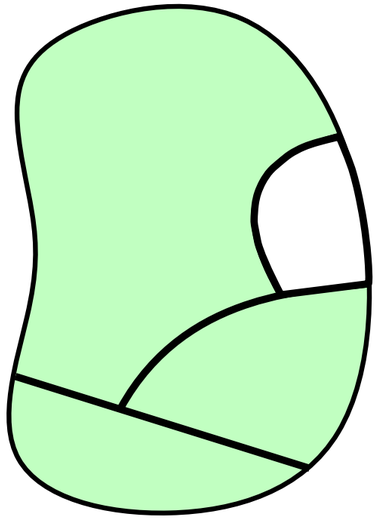
Entropy: Side Channels and Searching

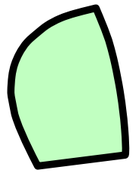
$i_0 \in I$

$i_1 \in I$

$i_2 \in I$

secret $s \in S$



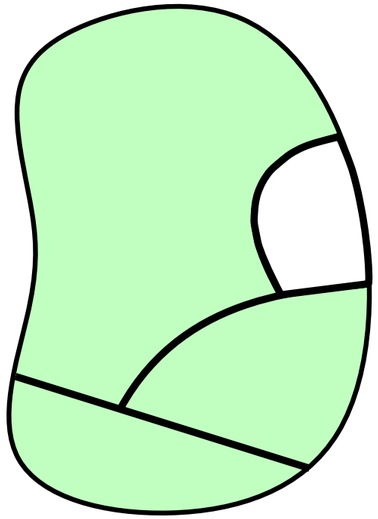
$p(s \in$  $)$

Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



secret $s \in S$

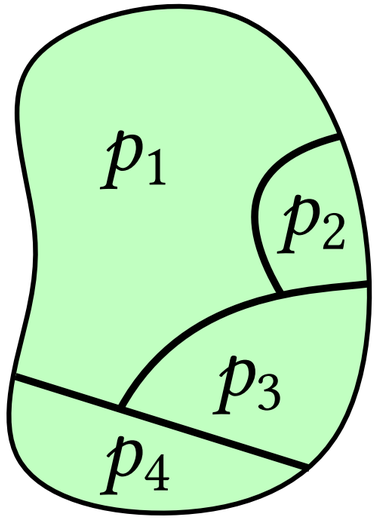
$$p(s \in \text{[shape]}) = \frac{|\text{[shape]}|}{|\text{[shape]}|}$$

Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



secret $s \in S$

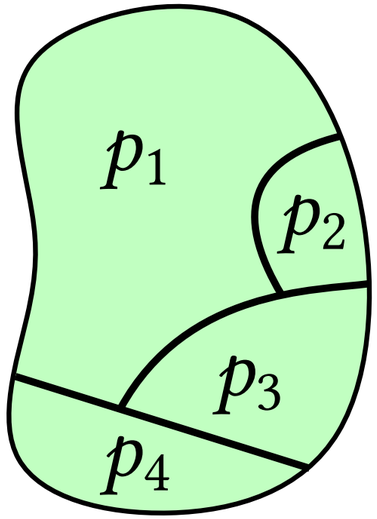
Entropy: Side Channels and Searching

secret $s \in S$

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



Quantify expected info gain measured in bits.

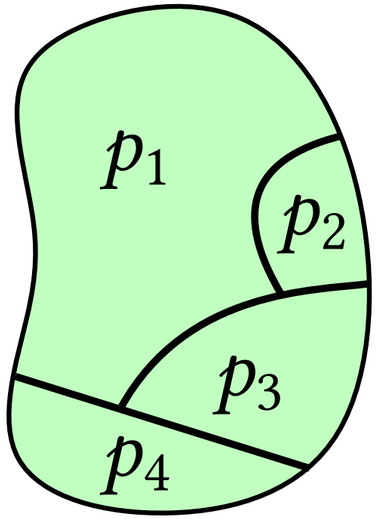
Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$

secret $s \in S$



Quantify expected info gain measured in bits.

$$\frac{1}{p_j}$$

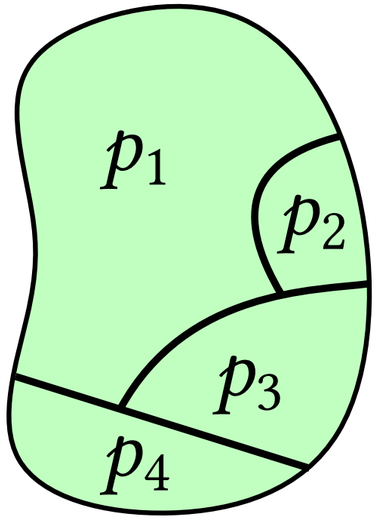
Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$

secret $s \in S$



Quantify expected info gain measured in bits.

$$\log_2 \frac{1}{p_j}$$

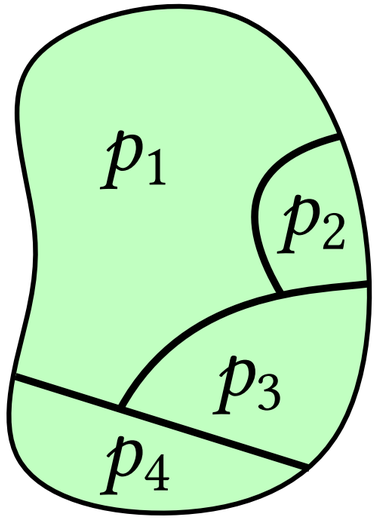
Entropy: Side Channels and Searching

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$

secret $s \in S$



Quantify expected info gain measured in bits.

$$\sum_{j=1}^{n} p_j \log_2 \frac{1}{p_j}$$

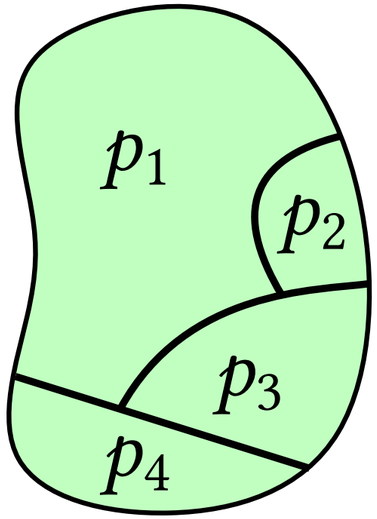
Entropy: Side Channels and Searching

secret $s \in S$

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



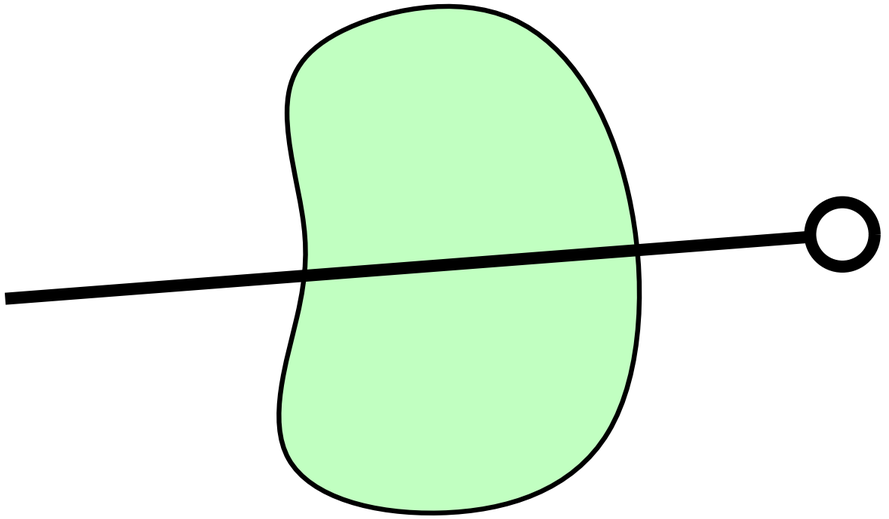
Quantify expected info gain measured in bits.

$$\mathcal{H} = \sum_{j=1}^{n} p_j \log_2 \frac{1}{p_j}$$

Entropy: Side Channels and Searching

$i_0 \in I$

secret $s \in S$



Quantify expected info gain measured in bits.

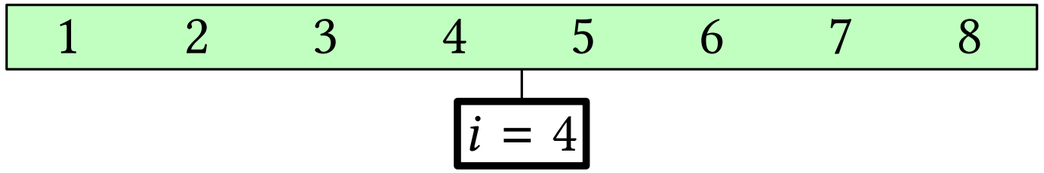
$$\mathcal{H} = \sum_{j=1}^{n} p_j \log_2 \frac{1}{p_j}$$

$o = 1 \Rightarrow s \leq i$

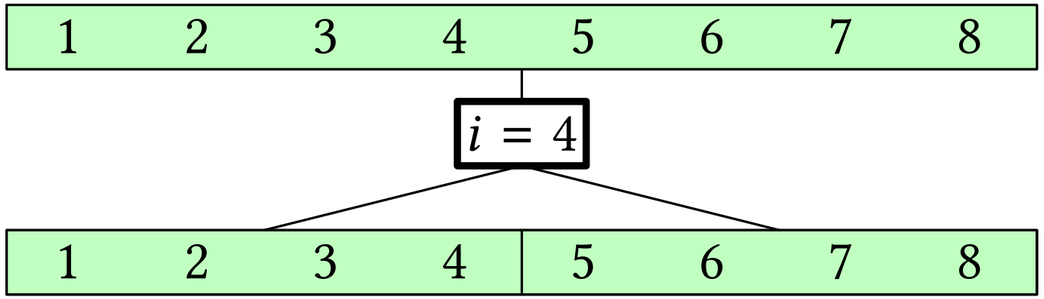
$o = 2 \Rightarrow s > i$

$o = 1 \Rightarrow s \leq i$ $o = 2 \Rightarrow s > i$

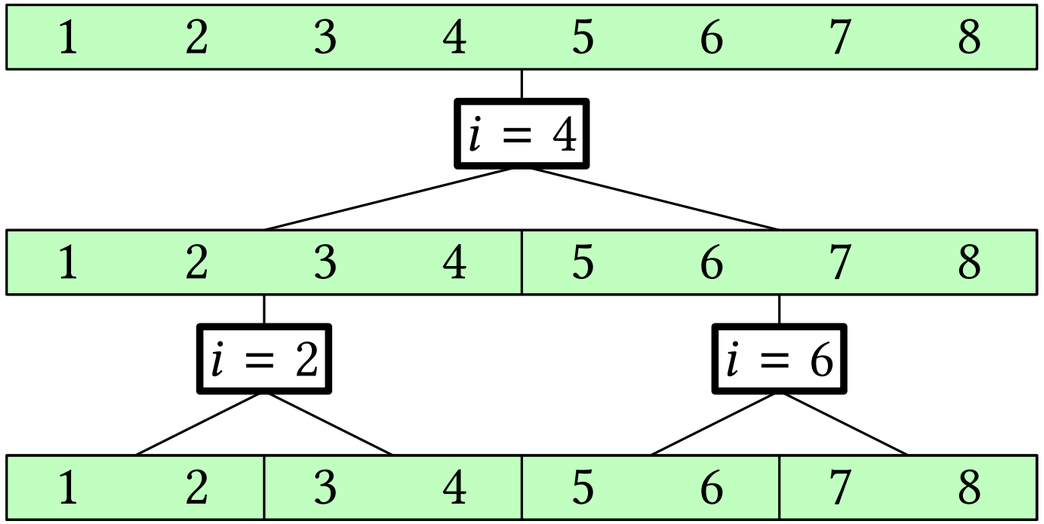
$o = 1 \Rightarrow s \leq i$ $o = 2 \Rightarrow s > i$



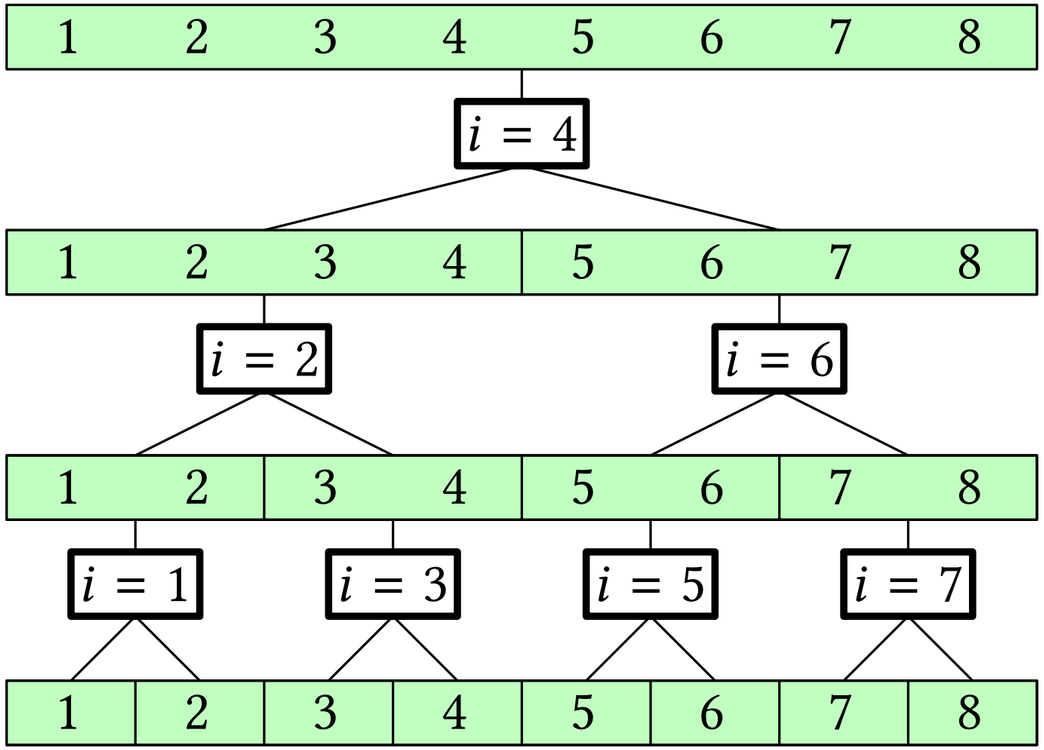
$o = 1 \Rightarrow s \leq i$ $o = 2 \Rightarrow s > i$



$o = 1 \Rightarrow s \leq i$ $o = 2 \Rightarrow s > i$



$o = 1 \Rightarrow s \leq i$ $o = 2 \Rightarrow s > i$



Entropy: Side Channels and Searching

$$\max \mathcal{H} \implies \text{Binary Search}$$

Entropy: Side Channels and Searching

$$\max \mathcal{H} \Rightarrow \text{Binary Search}$$

$$o = 1 \Rightarrow s \leq i \qquad o = 2 \Rightarrow s > i$$

Entropy: Side Channels and Searching

$$\max \mathcal{H} \Rightarrow \text{Binary Search}$$

$$o = 1 \Rightarrow s \leq i \qquad o = 2 \Rightarrow s > i$$

$$\max \mathcal{H} \Rightarrow \text{Optimal Search}$$

any program constraints

Entropy: Side Channels and Searching

$$\max \mathcal{H} \Rightarrow \text{Binary Search}$$

$$o = 1 \Rightarrow s \leq i \qquad o = 2 \Rightarrow s > i$$

$$\max \mathcal{H} \Rightarrow \text{Optimal Search}$$

any program constraints

How to maximize $\mathcal{H}$?

Overall Approach [CSF 2017]

$P(s, i)$

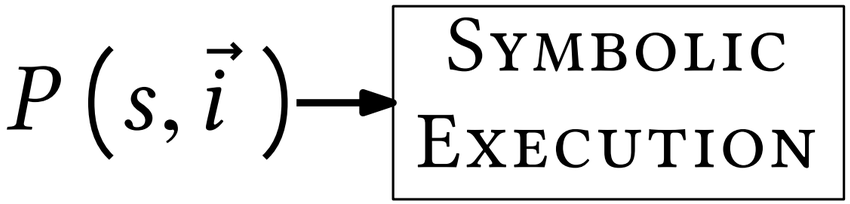
Overall Approach [CSF 2017]

$$P\left(s, \vec{i}\right)$$

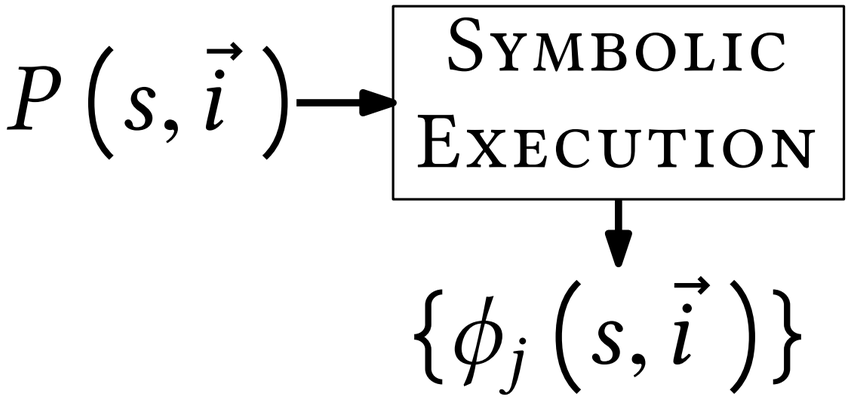
Overall Approach [CSF 2017]

$$P\left(s, \vec{i}\right)$$

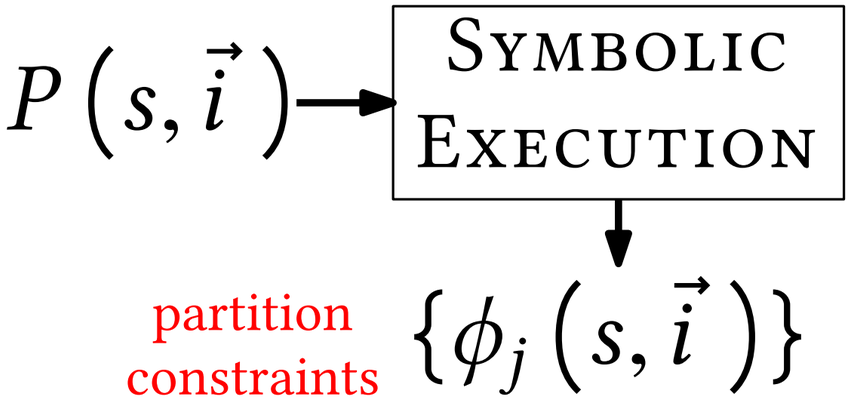
Overall Approach [CSF 2017]



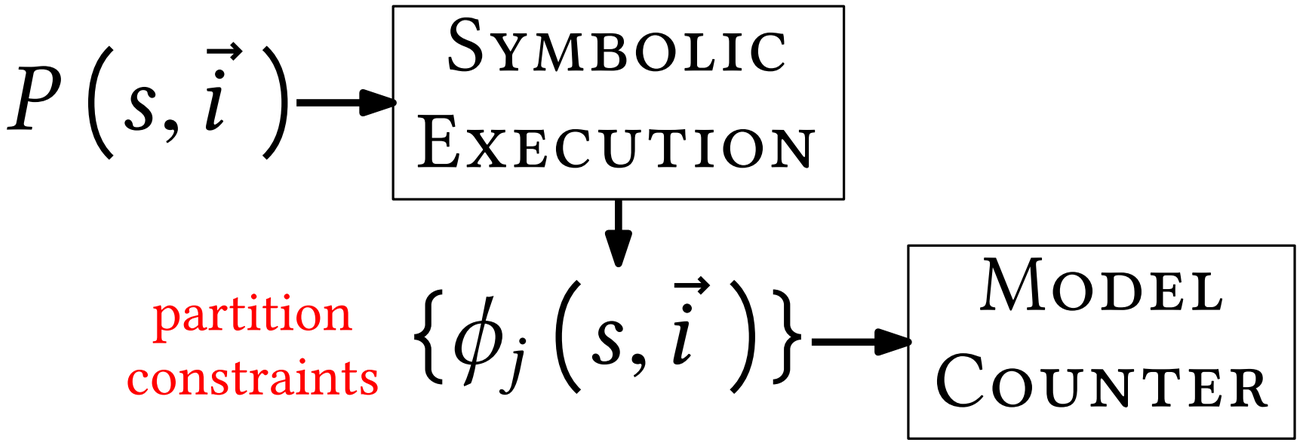
Overall Approach [CSF 2017]



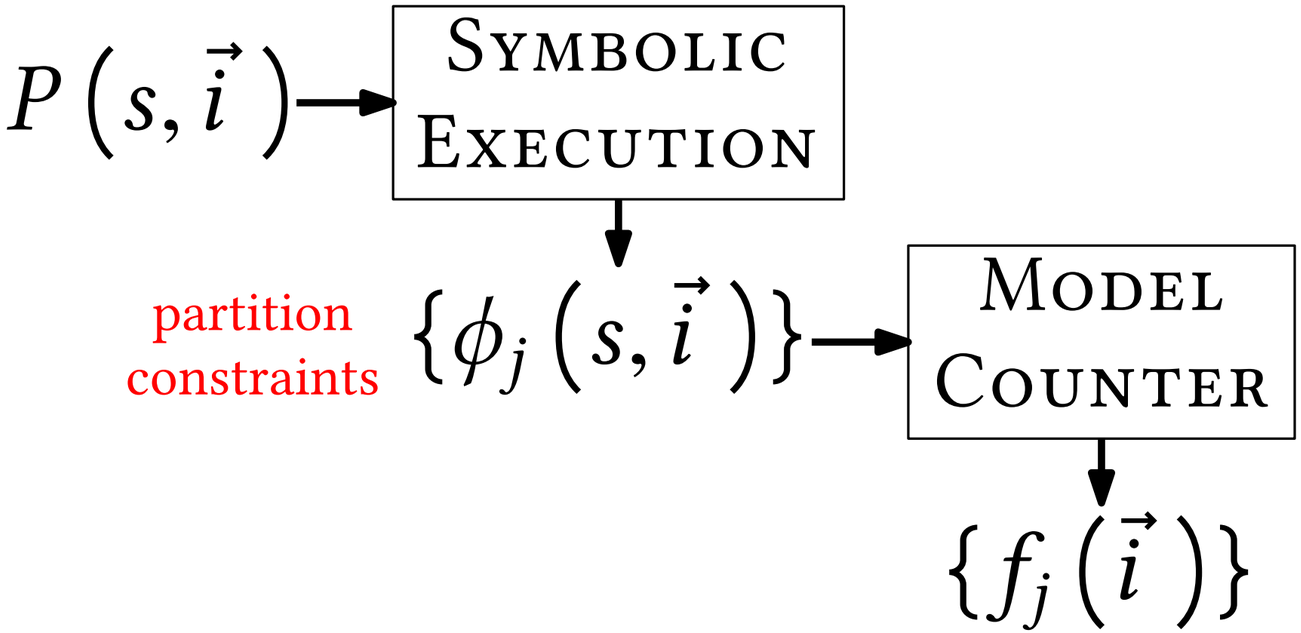
Overall Approach [CSF 2017]



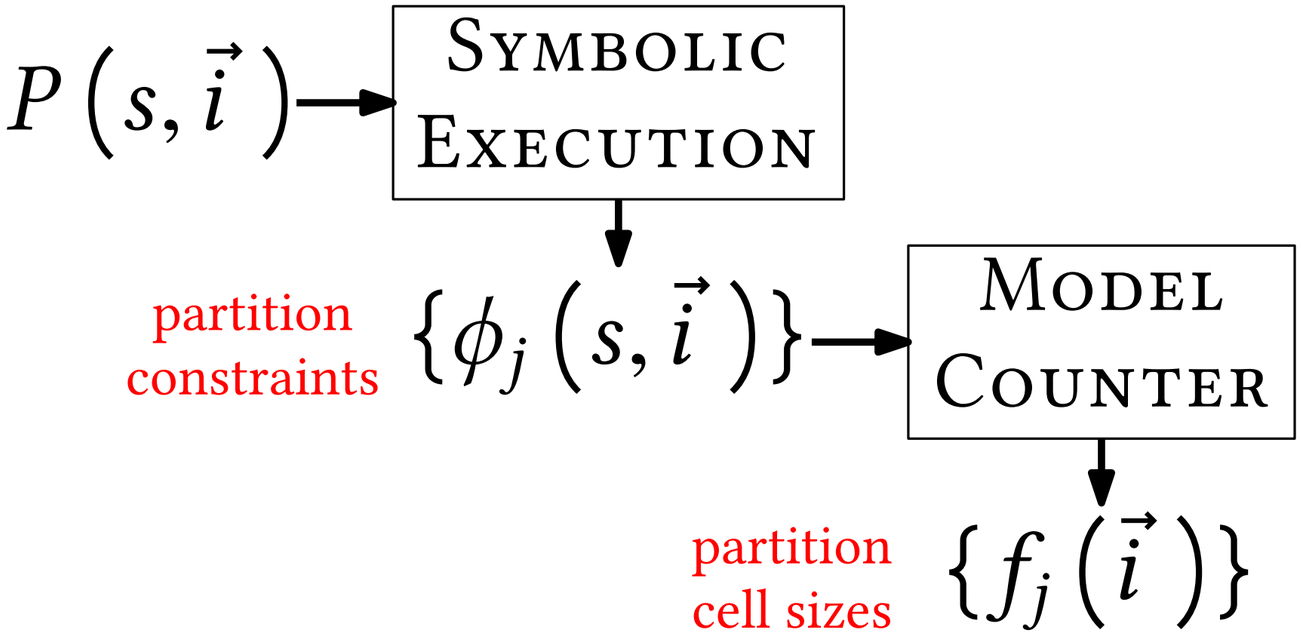
Overall Approach [CSF 2017]



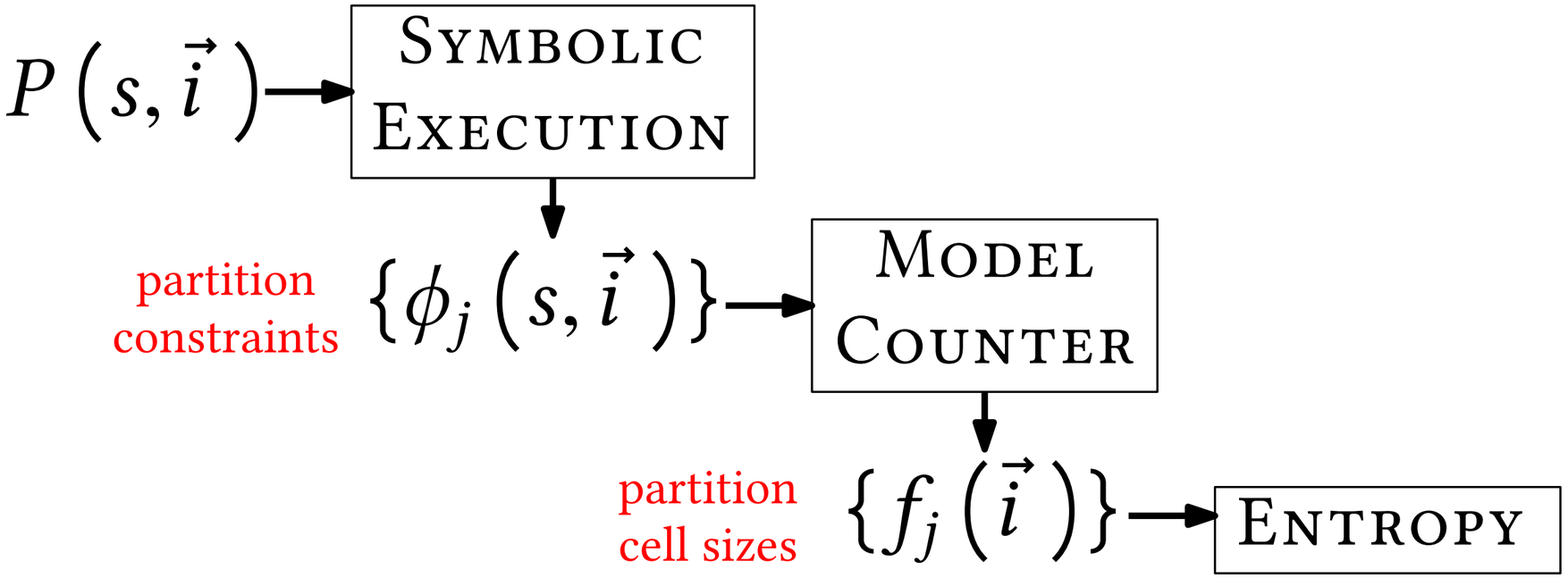
Overall Approach [CSF 2017]



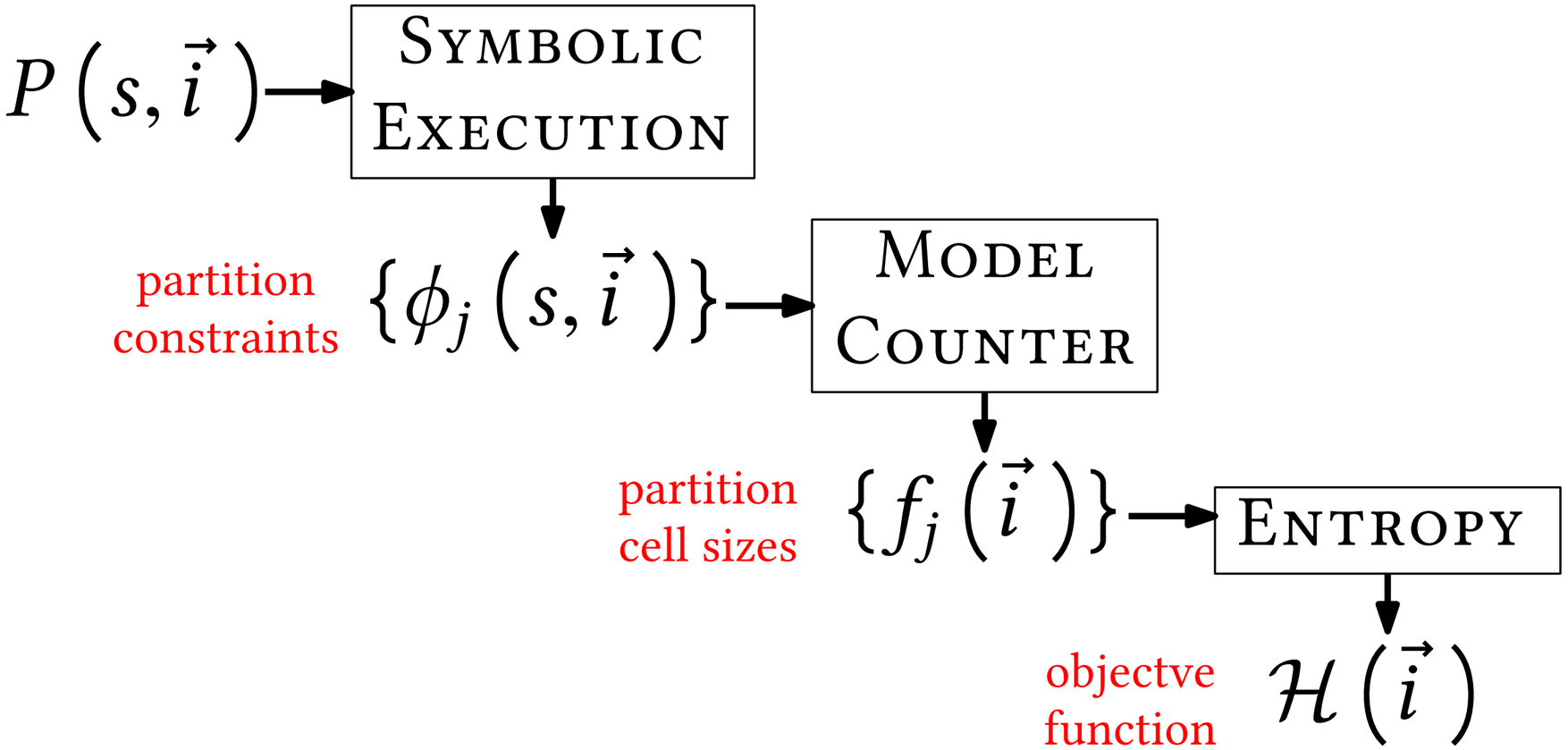
Overall Approach [CSF 2017]



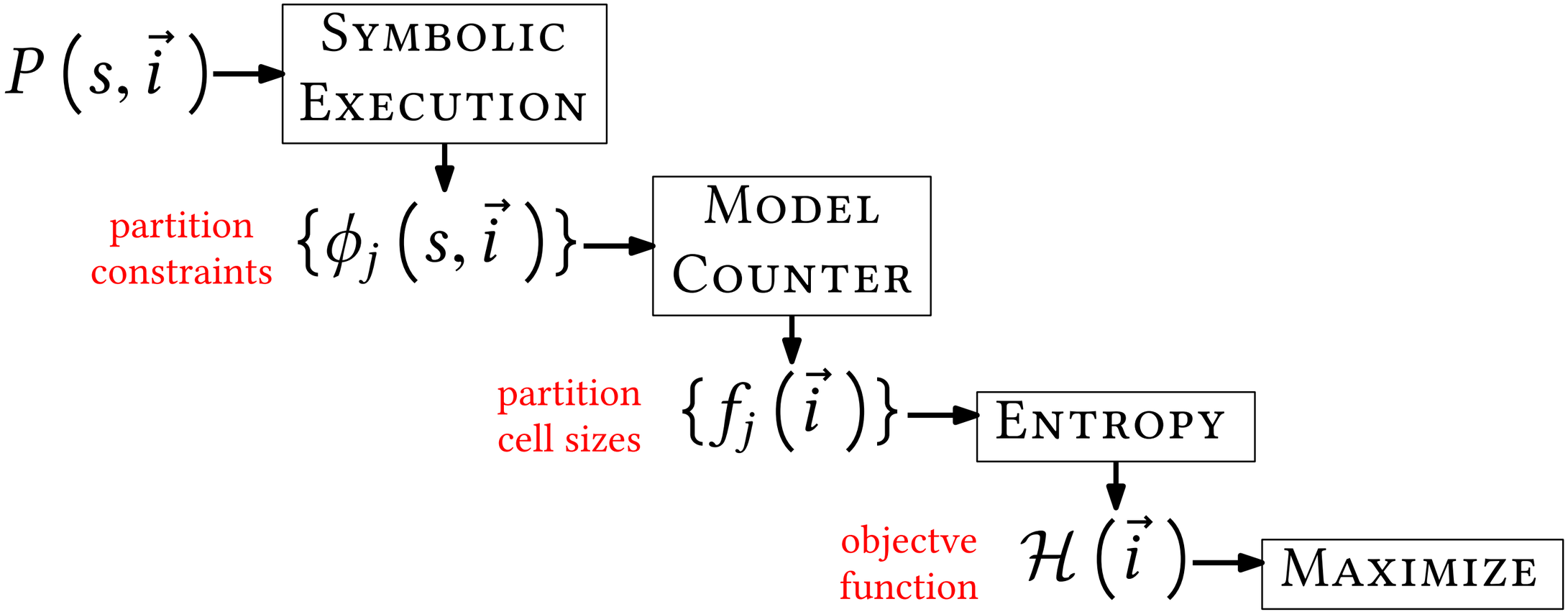
Overall Approach [CSF 2017]



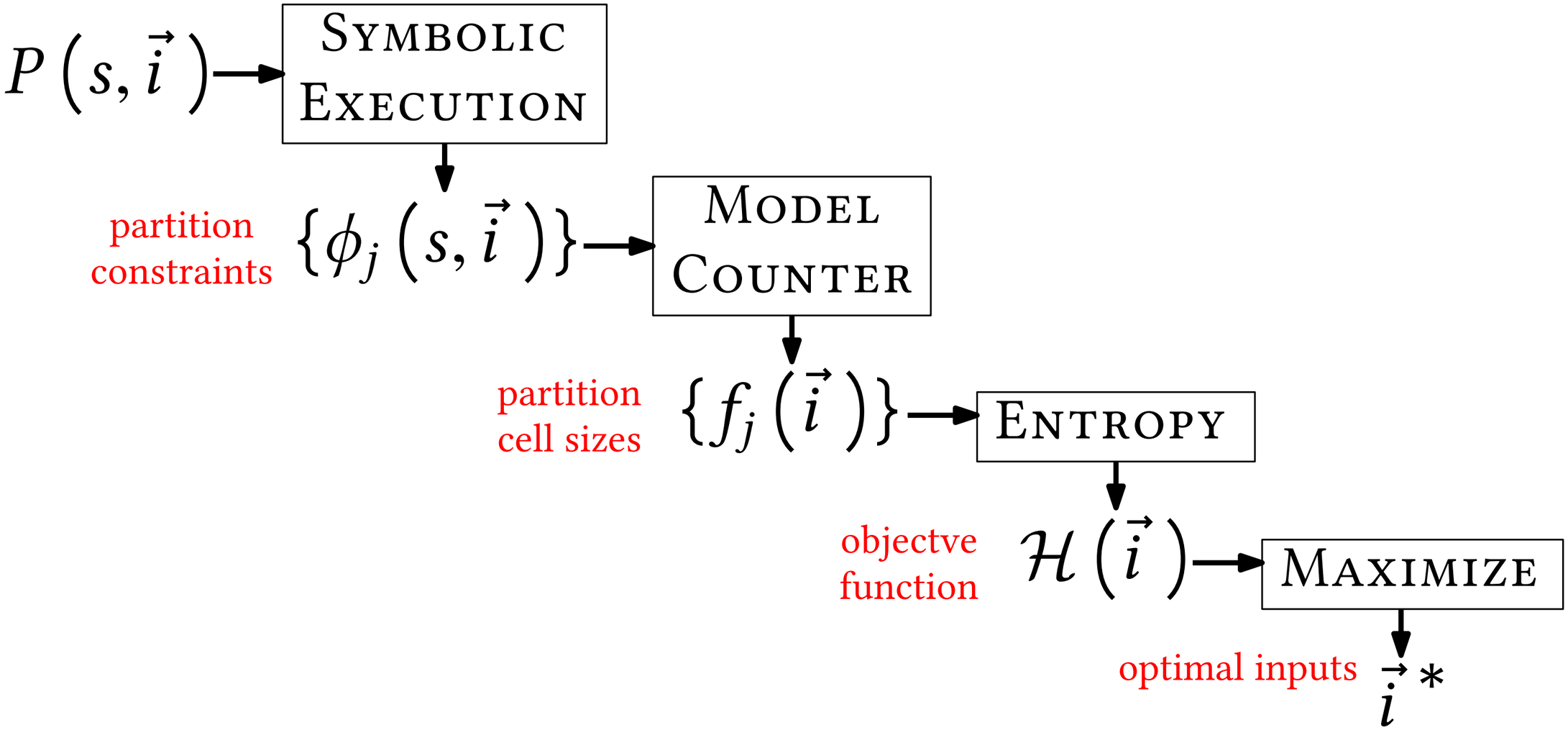
Overall Approach [CSF 2017]



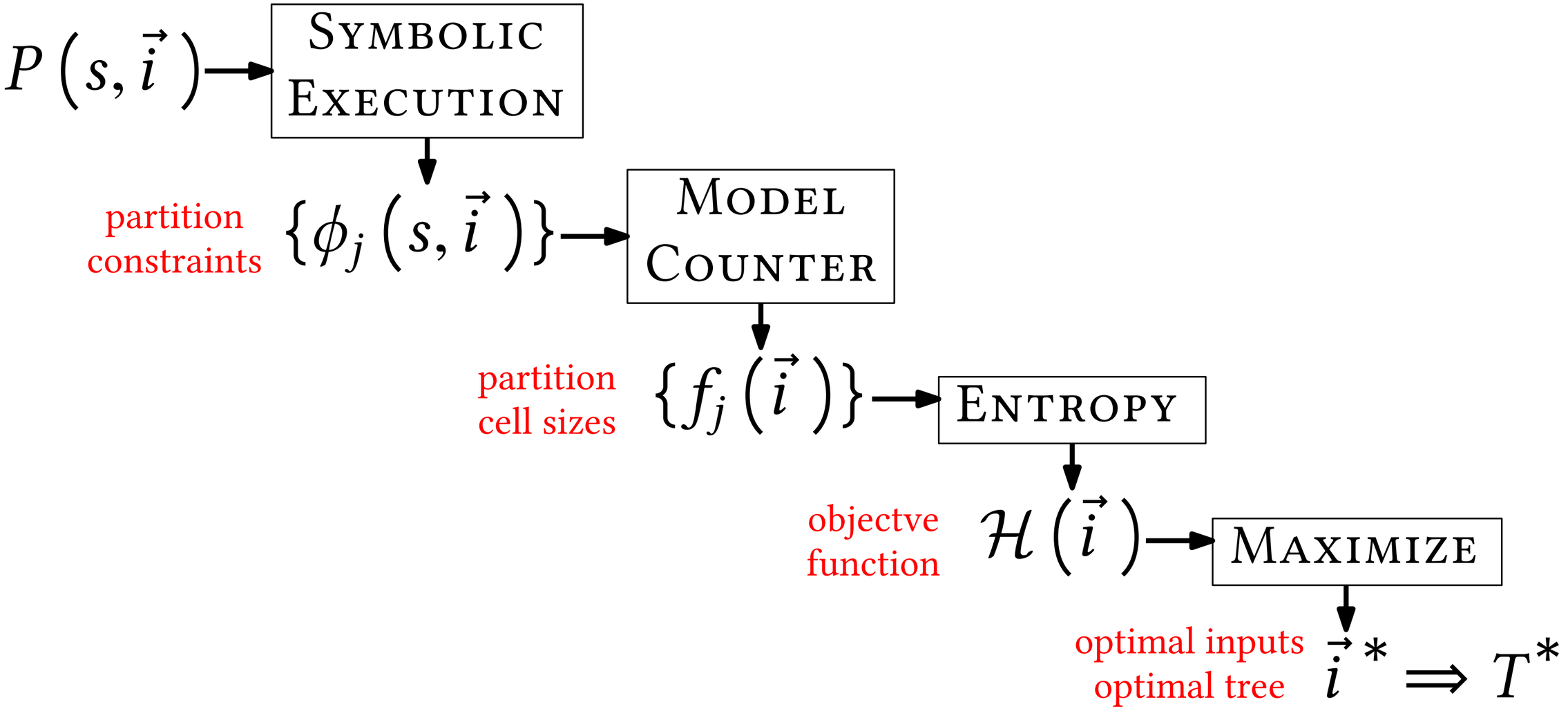
Overall Approach [CSF 2017]



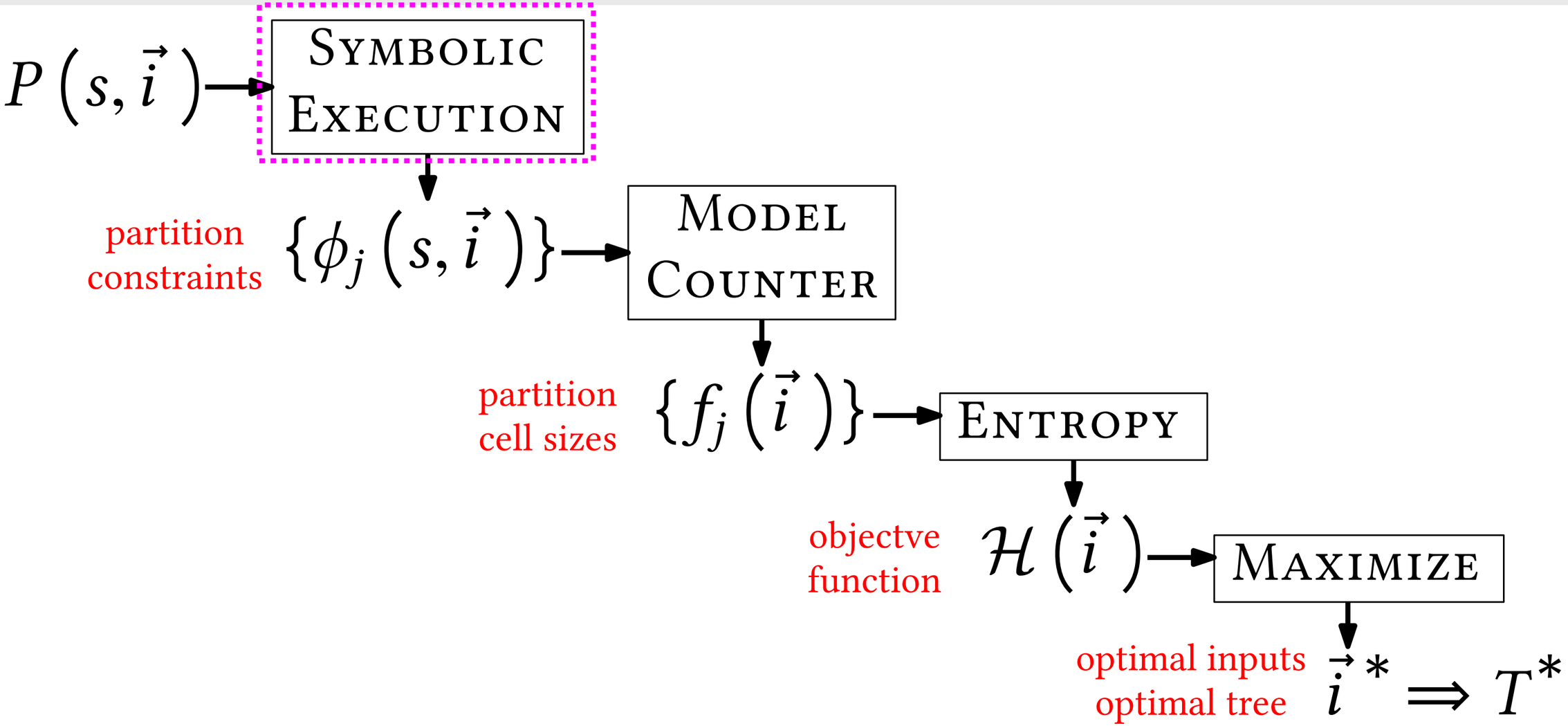
Overall Approach [CSF 2017]



Overall Approach [CSF 2017]



Overall Approach [CSF 2017]



Symbolic Execution

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:

```
if(c) then s1; else s2;
```

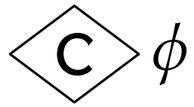
Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:


```
if(c) then s1; else s2;
```

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:



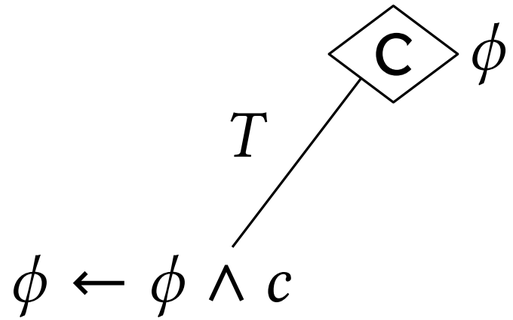
ϕ

```
if(c) then s1; else s2;
```

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:

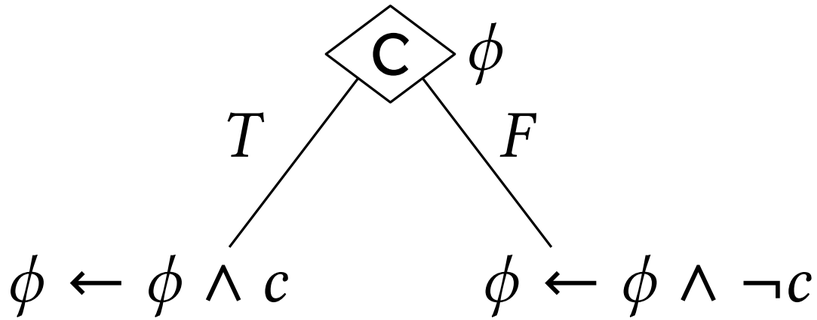
```
if(c) then s1; else s2;
```



Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:

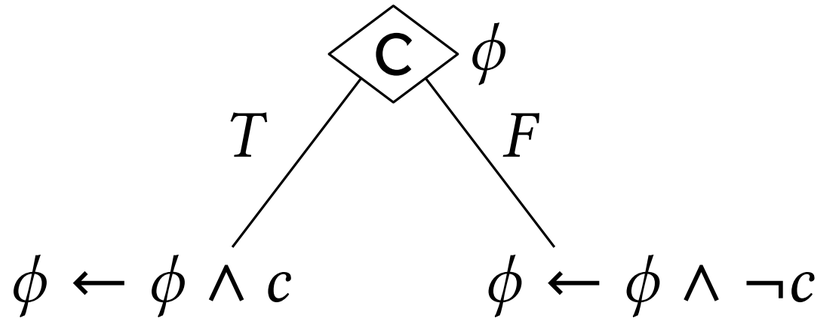
```
if(c) then s1; else s2;
```



Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:

```
if(c) then s1; else s2;
```

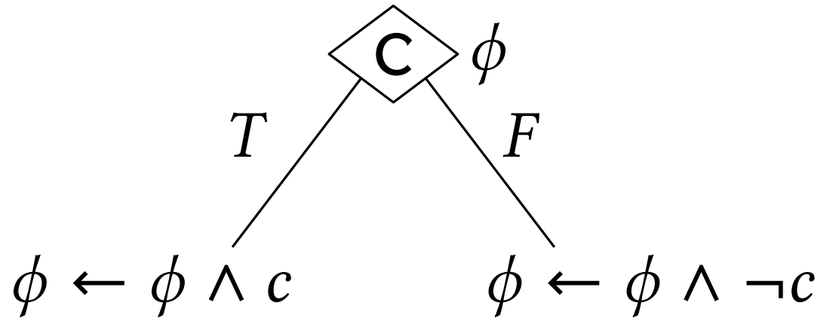


- Check satisfiability of ϕ using constraint solvers (like Z3).

Symbolic Execution

- Execute program on **symbolic** rather than concrete inputs.
- Maintain **path constraints**, PCs, ϕ_j over symbolic inputs.
- For branch instructions:

```
if(c) then s1; else s2;
```



- Check satisfiability of ϕ using constraint solvers (like Z3).
- Maintain **cost model** for every path constraint.

Symbolic Attack Tree via Symbolic Execution

Symbolic Attack Tree via Symbolic Execution

$$o = 1 \Longrightarrow s \leq i \qquad o = 2 \Longrightarrow s > i$$

Symbolic Attack Tree via Symbolic Execution

$o = 1 \Rightarrow s \leq i$ $\qquad$ $o = 2 \Rightarrow s > i$

$\boxed{i = i_0}$ 

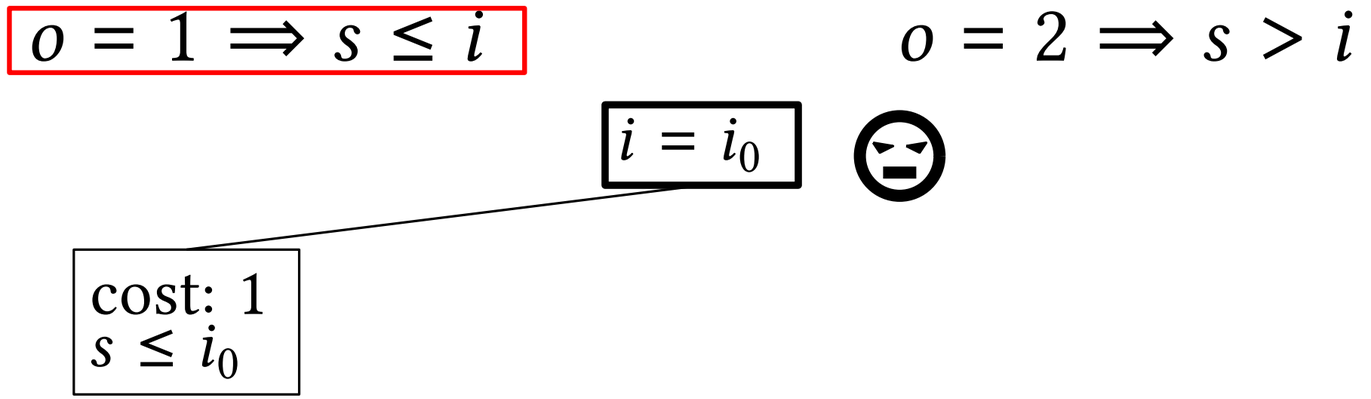
Symbolic Attack Tree via Symbolic Execution

$o = 1 \Longrightarrow s \leq i$

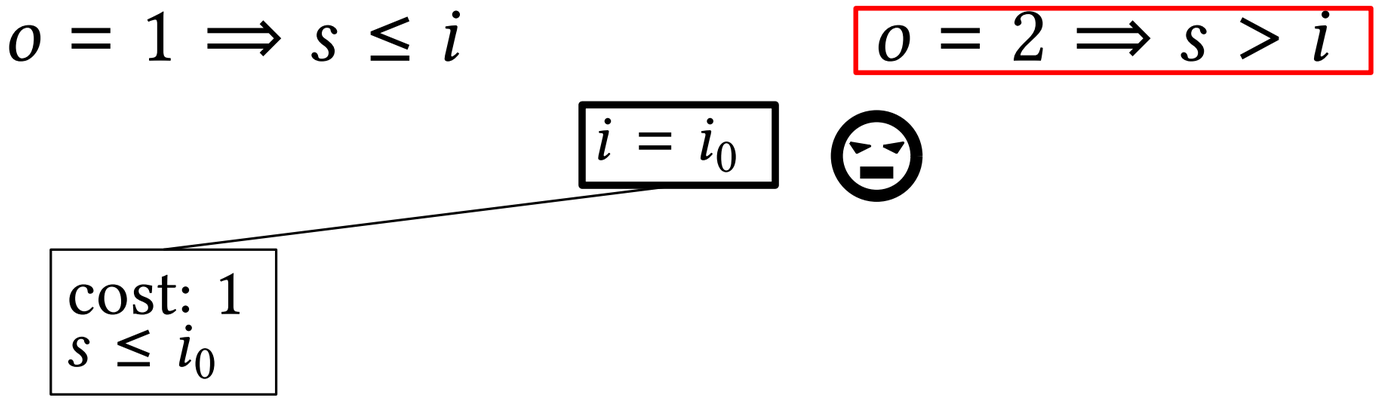
$o = 2 \Longrightarrow s > i$

$i = i_0$

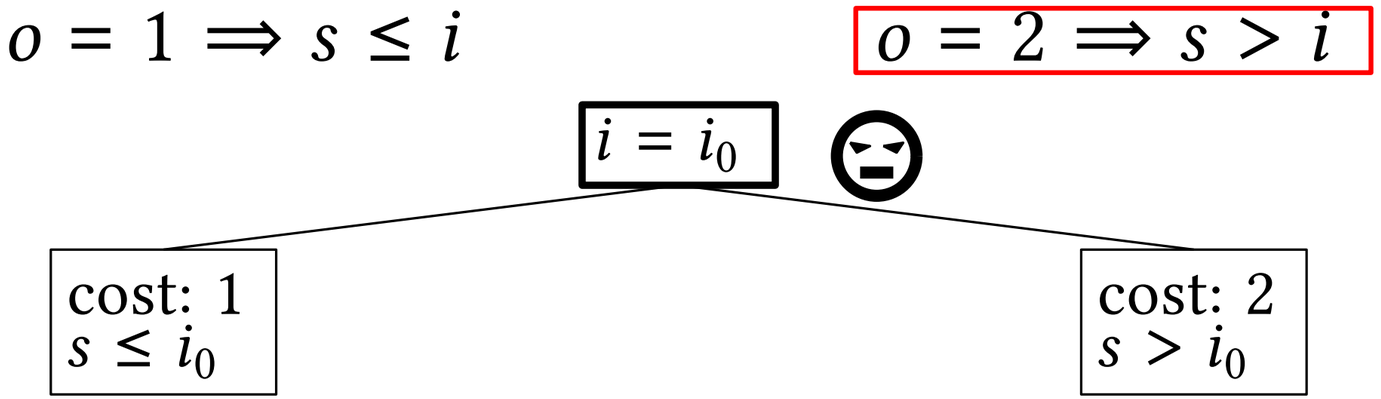
Symbolic Attack Tree via Symbolic Execution



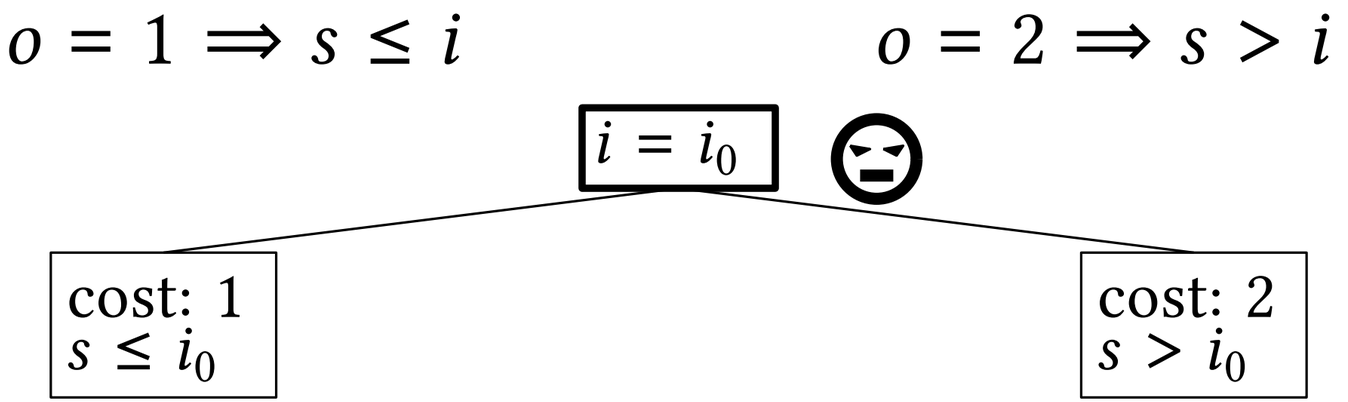
Symbolic Attack Tree via Symbolic Execution



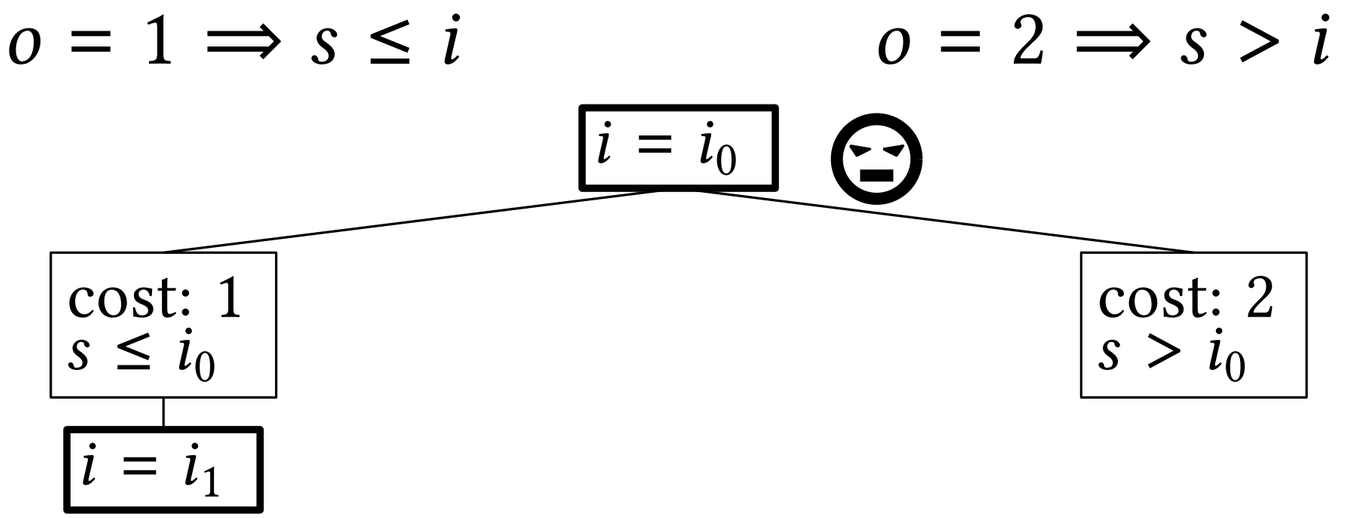
Symbolic Attack Tree via Symbolic Execution



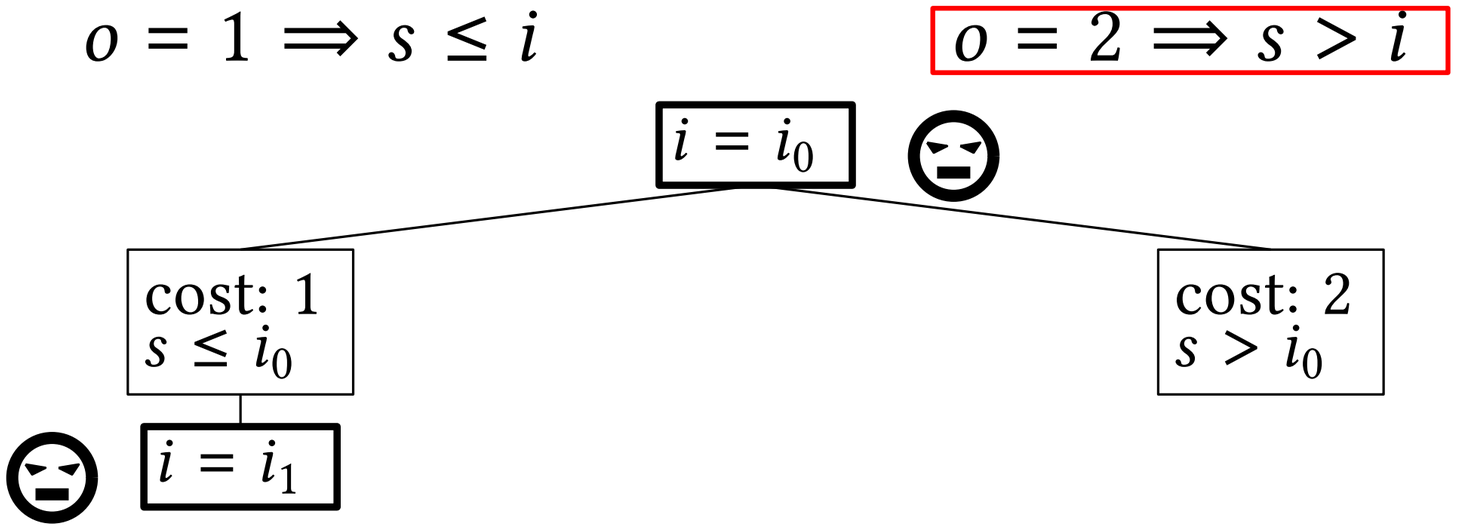
Symbolic Attack Tree via Symbolic Execution



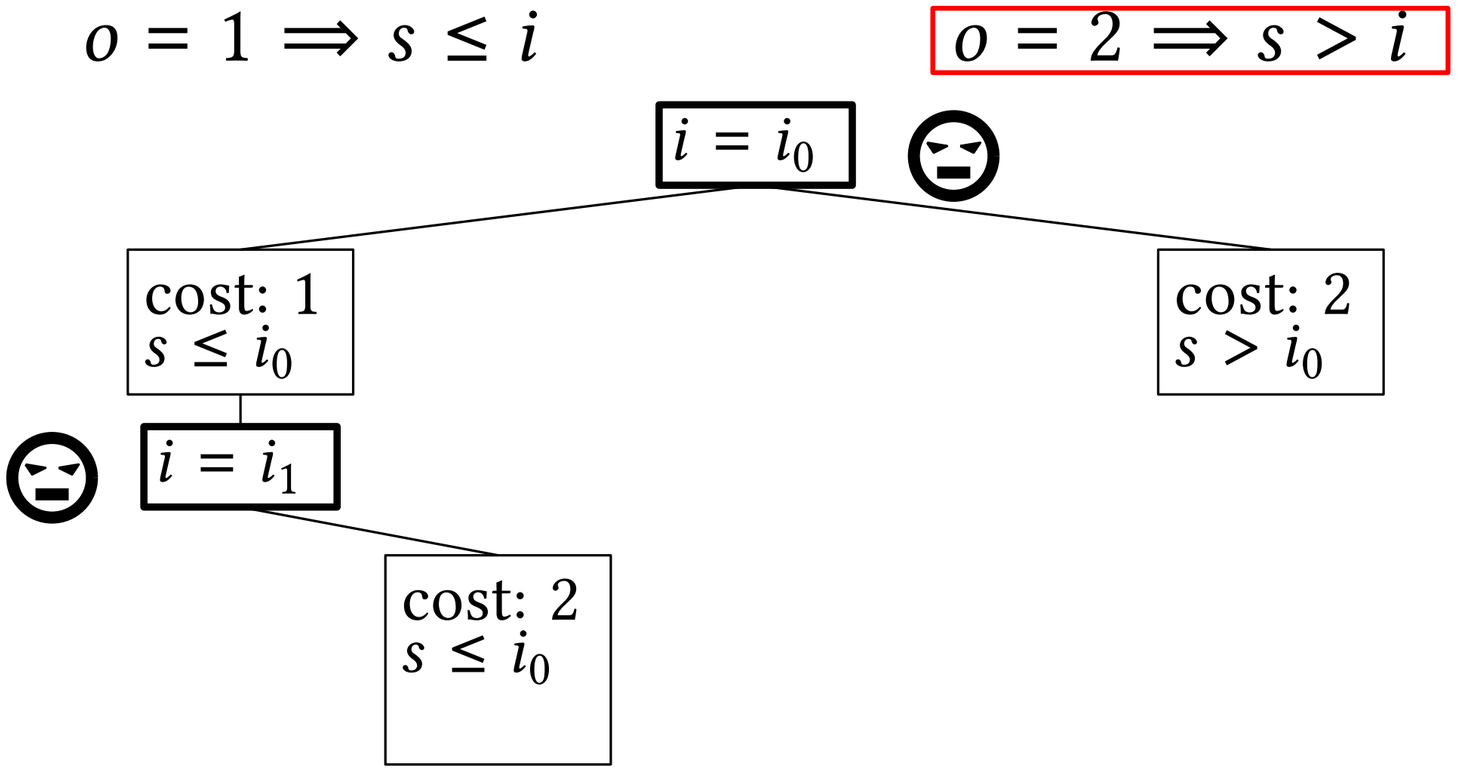
Symbolic Attack Tree via Symbolic Execution



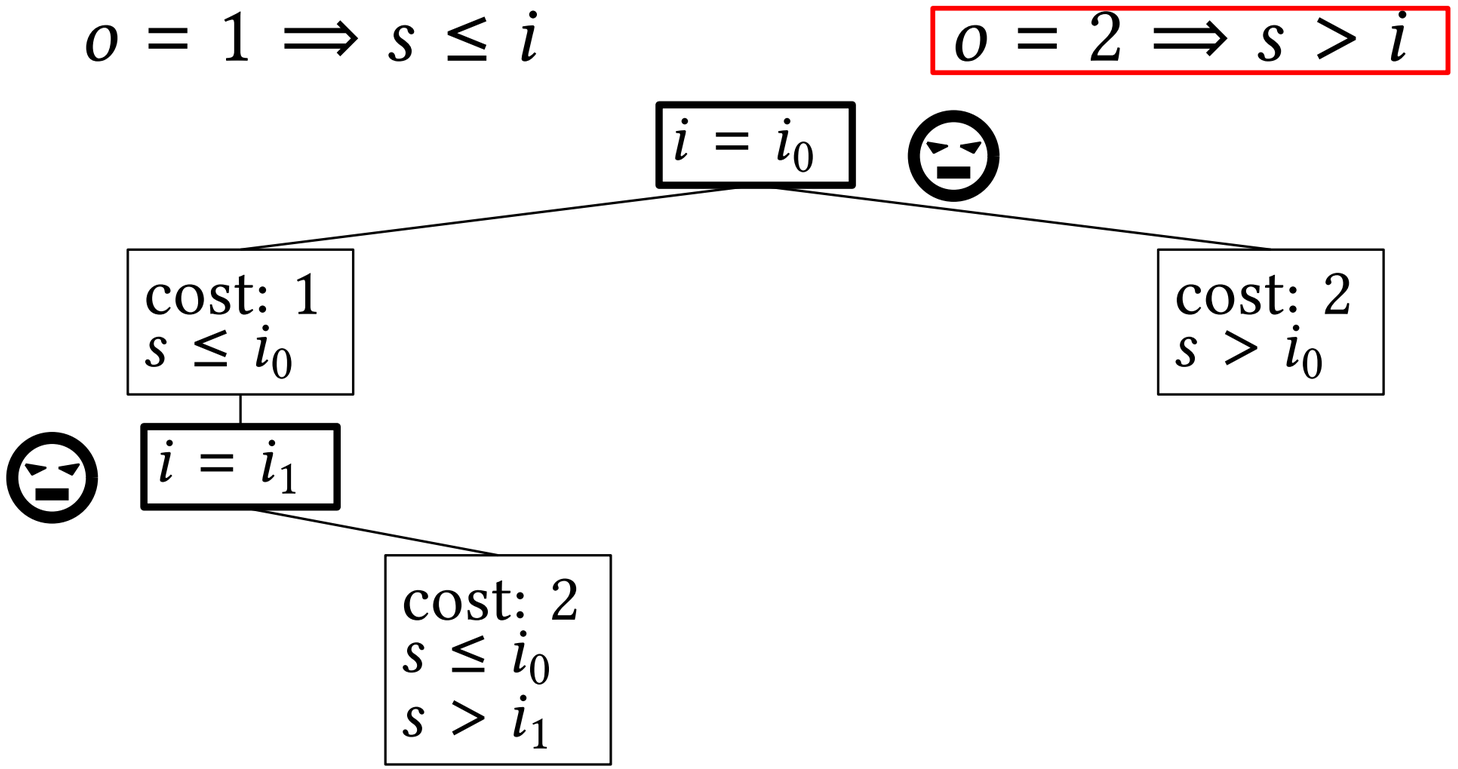
Symbolic Attack Tree via Symbolic Execution



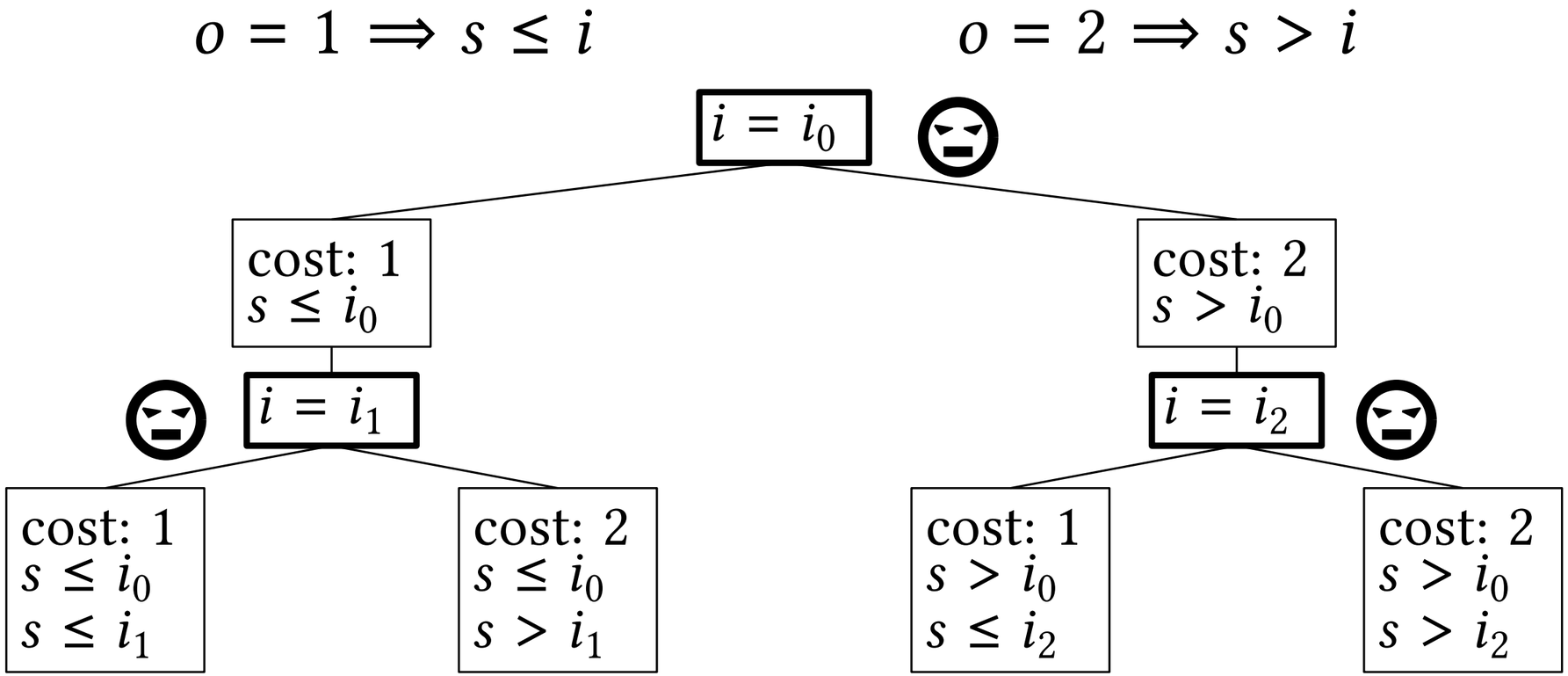
Symbolic Attack Tree via Symbolic Execution



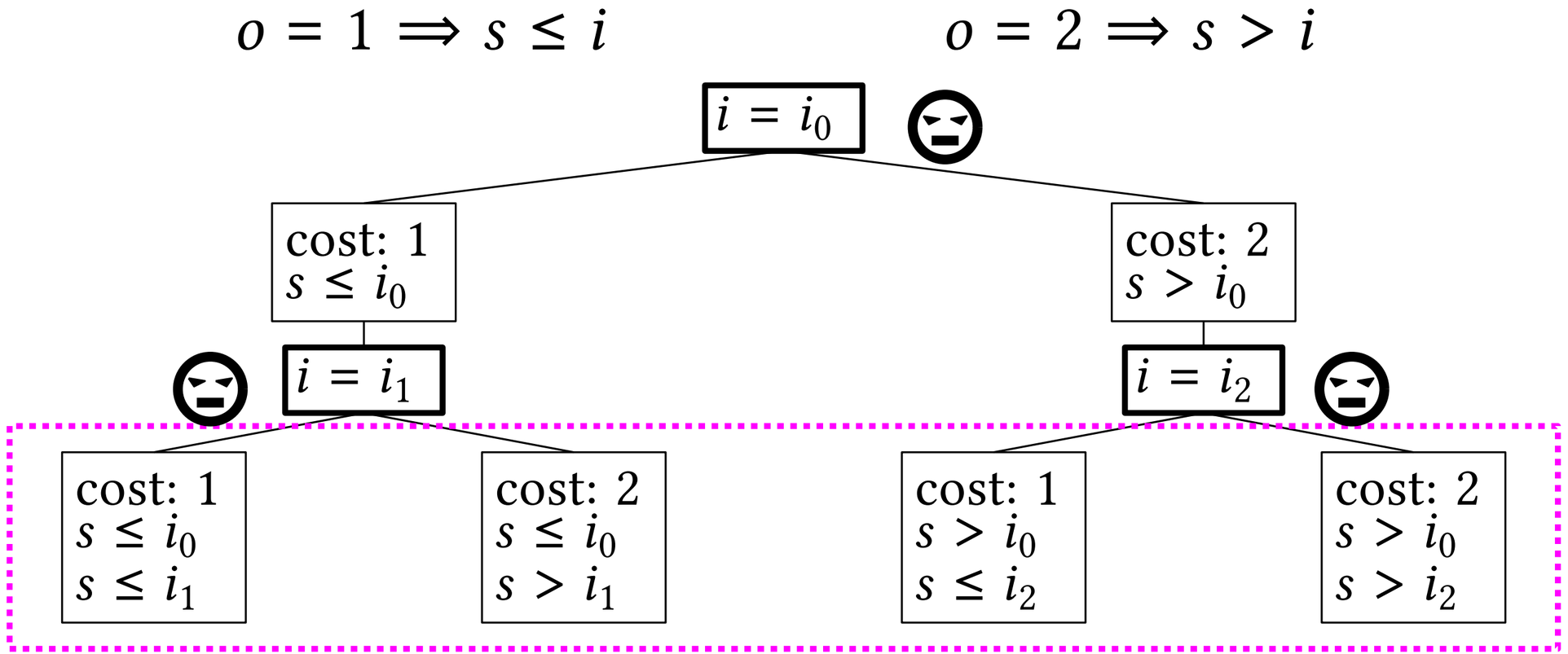
Symbolic Attack Tree via Symbolic Execution



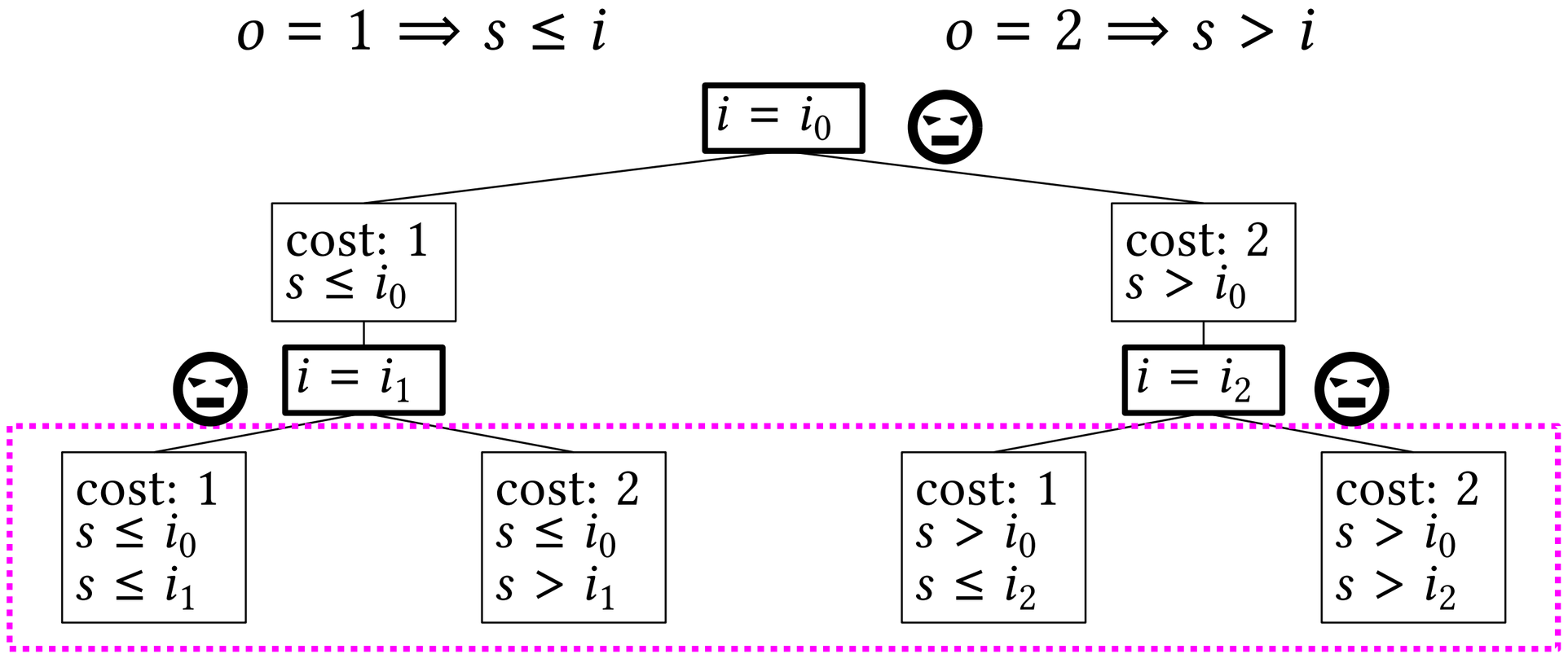
Symbolic Attack Tree via Symbolic Execution



Symbolic Attack Tree via Symbolic Execution

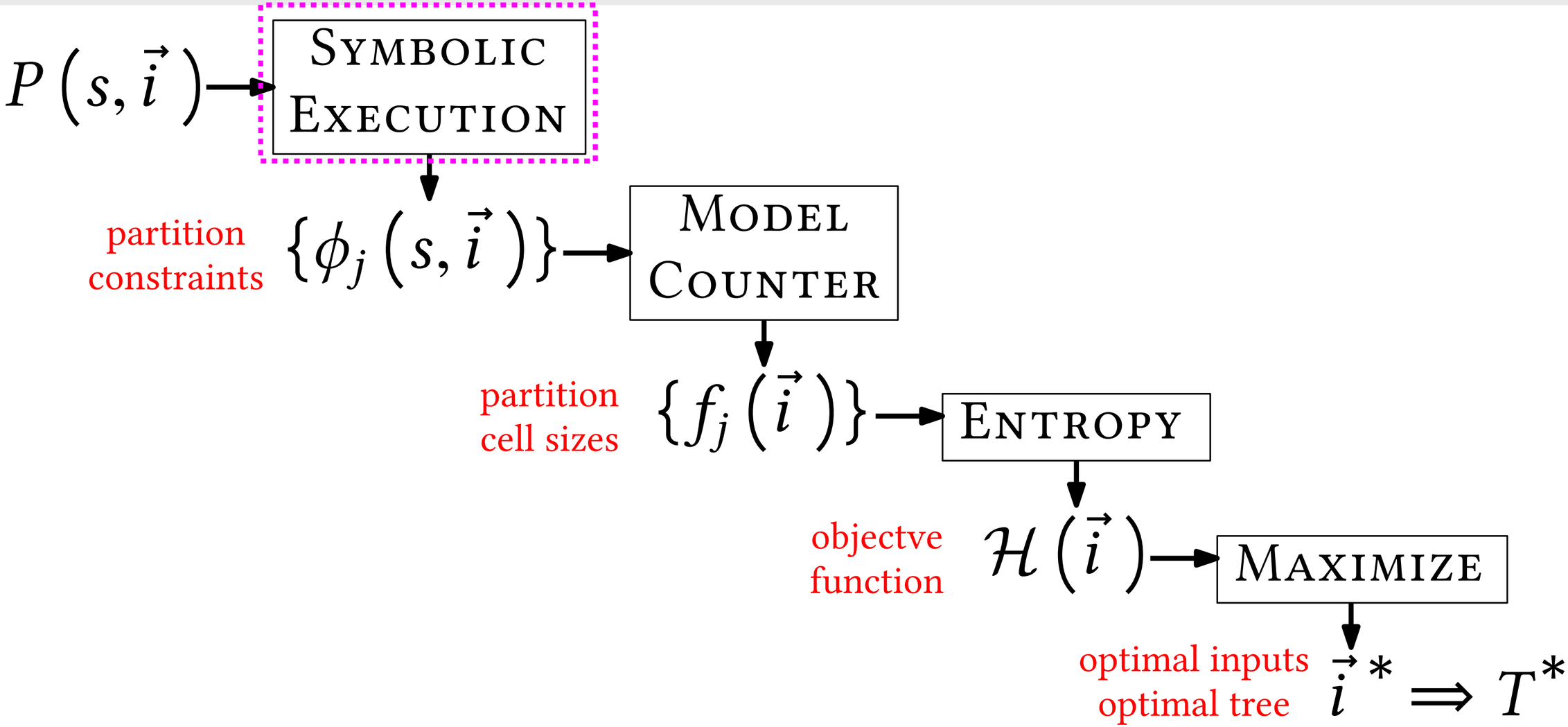


Symbolic Attack Tree via Symbolic Execution

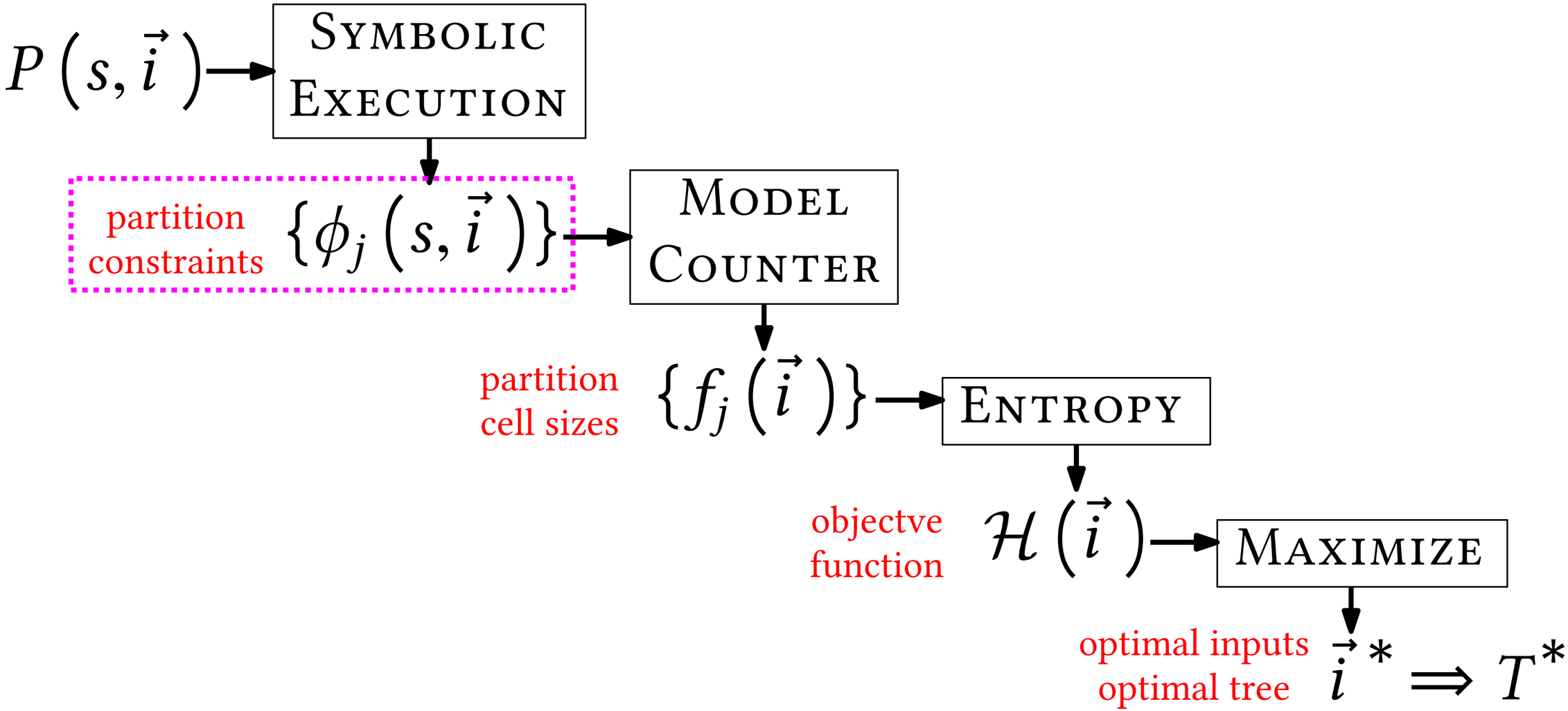


Set of leaf constraints define a **symbolic partition**.

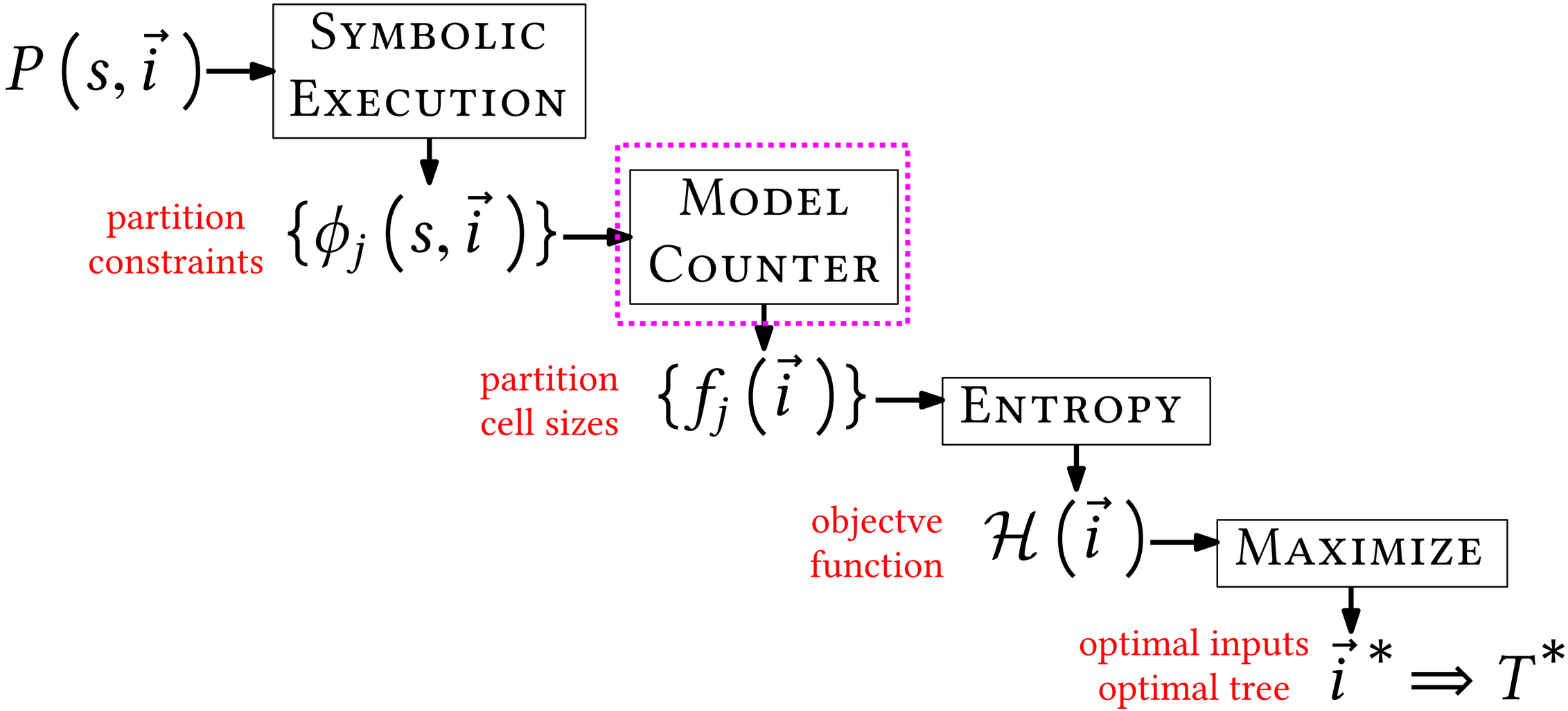
Overall Approach



Overall Approach

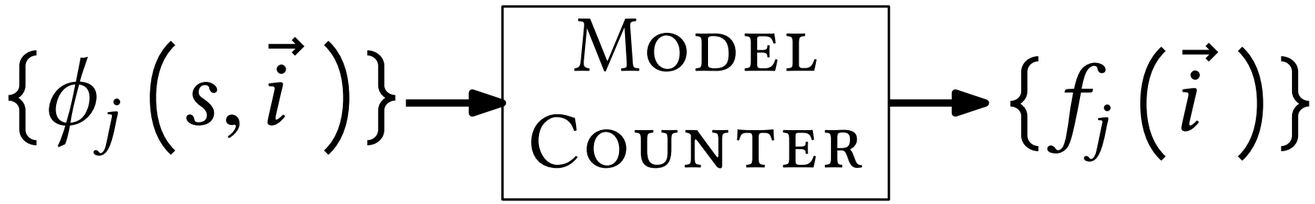


Overall Approach

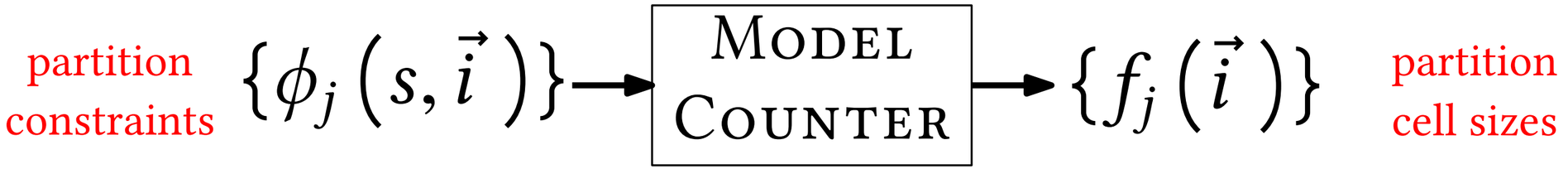


Partition Cell Sizes via Symbolic Model Counting

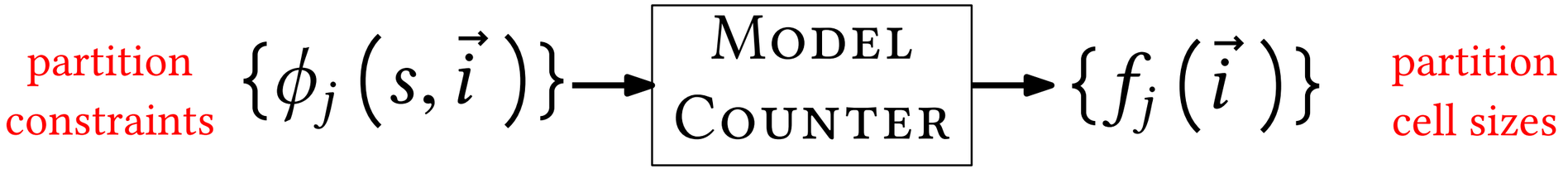
Partition Cell Sizes via Symbolic Model Counting



Partition Cell Sizes via Symbolic Model Counting

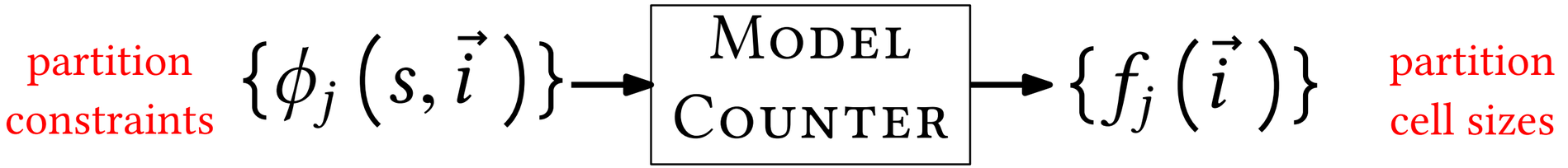


Partition Cell Sizes via Symbolic Model Counting



$f_j(\vec{i})$: size of partition cell j

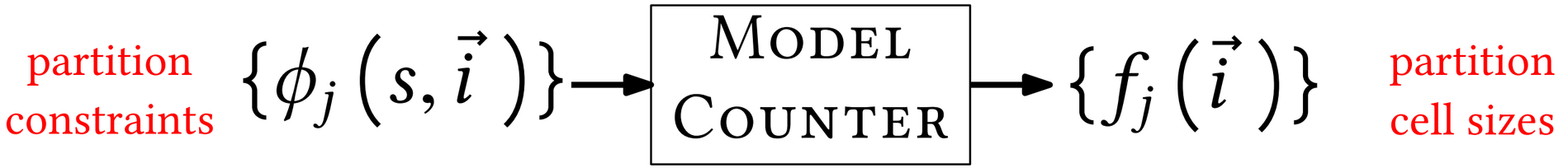
Partition Cell Sizes via Symbolic Model Counting



$$f_j(\vec{i}) : \text{size of partition cell } j$$

$$|\text{partition cell j}| = \#\text{ satisfying solutions (models) for } \phi(s, \vec{i})$$

Partition Cell Sizes via Symbolic Model Counting



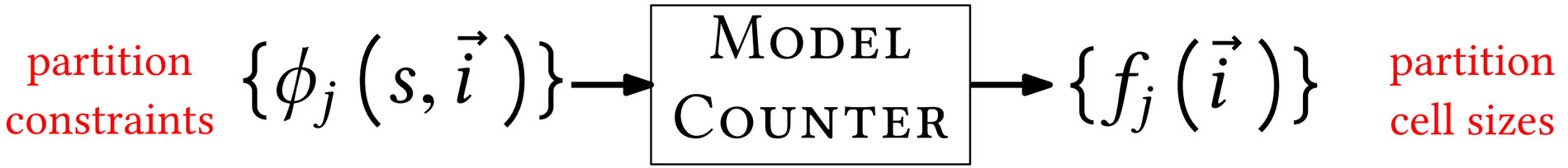
$$f_j(\vec{i}) : \text{size of partition cell } j$$

$$|\text{partition cell j}| = \#\text{ satisfying solutions (models) for } \phi(s, \vec{i})$$

Model Counting Constraint Solvers:

Barvinok: Linear Integer Arithmetic

Partition Cell Sizes via Symbolic Model Counting



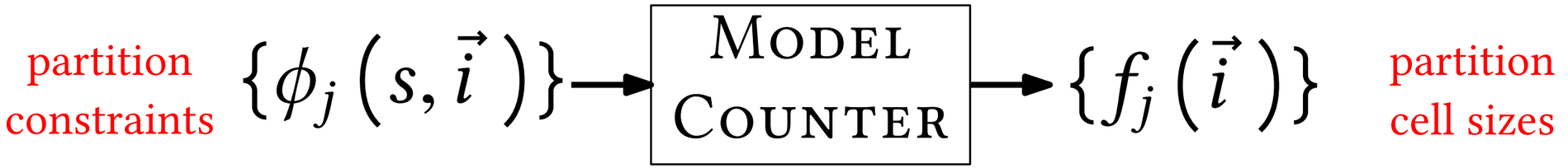
$$f_j(\vec{i}) : \text{size of partition cell } j$$

$$|\text{partition cell j}| = \#\text{ satisfying solutions (models) for } \phi(s, \vec{i})$$

Model Counting Constraint Solvers:

Barvinok: Linear Integer Arithmetic
ABC: Linear Integer Arithmetic + Strings [CAV '15]

Partition Cell Sizes via Symbolic Model Counting



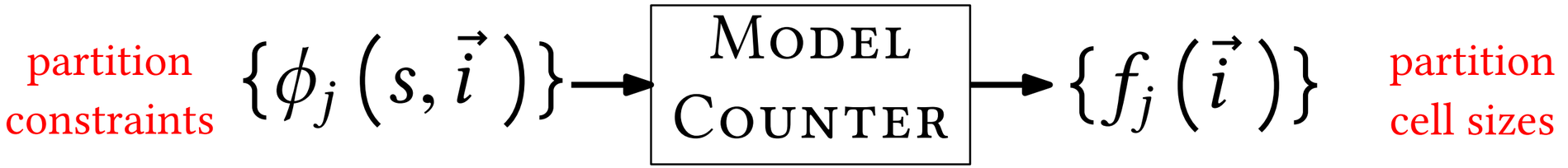
$$f_j(\vec{i}) : \text{size of partition cell } j$$

$$|\text{partition cell j}| = \#\text{ satisfying solutions (models) for } \phi(s, \vec{i})$$

Model Counting Constraint Solvers:

Barvinok: Linear Integer Arithmetic
ABC: Linear Integer Arithmetic + Strings [CAV '15]
SMC: Strings
LattE: Linear Integer Arithmetic

Partition Cell Sizes via Symbolic Model Counting



$$f_j(\vec{i}) : \text{size of partition cell } j$$

$$|\text{partition cell j}| = \#\text{ satisfying solutions (models) for } \phi(s, \vec{i})$$

Model Counting Constraint Solvers:

- Barvinok: Linear Integer Arithmetic
- ABC: Linear Integer Arithmetic + Strings [CAV '15]
- SMC: Strings
- LattE: Linear Integer Arithmetic

Partition Cell Sizes via Symbolic Model Counting

Partition Cell Sizes via Symbolic Model Counting

cost: 1
$s \leq i_0$
$s \leq i_1$

cost: 2
$s \leq i_0$
$s > i_1$

cost: 1
$s > i_0$
$s \leq i_2$

cost: 2
$s > i_0$
$s\ i_2$

Partition Cell Sizes via Symbolic Model Counting

cost: 1
$s \leq i_0$
$s \leq i_1$

cost: 2
$s \leq i_0$
$s > i_1$

cost: 1
$s > i_0$
$s \leq i_2$

cost: 2
$s > i_0$
$s\ i_2$

Partition Cell Sizes via Symbolic Model Counting

$$s \leq i_0$$
$$s > i_1$$

Partition Cell Sizes via Symbolic Model Counting

$$s \leq i_0$$
$$s > i_1$$

$\text{Barvinok}\left(\phi_j\left(s, \vec{i}\right), \vec{i}\right)$: piecewise polynomial function $f_j\left(\vec{i}\right)$

Partition Cell Sizes via Symbolic Model Counting

$$s \leq i_0$$
$$s > i_1$$

Barvinok$\left(\phi_j\left(s, \vec{i}\right), \vec{i}\right)$: piecewise polynomial function $f_j\left(\vec{i}\right)$

Partition Cell Sizes via Symbolic Model Counting

$$s \leq i_0$$
$$s > i_1$$

Barvinok$\left(\phi_j\left(s, \vec{i}\right), \vec{i}\right)$: piecewise polynomial function $f_j\left(\vec{i}\right)$

$$f\left(i_0, i_1\right) = \begin{cases} i_0 - i_1 & \text{if } 1 \leq i_1 \leq i_0 \leq 8 \\ i_0 & \text{if } i_1 < 1 \leq i_0 \leq 8 \\ 8 - i_1 & \text{if } 1 \leq i_1 \leq 8 \leq i_0 \\ 8 & \text{if } i_1 \leq 1 < 8 \leq i_0 \\ 0 & \text{otherwise} \end{cases}$$

Partition Cell Sizes via Symbolic Model Counting

$$s \leq i_0$$
$$s > i_1$$

Barvinok$\left(\phi_j\left(s, \vec{i}\right), \vec{i}\right)$: piecewise polynomial function $f_j\left(\vec{i}\right)$

$$f\left(i_0, i_1\right) = \begin{cases} i_0 - i_1 & \text{if } 1 \leq i_1 \leq i_0 \leq 8 \\ i_0 & \text{if } i_1 < 1 \leq i_0 \leq 8 \\ 8 - i_1 & \text{if } 1 \leq i_1 \leq 8 \leq i_0 \\ 8 & \text{if } i_1 \leq 1 < 8 \leq i_0 \\ 0 & \text{otherwise} \end{cases}$$

$$f\left(6, 2\right) = 4$$

Partition Cell Sizes via Symbolic Model Counting

$$s \leq i_0$$
$$s > i_1$$

Barvinok$\left(\phi_j\left(s, \vec{i}\right), \vec{i}\right)$: piecewise polynomial function $f_j\left(\vec{i}\right)$

$$f\left(i_0, i_1\right) = \begin{cases} i_0 - i_1 & \text{if } 1 \leq i_1 \leq i_0 \leq 8 \\ i_0 & \text{if } i_1 < 1 \leq i_0 \leq 8 \\ 8 - i_1 & \text{if } 1 \leq i_1 \leq 8 \leq i_0 \\ 8 & \text{if } i_1 \leq 1 < 8 \leq i_0 \\ 0 & \text{otherwise} \end{cases}$$

$f\left(6, 2\right) = 4$ $\qquad$ $f\left(5, -1\right) = 5$

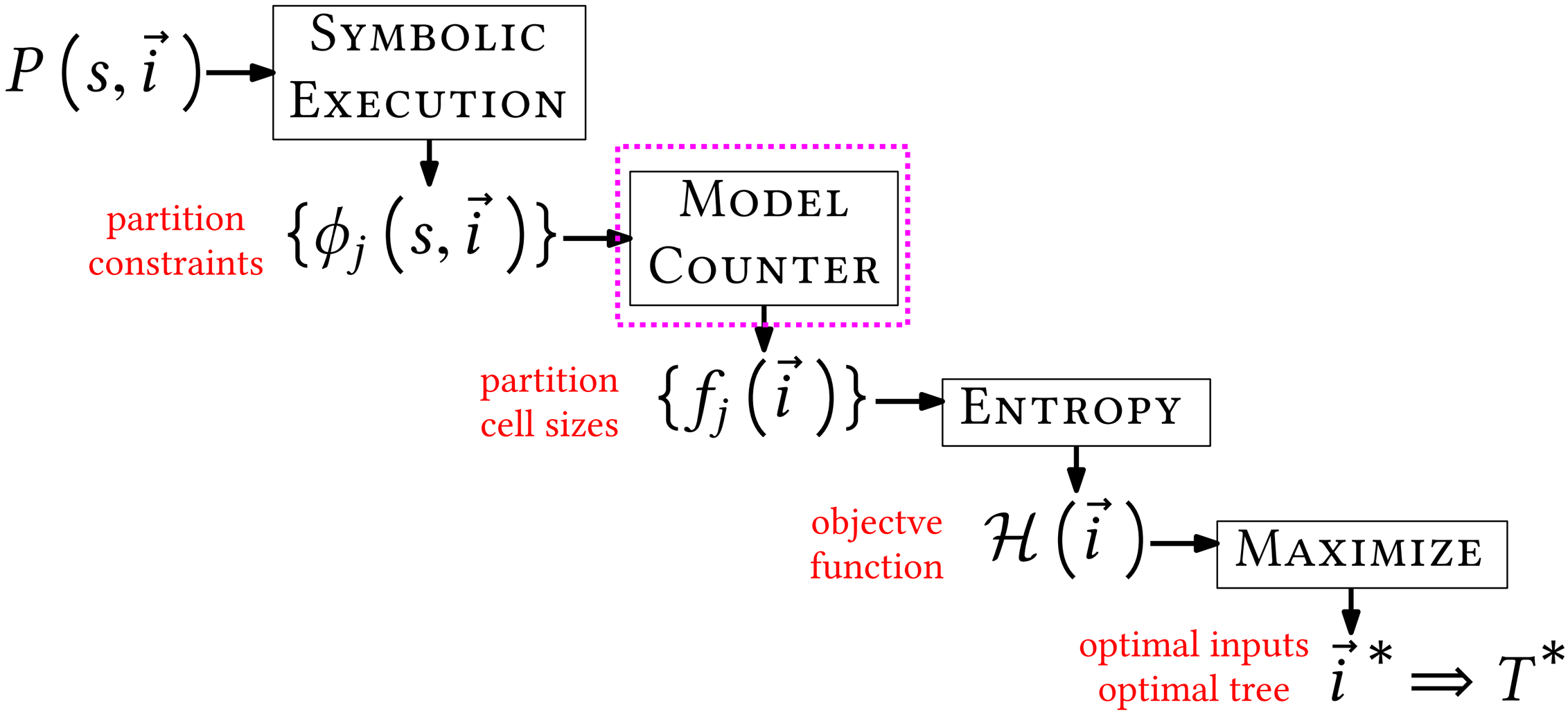
Partition Cell Sizes via Symbolic Model Counting

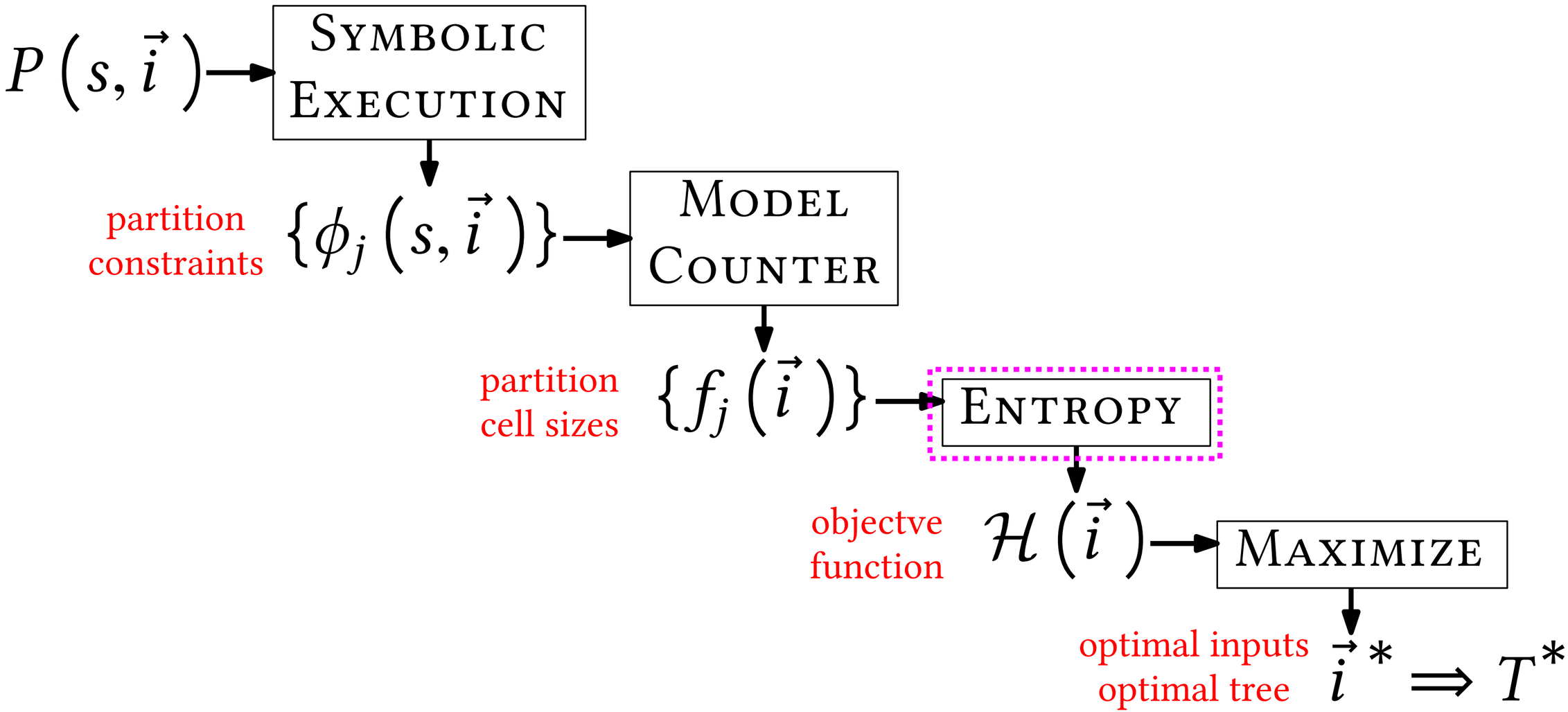
$$s \leq i_0$$
$$s > i_1$$

Barvinok$\left(\phi_j\left(s, \vec{i}\right), \vec{i}\right)$: piecewise polynomial function $f_j\left(\vec{i}\right)$

$$f\left(i_0, i_1\right) = \begin{cases} i_0 - i_1 & \text{if } 1 \leq i_1 \leq i_0 \leq 8 \\ i_0 & \text{if } i_1 < 1 \leq i_0 \leq 8 \\ 8 - i_1 & \text{if } 1 \leq i_1 \leq 8 \leq i_0 \\ 8 & \text{if } i_1 \leq 1 < 8 \leq i_0 \\ 0 & \text{otherwise} \end{cases}$$

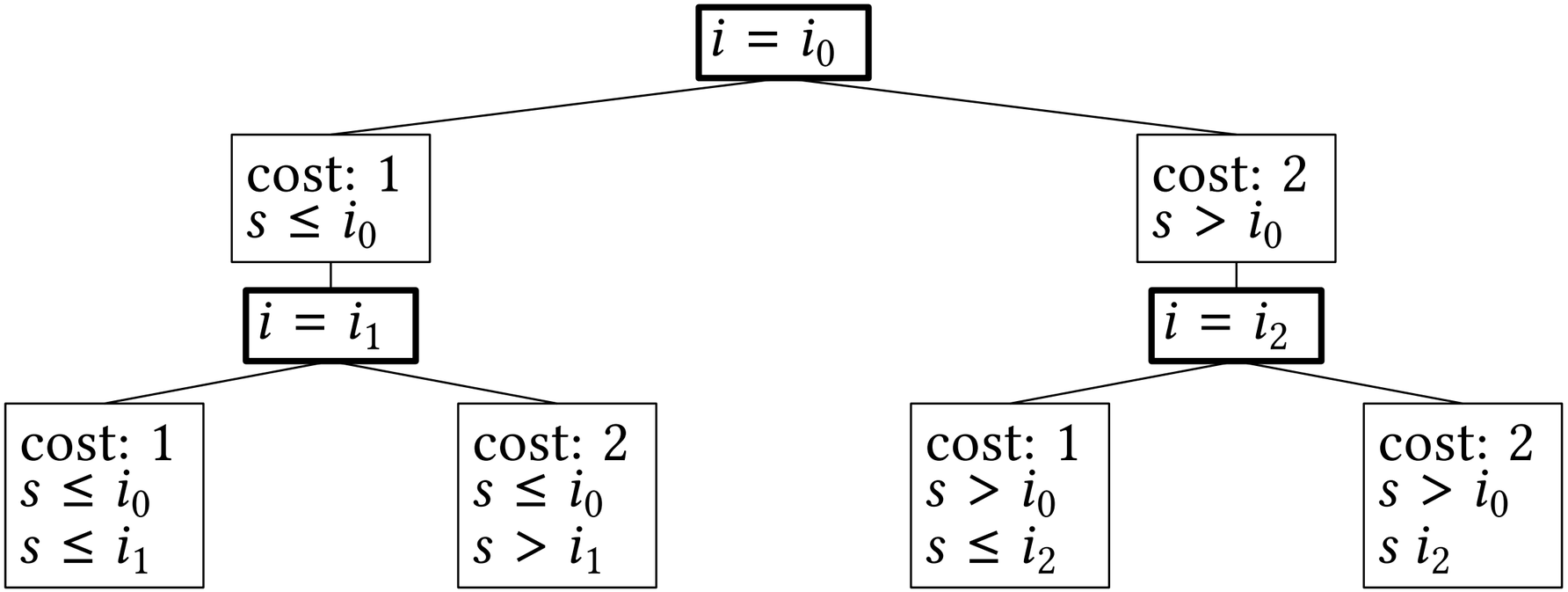
$f\left(6, 2\right) = 4$ $\quad$ $f\left(5, -1\right) = 5$ $\quad$ $f\left(3, 7\right) = 0$


$P\left(s, \vec{i}\right)$
Symbolic Execution
partition constraints
$\{\phi_j\left(s, \vec{i}\right)\}$
Model Counter
partition cell sizes
$\{f_j\left(\vec{i}\right)\}$
Entropy
objectve function
$\mathcal{H}\left(\vec{i}\right)$
Maximize
optimal inputs
optimal tree
$\vec{i}^{\,*} \Longrightarrow T^*$

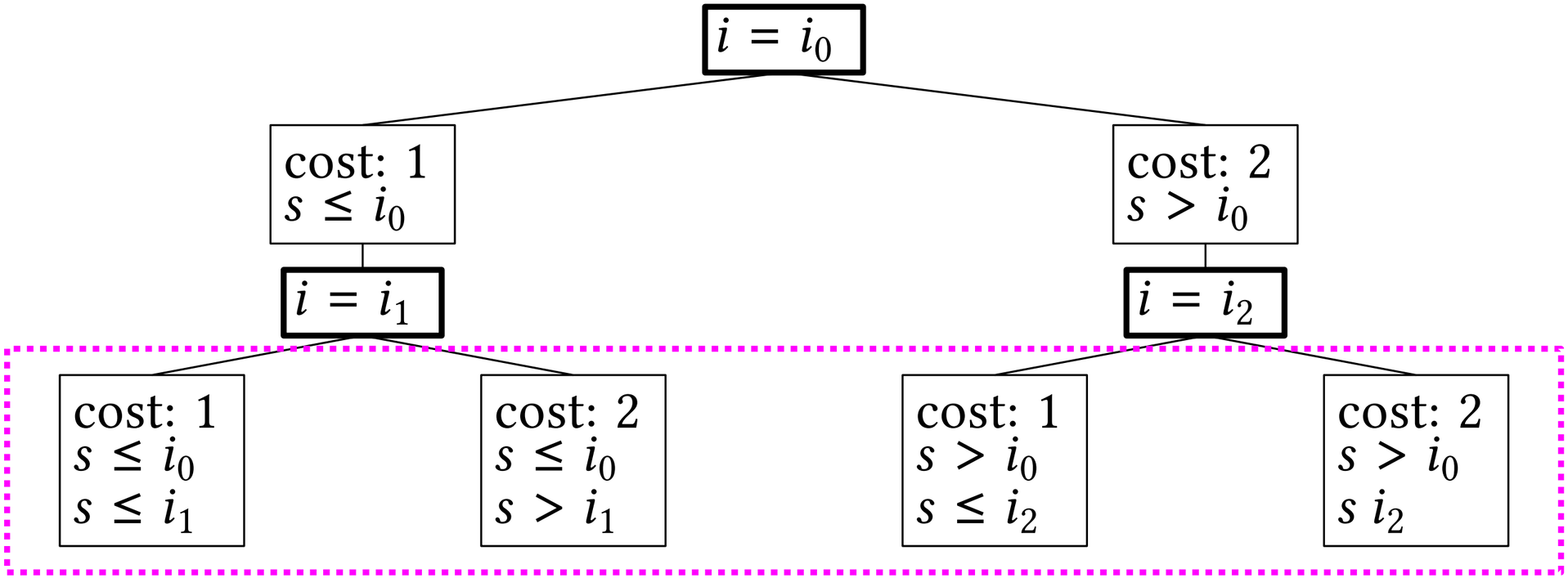

$P\left(s, \vec{i}\right)$
Symbolic Execution
partition constraints
$\{\phi_j\left(s, \vec{i}\right)\}$
Model Counter
partition cell sizes
$\{f_j\left(\vec{i}\right)\}$
Entropy
objectve function
$\mathcal{H}\left(\vec{i}\right)$
Maximize
optimal inputs
optimal tree
$\vec{i}^{\,*} \Longrightarrow T^*$

Computing Multi-Step Entropy Symbolically

Computing Multi-Step Entropy Symbolically



Computing Multi-Step Entropy Symbolically



Computing Multi-Step Entropy Symbolically

$s \leq i_0$
$s \leq i_1$

$s \leq i_0$
$s > i_1$

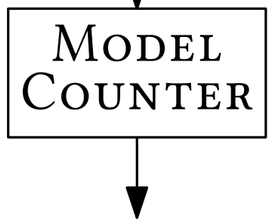
$s > i_0$
$s \leq i_2$

$s > i_0$
$s\ i_2$

Computing Multi-Step Entropy Symbolically

$s \leq i_0$
$s \leq i_1$

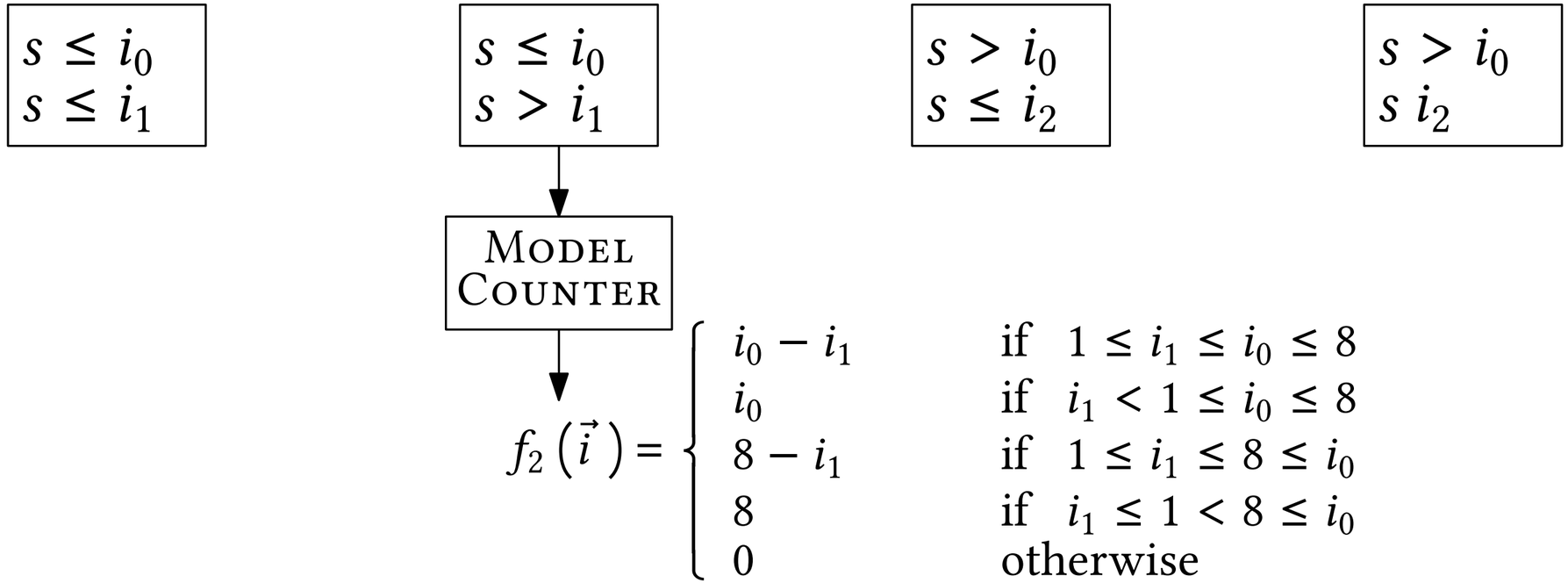
$s \leq i_0$
$s > i_1$



$s > i_0$
$s \leq i_2$

$s > i_0$
$s\ i_2$

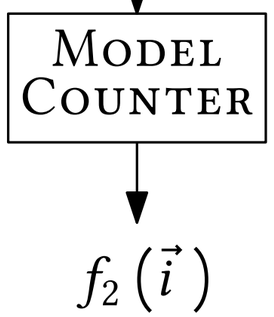
Computing Multi-Step Entropy Symbolically



Computing Multi-Step Entropy Symbolically

$s \leq i_0$
$s \leq i_1$

$s \leq i_0$
$s > i_1$

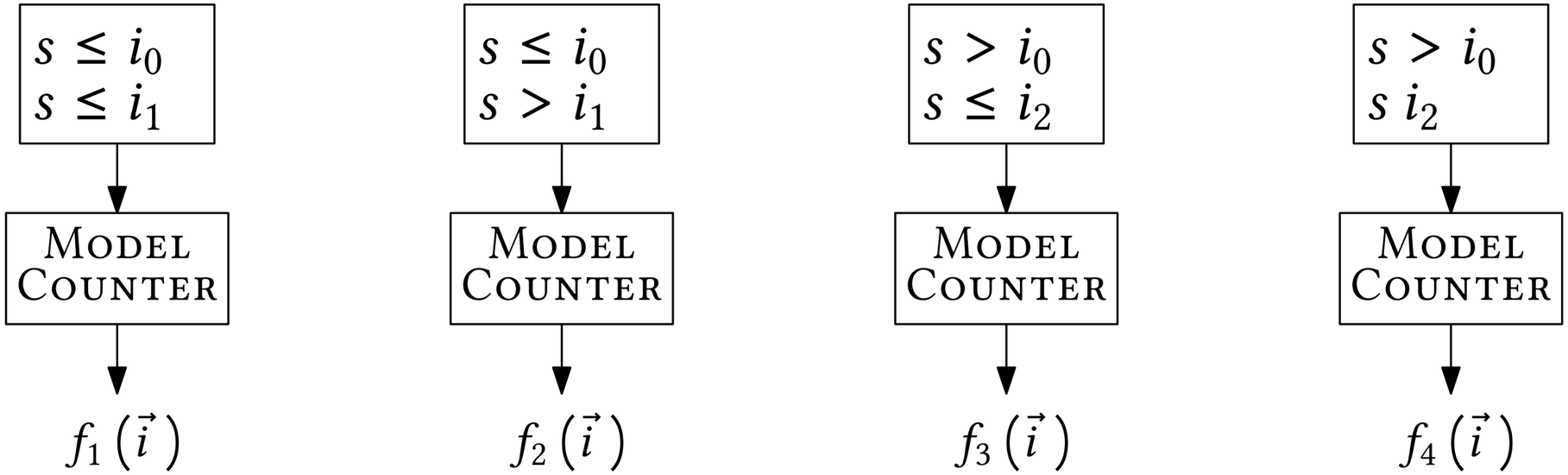


$f_2(\vec{i})$

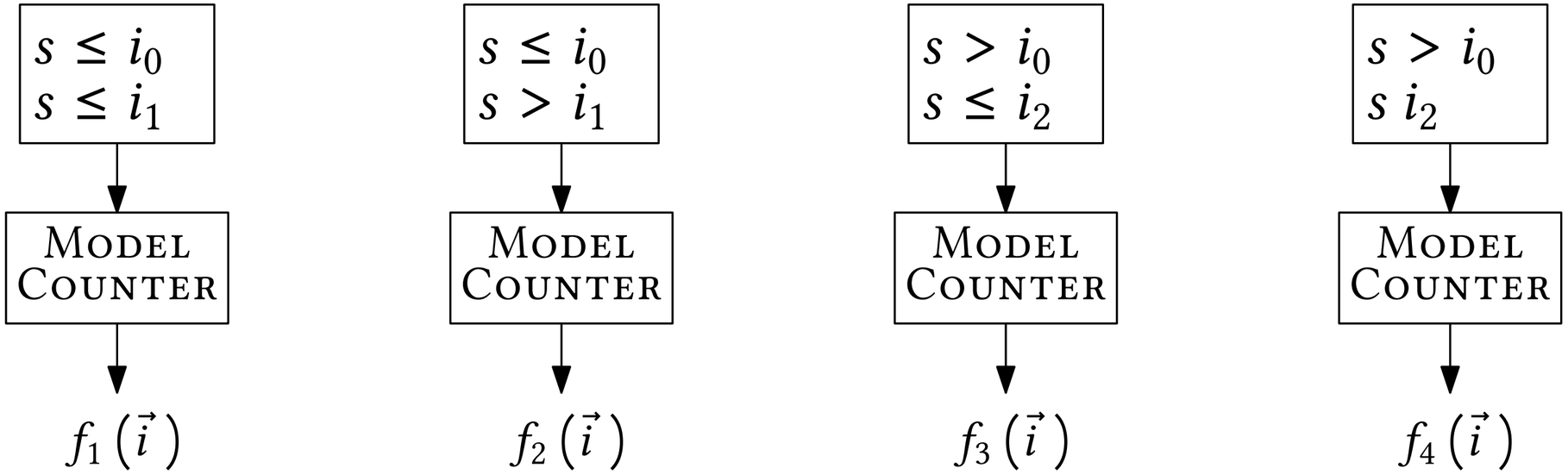
$s > i_0$
$s \leq i_2$

$s > i_0$
$s\ i_2$

Computing Multi-Step Entropy Symbolically

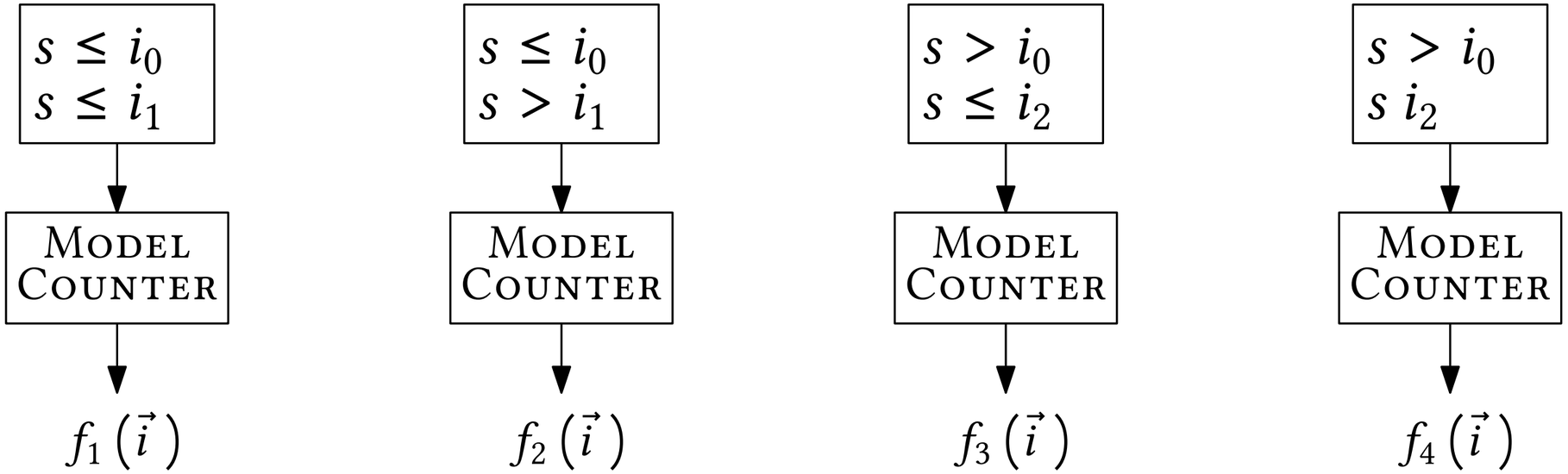


Computing Multi-Step Entropy Symbolically



$$p(s \in \square) = \frac{|\square|}{|\square|} \qquad p_j(\vec{i}) = \frac{f_j(\vec{i})}{|S|}$$

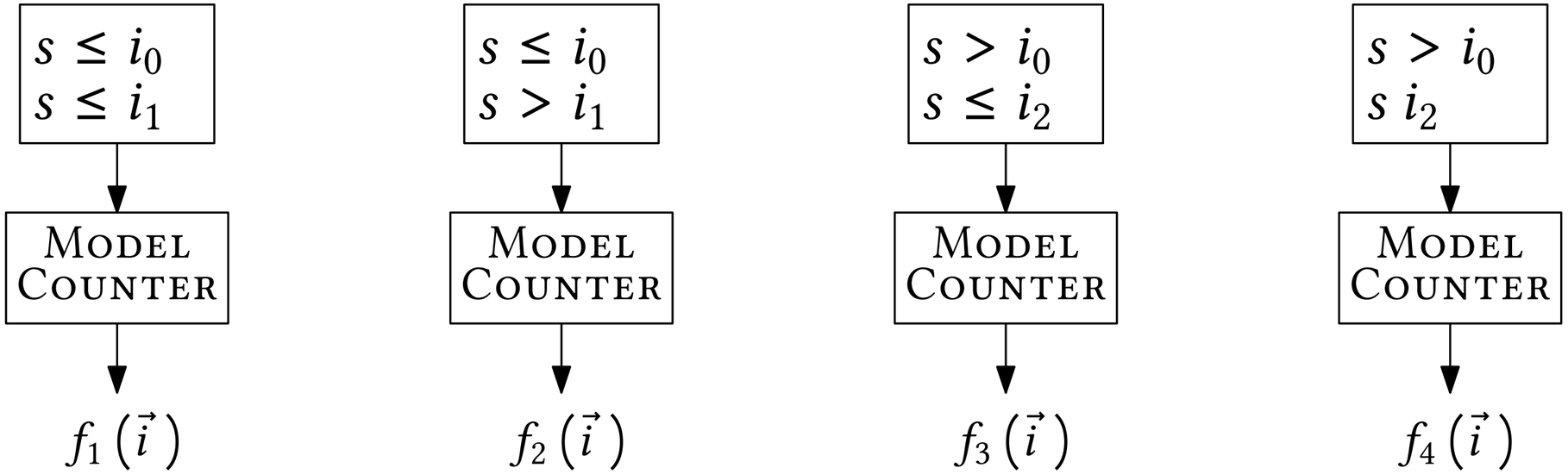
Computing Multi-Step Entropy Symbolically



$$p(s \in \square) = \frac{|\square|}{|\square|} \qquad p_j(\vec{i}) = \frac{f_j(\vec{i})}{|S|}$$

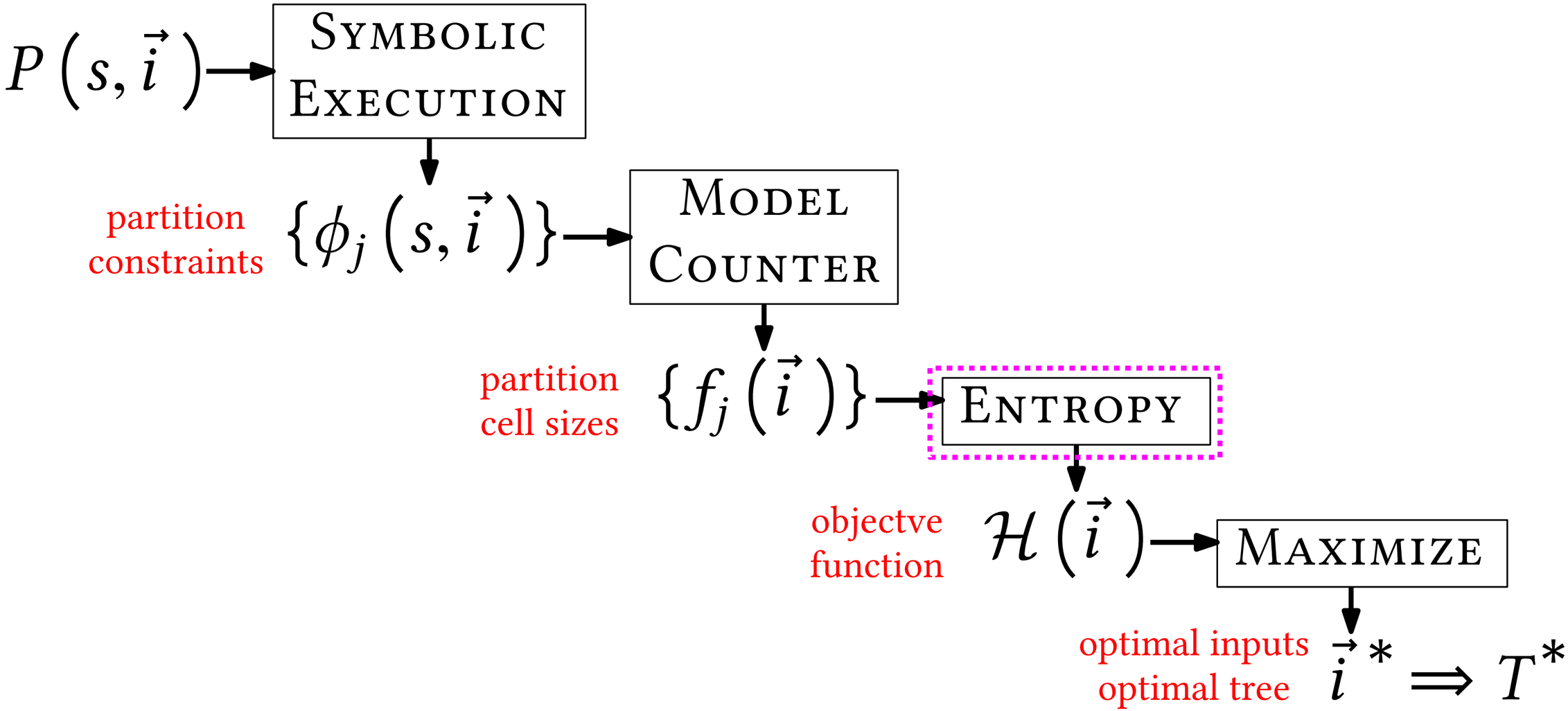
$$\mathcal{H}(\vec{i}) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j}$$

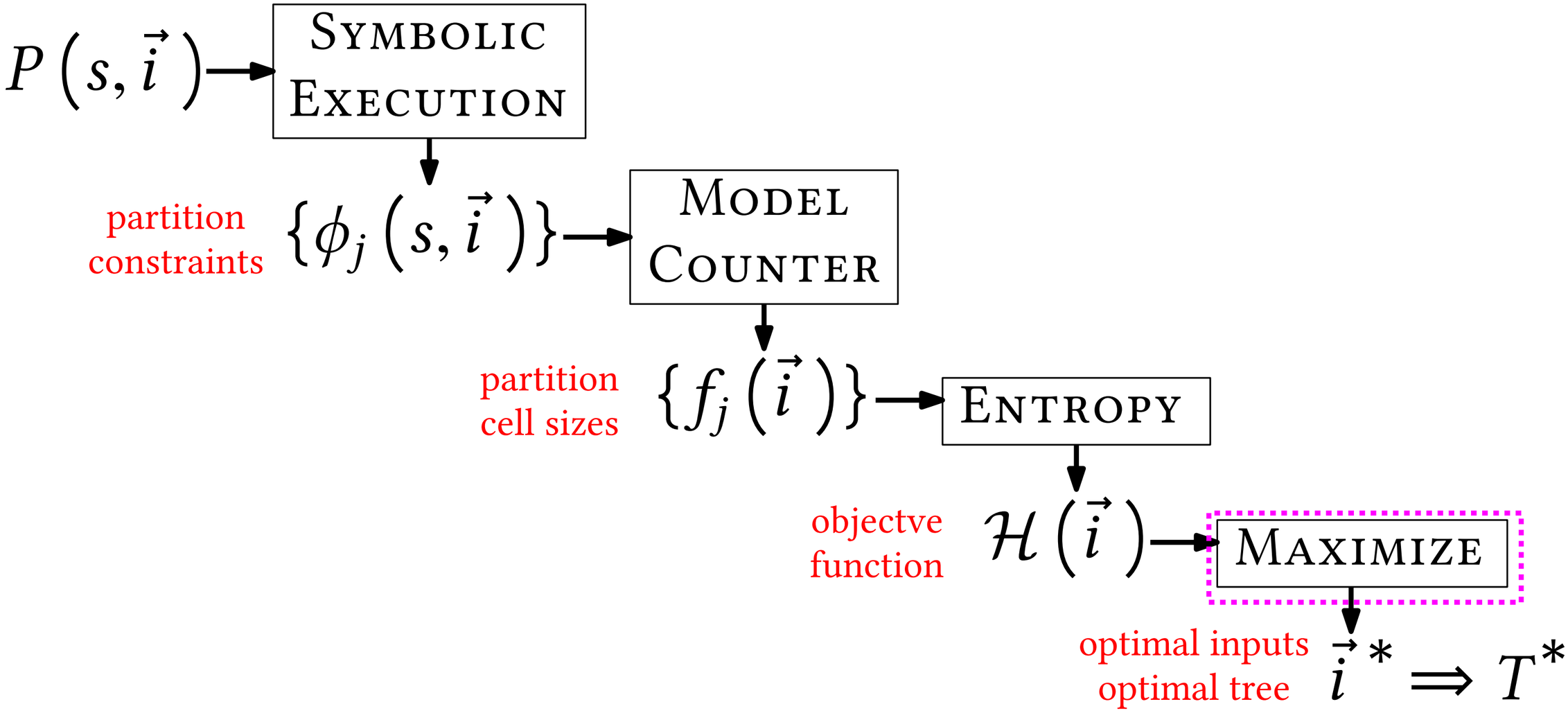
Computing Multi-Step Entropy Symbolically



$$p(s \in \square) = \frac{|\square|}{|\square|} \qquad p_j(\vec{i}) = \frac{f_j(\vec{i})}{|S|}$$

$$\mathcal{H}(\vec{i}) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i})}{8} \log_2 \frac{8}{f_1(\vec{i})} + \frac{f_2(\vec{i})}{8} \log_2 \frac{8}{f_2(\vec{i})} + \frac{f_3(\vec{i})}{8} \log_2 \frac{8}{f_3(\vec{i})} + \frac{f_4(\vec{i})}{8} \log_2 \frac{8}{f_4(\vec{i})}$$


$P\left(s, \vec{i}\right)$
Symbolic Execution
partition constraints
$\{\phi_j\left(s, \vec{i}\right)\}$
Model Counter
partition cell sizes
$\{f_j\left(\vec{i}\right)\}$
Entropy
objectve function
$\mathcal{H}\left(\vec{i}\right)$
Maximize
optimal inputs
optimal tree
$\vec{i}^* \Rightarrow T^*$


$P\left(s, \vec{i}\right)$
Symbolic Execution
partition constraints
$\{\phi_j\left(s, \vec{i}\right)\}$
Model Counter
partition cell sizes
$\{f_j\left(\vec{i}\right)\}$
Entropy
objectve function
$\mathcal{H}\left(\vec{i}\right)$
Maximize
optimal inputs
optimal tree
$\vec{i}^{\,*} \Rightarrow T^*$

Numeric Maximization

$$\mathcal{H}\left(\vec{i}\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i})}{8} \log_2 \frac{8}{f_1(\vec{i})} + \frac{f_2(\vec{i})}{8} \log_2 \frac{8}{f_2(\vec{i})} + \frac{f_3(\vec{i})}{8} \log_2 \frac{8}{f_3(\vec{i})} + \frac{f_4(\vec{i})}{8} \log_2 \frac{8}{f_4(\vec{i})}$$

Numeric Maximization

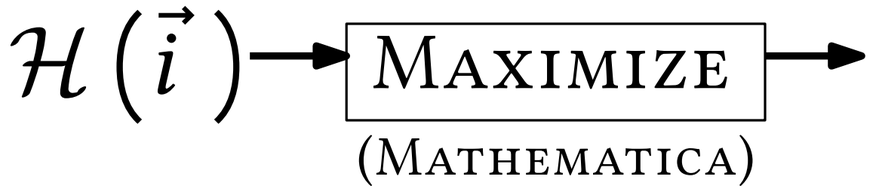
$$\mathcal{H}\left(\vec{i}\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i})}{8} \log_2 \frac{8}{f_1(\vec{i})} + \frac{f_2(\vec{i})}{8} \log_2 \frac{8}{f_2(\vec{i})} + \frac{f_3(\vec{i})}{8} \log_2 \frac{8}{f_3(\vec{i})} + \frac{f_4(\vec{i})}{8} \log_2 \frac{8}{f_4(\vec{i})}$$

objectve function $\mathcal{H}\left(\vec{i}\right) \rightarrow$ MAXIMIZE

Numeric Maximization

$$\mathcal{H}\left(\vec{i}\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i})}{8} \log_2 \frac{8}{f_1(\vec{i})} + \frac{f_2(\vec{i})}{8} \log_2 \frac{8}{f_2(\vec{i})} + \frac{f_3(\vec{i})}{8} \log_2 \frac{8}{f_3(\vec{i})} + \frac{f_4(\vec{i})}{8} \log_2 \frac{8}{f_4(\vec{i})}$$

objectve function



Numeric Maximization

$$\mathcal{H}\left(\vec{i}\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i})}{8} \log_2 \frac{8}{f_1(\vec{i})} + \frac{f_2(\vec{i})}{8} \log_2 \frac{8}{f_2(\vec{i})} + \frac{f_3(\vec{i})}{8} \log_2 \frac{8}{f_3(\vec{i})} + \frac{f_4(\vec{i})}{8} \log_2 \frac{8}{f_4(\vec{i})}$$

objectve function

$$\mathcal{H}\left(\vec{i}\right) \rightarrow \boxed{\text{Maximize}} \rightarrow \vec{i}^{\,*} \Longrightarrow T^*$$

(Mathematica)

optimal inputs
optimal tree

Numeric Maximization

$$\mathcal{H}\left(\vec{i}\,\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i}\,)}{8} \log_2 \frac{8}{f_1(\vec{i}\,)} + \frac{f_2(\vec{i}\,)}{8} \log_2 \frac{8}{f_2(\vec{i}\,)} + \frac{f_3(\vec{i}\,)}{8} \log_2 \frac{8}{f_3(\vec{i}\,)} + \frac{f_4(\vec{i}\,)}{8} \log_2 \frac{8}{f_4(\vec{i}\,)}$$

objectve function $\mathcal{H}\left(\vec{i}\,\right) \rightarrow$ MAXIMIZE (MATHEMATICA) $\rightarrow \vec{i}^{\,*} \Longrightarrow T^*$ optimal inputs, optimal tree

$$\vec{i}^{\,*} = (i_0^*, i_1^*, i_2^*)$$

Numeric Maximization

$$\mathcal{H}\left(\vec{i}\,\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i}\,)}{8} \log_2 \frac{8}{f_1(\vec{i}\,)} + \frac{f_2(\vec{i}\,)}{8} \log_2 \frac{8}{f_2(\vec{i}\,)} + \frac{f_3(\vec{i}\,)}{8} \log_2 \frac{8}{f_3(\vec{i}\,)} + \frac{f_4(\vec{i}\,)}{8} \log_2 \frac{8}{f_4(\vec{i}\,)}$$

objectve function $\mathcal{H}\left(\vec{i}\,\right) \rightarrow$ MAXIMIZE (MATHEMATICA) $\rightarrow \vec{i}^{\,*} \Longrightarrow T^*$ optimal inputs optimal tree

$$\vec{i}^{\,*} = (i_0^*, i_1^*, i_2^*)$$

$$\vec{i}^{\,*} = (4, 2, 6)$$

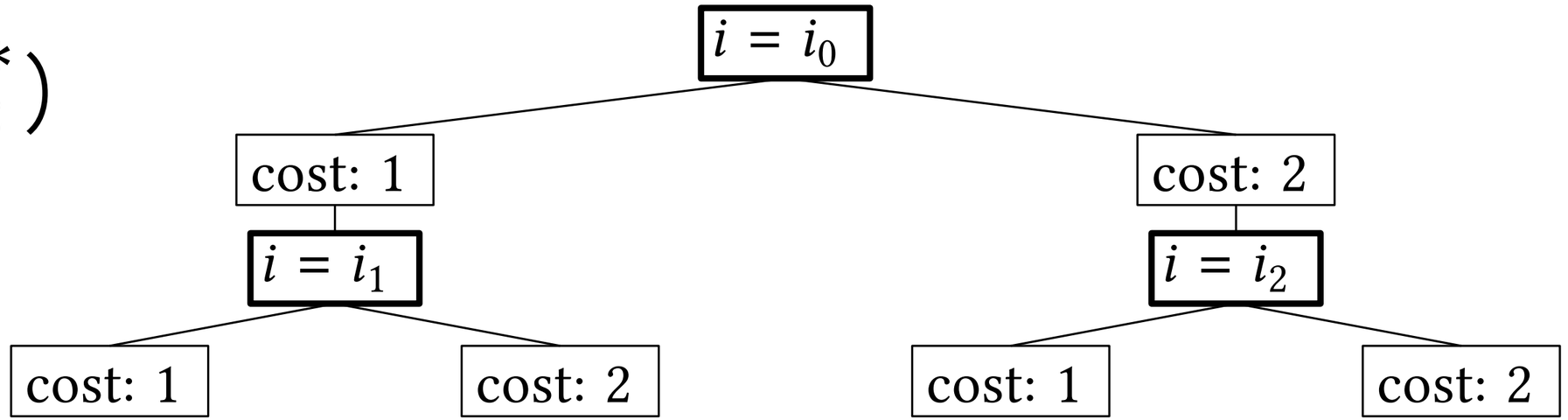
Numeric Maximization

$$\mathcal{H}\left(\vec{i}\,\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i}\,)}{8} \log_2 \frac{8}{f_1(\vec{i}\,)} + \frac{f_2(\vec{i}\,)}{8} \log_2 \frac{8}{f_2(\vec{i}\,)} + \frac{f_3(\vec{i}\,)}{8} \log_2 \frac{8}{f_3(\vec{i}\,)} + \frac{f_4(\vec{i}\,)}{8} \log_2 \frac{8}{f_4(\vec{i}\,)}$$

objectve function $\mathcal{H}\left(\vec{i}\,\right) \rightarrow$ MAXIMIZE (MATHEMATICA) $\rightarrow \vec{i}^{\,*} \Longrightarrow T^*$ optimal inputs optimal tree

$$\vec{i}^{\,*} = (i_0^*, i_1^*, i_2^*)$$

$$\vec{i}^{\,*} = (4, 2, 6)$$



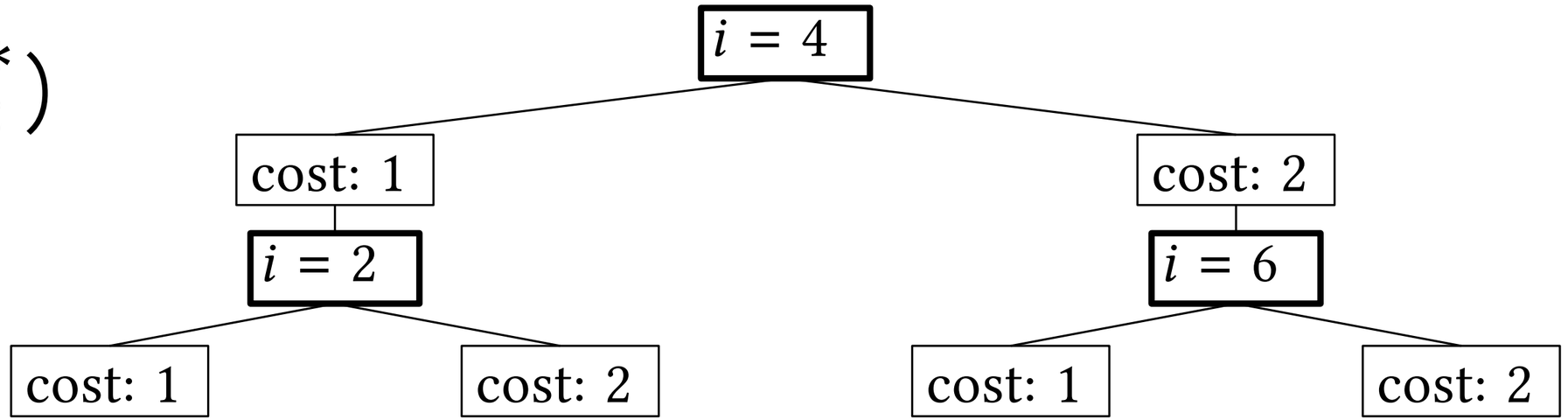
Numeric Maximization

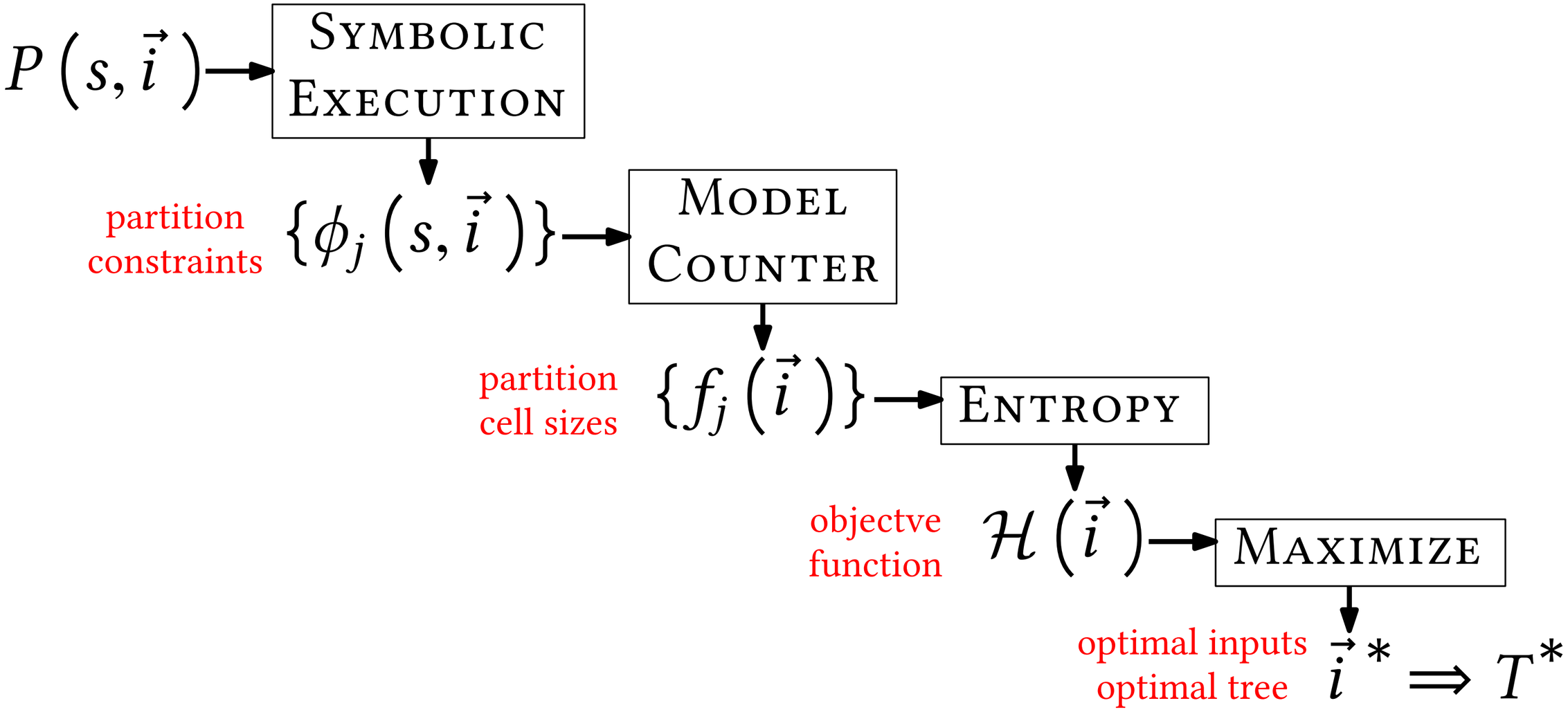
$$\mathcal{H}\left(\vec{i}\,\right) = \sum_{j=1}^{n} p_i \log_2 \frac{1}{p_j} = \frac{f_1(\vec{i}\,)}{8} \log_2 \frac{8}{f_1(\vec{i}\,)} + \frac{f_2(\vec{i}\,)}{8} \log_2 \frac{8}{f_2(\vec{i}\,)} + \frac{f_3(\vec{i}\,)}{8} \log_2 \frac{8}{f_3(\vec{i}\,)} + \frac{f_4(\vec{i}\,)}{8} \log_2 \frac{8}{f_4(\vec{i}\,)}$$

objectve function $\mathcal{H}\left(\vec{i}\,\right) \rightarrow$ MAXIMIZE (MATHEMATICA) $\rightarrow \vec{i}^{\,*} \Longrightarrow T^*$ optimal inputs optimal tree

$$\vec{i}^{\,*} = (i_0^*, i_1^*, i_2^*)$$

$$\vec{i}^{\,*} = (4, 2, 6)$$




$P\left(s, \vec{i}\right)$
Symbolic Execution
partition constraints
$\{\phi_j\left(s, \vec{i}\right)\}$
Model Counter
partition cell sizes
$\{f_j\left(\vec{i}\right)\}$
Entropy
objectve function
$\mathcal{H}\left(\vec{i}\right)$
Maximize
optimal inputs
optimal tree
$\vec{i}^{\,*} \Rightarrow T^*$

Other Methods

Other Methods

My approach: reduce attack synthesis to **numeric optimization** problem.

Other Methods

My approach: reduce attack synthesis to **numeric optimization** problem.

Limited by model counter.

Numeric optimum not guaranteed.

Other Methods

My approach: reduce attack synthesis to **numeric optimization** problem.

- Limited by model counter.
- Numeric optimum not guaranteed.

MaxSMT: reduce attack synthesis to **Max SAT** problem.

Other Methods

My approach: reduce attack synthesis to **numeric optimization** problem.

- Limited by model counter.
- Numeric optimum not guaranteed.

MaxSMT: reduce attack synthesis to **Max SAT** problem.

- Reduces everything to bits.
- Upper bound on information leakage.

Other Methods

My approach: reduce attack synthesis to **numeric optimization** problem.

- Limited by model counter.
- Numeric optimum not guaranteed.

MaxSMT: reduce attack synthesis to **Max SAT** problem.

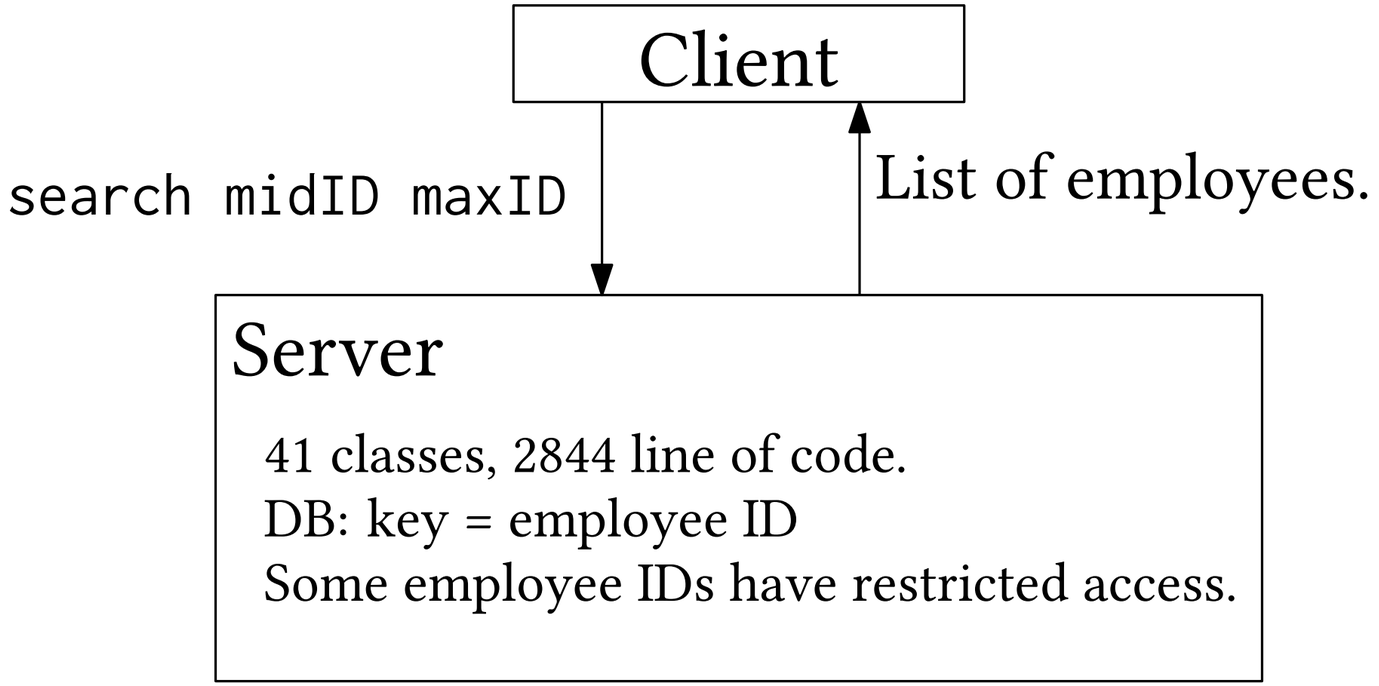
- Reduces everything to bits.
- Upper bound on information leakage.

MARCO: reduce attack synthesis to **Maximum SAT Subsets** problem.

- Reduces everything to bits.
- Exact optimal information leakage guaranteed.

Case Study: LawDB

From DARPA Space-Time Analysis for Cybersecurity (STAC)



LawDB Partition Constraints

$$
\begin{aligned}
&\{h \leq l_1 \wedge 85 \leq l_1 \wedge 64 \leq l_1 \wedge 64 \geq l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h > l_1 \wedge 85 \leq l_1 \wedge 64 \leq l_1 \wedge 64 \geq l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&85 > l_1 \wedge 64 \leq l_1 \wedge 64 \geq l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&64 > l_1 \wedge 64 \geq l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h \leq l_1 \wedge 85 \leq l_1 \wedge 85 \geq l_2 \wedge 64 < l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h > l_1 \wedge 85 \leq l_1 \wedge 85 \geq l_2 \wedge 64 < l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&85 > l_1 \wedge 85 \geq l_2 \wedge 64 < l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h \leq l_1 \wedge h \geq l_2 \wedge 85 < l_2 \wedge 64 < l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h > l_1 \wedge h \geq l_2 \wedge 85 < l_2 \wedge 64 < l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h < l_2 \wedge 85 < l_2 \wedge 64 < l_2 \wedge h > 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&85 \leq l_1 \wedge h \leq l_1 \wedge 64 \leq l_1 \wedge 64 \geq l_2 \wedge h \leq 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&85 > l_1 \wedge h \leq l_1 \wedge 64 \leq l_1 \wedge 64 \geq l_2 \wedge h \leq 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&h > l_1 \wedge 64 \leq l_1 \wedge 64 \geq l_2 \wedge h \leq 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&64 > l_1 \wedge 64 \geq l_2 \wedge h \leq 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,\\
&85 \leq l_1 \wedge h \leq l_1 \wedge h \geq l_2 \wedge 64 < l_2 \wedge h \leq 85 \wedge h > 64 \wedge h \neq 85 \wedge h \neq 64,
\end{aligned}
$$

Case Study: LawDB, DB size = 100

Keep pushing tree deeper until partitions have size 1.

	Tree depth	Time
Numeric	7	57s
MaxSMT	17	21s
MARCO	7	2m 36s

Case Study: LawDB, DB size = 100

Keep pushing tree deeper until partitions have size 1.

	Tree depth	Time
Numeric	7	57s
MaxSMT	17	21s
MARCO	7	2m 36s

Case Study: LawDB, DB size = 100

Keep pushing tree deeper until partitions have size 1.

	Tree depth	Time
Numeric	7	57s
MaxSMT	17	21s
MARCO	7	2m 36s

Proposed Experiments

DARPA Space-Time Analysis for Cybersecurity (STAC)

Canonical Side-Channel Vulnerability Benchmark

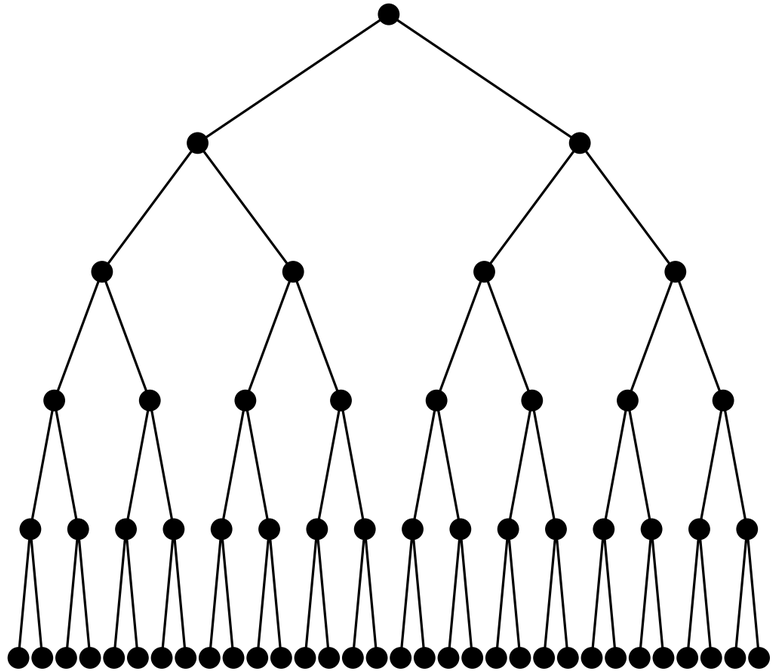
https://github.com/Apogee-Research/STAC/

7 Applications, 1 to 3 variants each

14 total programs

Challenges and Solutions

Fully Offline Static



Quantify over all $s \in S$

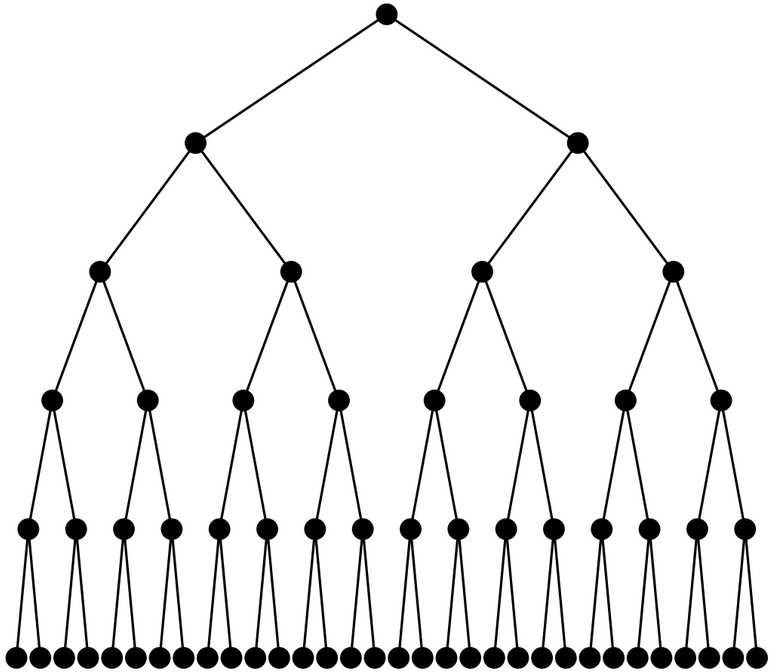
Exponential blowup

Cost model: overly ideal, not realistic

Ignores HW / OS properties

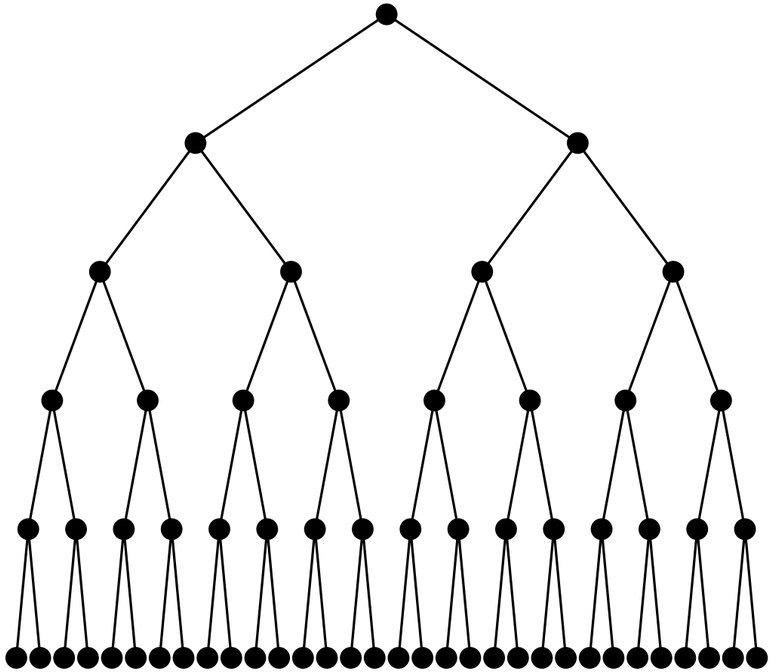
Challenges and Solutions

Fully Offline Static



Quantify over all $s \in S$
Exponential blowup
Cost model: overly ideal, not realistic
Ignores HW / OS properties

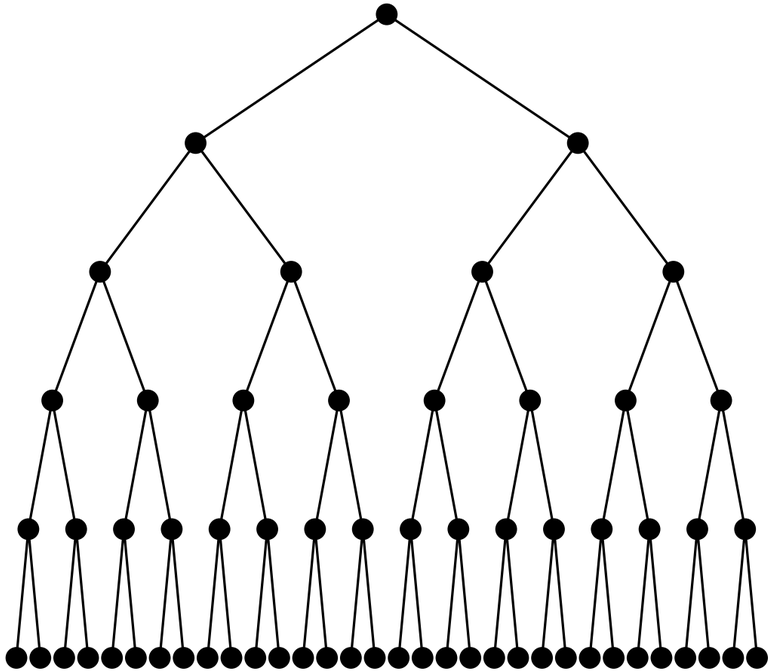
Static + Dynamic



Real system has one $s \in S$

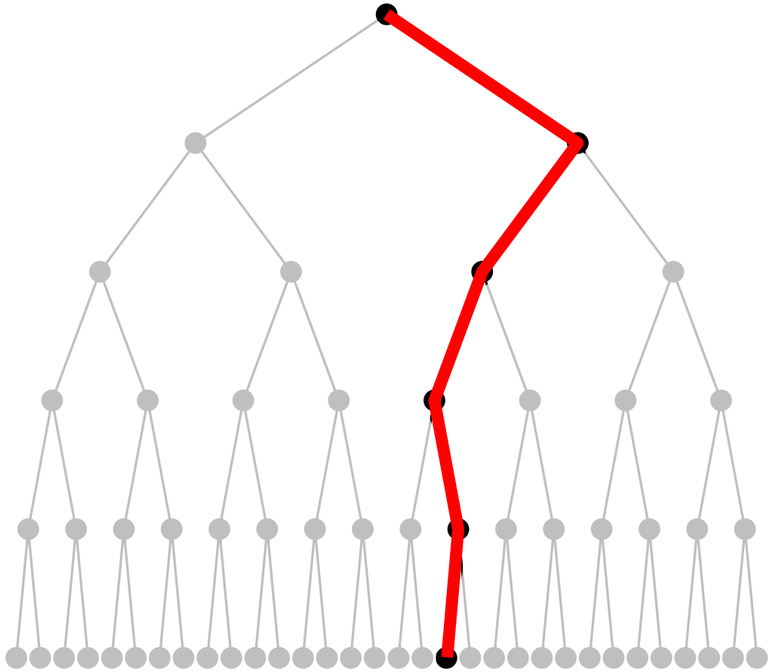
Challenges and Solutions

Fully Offline Static



Quantify over all $s \in S$
Exponential blowup
Cost model: overly ideal, not realistic
Ignores HW / OS properties

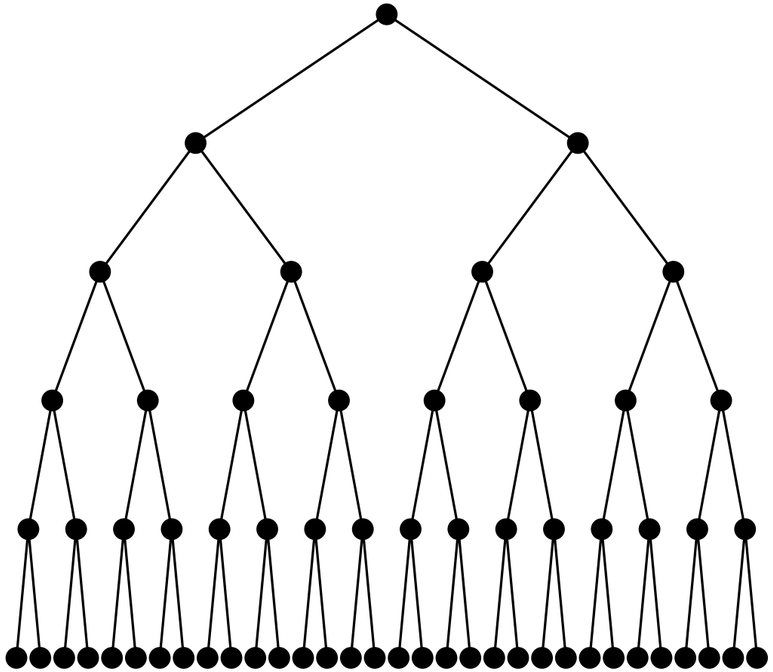
Static + Dynamic



Real system has one $s \in S$

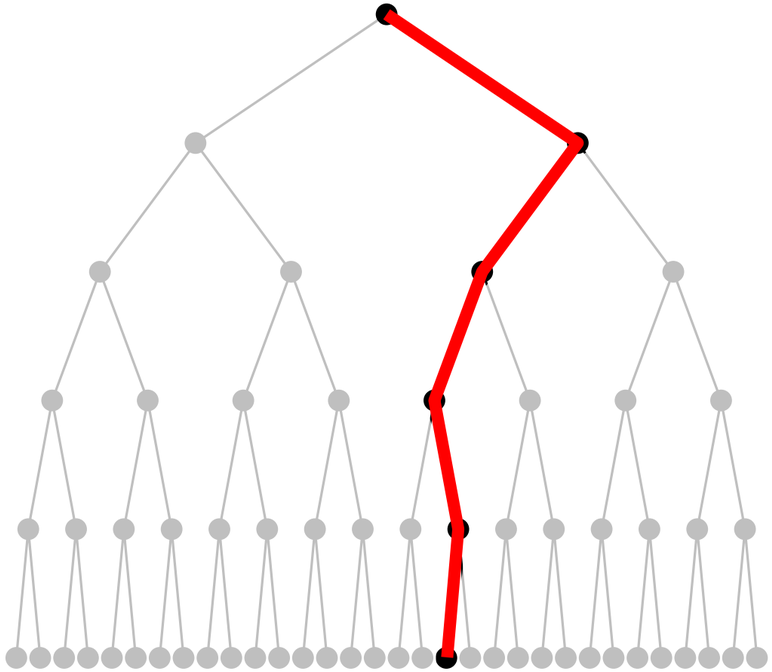
Challenges and Solutions

Fully Offline Static



Quantify over all $s \in S$
Exponential blowup
Cost model: overly ideal, not realistic
Ignores HW / OS properties

Static + Dynamic



Real system has one $s \in S$
Put program on a real system
Dynamic cost profiling

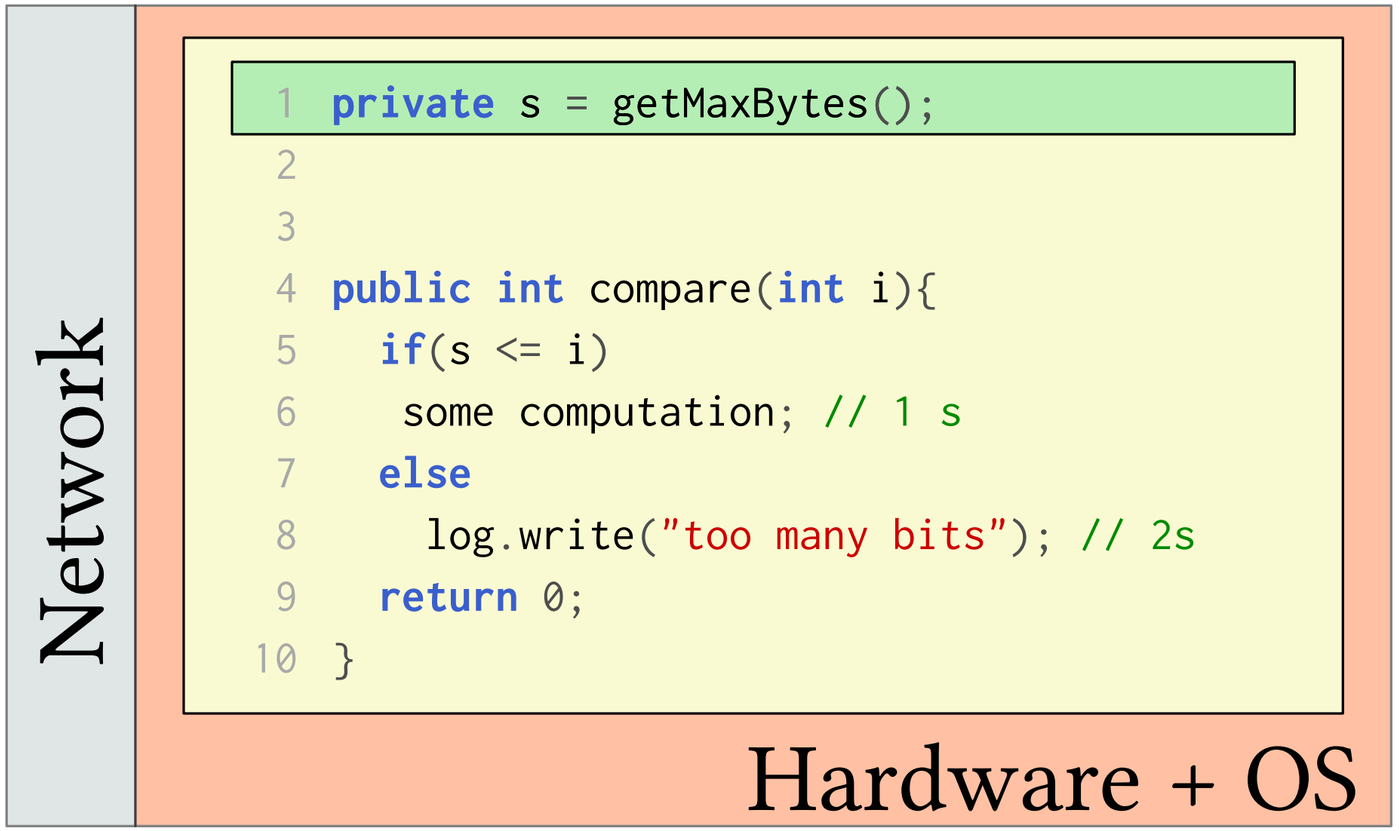
Online Attack Synthesis

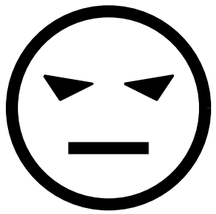
Online Attack Synthesis Proposed Work

```
1  private s = getMaxBytes();
2
3
4  public int compare(int i){
5    if(s <= i)
6     some computation; // 1 s
7    else
8      log.write("too many bits"); // 2s
9    return 0;
10 }
```

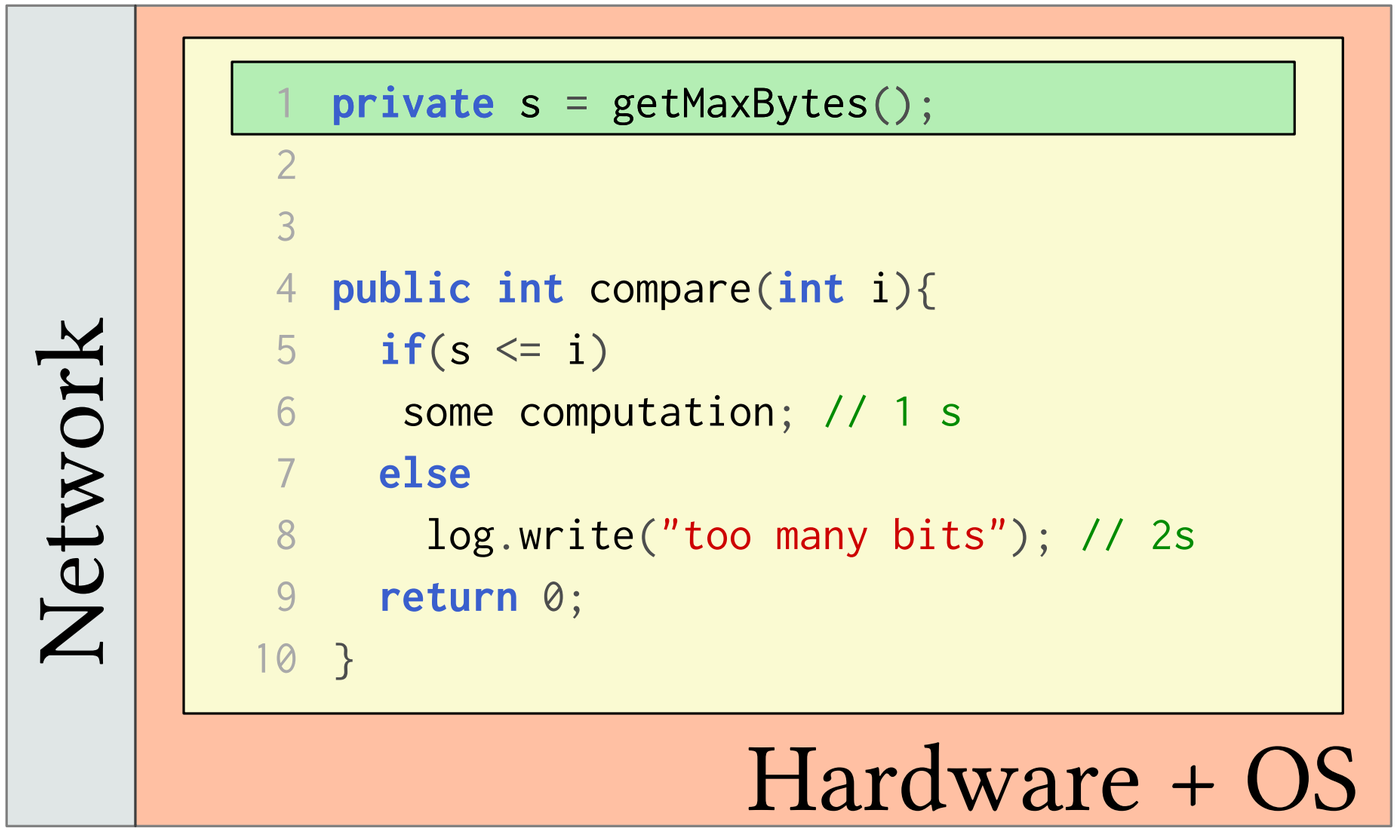
```
1  private s = getMaxBytes();
2
3
4  public int compare(int i){
5    if(s <= i)
6     some computation; // 1 s
7    else
8      log.write("too many bits"); // 2s
9    return 0;
10 }
```

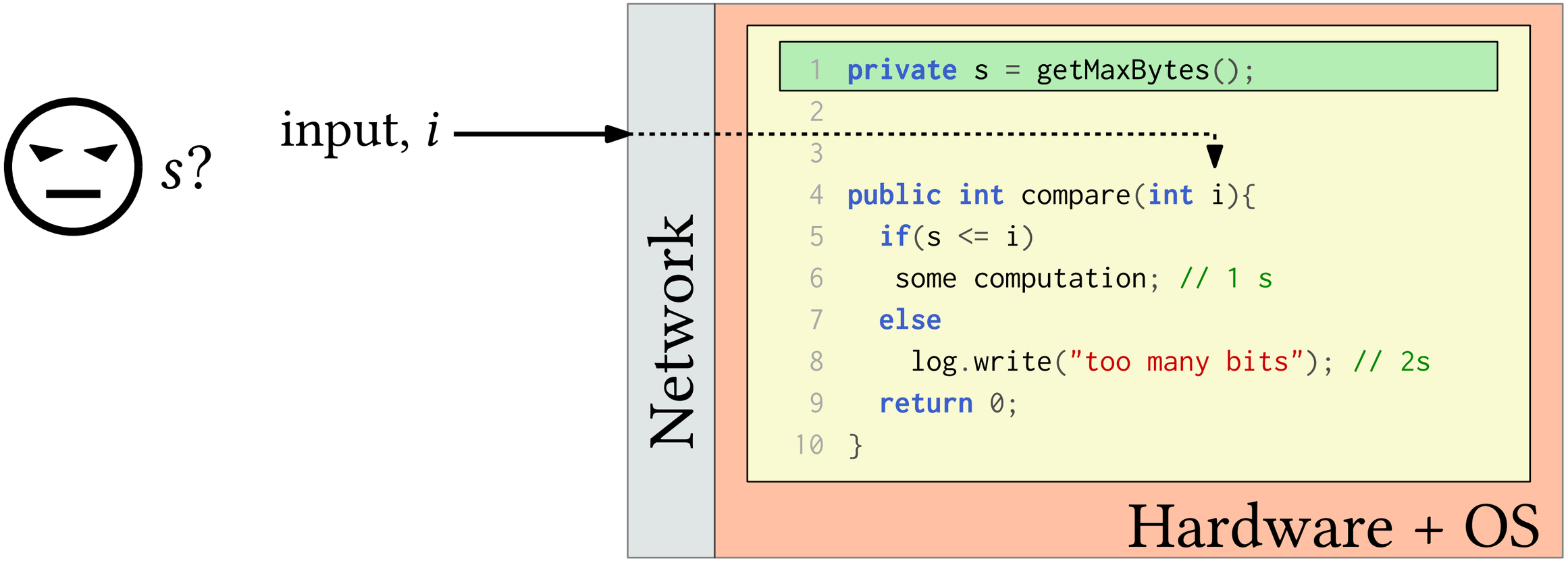
Hardware + OS

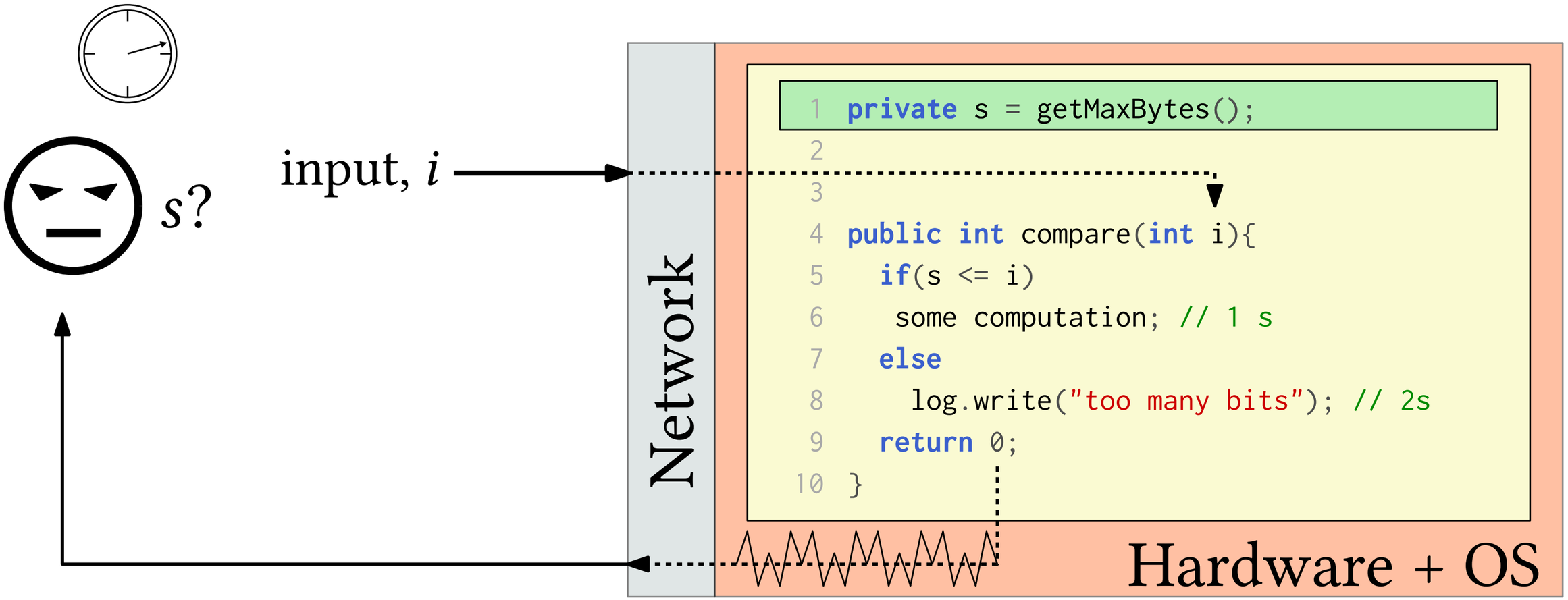

Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

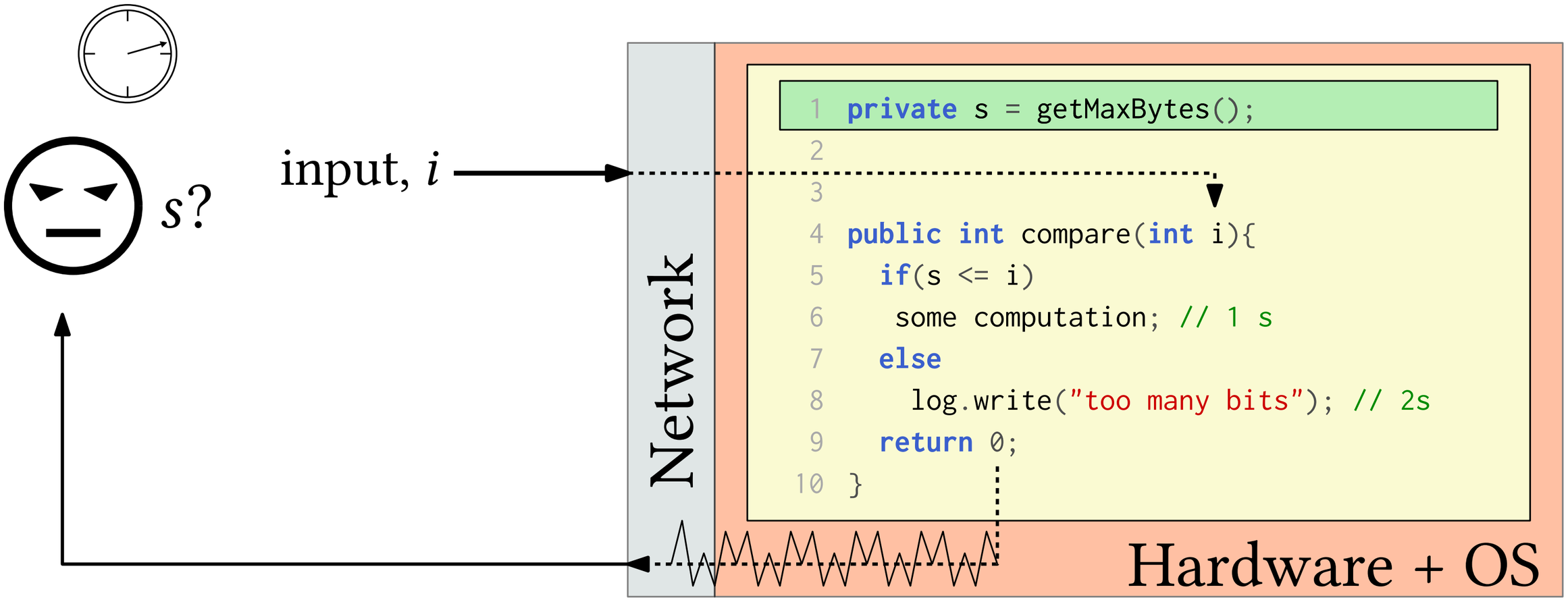


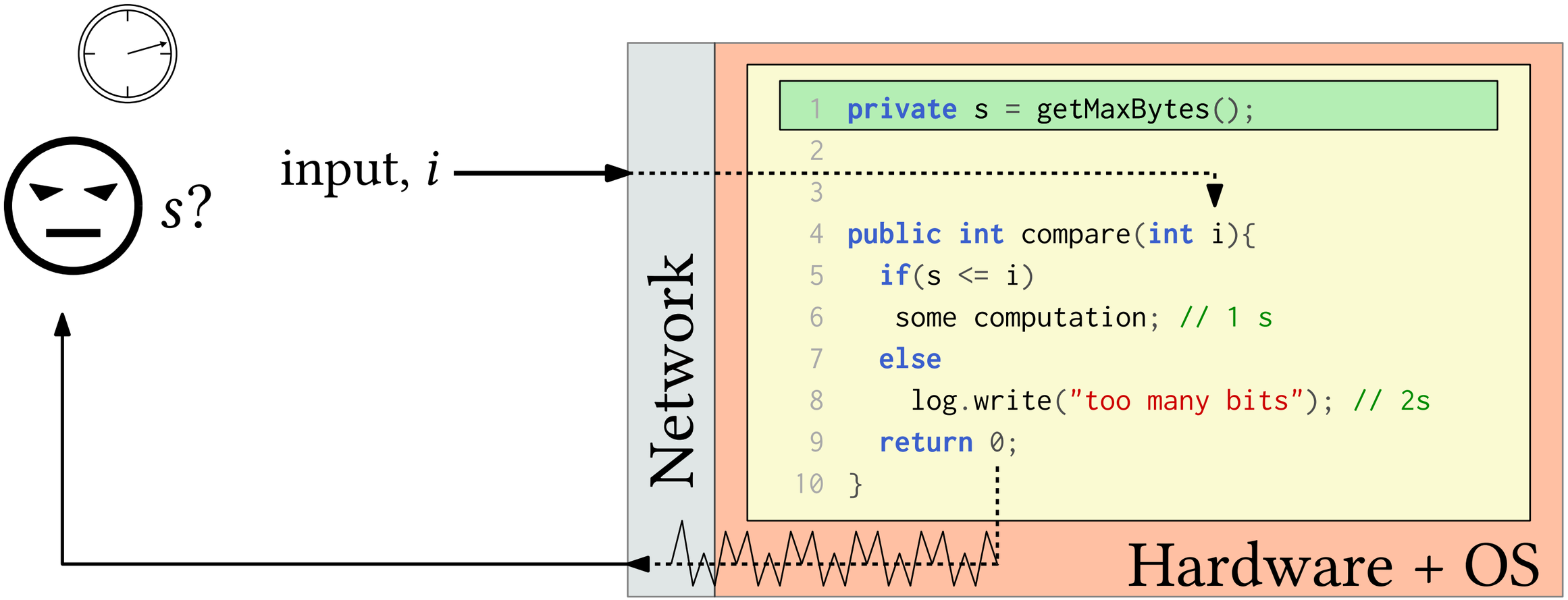
s?

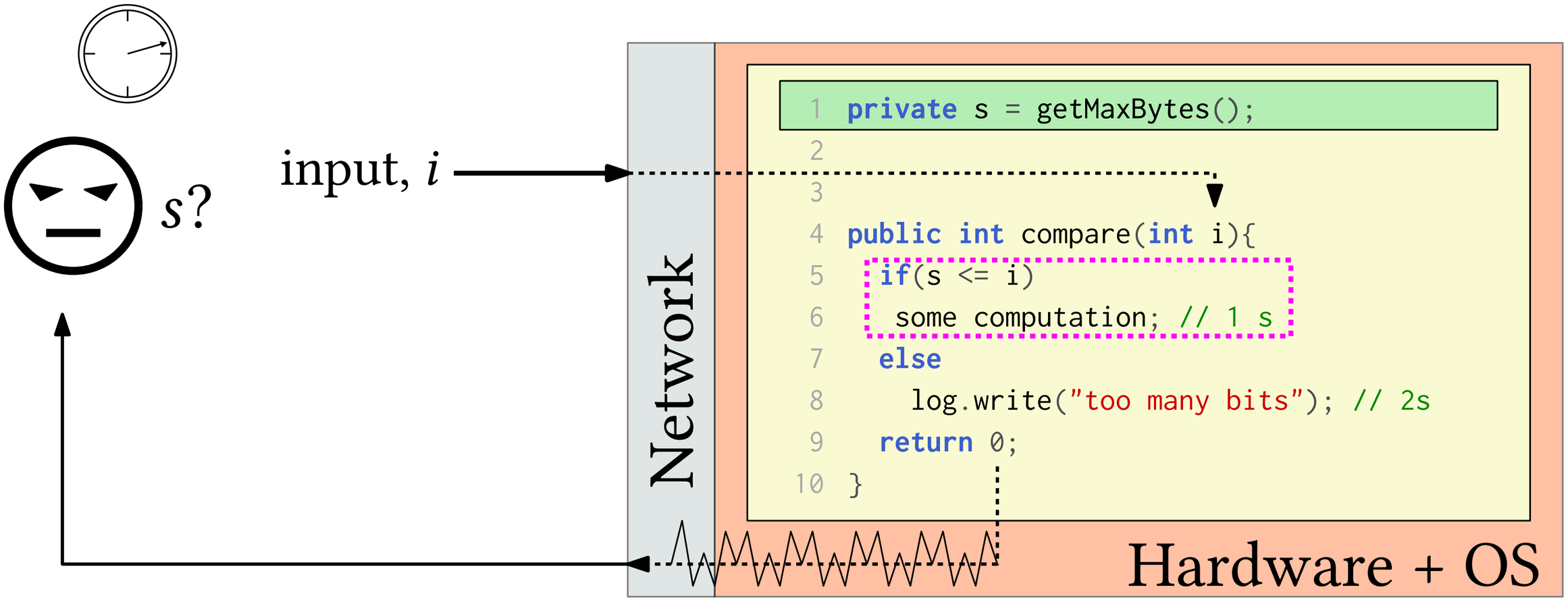

Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

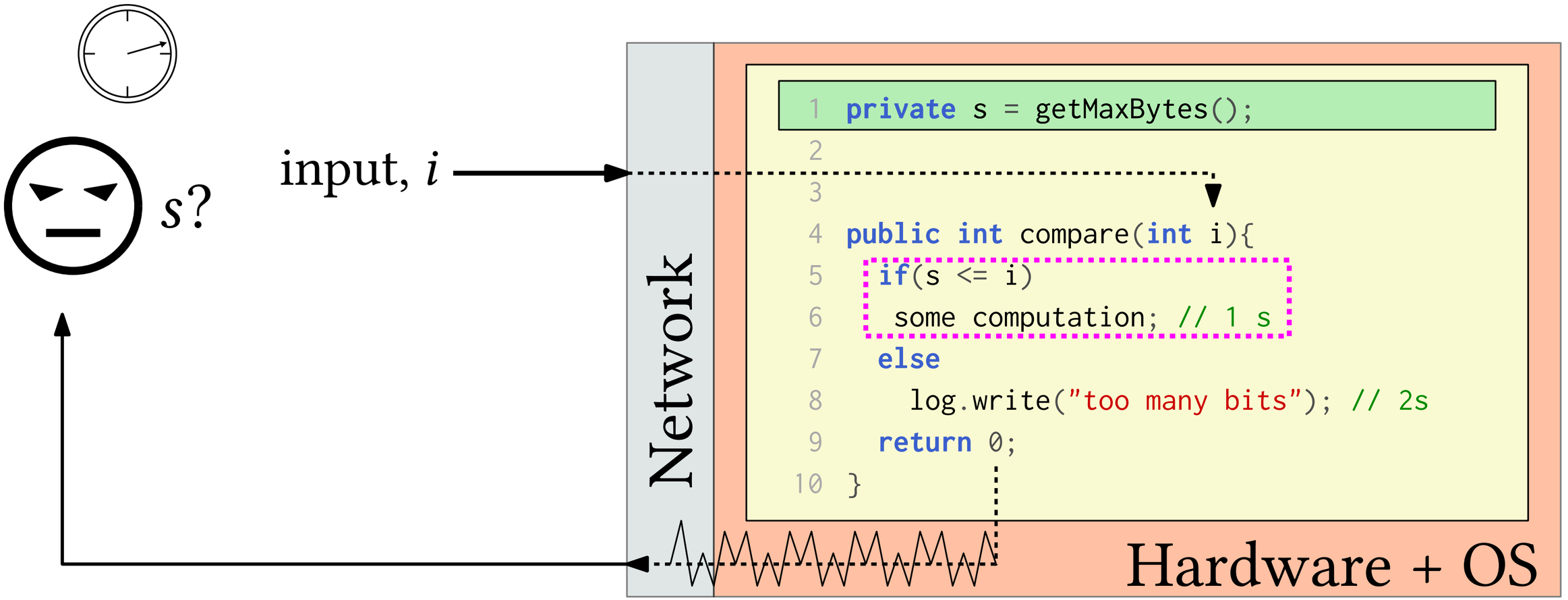

s?
input, i
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS


input, i
s?
Network
private s = getMaxBytes();
public int compare(int i){
if(s <= i)
some computation; // 1 s
else
log.write("too many bits"); // 2s
return 0;
}
Hardware + OS

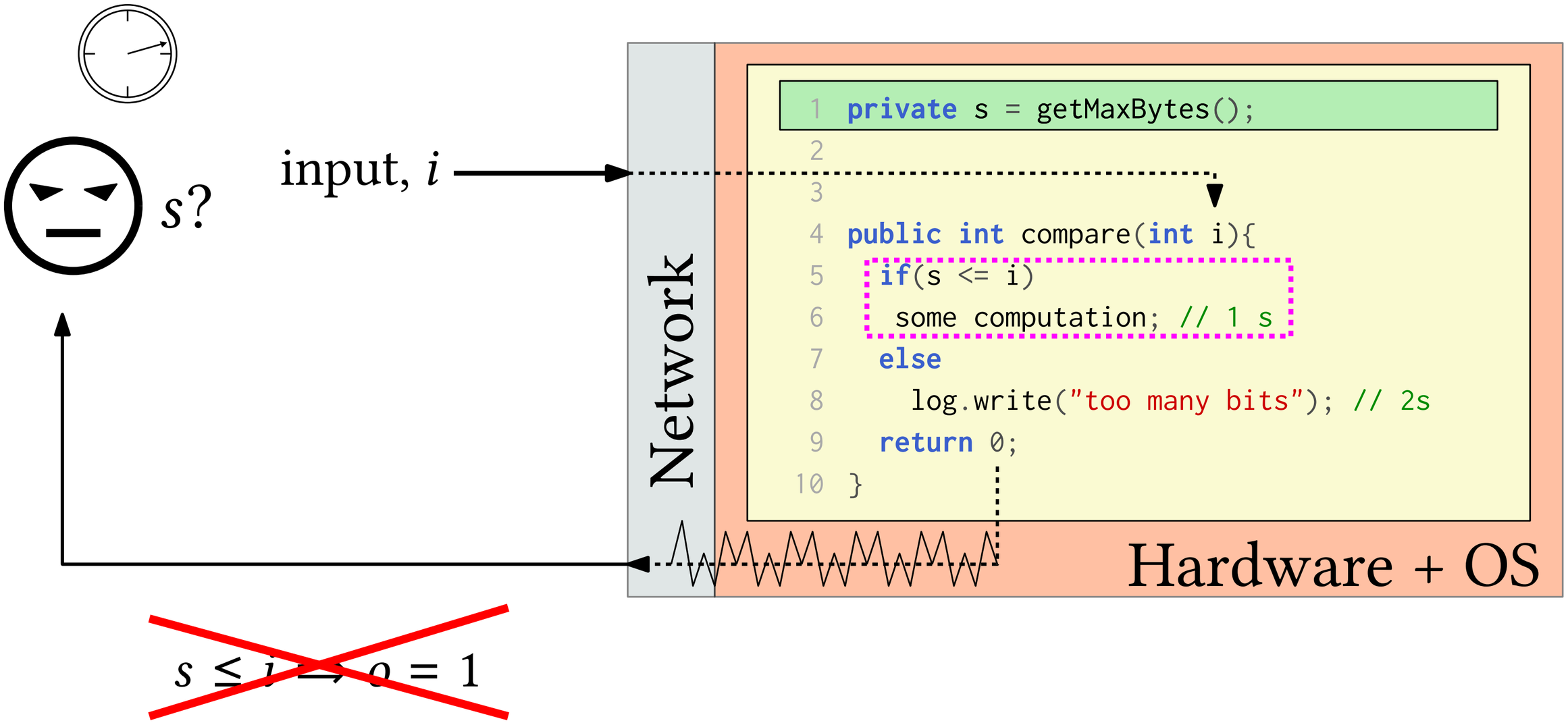

input, i
s?
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

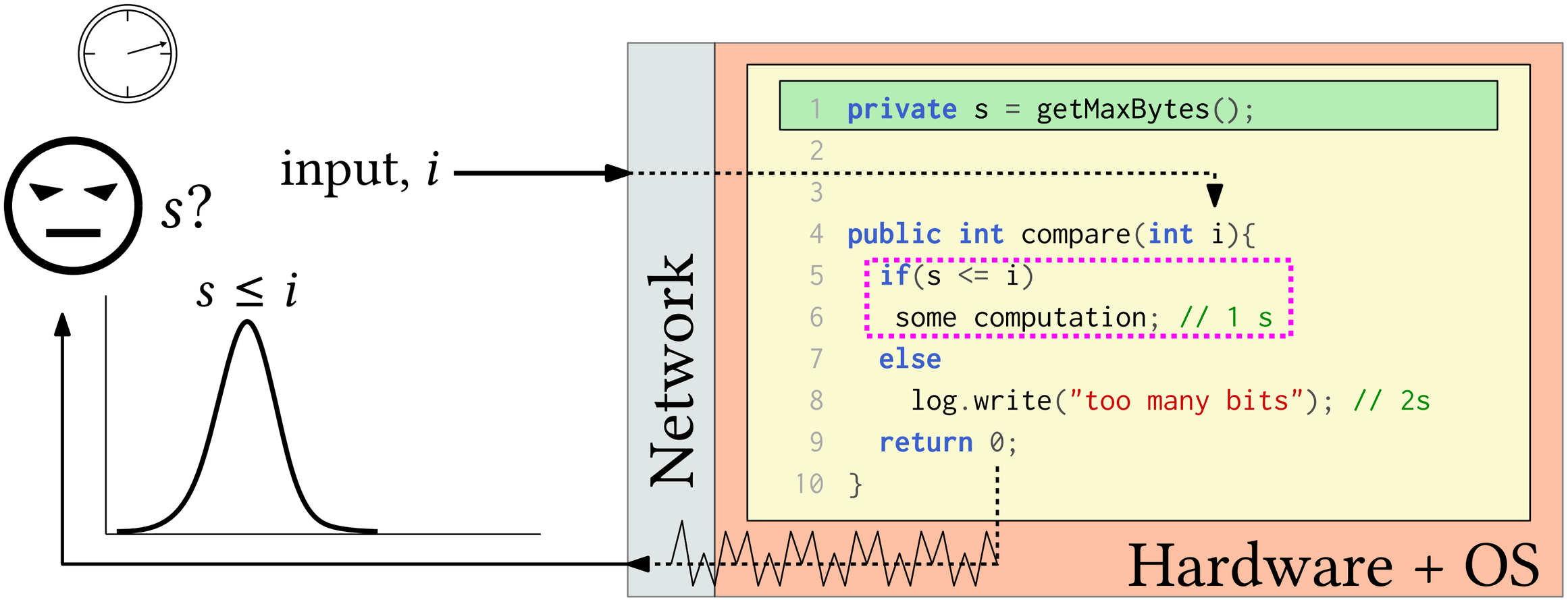

s?
input, i
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

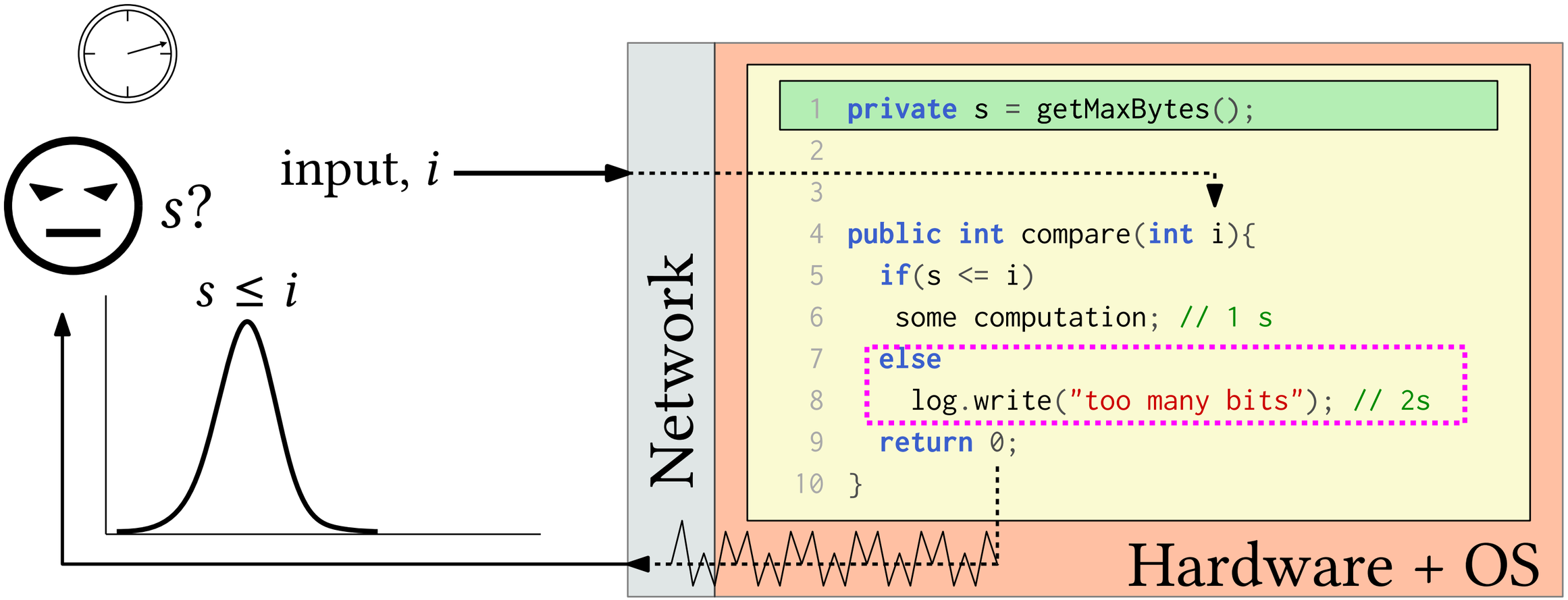

s?
input, i
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

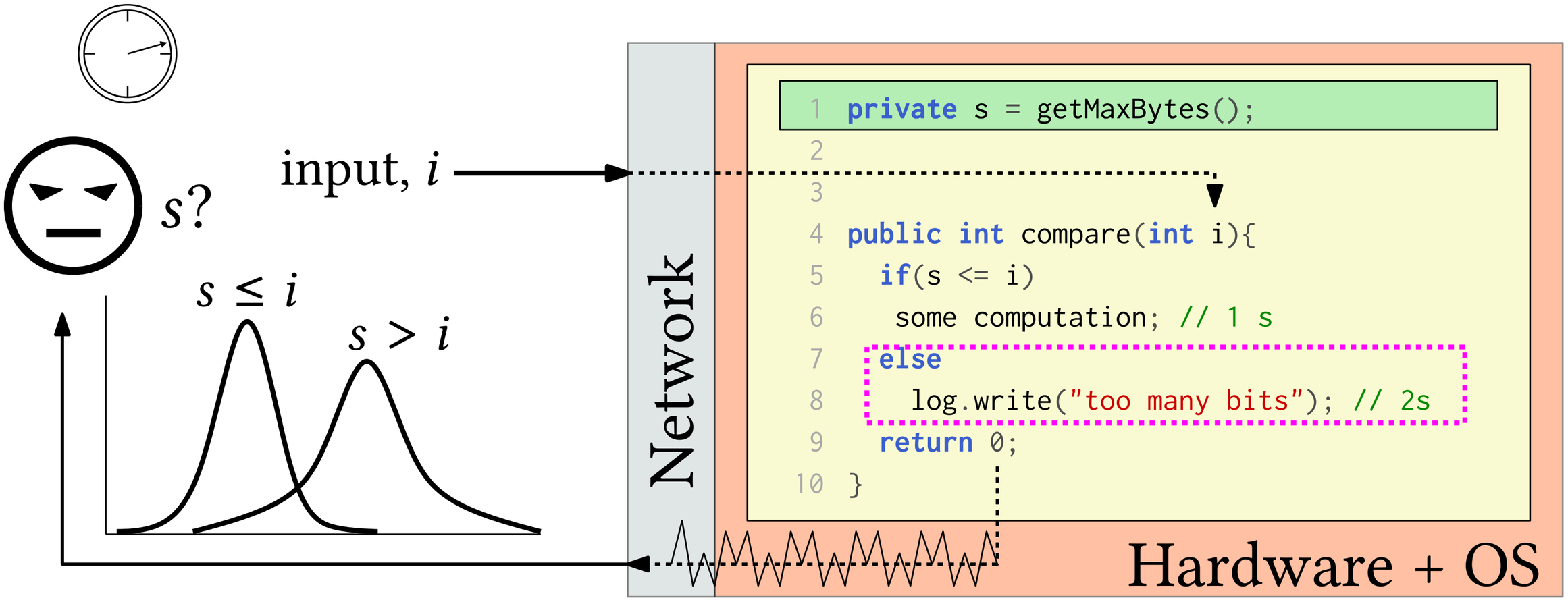


$$s \leq i \Longrightarrow o = 1$$


s?
input, i
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS
s ≤ i ⟶ o = 1


input, i
s?
$s \leq i$
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

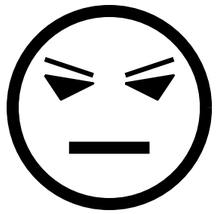

input, i
s?
$s \leq i$
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS


s?
input, i
s ≤ i
s > i
Network
1 private s = getMaxBytes();
2
3
4 public int compare(int i){
5 if(s <= i)
6 some computation; // 1 s
7 else
8 log.write("too many bits"); // 2s
9 return 0;
10 }
Hardware + OS

Challenges: Uncertainty Everywhere

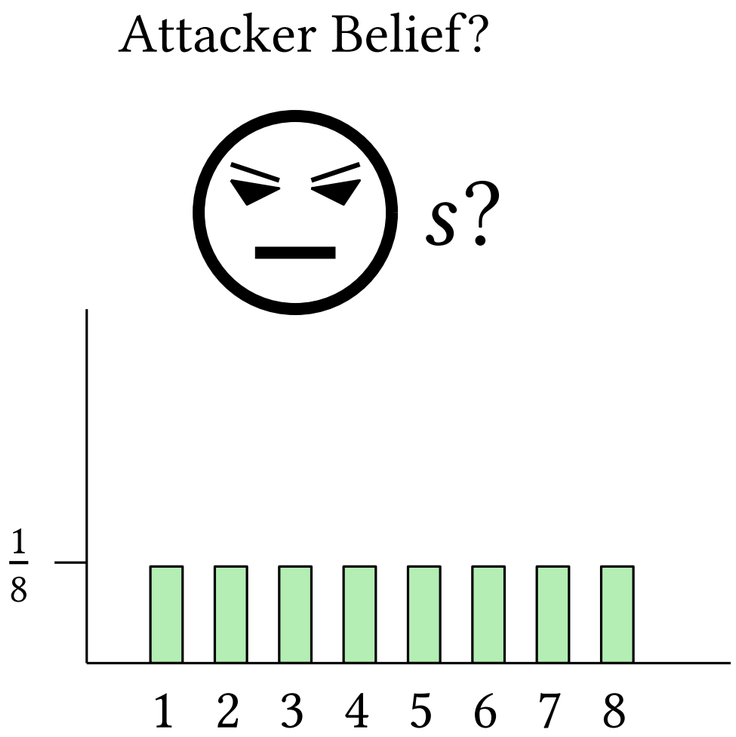
Challenges: Uncertainty Everywhere

Attacker Belief?



s?

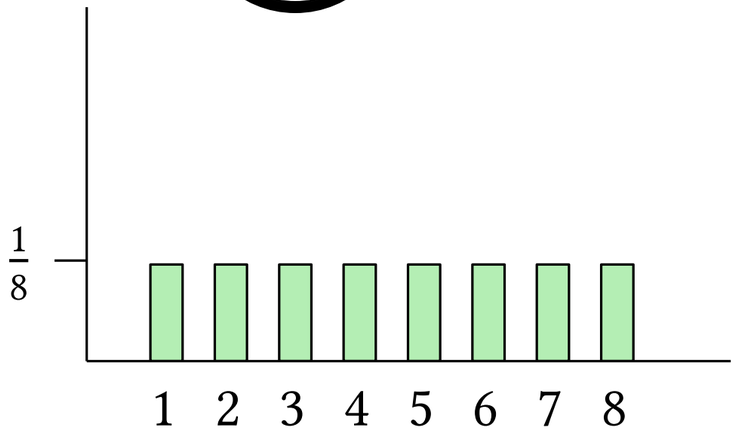
Challenges: Uncertainty Everywhere



Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$

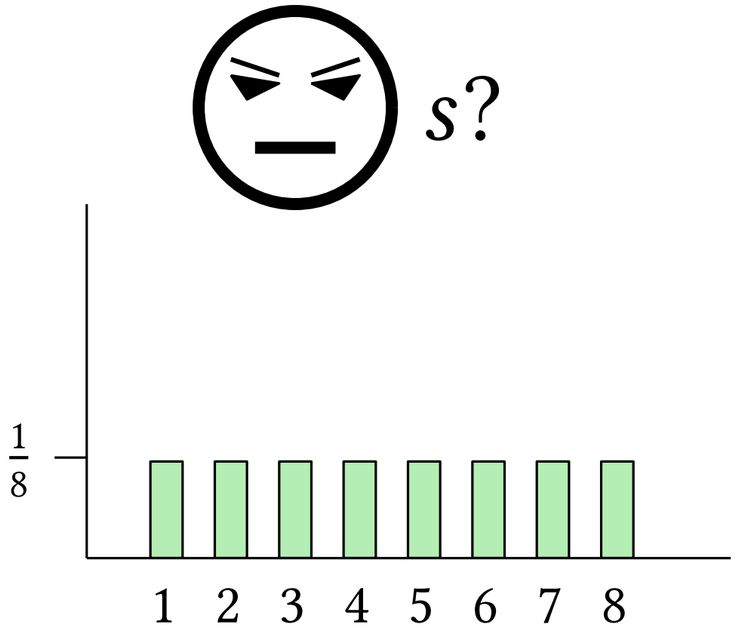


Input Choice?

i^*

Challenges: Uncertainty Everywhere

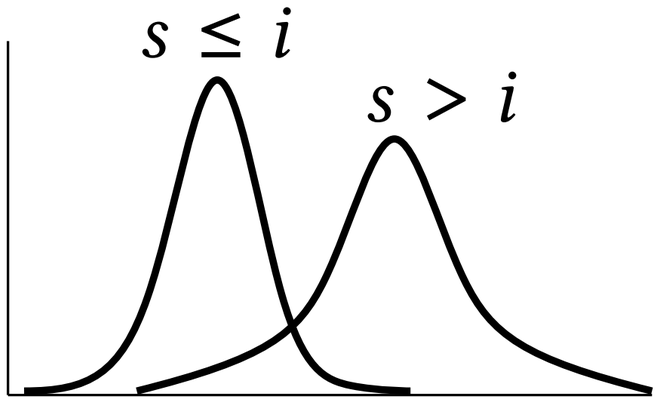
Attacker Belief?



Input Choice?

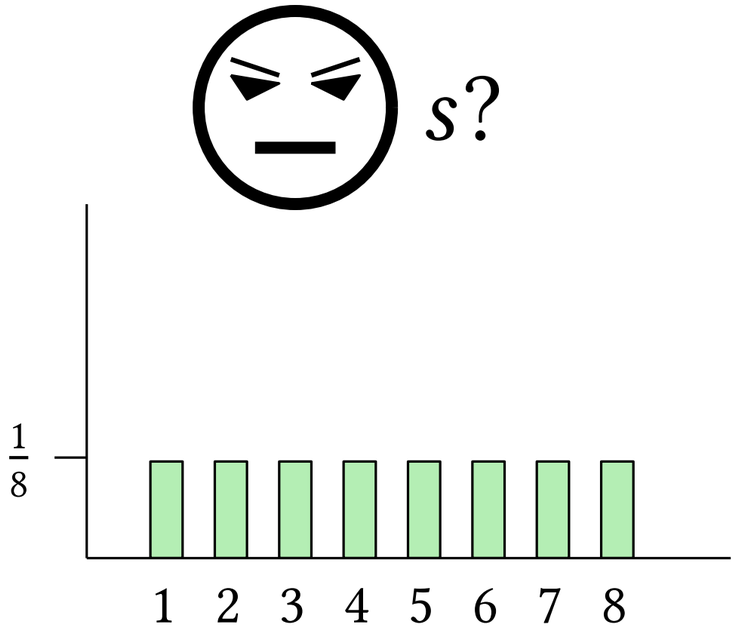
i^*

Observation noise?



Challenges: Uncertainty Everywhere

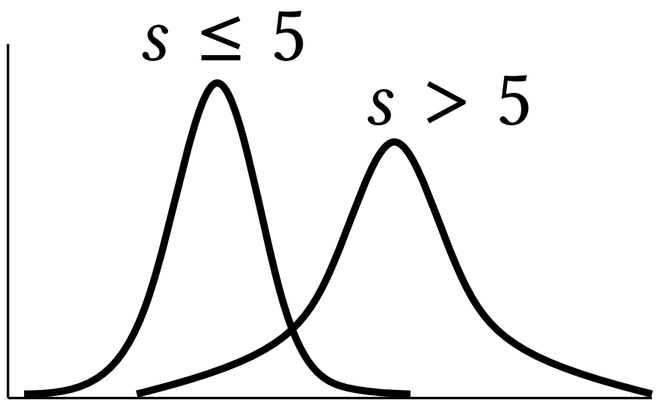
Attacker Belief?



Input Choice?

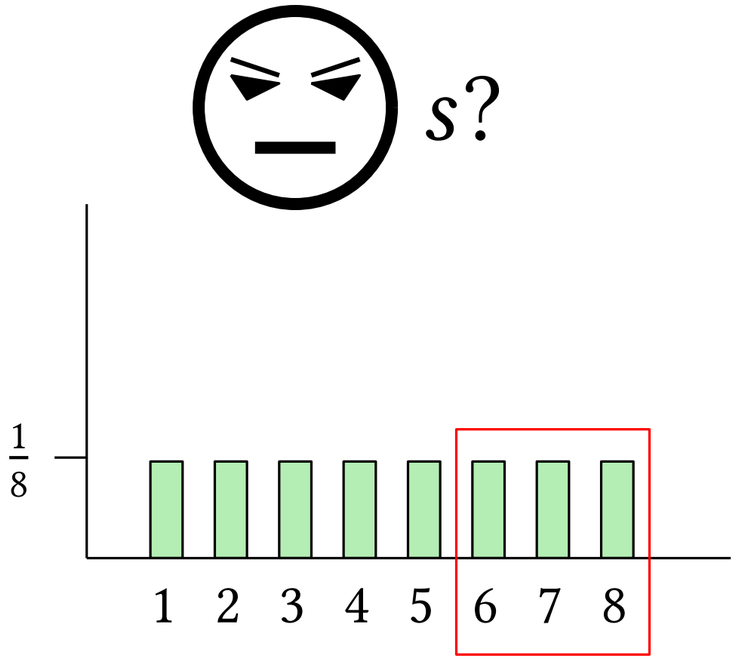
$$i^* = 5$$

Observation noise?



Challenges: Uncertainty Everywhere

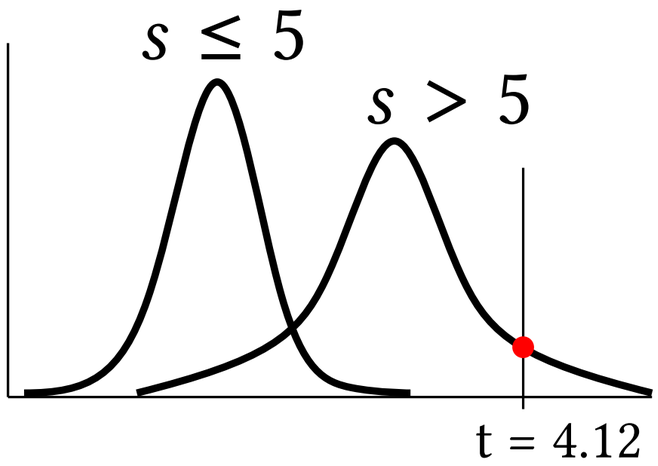
Attacker Belief?



Input Choice?

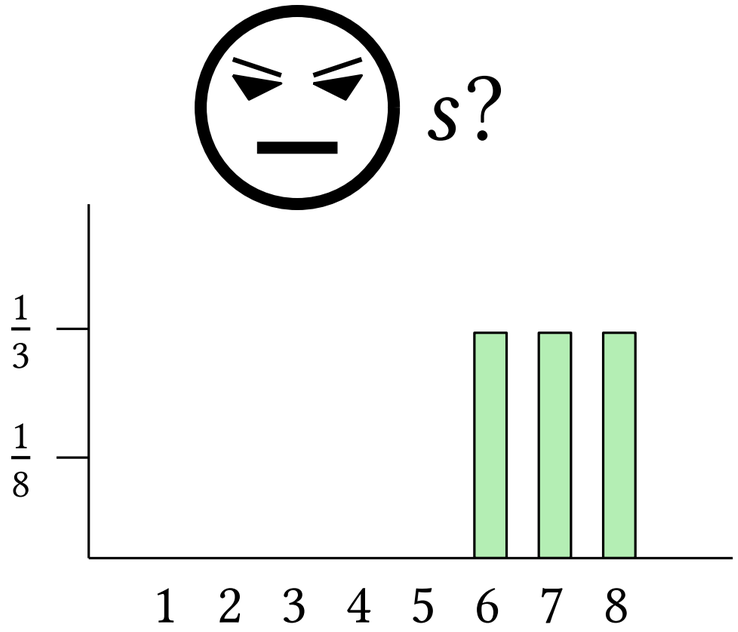
$$i^* = 5$$

Observation noise?



Challenges: Uncertainty Everywhere

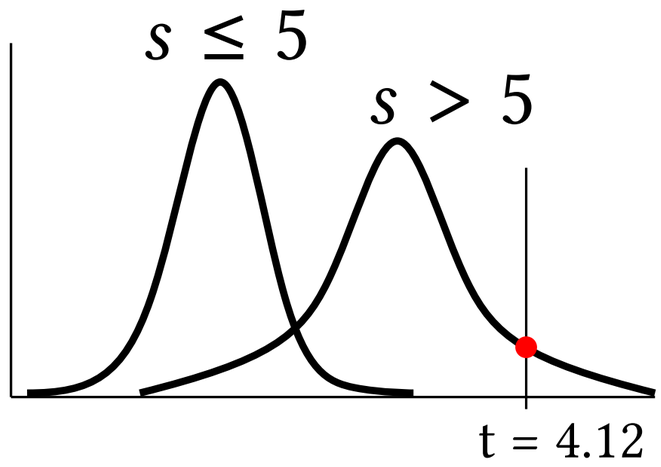
Attacker Belief?



Input Choice?

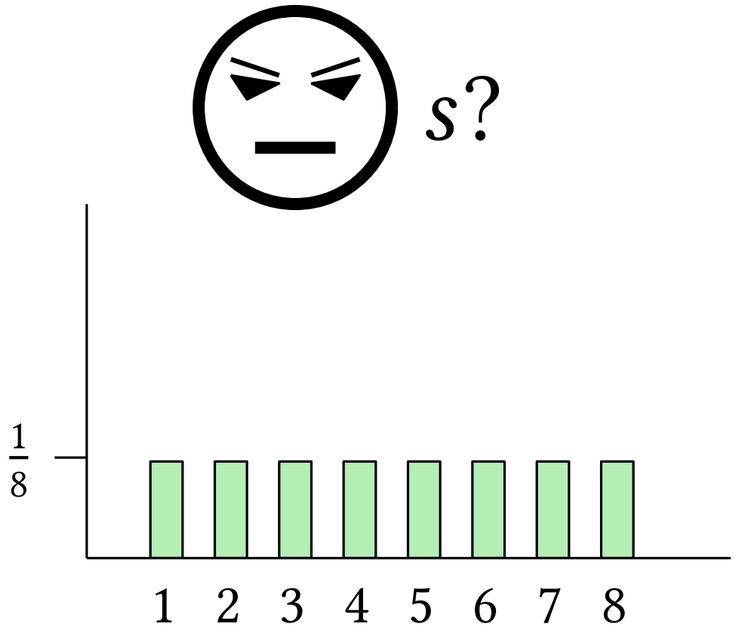
$$i^* = 5$$

Observation noise?



Challenges: Uncertainty Everywhere

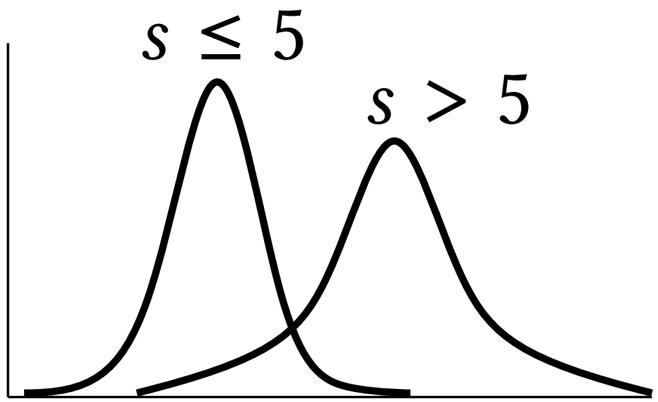
Attacker Belief?



Input Choice?

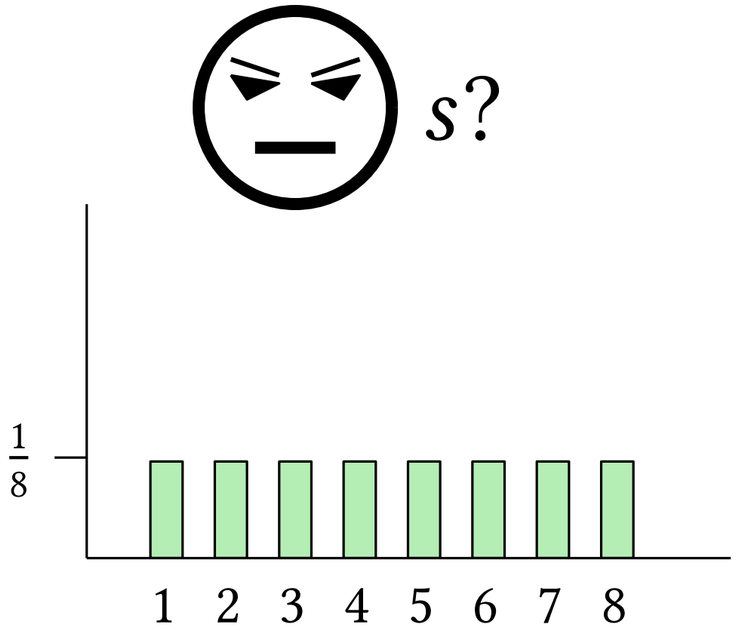
$$i^* = 5$$

Observation noise?



Challenges: Uncertainty Everywhere

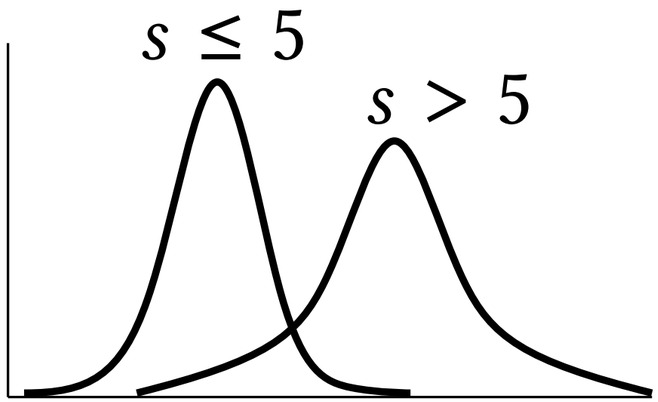
Attacker Belief?



Input Choice?

$$i^* = 5$$

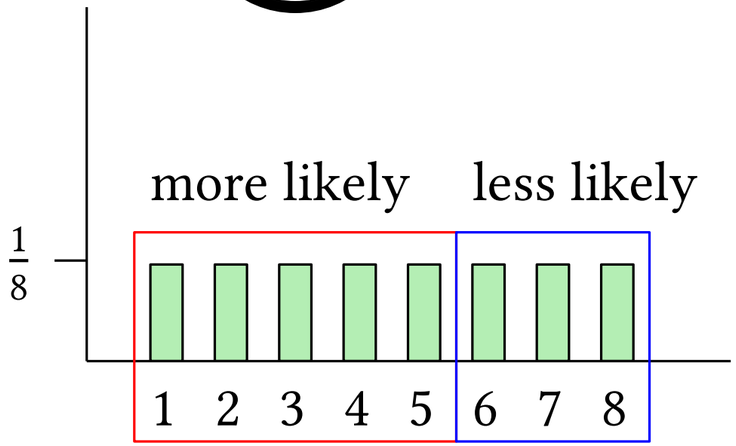
Observation noise?



Challenges: Uncertainty Everywhere

Attacker Belief?

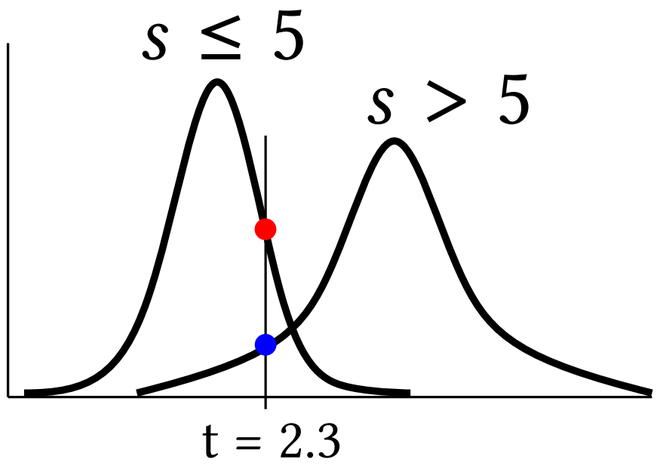
$s?$



Input Choice?

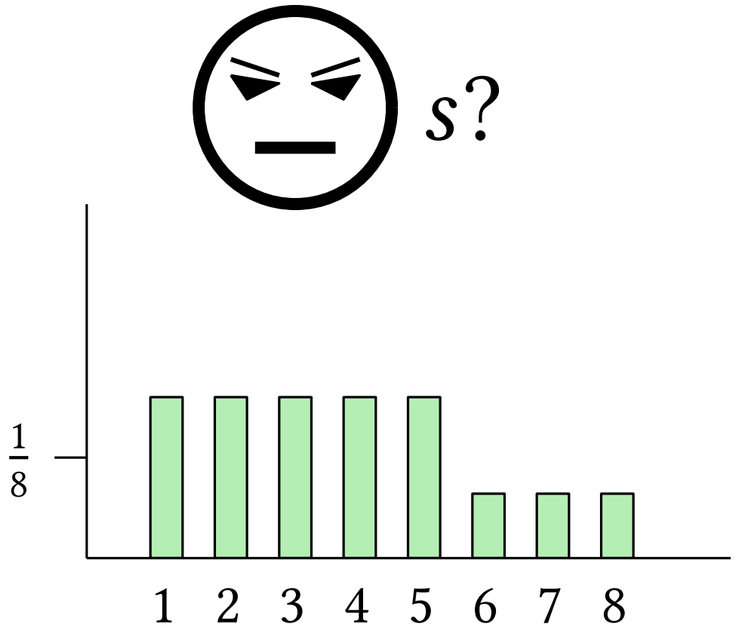
$$i^* = 5$$

Observation noise?



Challenges: Uncertainty Everywhere

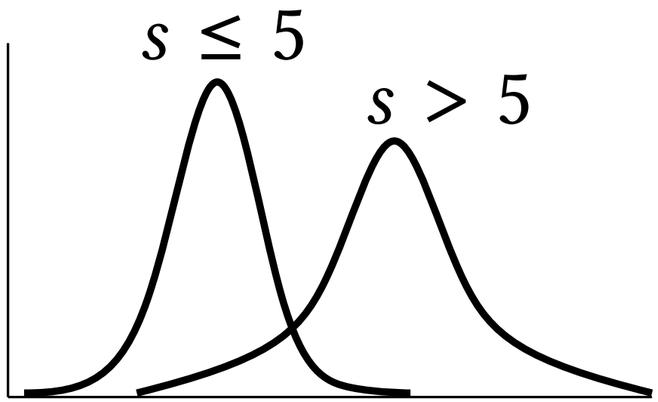
Attacker Belief?



Input Choice?

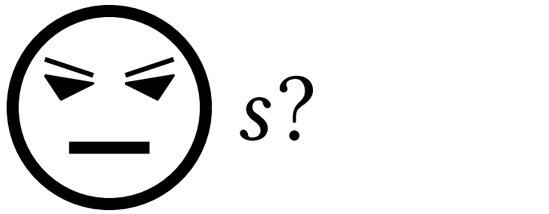
$$i^* = 5$$

Observation noise?

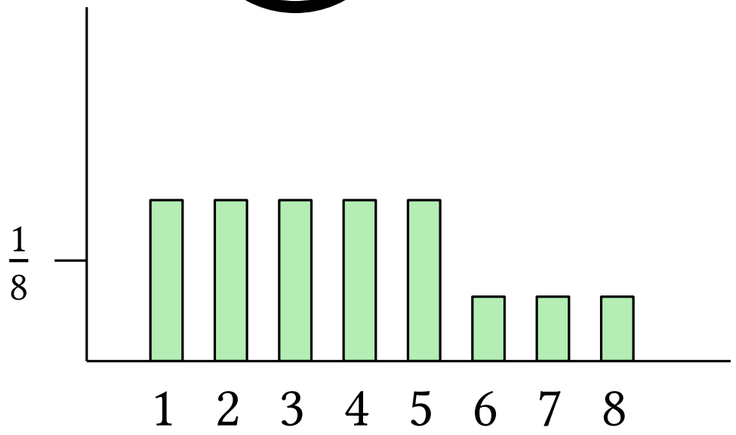


Challenges: Uncertainty Everywhere

Attacker Belief?



$s?$

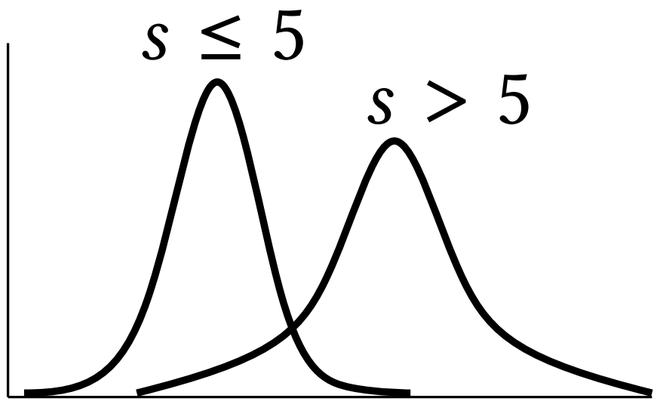


$p(s|o, i^*)$

Input Choice?

$i^* = 5$

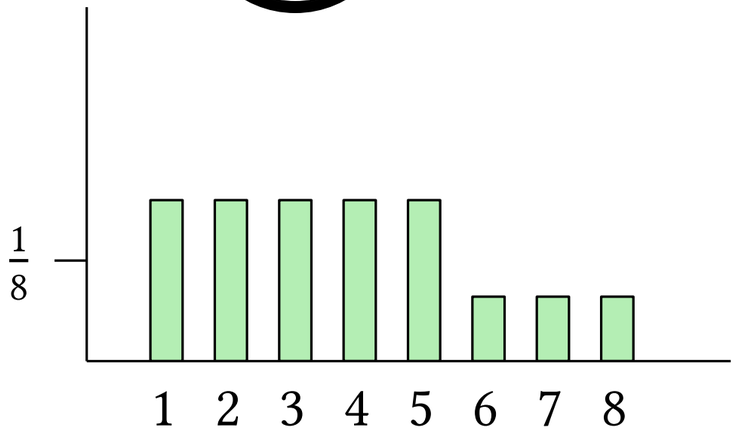
Observation noise?



Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$

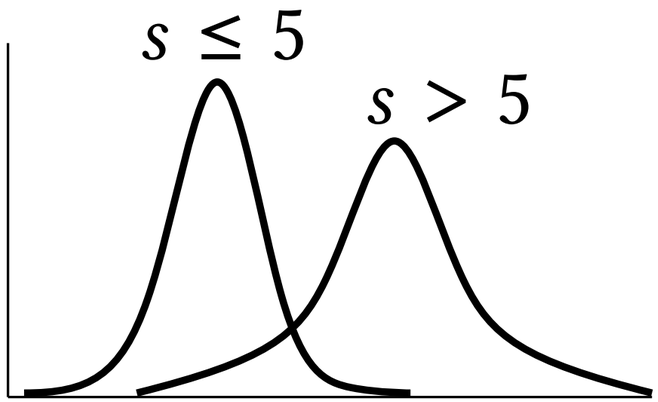


$p(s|o, i^*)$

Input Choice?

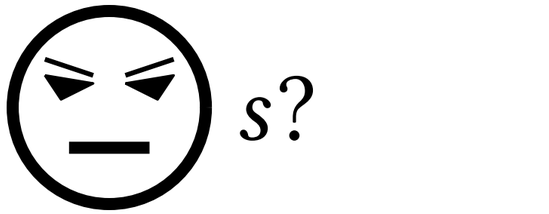
$$i^* = 5$$

Observation noise?

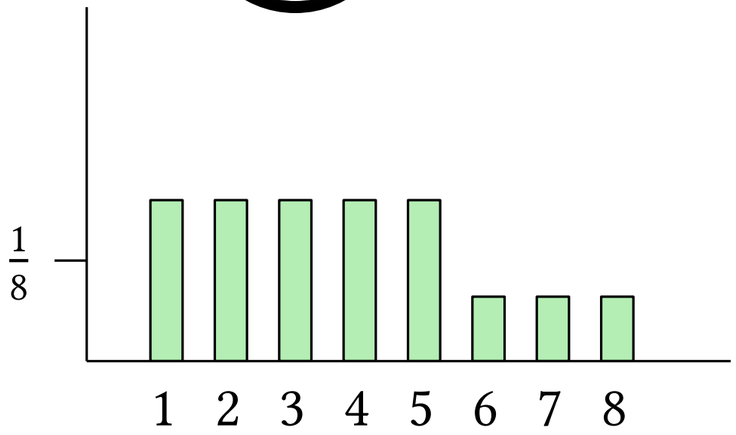


Challenges: Uncertainty Everywhere

Attacker Belief?



$s?$

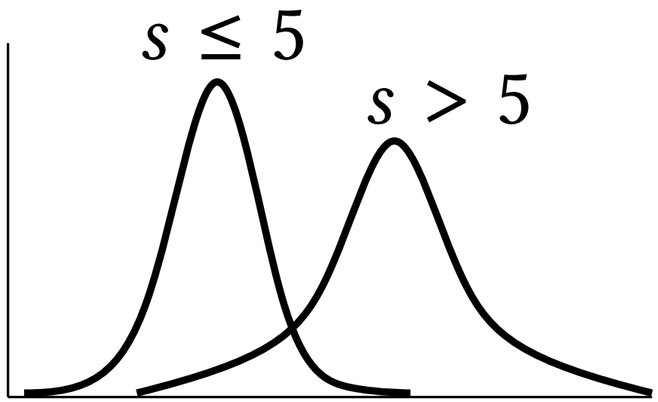


$p(s|o, i^*)$

Input Choice?

$i^* = 5$

Observation noise?

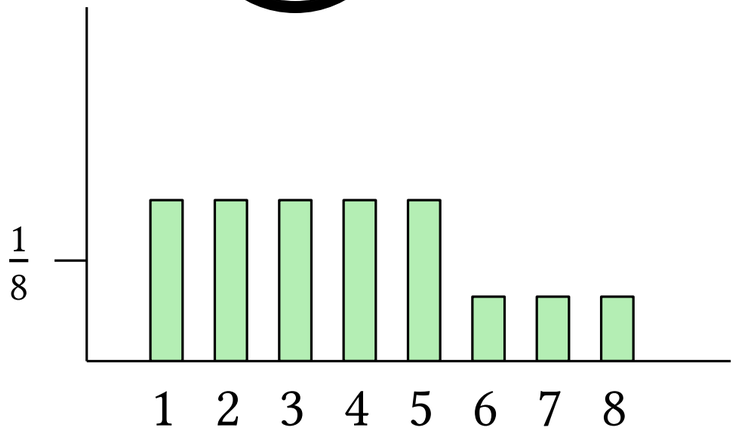


$p(o|s, i)$

Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$

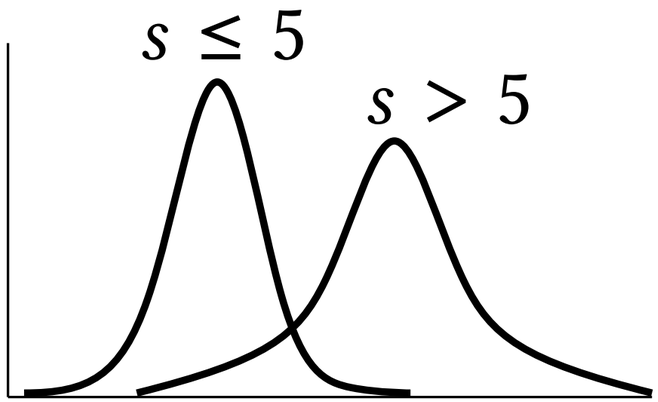


$p(s|o, i^*)$

Input Choice?

$i^* = 5$

Observation noise?

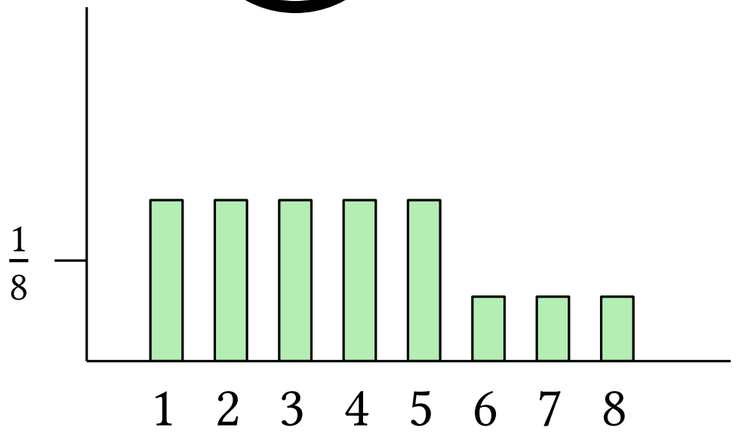


$p(o|s, i)$

Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$



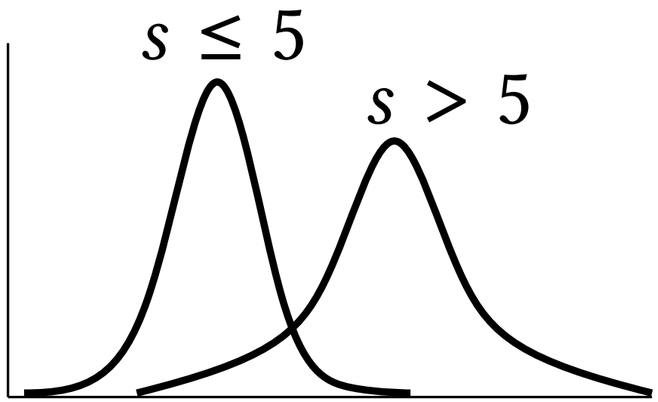
$p(s|o, i^*)$

Input Choice?

$i^* = 5$

$p(o|s, i)$

Observation noise?



$p(o|s, i)$

Challenges: Uncertainty Everywhere

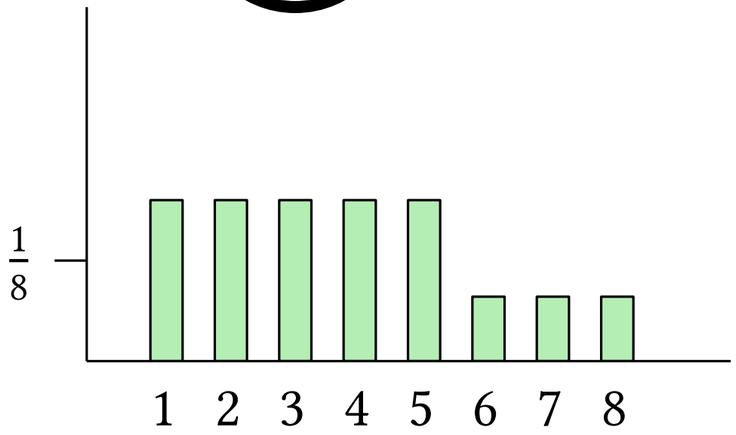
Attacker Belief?

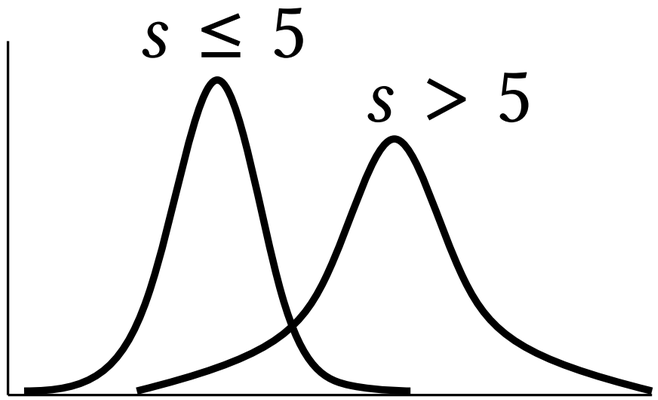
Input Choice?

Observation noise?

$s?$

$i^* = 5$





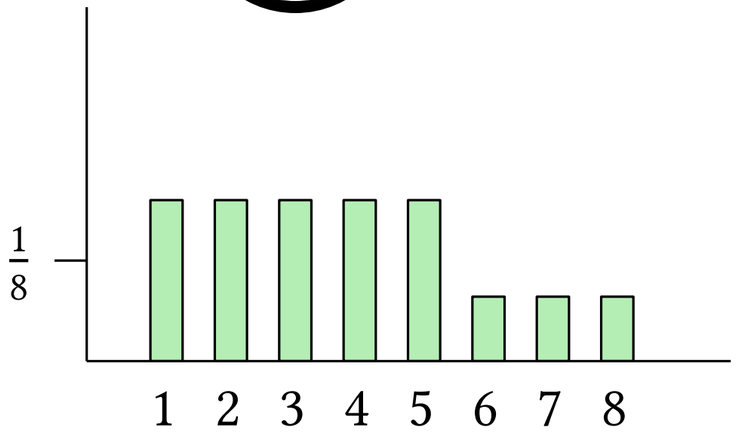
$p(o|s, i^*)$

$p(s|o, i^*)$ ← $p(o|s, i)$

Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$



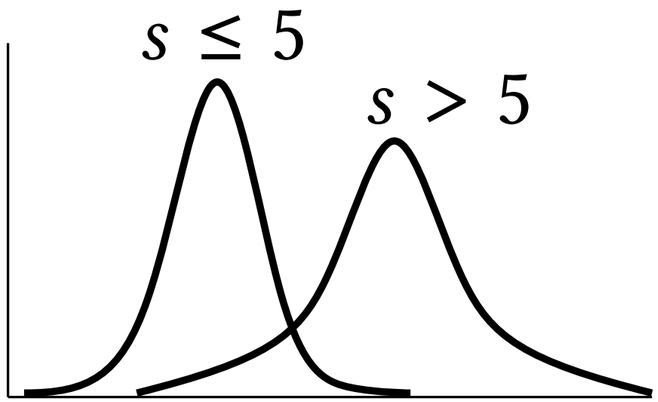
$p(s|o, i^*)$

Input Choice?

$i^* = 5$

$p(o|s, i^*)$

Observation noise?

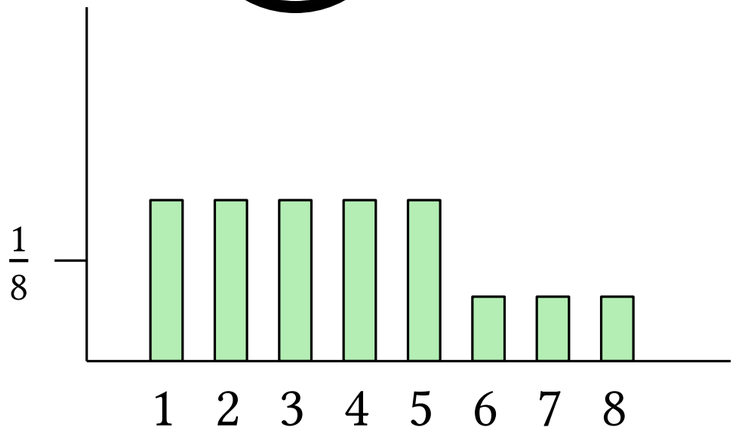


$p(o|s, i)$

Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$



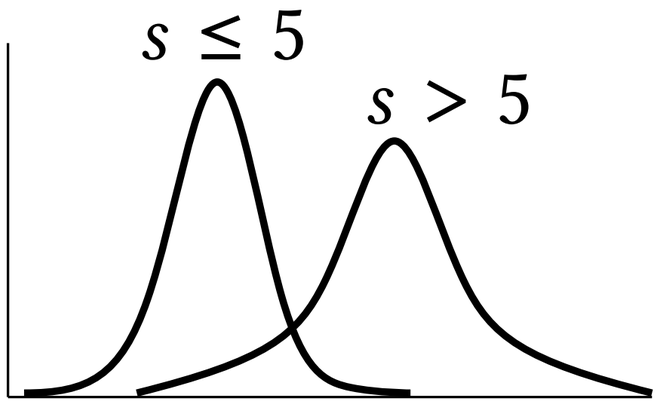
$p(s|o, i^*)$

Input Choice?

$i^* = 5$

$p(s|o, i^*)$

Observation noise?

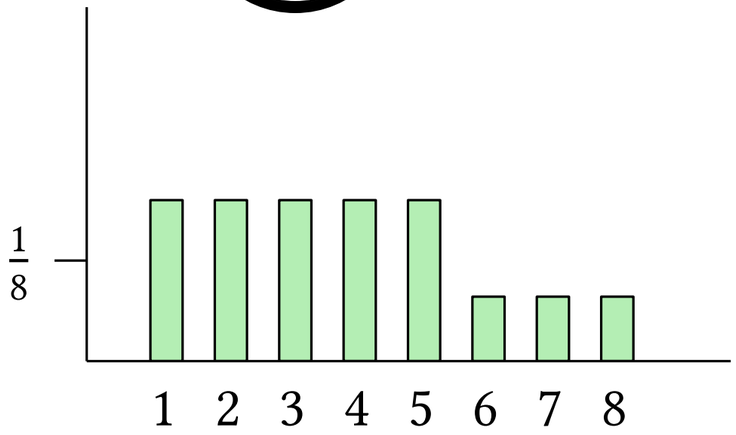


$p(o|s, i)$

Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$

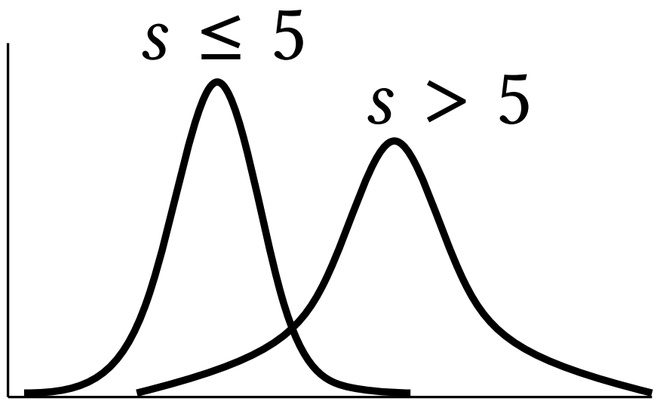


$p(s|o, i^*)$

Input Choice?

$i^* = 5$

Observation noise?



$p(o|s, i)$

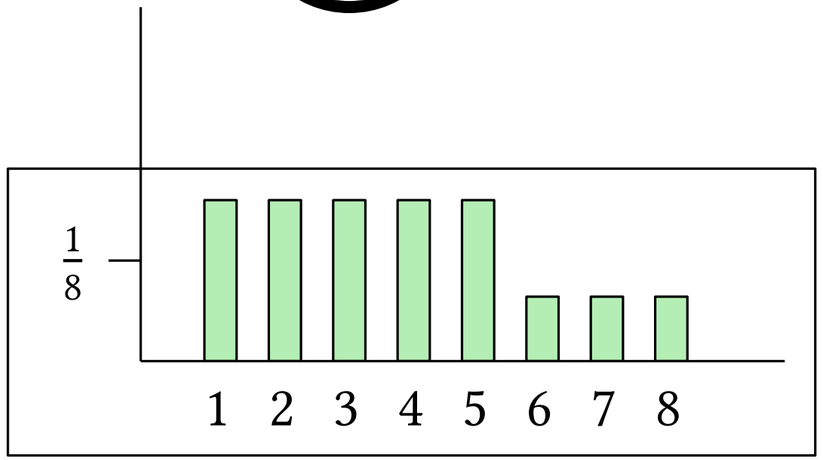
$p(s|o, i^*)$

Bayes Rule

Challenges: Uncertainty Everywhere

Attacker Belief?

$s?$

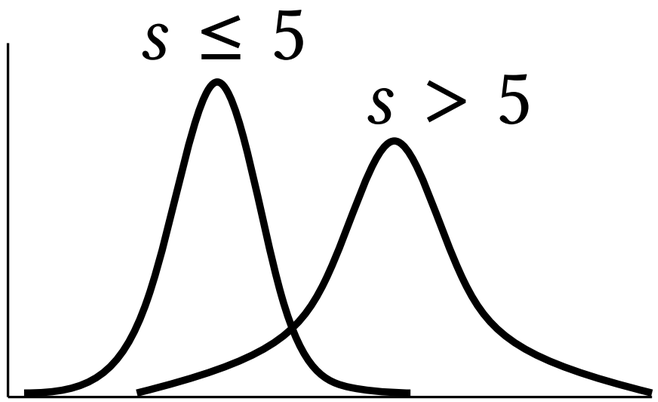


$p(s|o, i^*)$

Input Choice?

$i^* = 5$

Observation noise?



$p(o|s, i)$

$p(s|o, i^*)$

Bayes Rule

Challenges: Uncertainty Everywhere

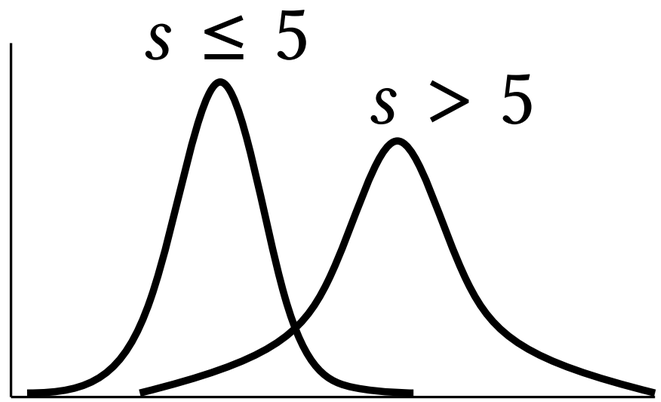
Attacker Belief?

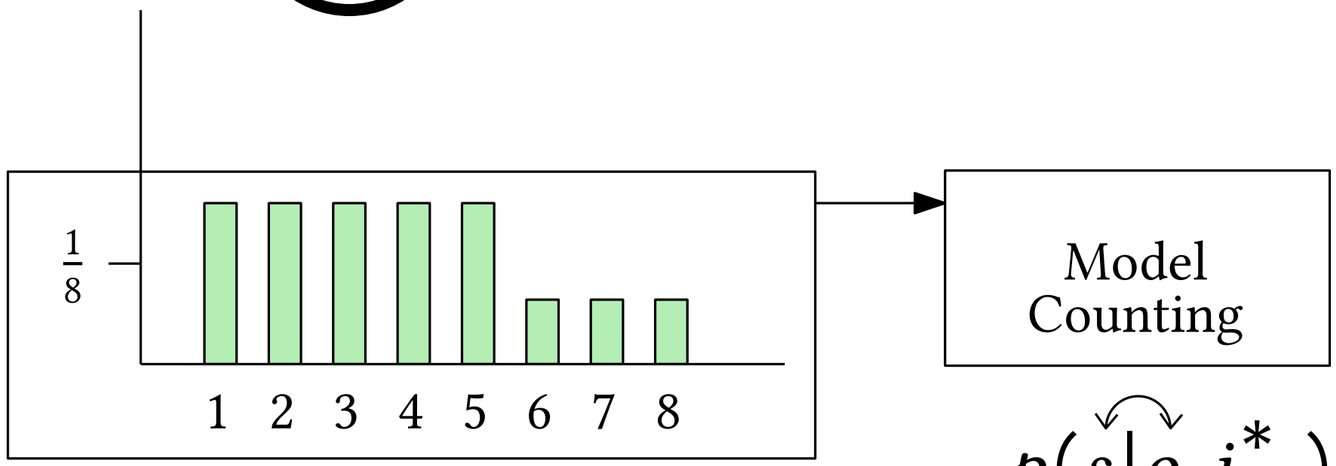
Input Choice?

Observation noise?

$s?$

$i^* = 5$





Model Counting

$p(s|o, i^*)$

$p(s|o, i^*)$

Bayes Rule

$p(o|s, i)$

Challenges: Uncertainty Everywhere

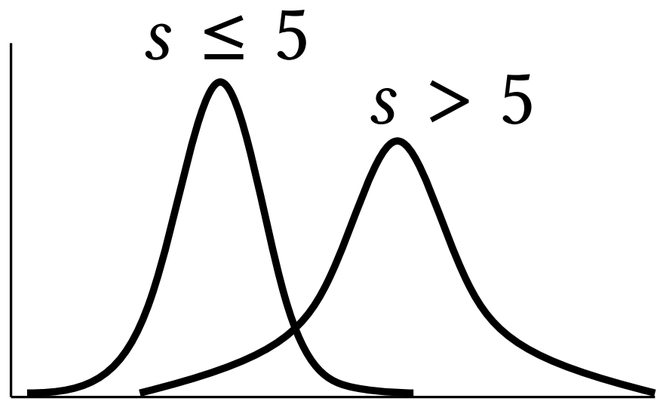
Attacker Belief?

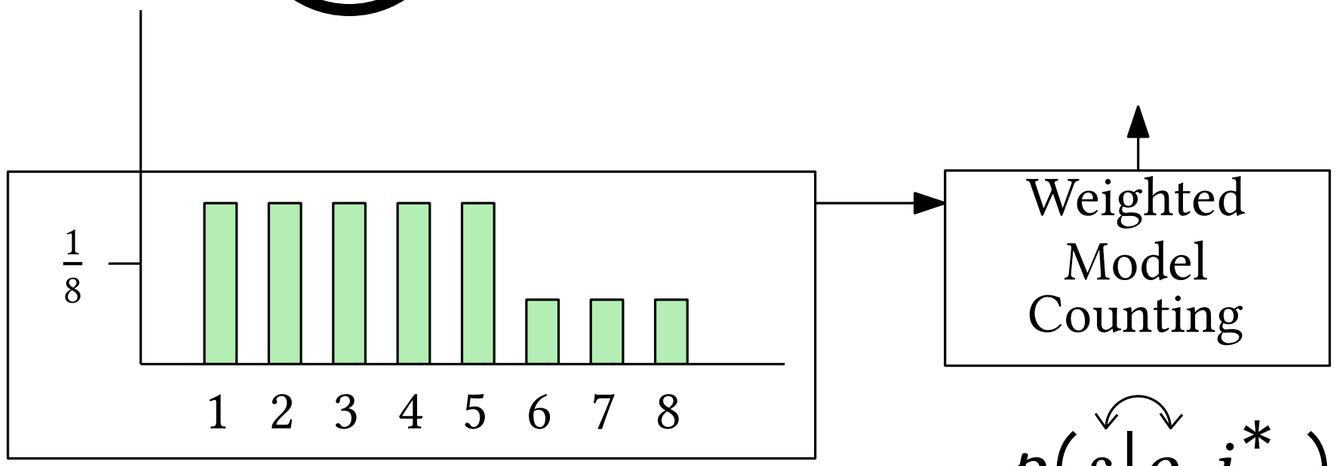
Input Choice?

Observation noise?

$s?$

$i^* = 5$





Weighted Model Counting

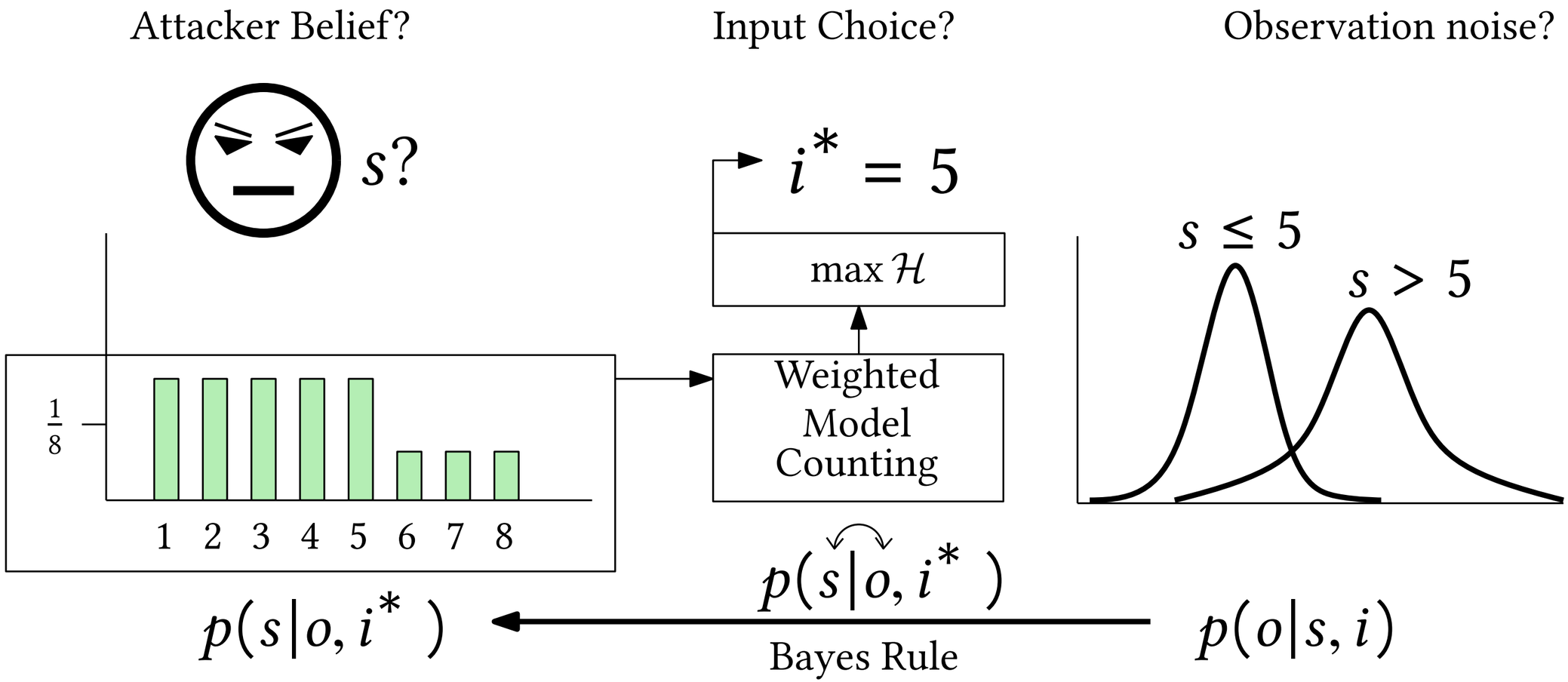
$p(s|o, i^*)$

$p(s|o, i^*)$

$p(o|s, i)$

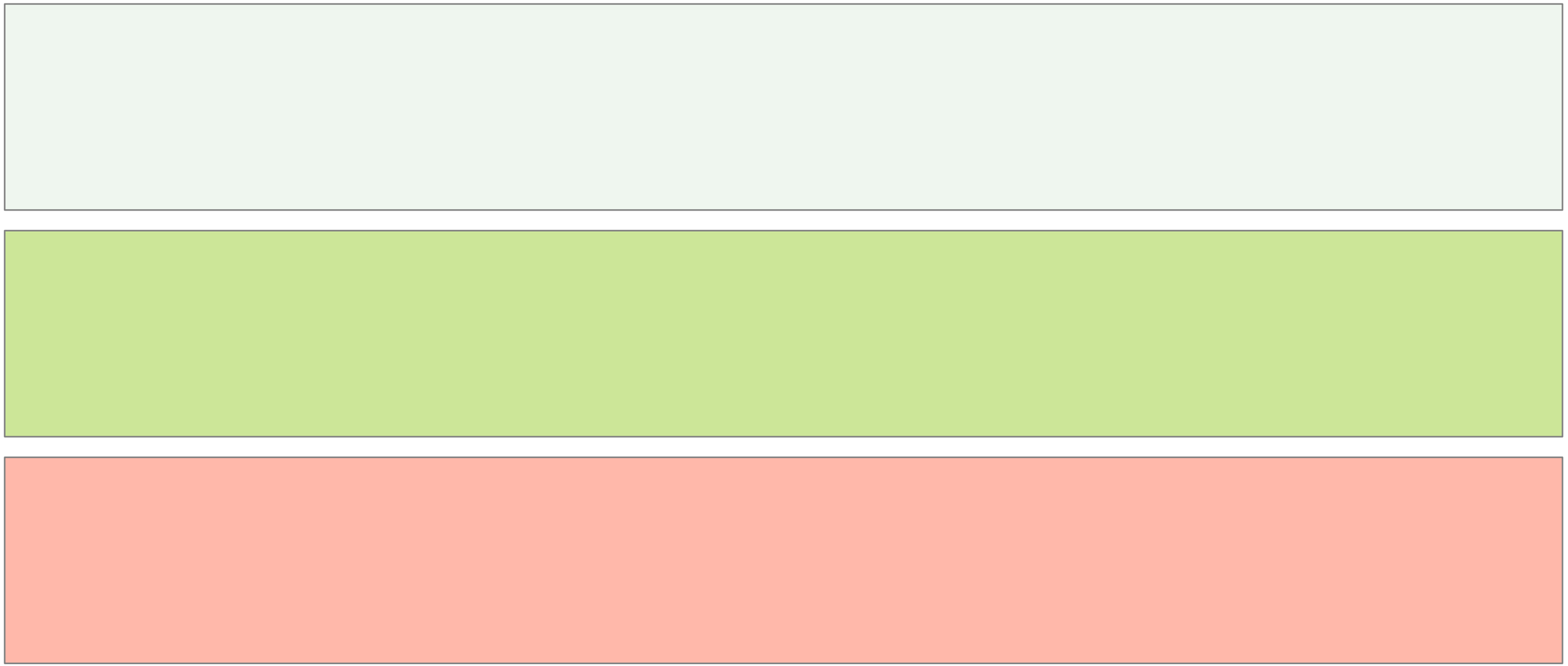
Bayes Rule

Challenges: Uncertainty Everywhere



Proposed Approach

Proposed Approach



Proposed Approach

1. Offline Static Analysis

Proposed Approach

1. Offline Static Analysis

2. Offline Dynamic Analysis

Proposed Approach

1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis

Proposed Approach

1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis

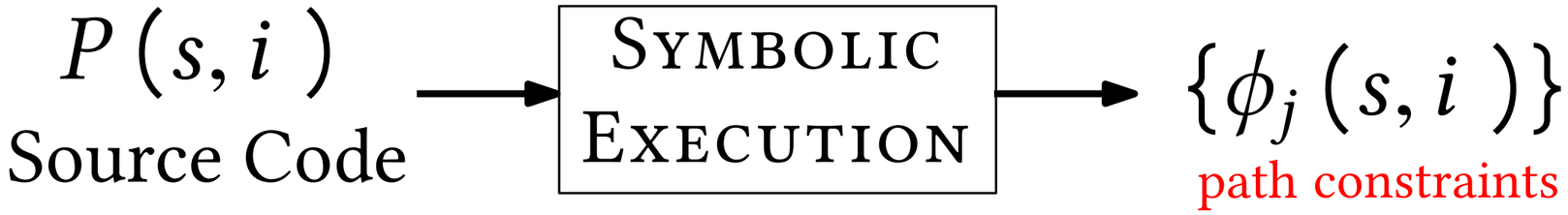
Proposed Approach

Proposed Approach

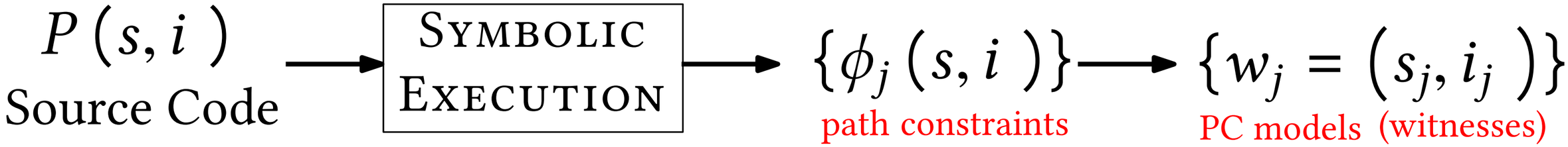
$$P(s, i)$$

Source Code

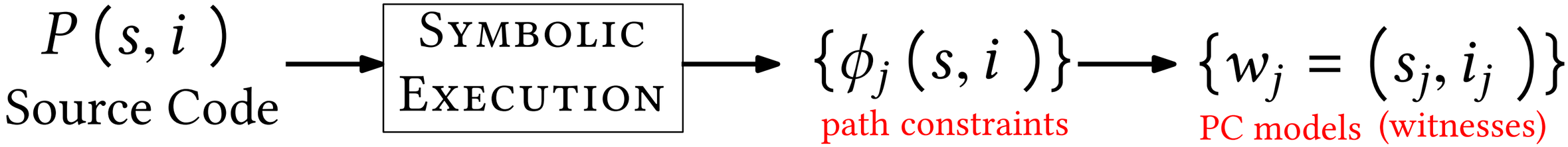
Proposed Approach



Proposed Approach

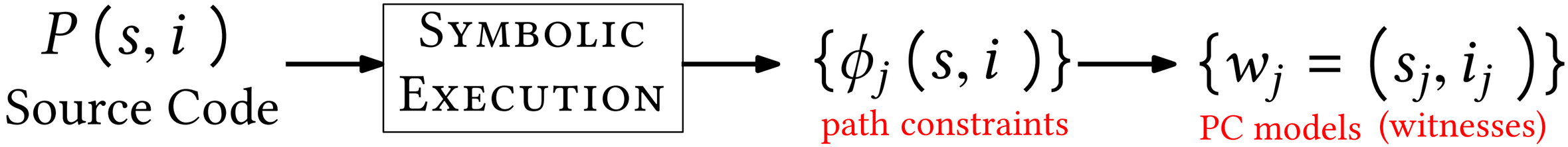


Proposed Approach



Idea: each PC characterizes an observable program behavior

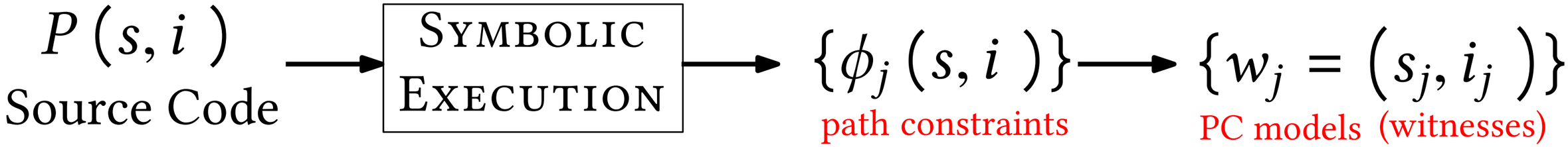
Proposed Approach



Idea: each PC characterizes an observable program behavior

$$(s_j, i_j) \models \phi_j \qquad (s'_j, i'_j) \models \phi_j$$

Proposed Approach

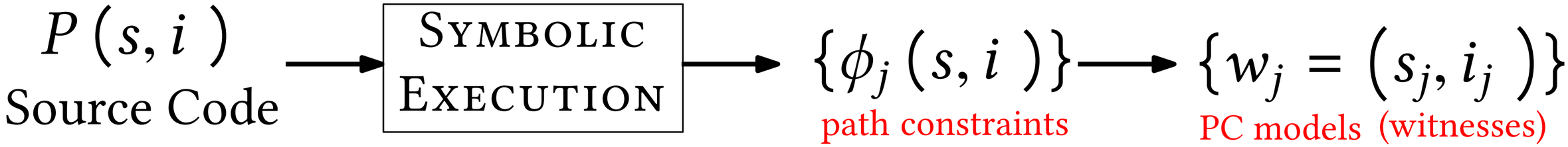


Idea: each PC characterizes an observable program behavior

$$(s_j, i_j) \models \phi_j \qquad (s'_j, i'_j) \models \phi_j$$

$$P(s_j, i_j) \qquad P(s'_j, i'_j)$$

Proposed Approach

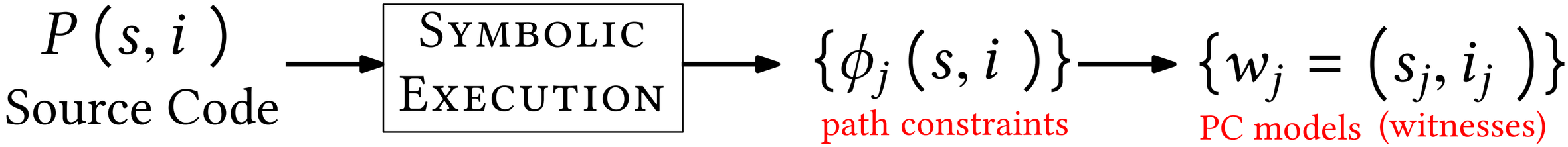


Idea: each PC characterizes an observable program behavior

$$(s_j, i_j) \models \phi_j \qquad (s'_j, i'_j) \models \phi_j$$

$$P(s_j, i_j) \quad ? \; 😒 \; ? \quad P(s'_j, i'_j)$$

Proposed Approach



Idea: each PC characterizes an observable program behavior

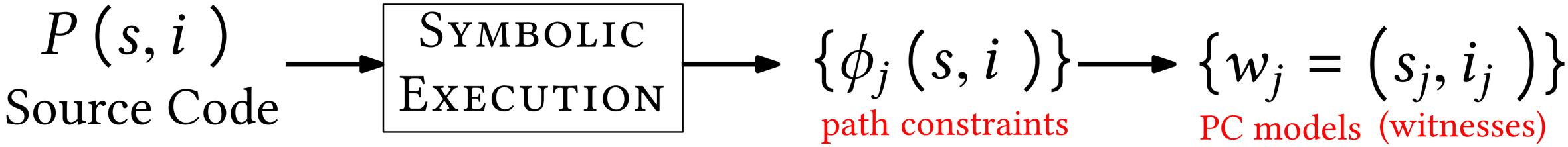
$$(s_j, i_j) \models \phi_j \qquad (s'_j, i'_j) \models \phi_j$$

$$P(s_j, i_j) \quad ? \; 😒 \; ? \quad P(s'_j, i'_j)$$

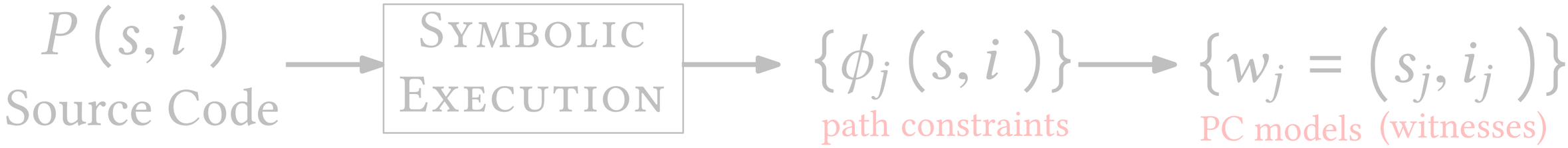
$\phi_j(s, i)$ characterizes observationally indistinguishable behaviors

$P(s_j, i_j)$ is a representative of all behaviors in that class

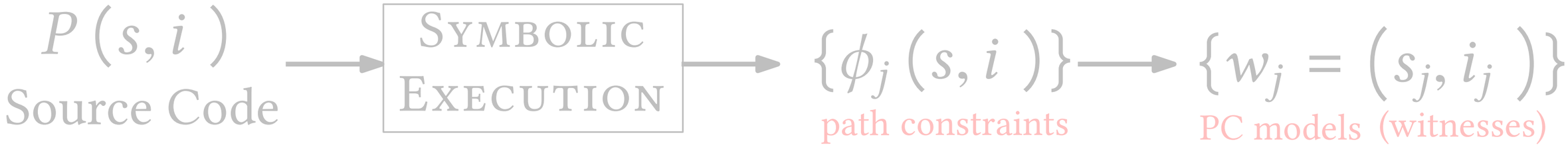
Proposed Approach



Proposed Approach



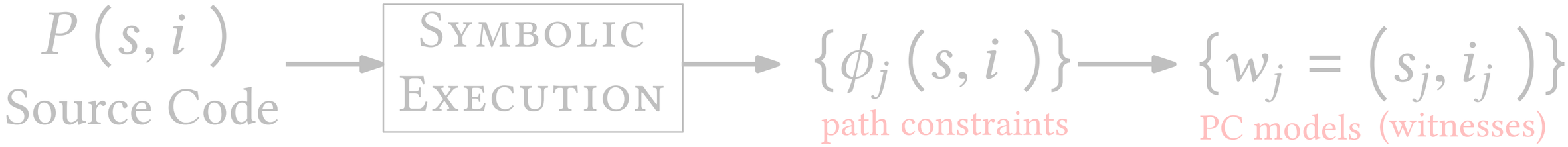
Proposed Approach



2. Offline Dynamic Analysis

3. Online Attack Synthesis

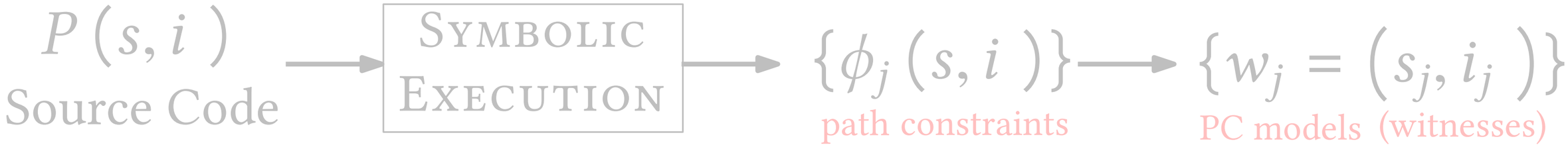
Proposed Approach



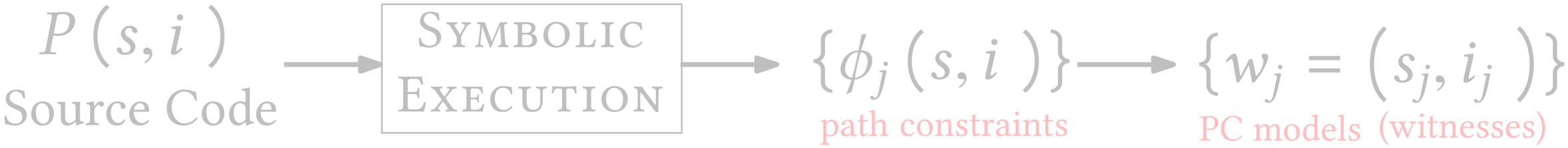
2. Offline Dynamic Analysis

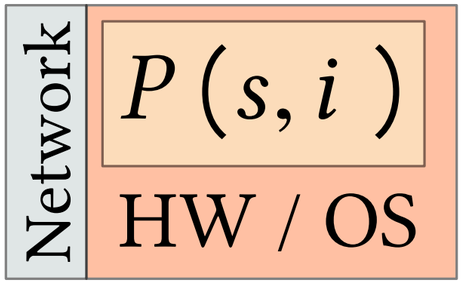
3. Online Attack Synthesis

Proposed Approach



Proposed Approach





Proposed Approach

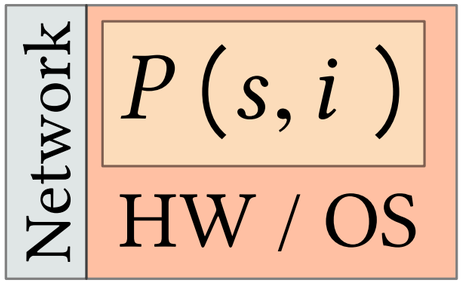
$P(s, i)$
Source Code

→ SYMBOLIC EXECUTION →

$\{\phi_j(s, i)\}$
path constraints

→ $\{w_j = (s_j, i_j)\}$
PC models (witnesses)

$$\{w_j = (s_i, i_j)\}$$



Proposed Approach

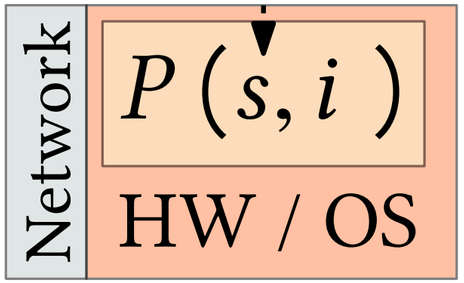
$P(s, i)$
Source Code

→ SYMBOLIC EXECUTION →

$\{\phi_j(s, i)\}$
path constraints

→ $\{w_j = (s_j, i_j)\}$
PC models (witnesses)

$\{w_j = (s_i, i_j)\}$



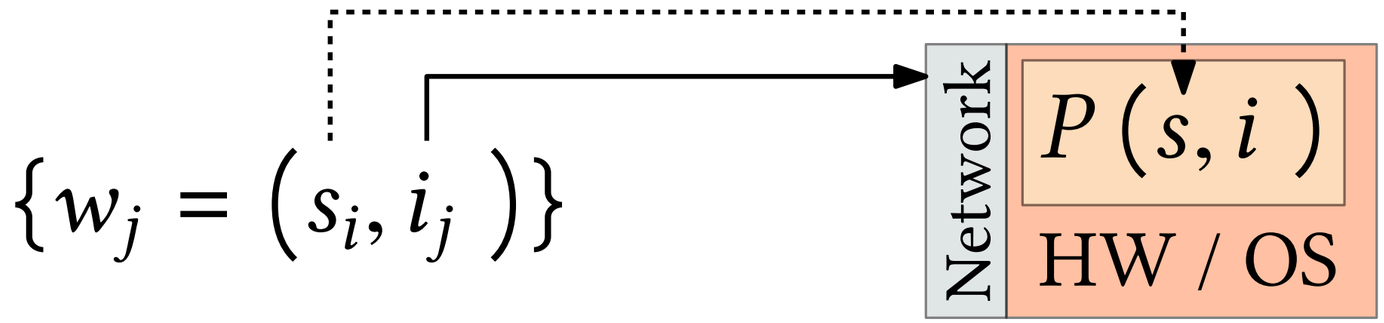
Proposed Approach

$P(s, i)$
Source Code

→ SYMBOLIC EXECUTION →

$\{\phi_j(s, i)\}$
path constraints

→ $\{w_j = (s_j, i_j)\}$
PC models (witnesses)



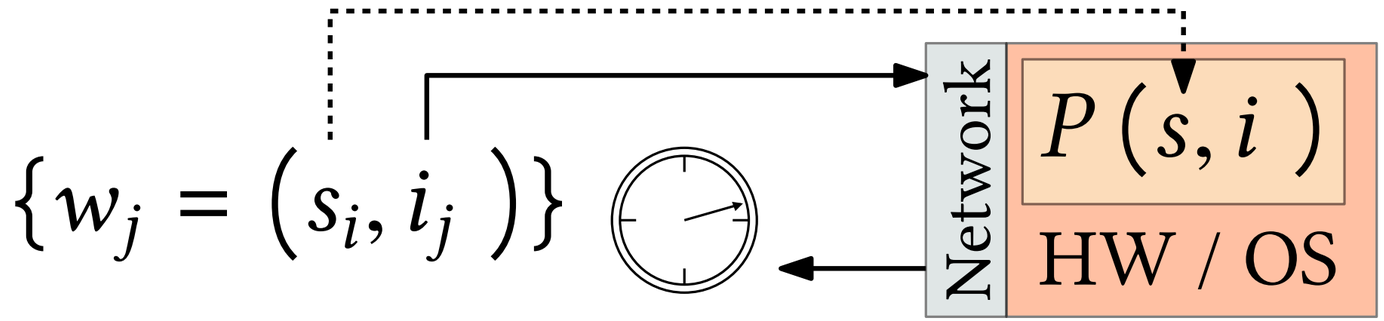
Proposed Approach

$P(s, i)$
Source Code

→ SYMBOLIC EXECUTION →

$\{\phi_j(s, i)\}$
path constraints

→ $\{w_j = (s_j, i_j)\}$
PC models (witnesses)



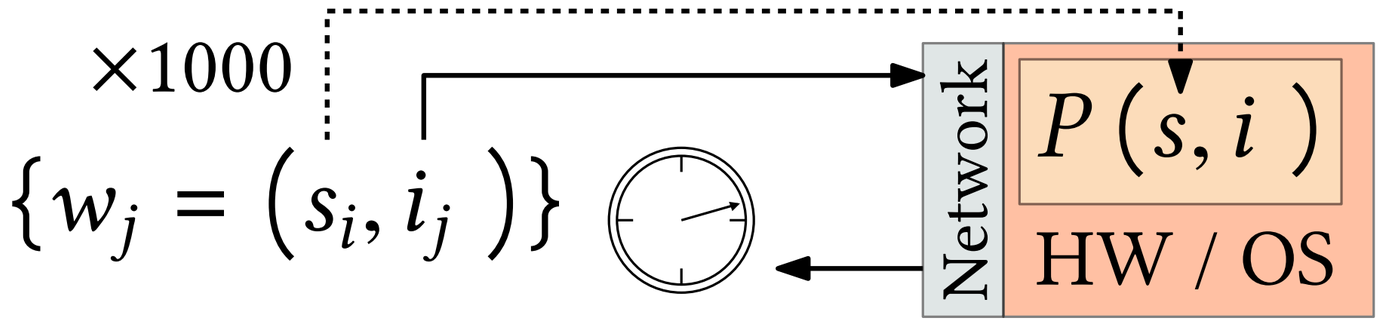
Proposed Approach

$P(s, i)$
Source Code

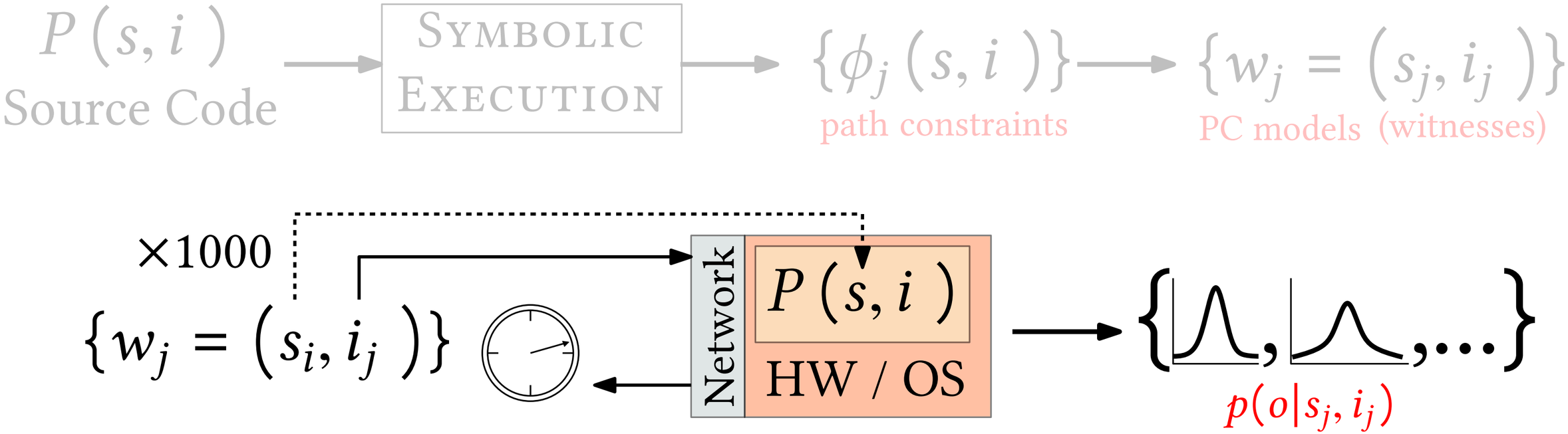
→ SYMBOLIC EXECUTION →

$\{\phi_j(s, i)\}$
path constraints

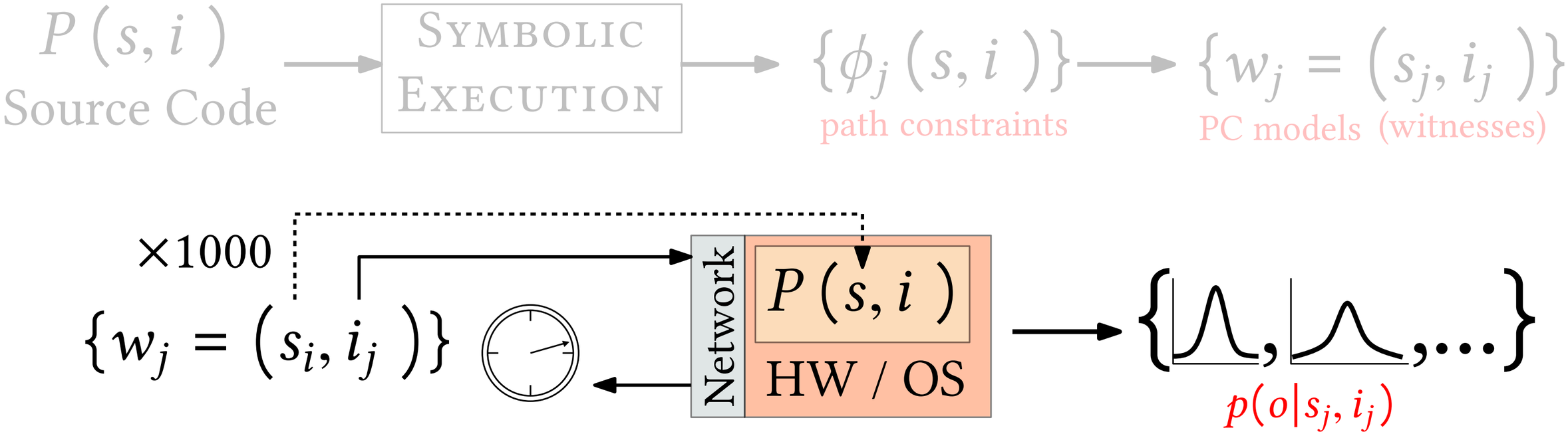
→ $\{w_j = (s_j, i_j)\}$
PC models (witnesses)



Proposed Approach

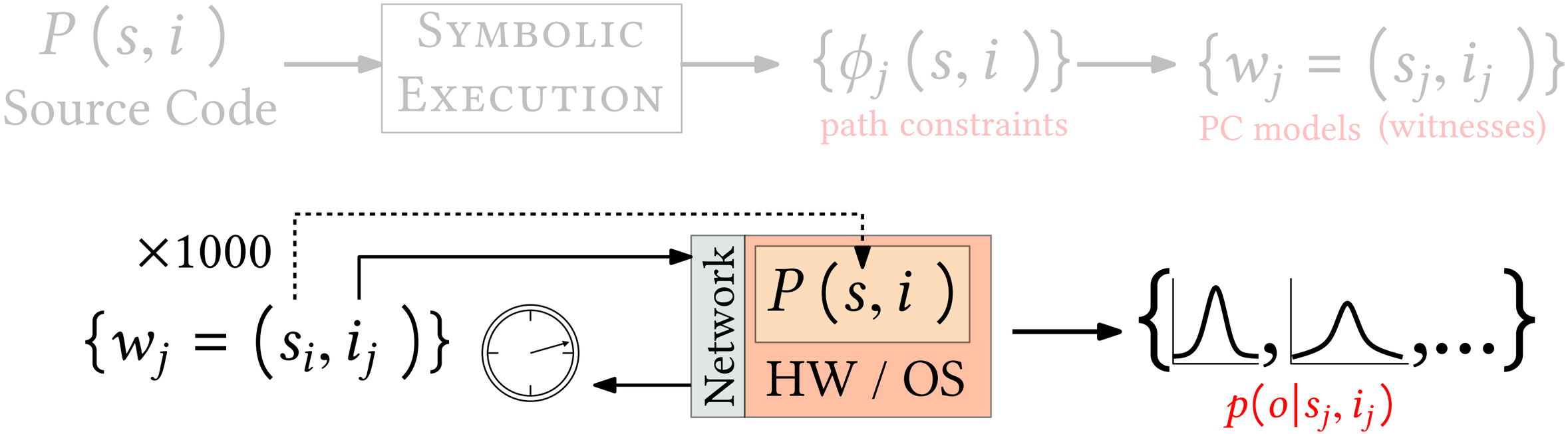


Proposed Approach

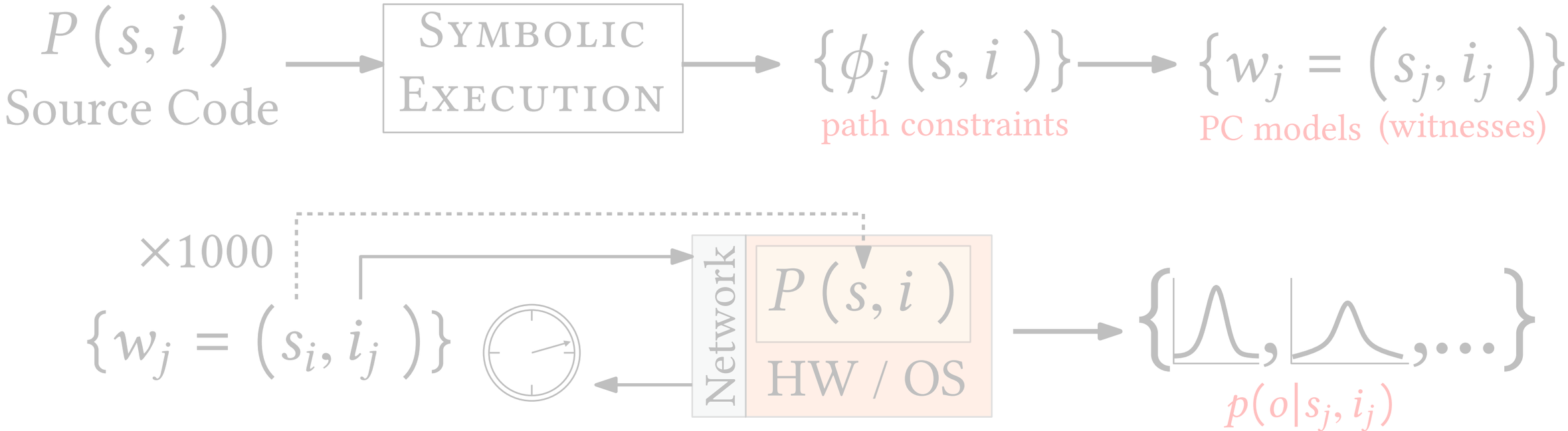


Idea: characterize effect of noise on each class of program behaviors using the witness for that behavior.

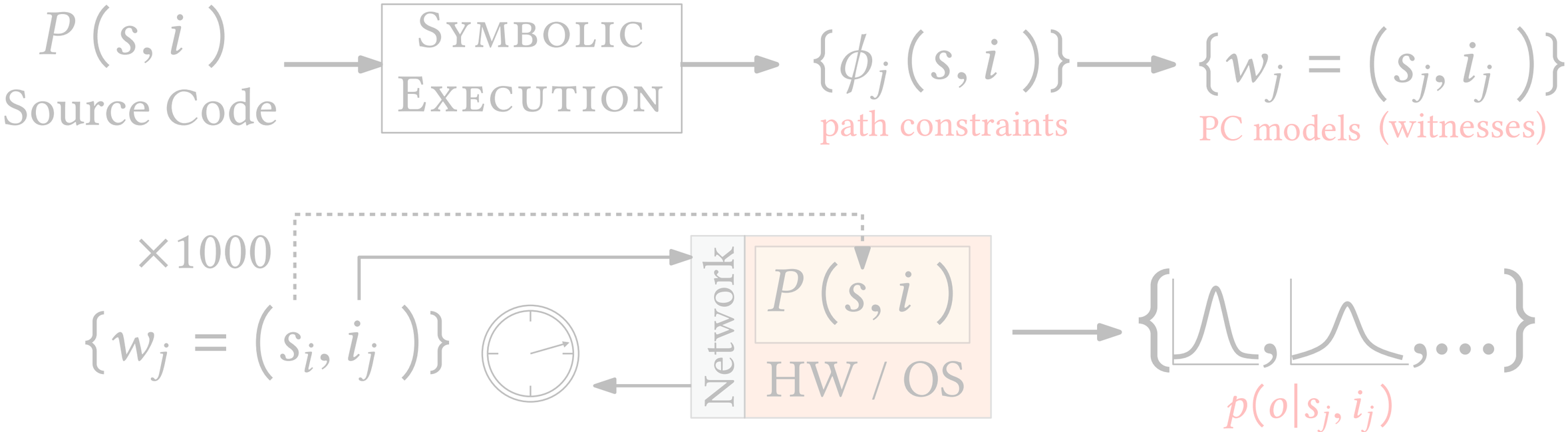
Proposed Approach



Proposed Approach

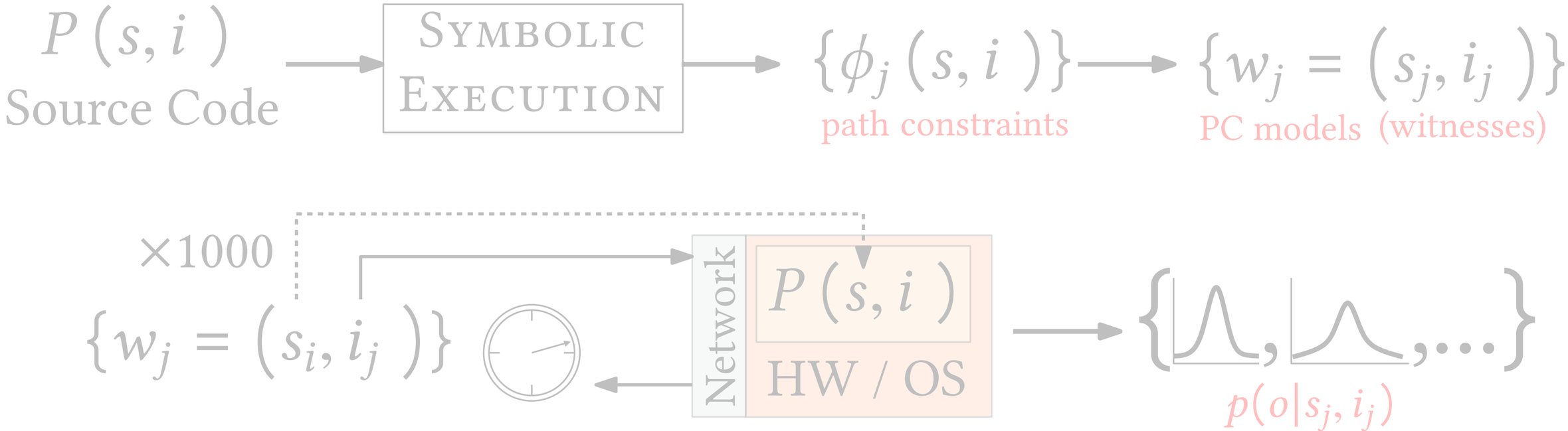


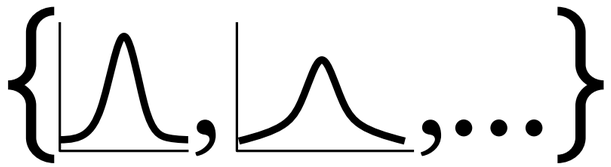
Proposed Approach



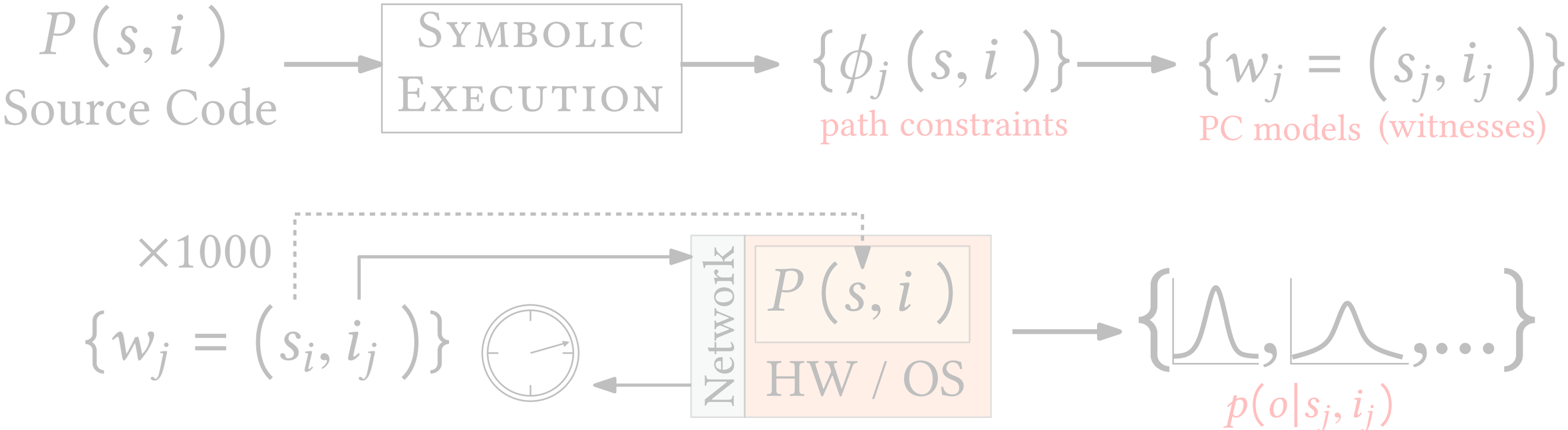
3. Online Attack Synthesis

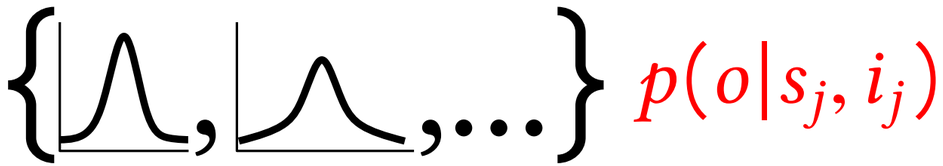
Proposed Approach





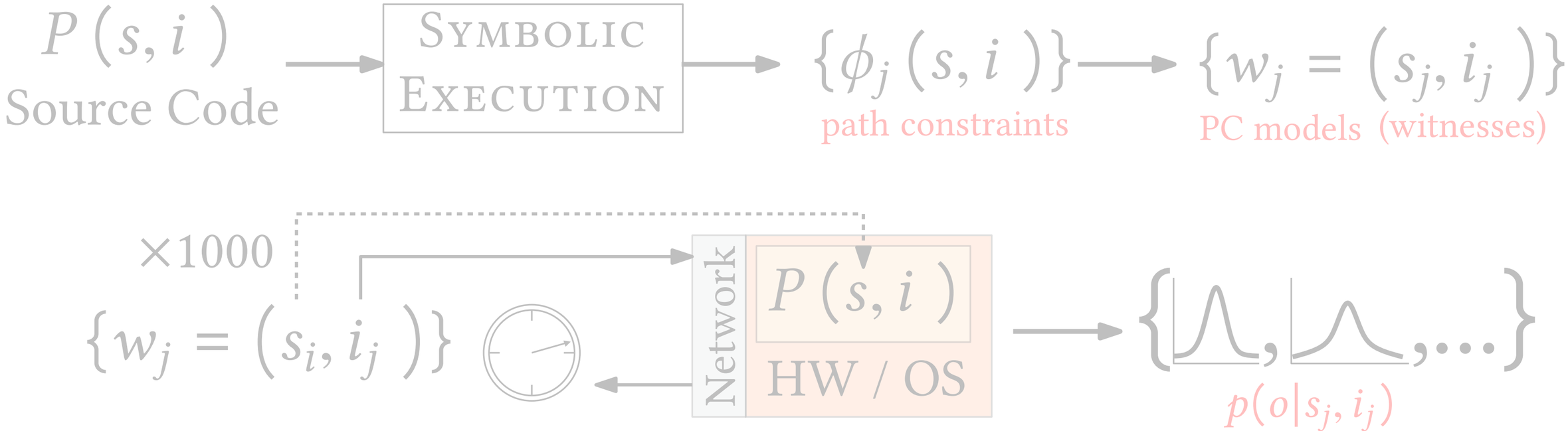
Proposed Approach

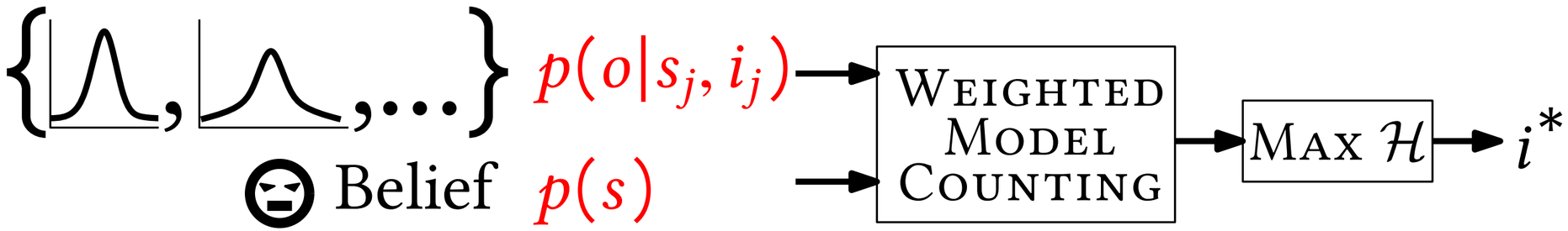




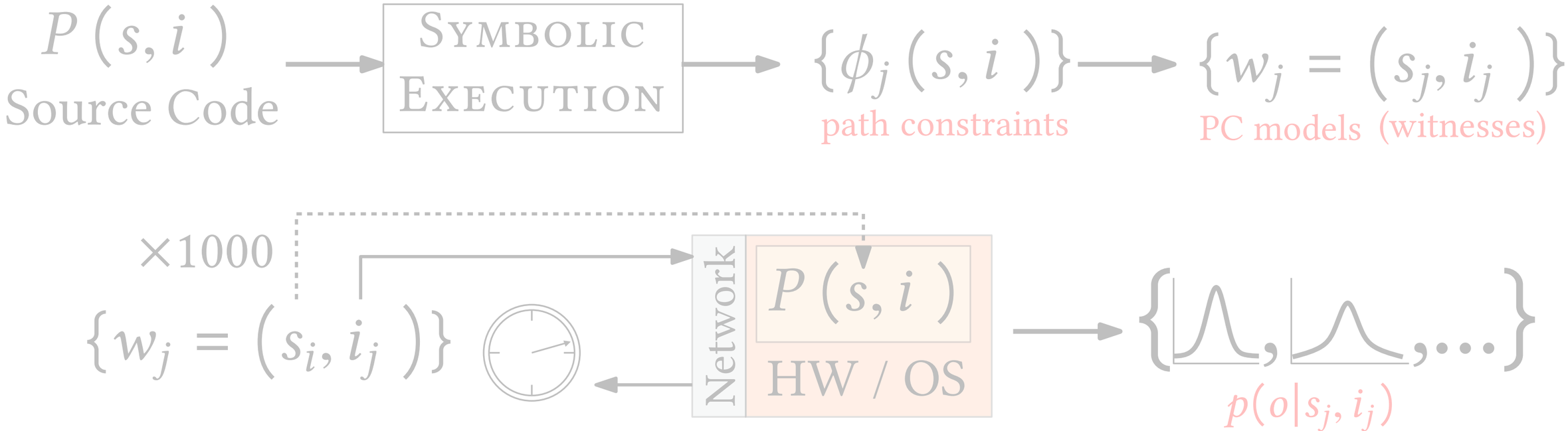
Belief $p(s)$

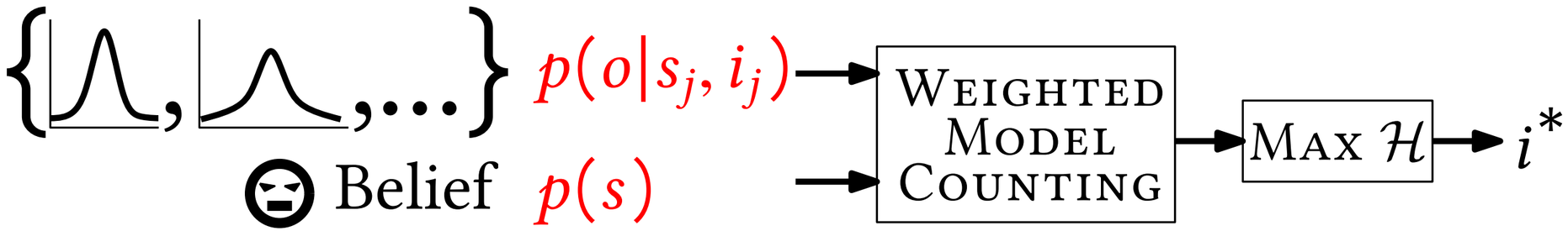
Proposed Approach



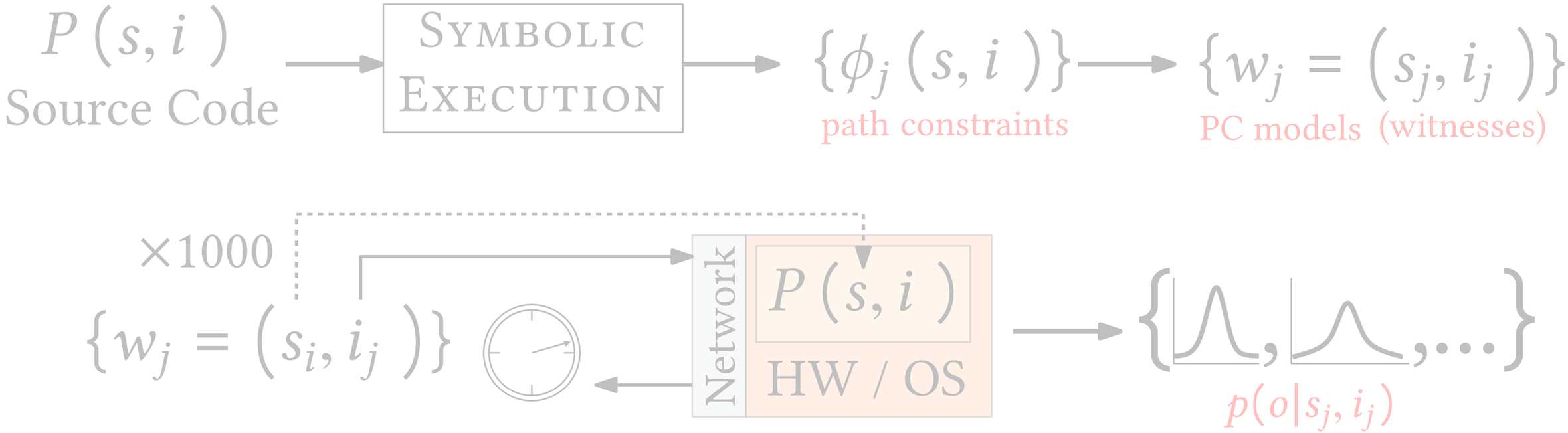


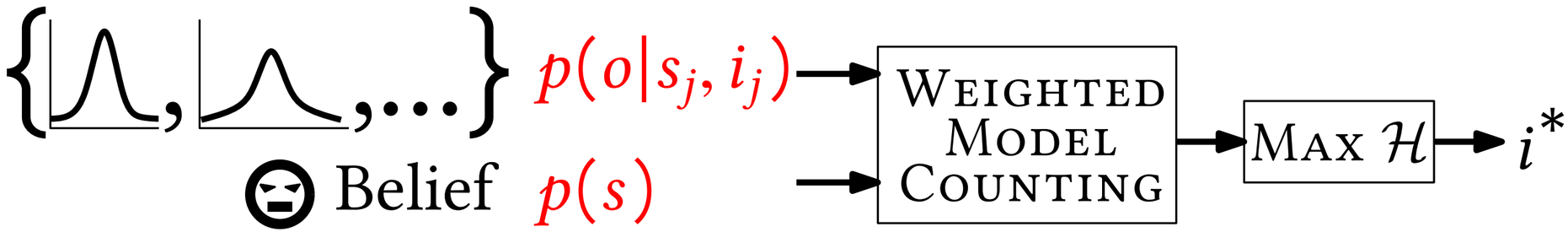
Proposed Approach



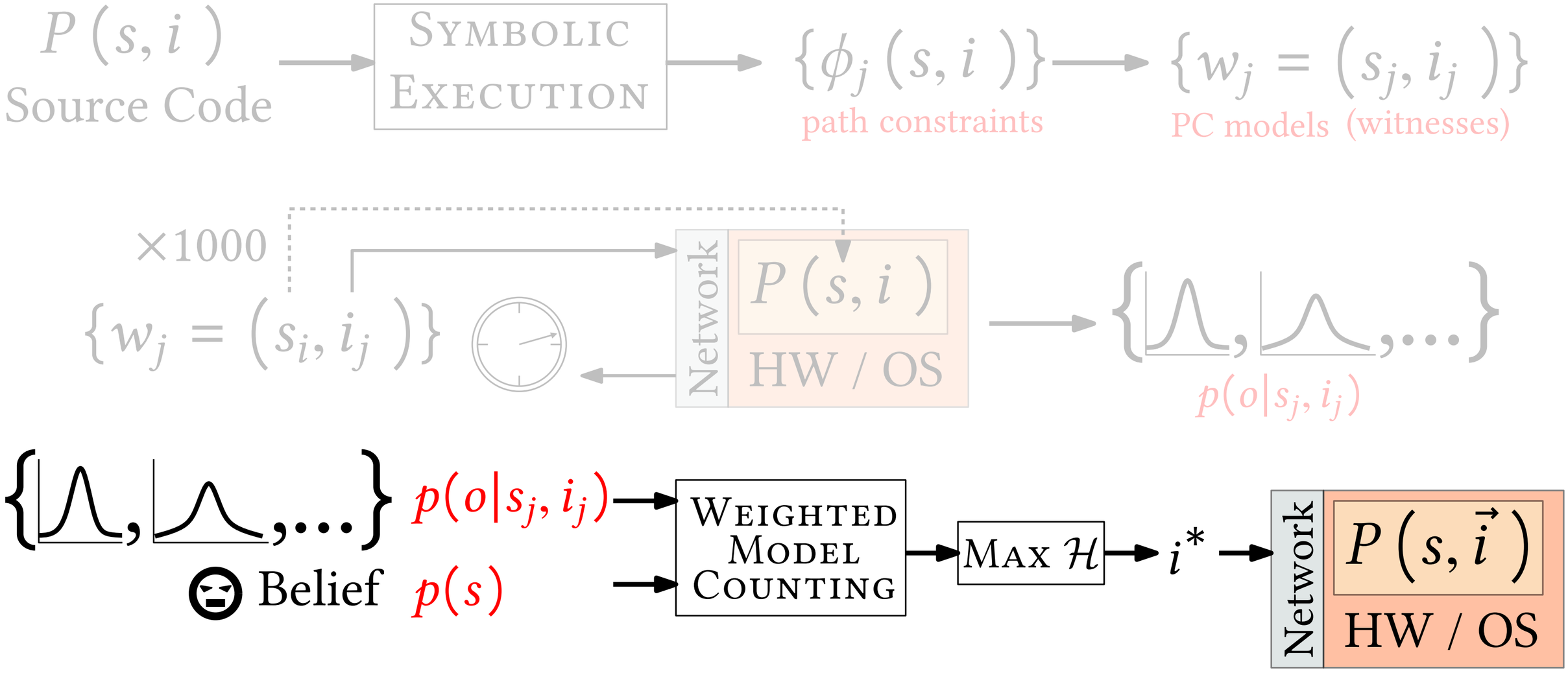


Proposed Approach

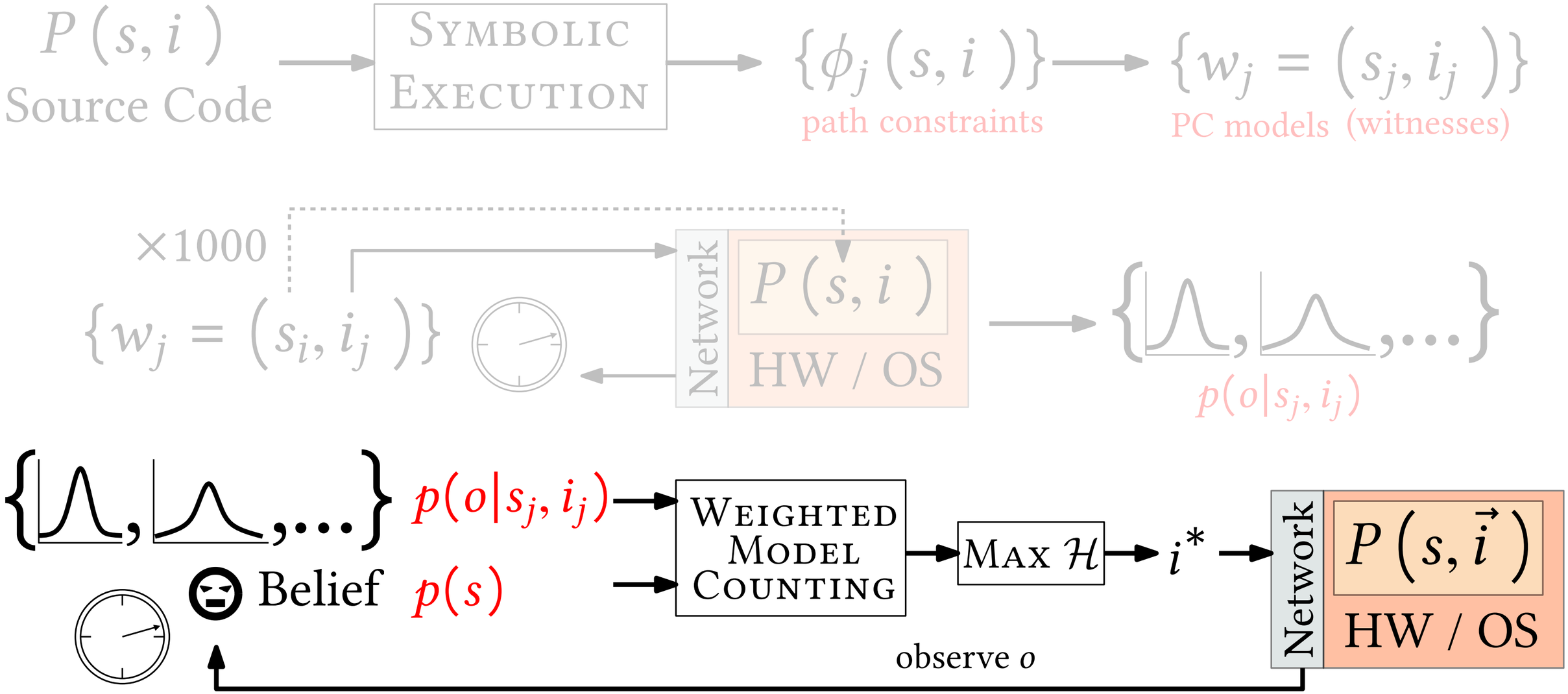




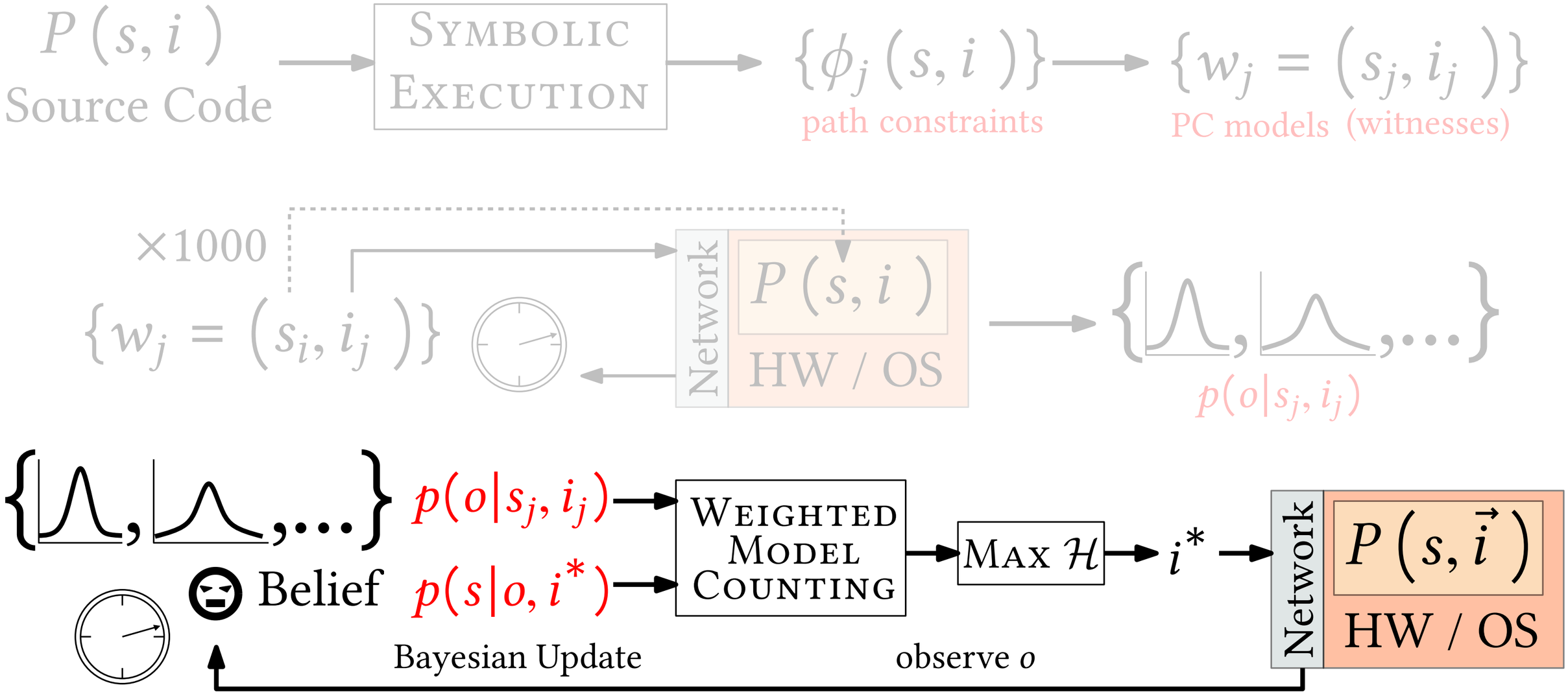
Proposed Approach



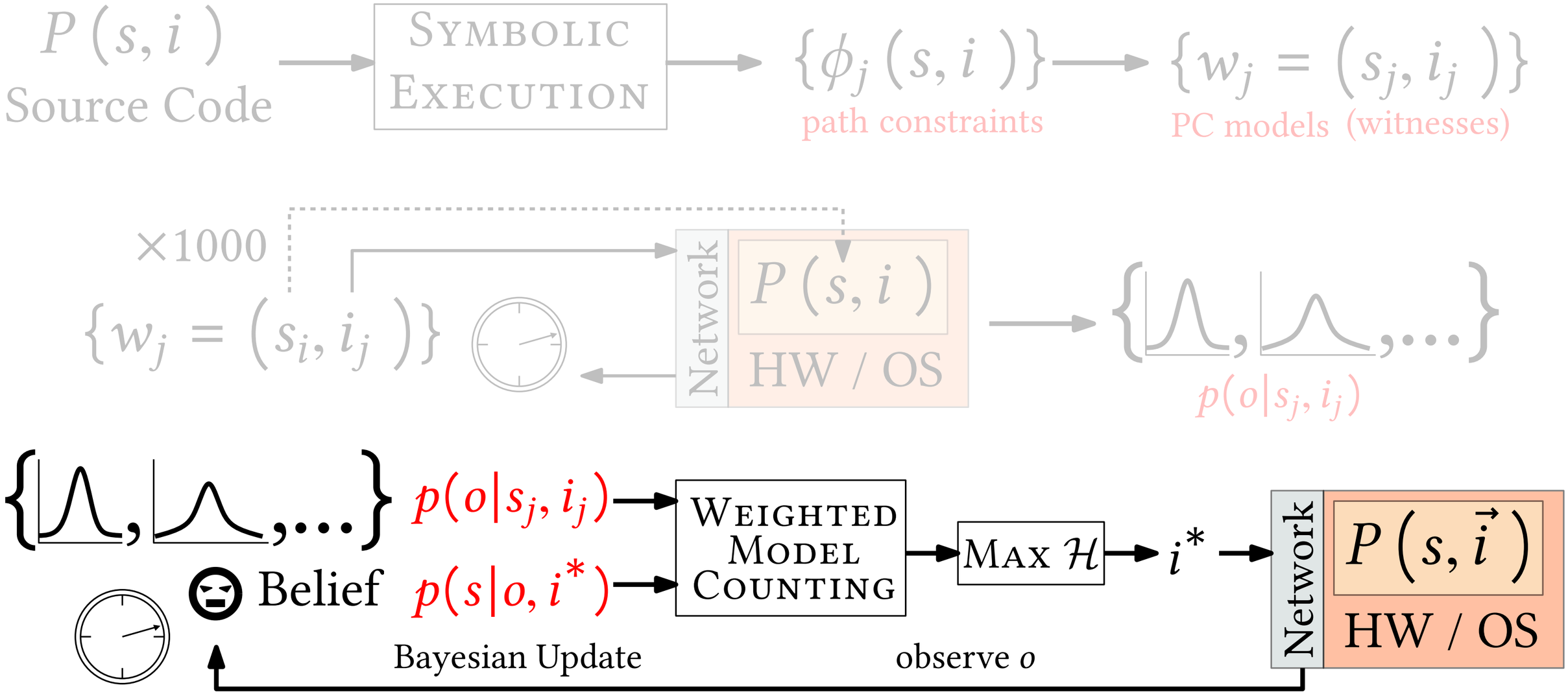
Proposed Approach



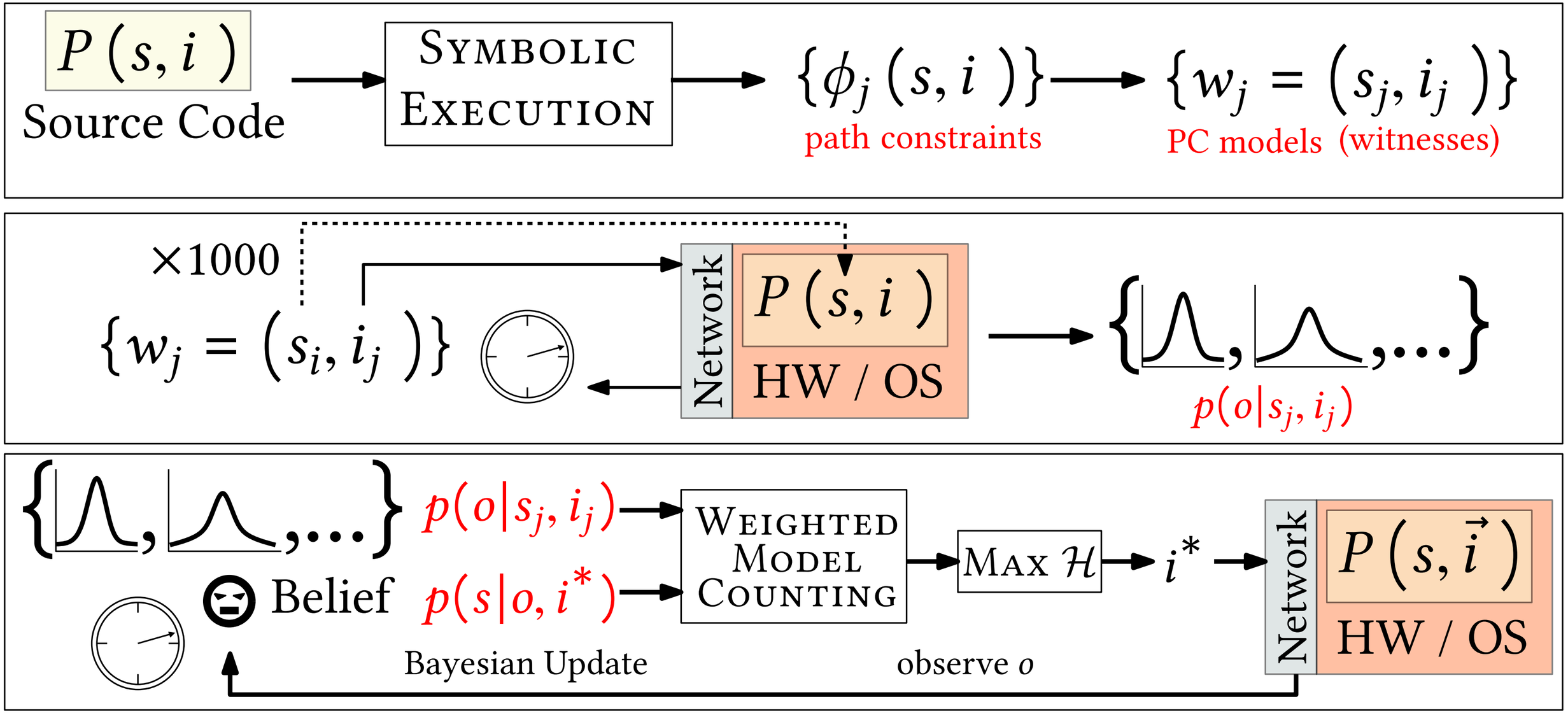
Proposed Approach



Proposed Approach

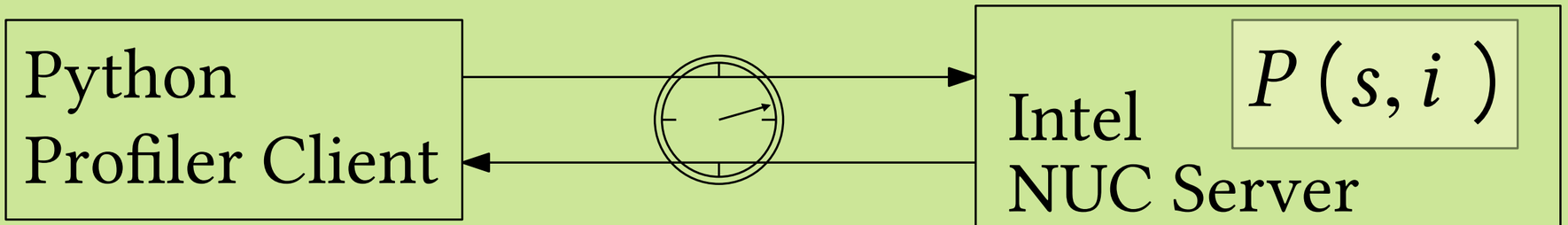


Proposed Approach



Prototype Implementation

NASA Symbolic PathFinder (SPF) + Z3 Constraint Solver



Barvinok
Weighted Symbolic
Model Counting

Mathematica
Symbolic Entropy Computation
Numeric Maximization

Proposed Experiments

DARPA Space-Time Analysis for Cybersecurity (STAC)

Canonical Side-Channel Vulnerability Benchmark

https://github.com/Apogee-Research/STAC/

7 Applications, 1 to 3 variants each

14 total programs

Compare the two approaches.

Proposed Work Summary

1. Offline Static Analysis

2. Offline Dynamic Analysis

3. Online Attack Synthesis

Publications

- Aydin, **Bang**, Bultan: Automata-Based Model Counting for String Constraints. CAV ’15.
- **Bang**, Aydin, Bultan: Automatically Computing Path Complexity of Programs. FSE ’15.
- **Bang**, Aydin, Phan, Pasareanu, Bultan: String Analysis for Side Channels with Segmented Oracles. FSE ’16.
- Phan, **Bang**, Pasareanu, Malacaria, Bultan: Synthesis of Adaptive Side-Channel Attacks. CSF ’17.

Timeline

Fall 2017:

- Unified theoretical model for side-channel techniques from my work.
- Incorporate feedback from committee.
- Improve prototype implementation.

Winter 2018:

- Finish implementation.
- Finish all experiments.

Spring 2018:

- Complete dissertaion draft by April.
- Defend dissertation in May.

Thanks!

Questions?

Thanks!

Questions?