Attack Synthesis for Strings using Meta-heuristics

JPF Workshop 2018

Seemanta Saha[*], Ismet Burak Kadron[*], William Eiers[*], Lucas Bang[+], Tevfik Bultan[*]

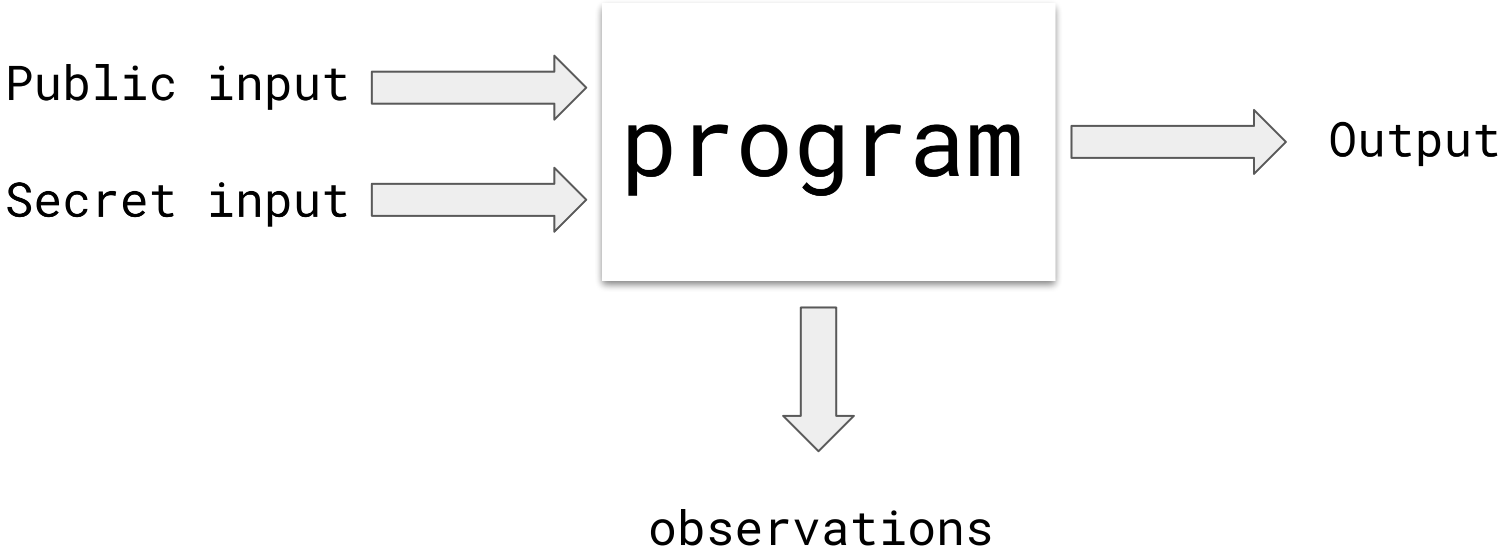
[*] University of California Santa Barbara
[+] Harvey Mudd College

Software Side-Channel Attack

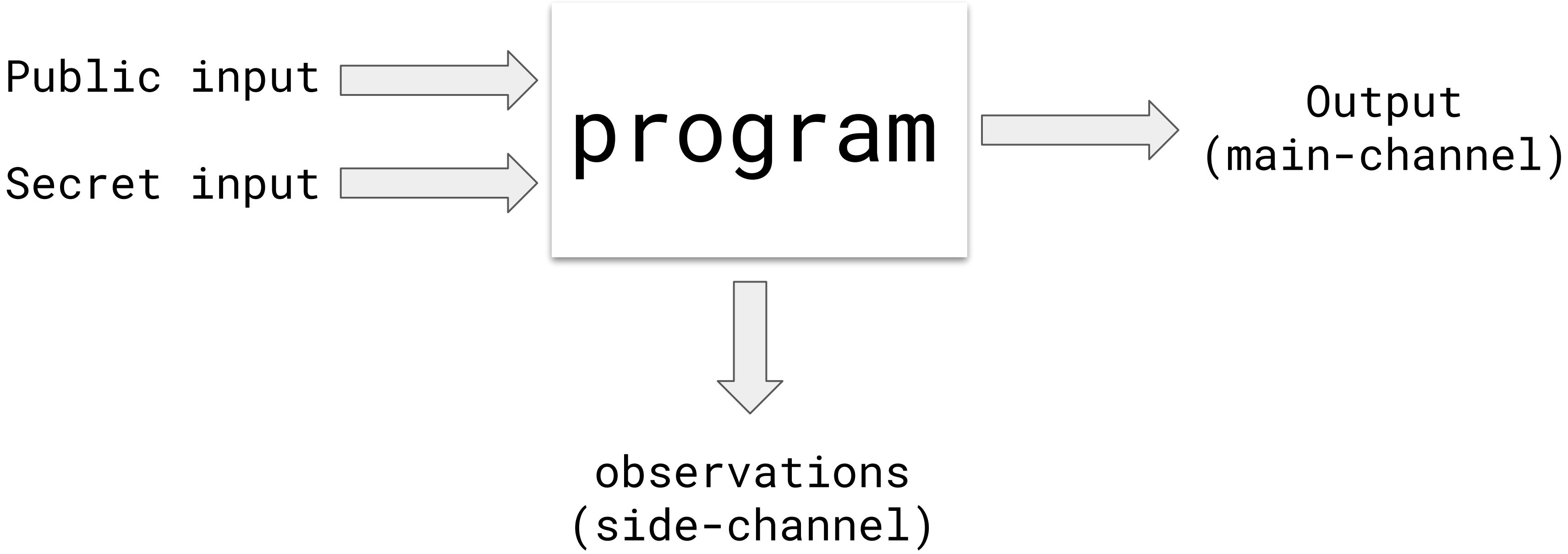
Software Side-Channel Attack

Software Side-Channel Attack

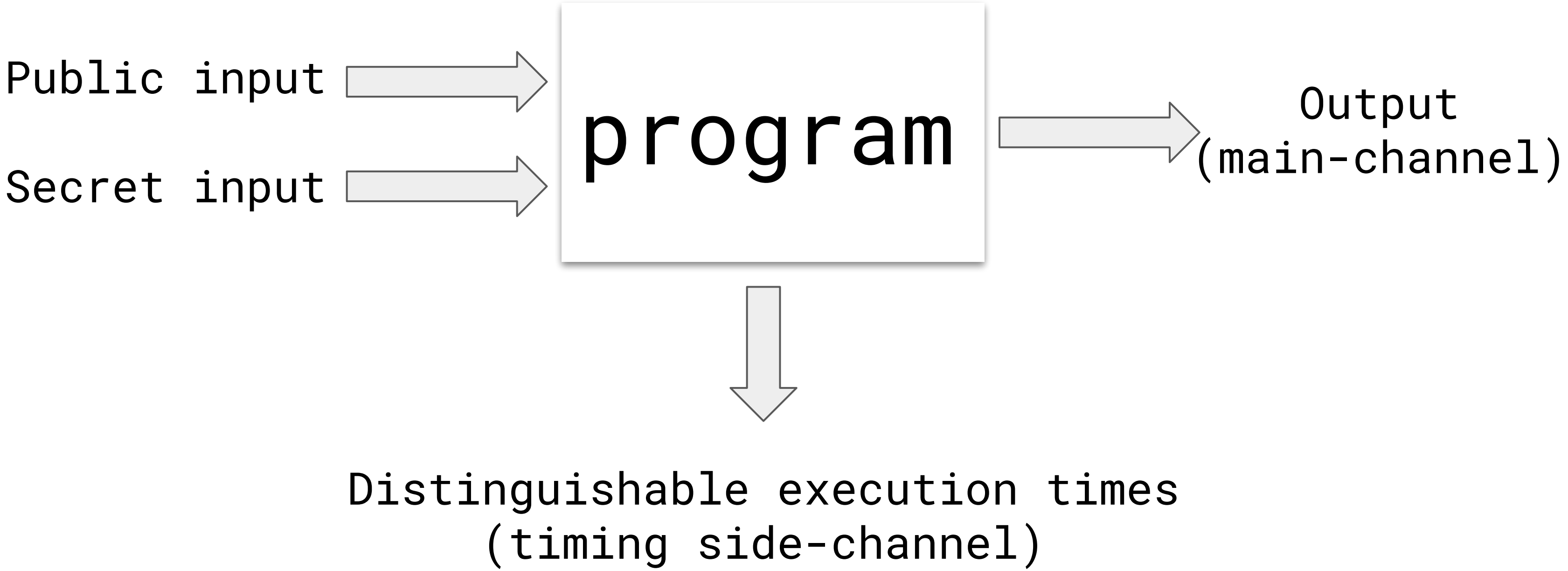
Software Side-Channel Attack



Software Side-Channel Attack



Timing Side-Channel Attack



Timing Side-channel in Password Checking Function

Public input (l)

Secret input (h)

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false/true

distinguishable execution time

Timing Side-channel in Password Checking Function

Public input (l)

Secret input (h)

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false/true

distinguishable execution time

Let’s consider one loop iteration takes 1 millisecond

Timing Side-channel in Password Checking Function

l = "XXXXXXXXXX"

h = "PATHFINDER"

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false

Execution time = 5 milliseconds

Timing Side-channel in Password Checking Function

l = “MATHFINDER”

h = “PATHFINDER”

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false

Execution time = 5 milliseconds

Timing Side-channel in Password Checking Function

l = "PXXXXXXXXX"

h = "PATHFINDER"

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false

Execution time = 6 milliseconds

Timing Side-channel in Password Checking Function

l = “PAXXXXXXXX”

h = “PATHFINDER”

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false

Execution time = 7 milliseconds

Timing Side-channel in Password Checking Function

l = “PATXXXXXXX”

h = “PATHFINDER”

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false

Execution time = 8 milliseconds

Timing Side-channel in Password Checking Function

l = “PATHXXXXXX”

h = “PATHFINDER”

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

false

Execution time = 9 milliseconds

Timing Side-channel in Password Checking Function

l = “PATHFINDER”

h = “PATHFINDER”

```
public boolean passwordChecker(String h,
String l) {

    for (int i = 0; i < h.length(); i++) {

        if (h[i] != l[i])
            return false
    }

    return true
}
```

true

Execution time = 15 milliseconds

Timing Side-channel in Password Checking Function

- known as segment attack vulnerability:
  - attacker reveals the secret input segment (character) by segment (character)

Timing Side-channel in Password Checking Function

- known as segment attack vulnerability:
  - attacker reveals the secret input segment (character) by segment (character)
- this vulnerability was present in
  - **Google KeyCzar library**

Timing attack in Google Keyczar library

Filed under: Crypto, Hacking, Network, Protocols, python, Security — Nate Lawson @ 11:30 pm

I recently found a security flaw in the Google Keyczar crypto library. The impact was that an attacker could forge signatures for data that was "signed" with the SHA-1 HMAC algorithm (the default algorithm).

Firstly, I'm really glad to see more high-level libraries being developed so that programmers don't have to work directly with algorithms. Keyczar is definitely a step in the right direction. Thanks to all the people who developed it. Also, thanks to Stephen Weis for responding quickly to address this issue after I notified him (Python fix and Java fix).

Timing Side-channel in Password Checking Function

- known as segment attack vulnerability
  - attacker reveals the secret input segment (character) by segment (character)
- this vulnerability was present in
  - **Google KeyCzar library**, **OpenID**, etc.

Timing attack in Google Keyczar library

Filed under: Crypto, Hacking, Network, Protocols, python, Security — Nate Lawson @ 11:30 pm

I recently found a security flaw in the Google Keyczar crypto library. The impact was that an attacker could forge signatures for data that was "signed" with the SHA-1 HMAC algorithm (the default algorithm).

Firstly, I'm really glad to see more high-level libraries being developed so that programmers don't have to work directly with algorithms. Keyczar is definitely a step in the right direction. Thanks to all the people who developed it. Also, thanks to Stephen Weis for responding quickly to address this issue after I notified him (Python fix and Java fix).

[security] Widespread Timing Vulnerabilities in OpenID implementations

Taylor Nelson taylor at rootlabs.com
Tue Jul 13 20:32:50 UTC 2010

- Next message: [security] Widespread Timing Vulnerabilities in OpenID implementations
- **Messages sorted by:** [date] [thread] [subject] [author]

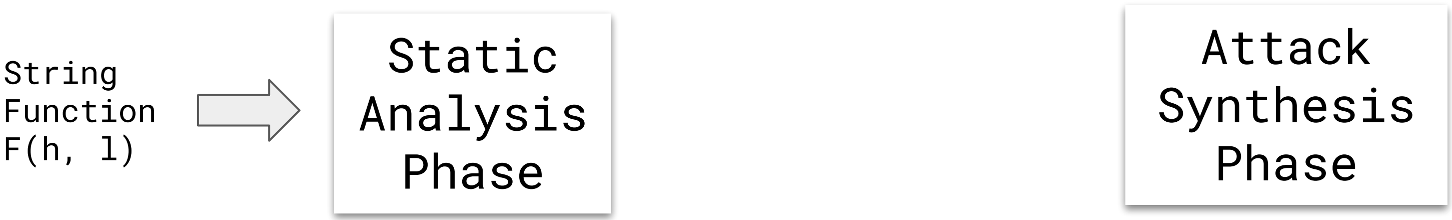
```
Every OpenID implementation I have checked this far has contained timing
dependent compares in the HMAC verification, allowing a remote attacker
to forge valid tokens.

In JOpenId:
There is a timing vulnerability in thegetAuthentication  function in
trunk/JOpenId/src/org/expressme/openid/OpenIdManager.java
```

Attack Synthesis Overview

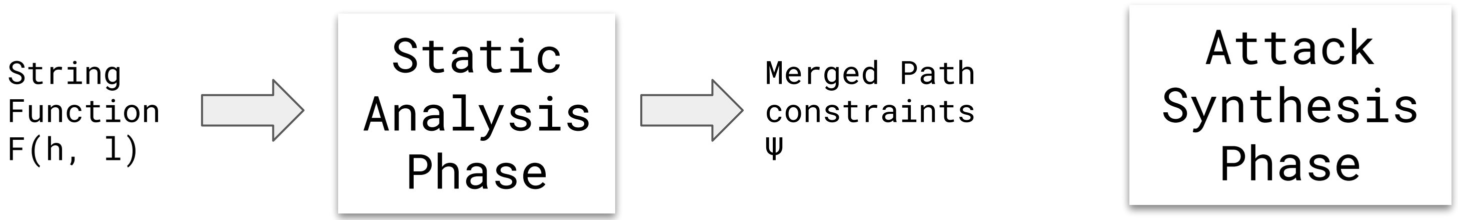
Static Analysis Phase

Attack Synthesis Phase

Attack Synthesis Overview

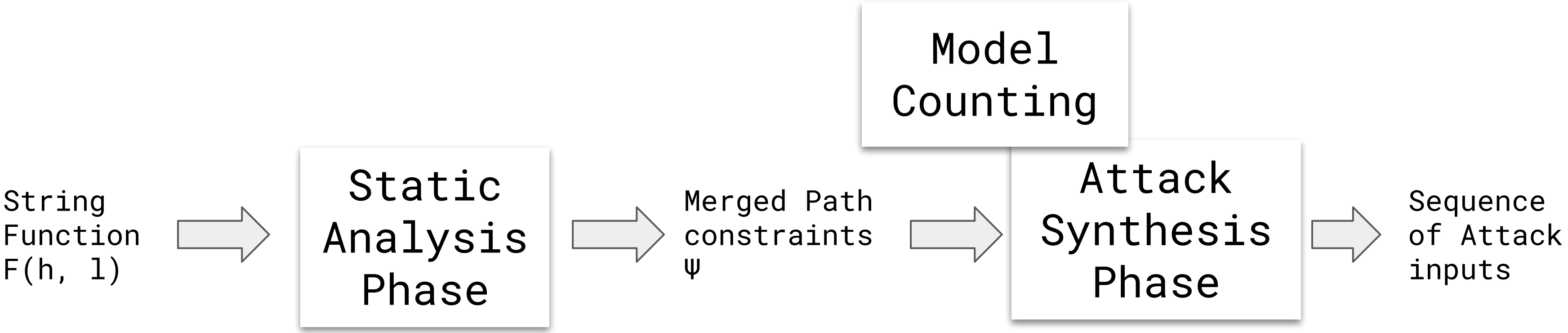


Attack Synthesis Overview

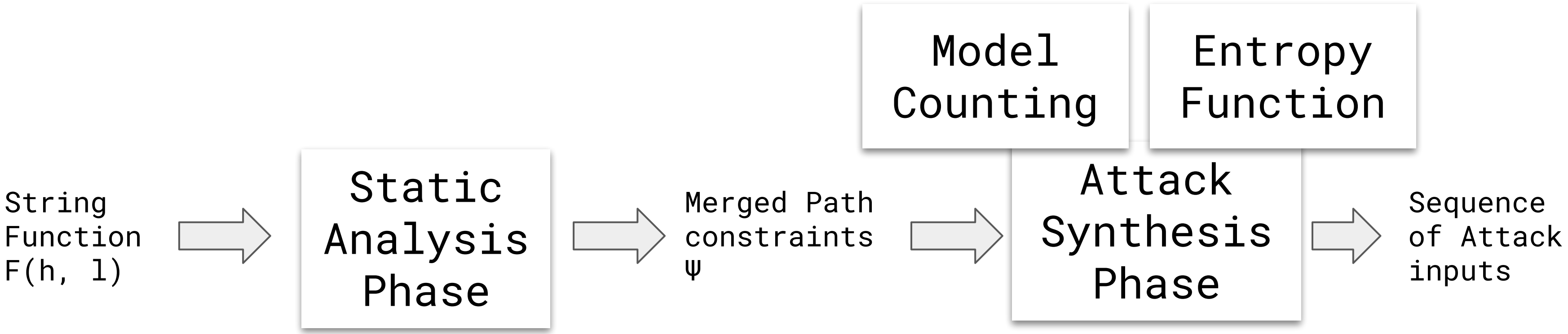


Attack Synthesis Overview

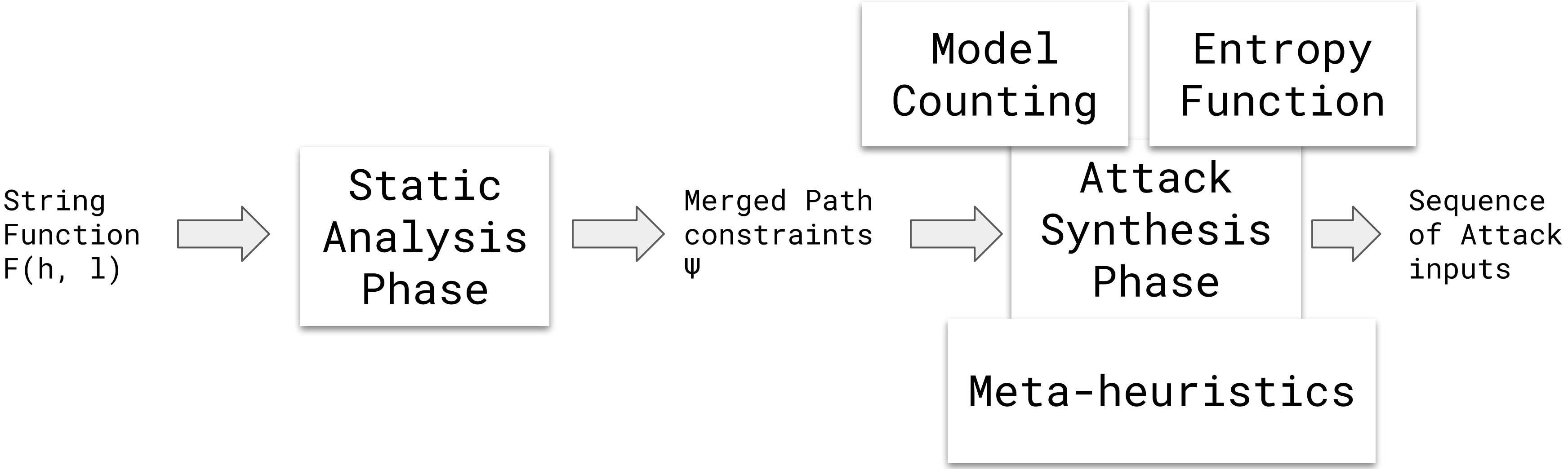
Attack Synthesis Overview



Attack Synthesis Overview



Attack Synthesis Overview



Static Analysis Phase

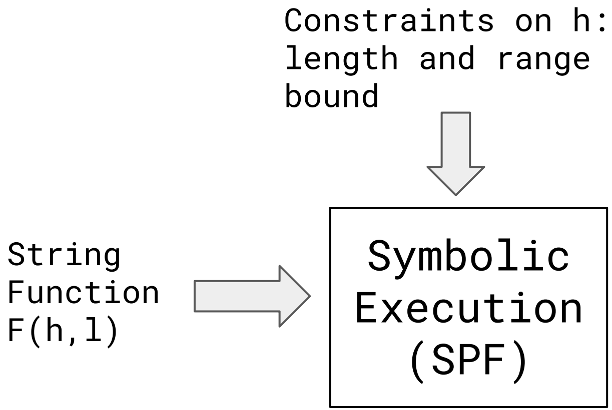
String
Function
F(h,l)

Static Analysis Phase

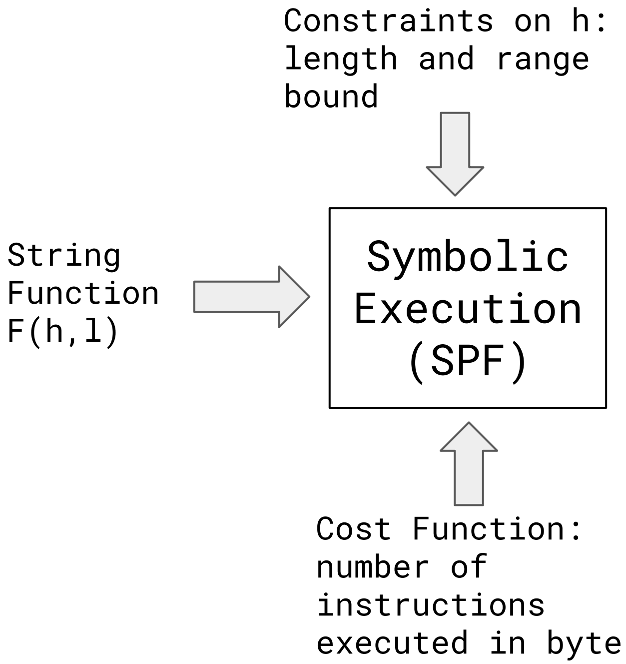
Constraints on h:
length and range
bound

String
Function
F(h,l)

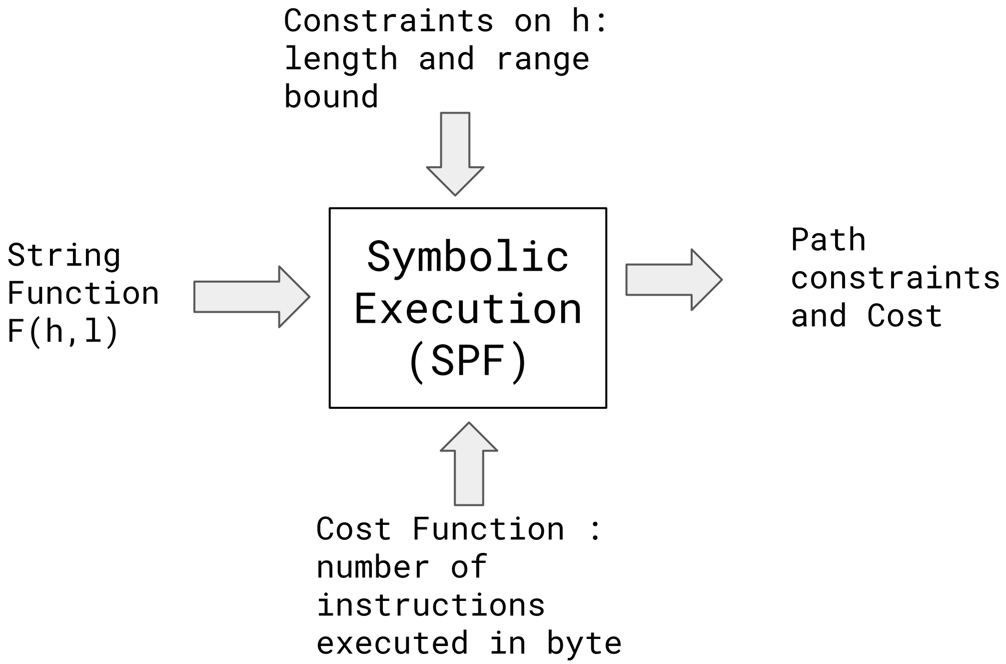
Static Analysis Phase



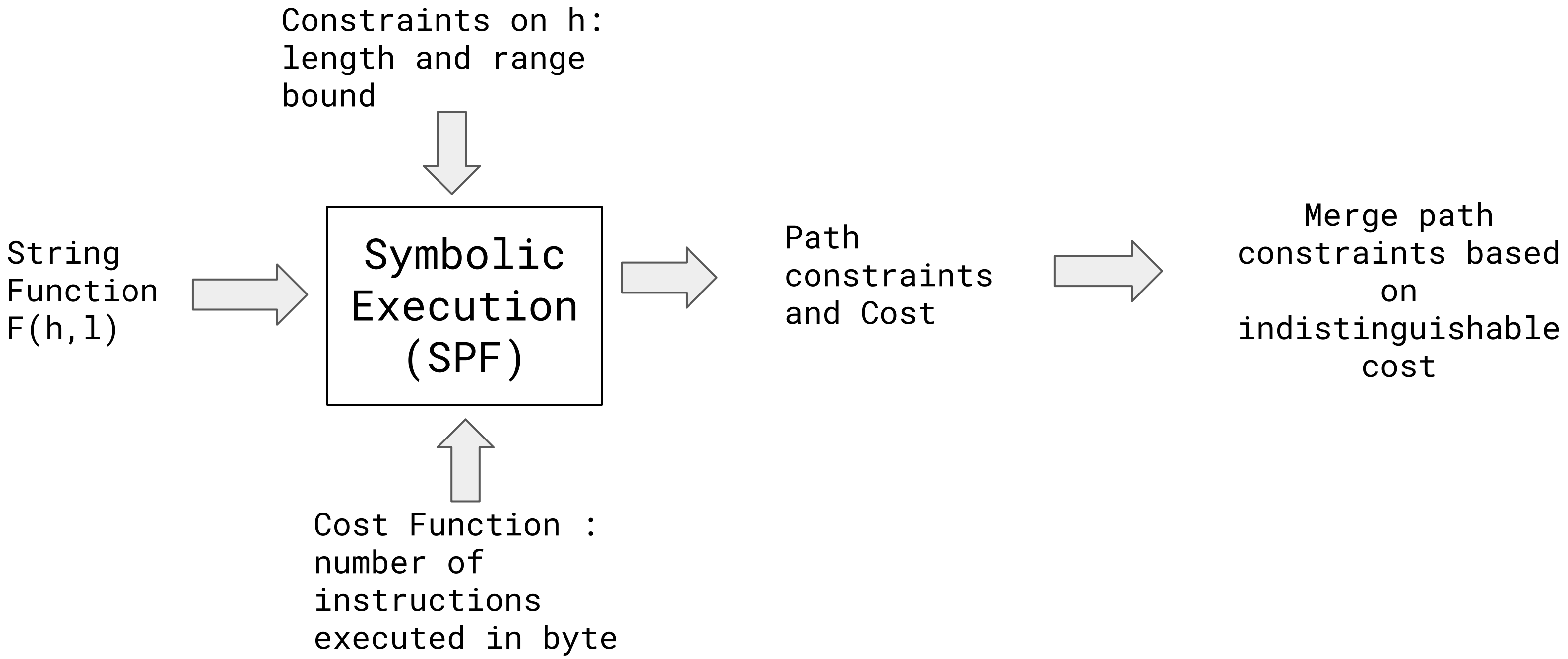
Static Analysis Phase



Static Analysis Phase



Static Analysis Phase



Path Constraints for Password Checking Function

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

Length of public input (l) = 4
Length of secret input (h) = 4

Goal: Attack Synthesis

Generate Sequence of inputs revealing information about the secret value

Attack Synthesis

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

Attack Synthesis

Unknown Secret:
“PATH”

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge$ $charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

Attack Synthesis

Unknown Secret: "PATH"

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ

Solve Constraint

Get a Random Model

attack input: "ABCD"

Attack Synthesis

Unknown Secret: "PATH"

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ

Solve Constraint

Get a Random Model

attack input: "ABCD"

Infer constraint based on observation

Attack Synthesis

Unknown Secret: “PATH”

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ

Solve Constraint

Get a Random Model

attack input: “ABCD”

Infer constraint based on observation

Inferred constraint:
h[0] != l[0]

Attack Synthesis

Unknown Secret: "PATH"

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ

Solve Constraint

Get a Random Model

attack input: "ABCD"

Infer constraint based on observation

Inferred constraint:
h[0] != l[0]

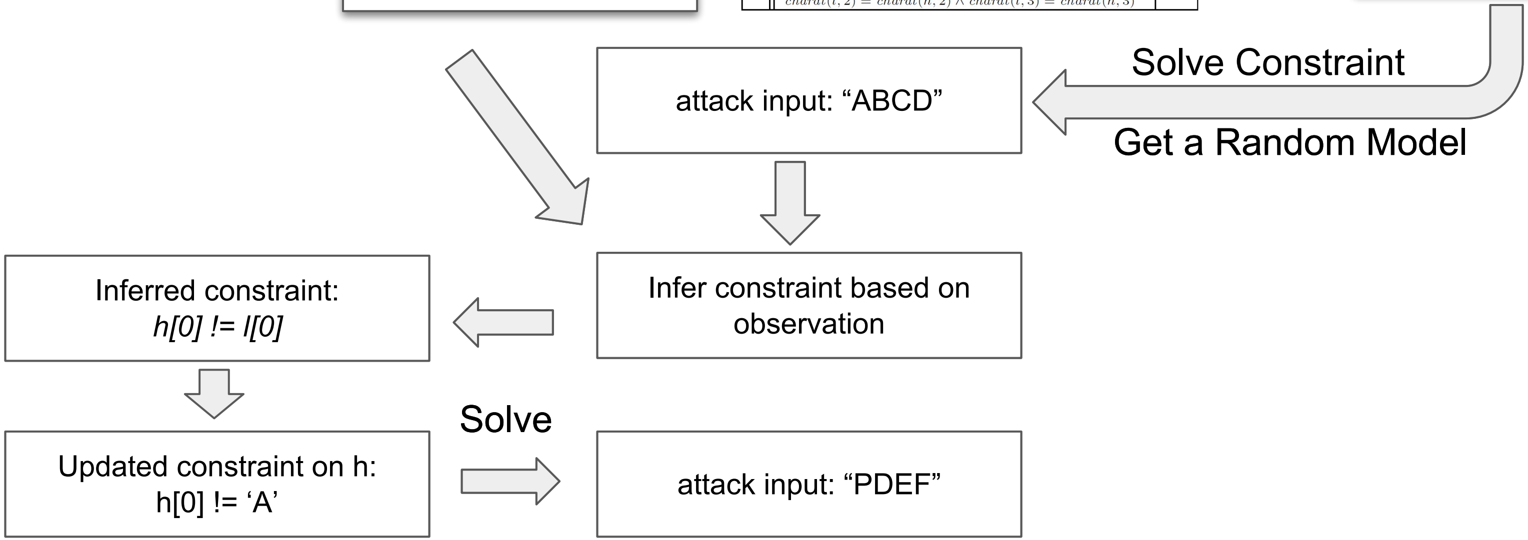
Updated constraint on h:
h[0] != 'A'

Attack Synthesis

Unknown Secret:
“PATH”

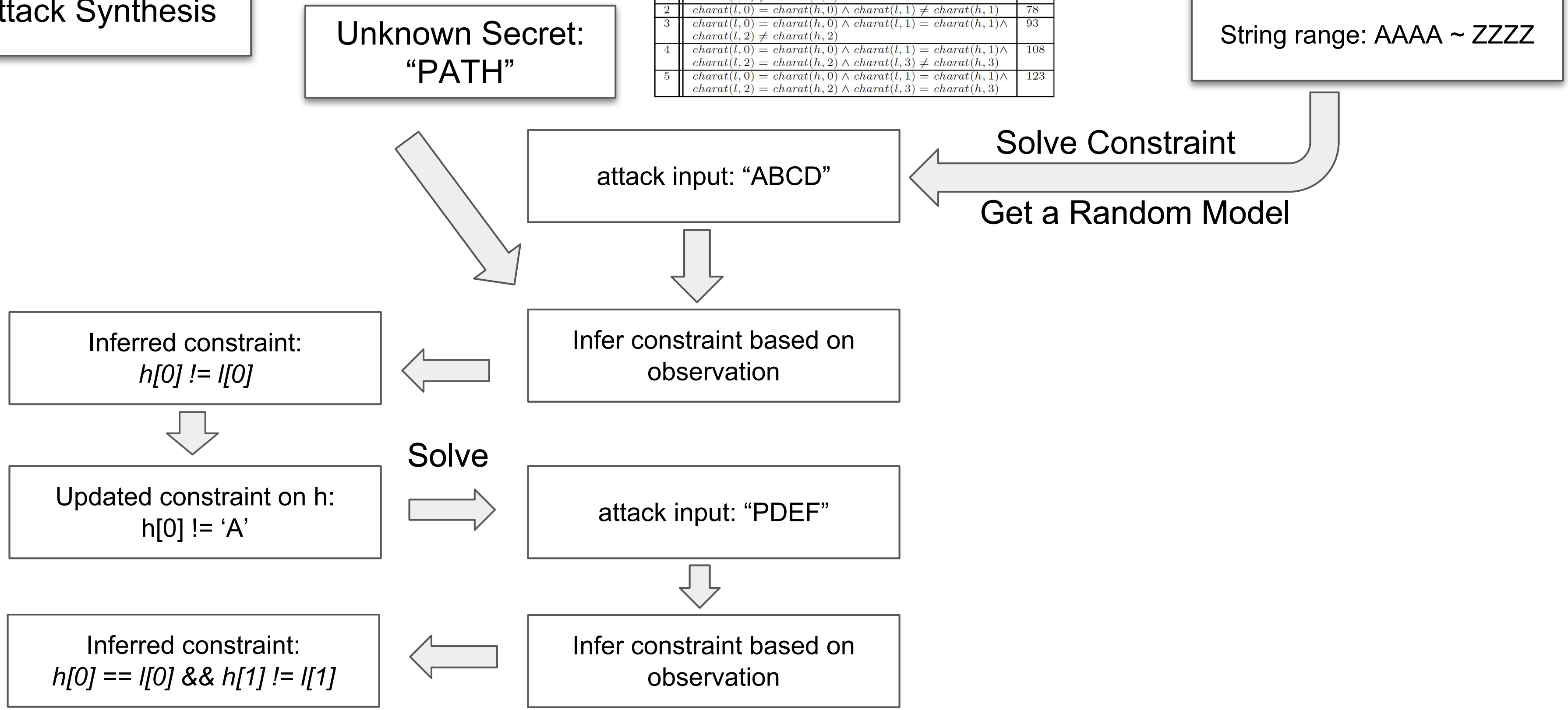
i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ



Attack Synthesis

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

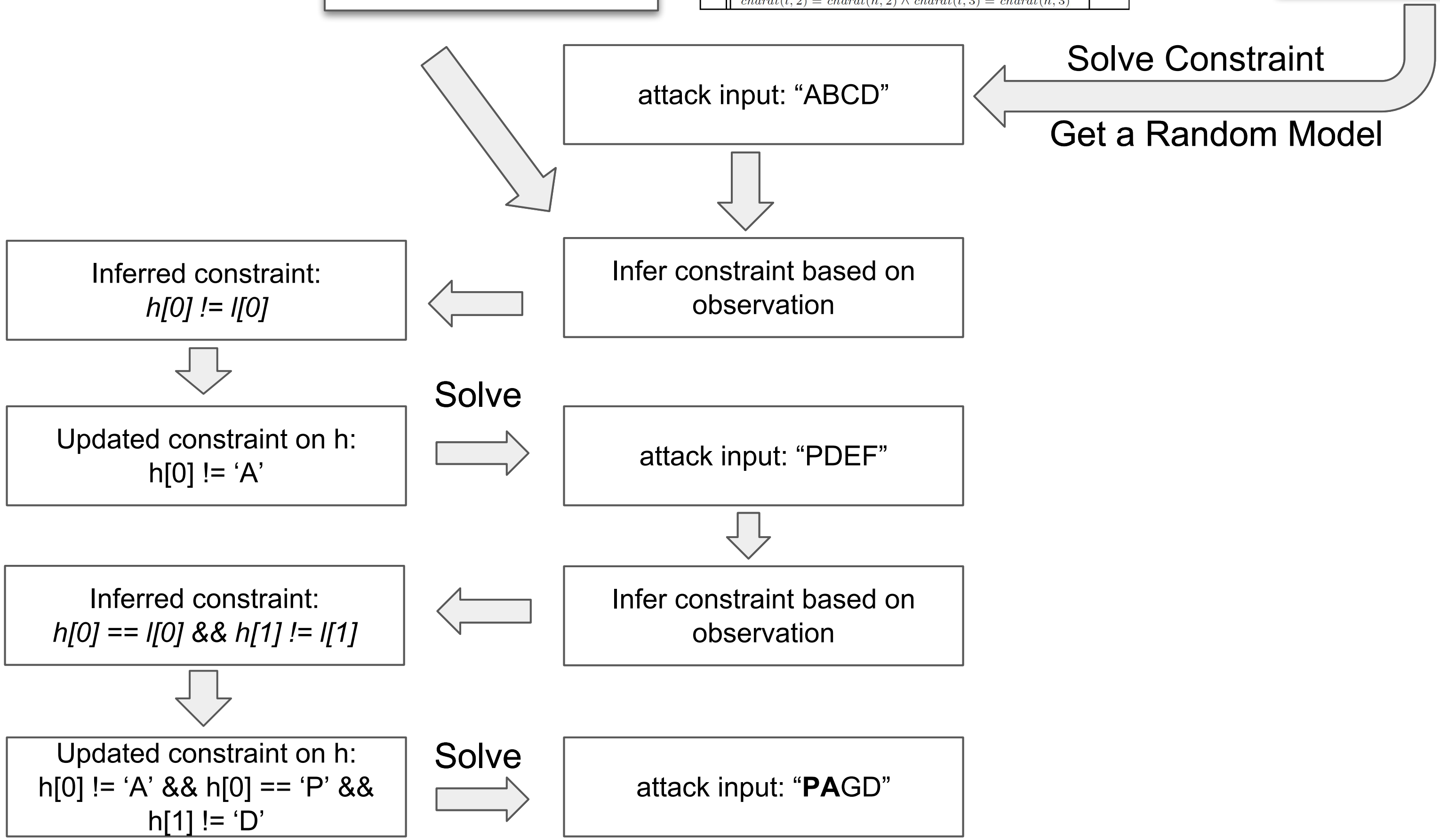


Attack Synthesis

Unknown Secret: "PATH"

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ



Attack Synthesis

Unknown Secret: "PATH"

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

String range: AAAA ~ ZZZZ

Solve Constraint

Get a Random Model

attack input: "ABCD"

Infer constraint based on observation

Inferred constraint:
h[0] != l[0]

Updated constraint on h:
h[0] != 'A'

Solve

attack input: "PDEF"

Infer constraint based on observation

Inferred constraint:
h[0] == l[0] && h[1] != l[1]

Updated constraint on h:
h[0] != 'A' && h[0] == 'P' &&
h[1] != 'D'

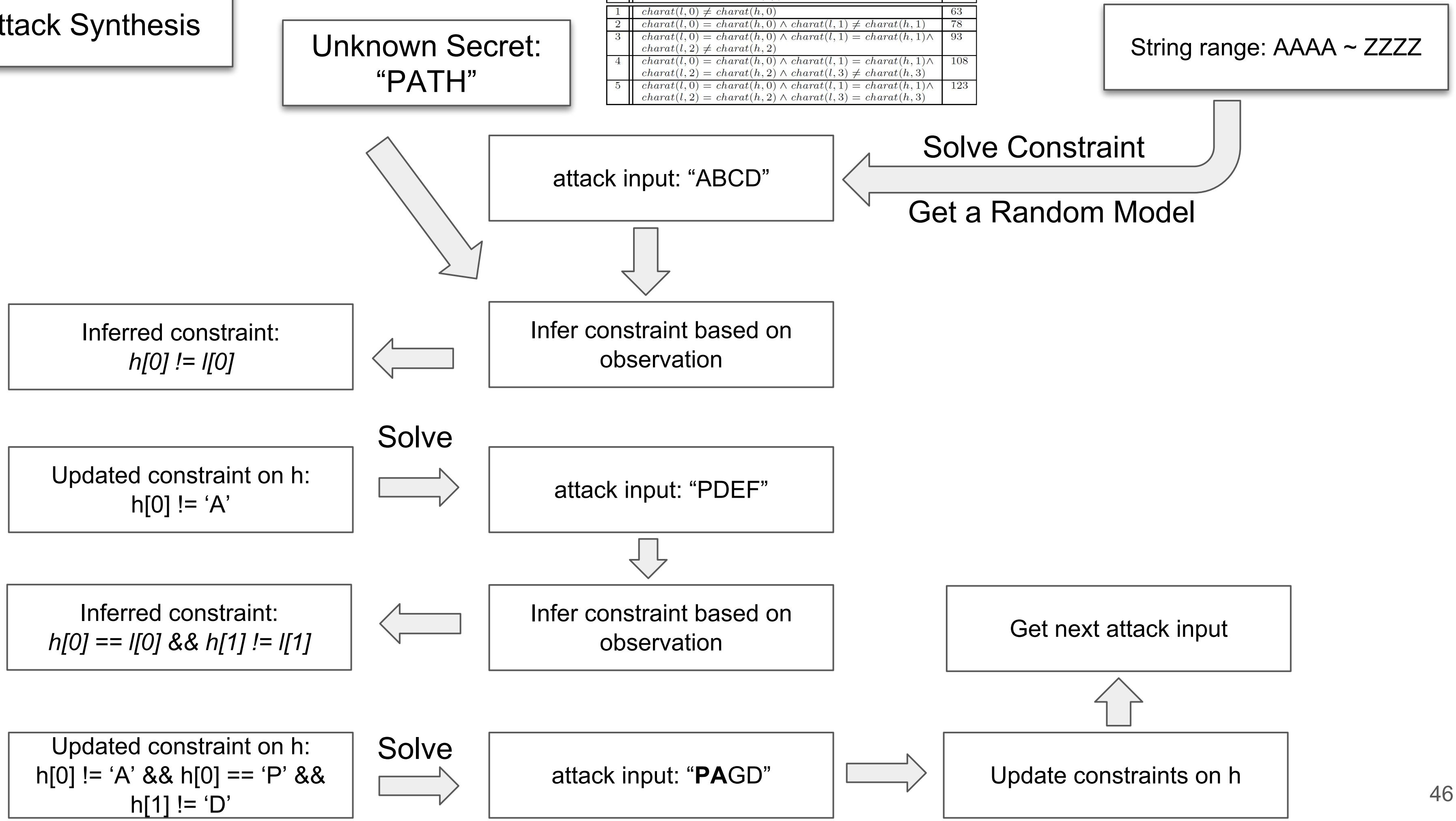
Solve

attack input: "**PA**GD"

Update constraints on h

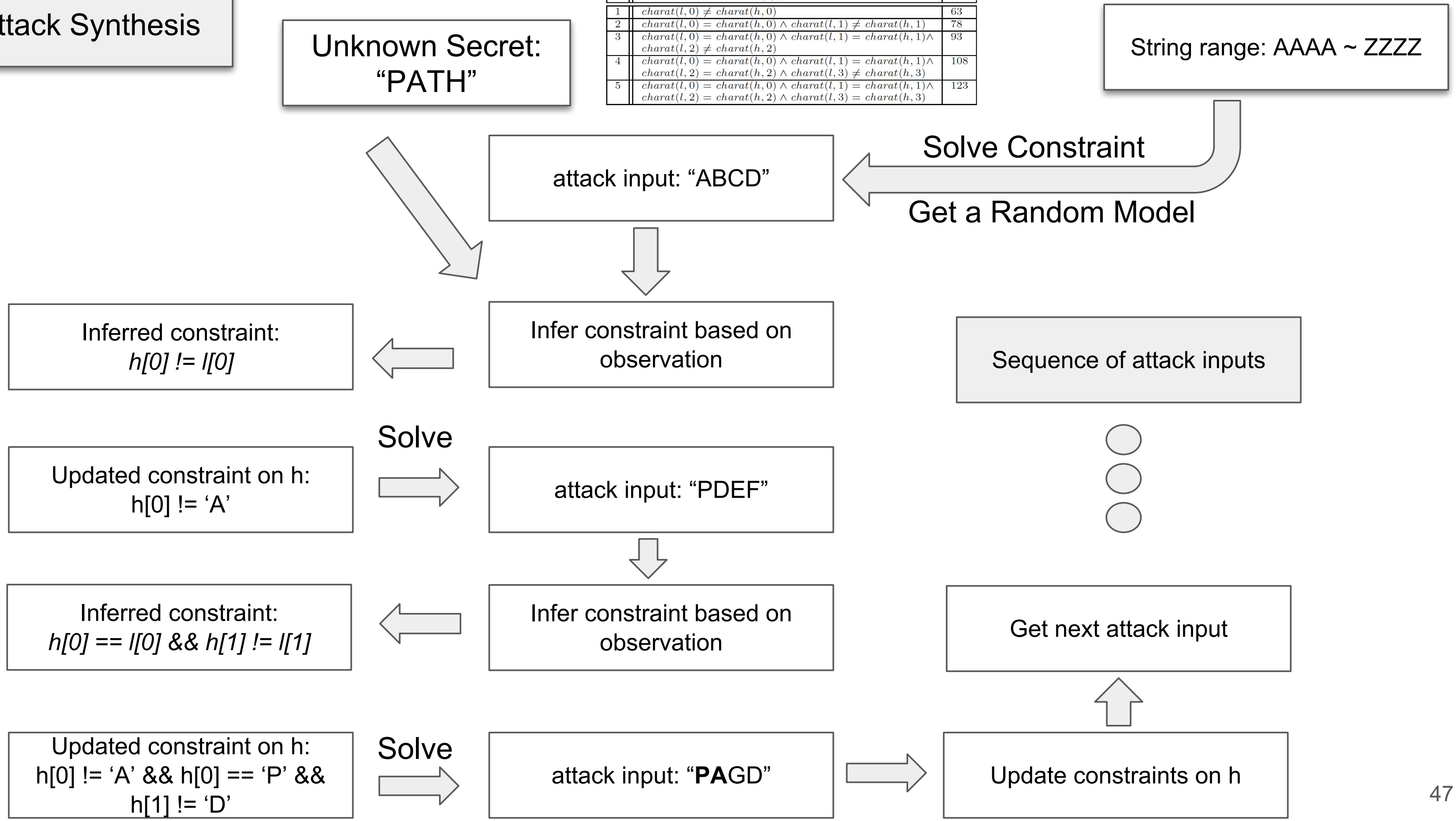
Attack Synthesis

i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123

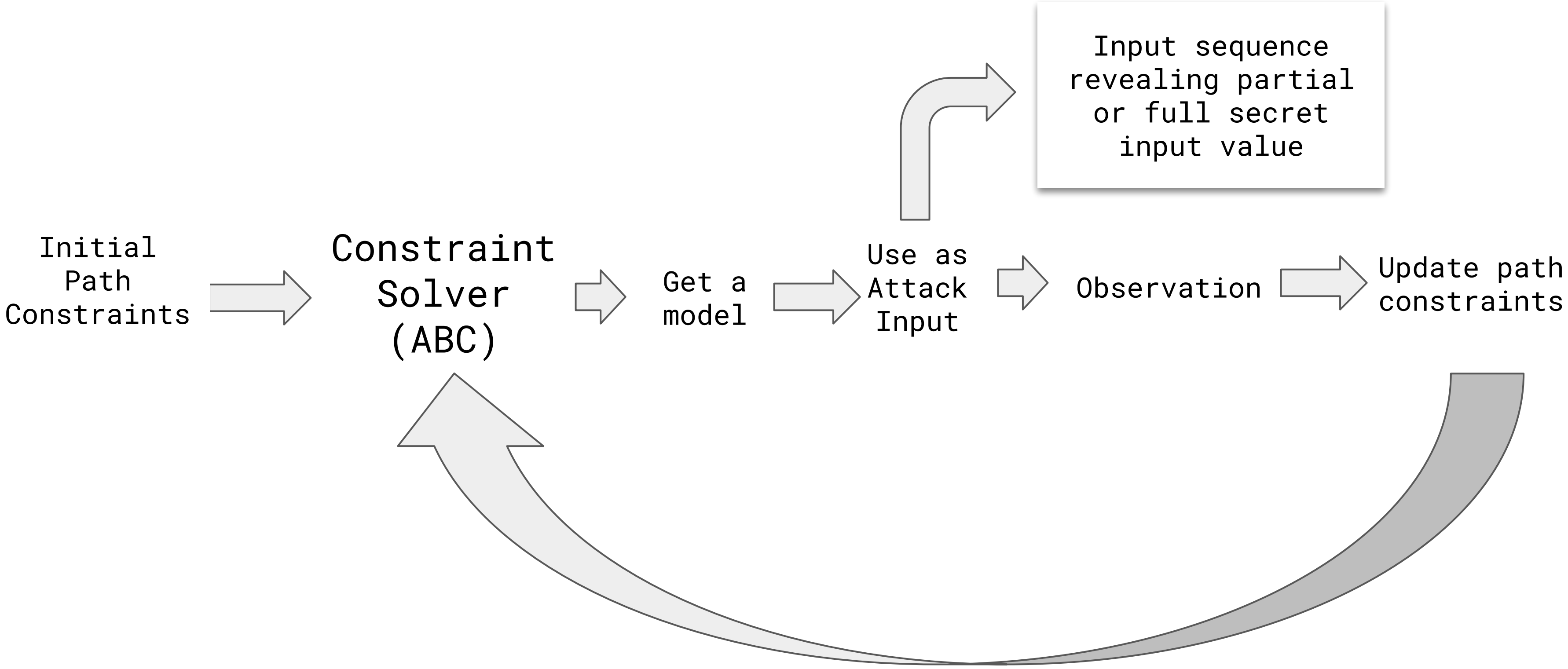


Attack Synthesis

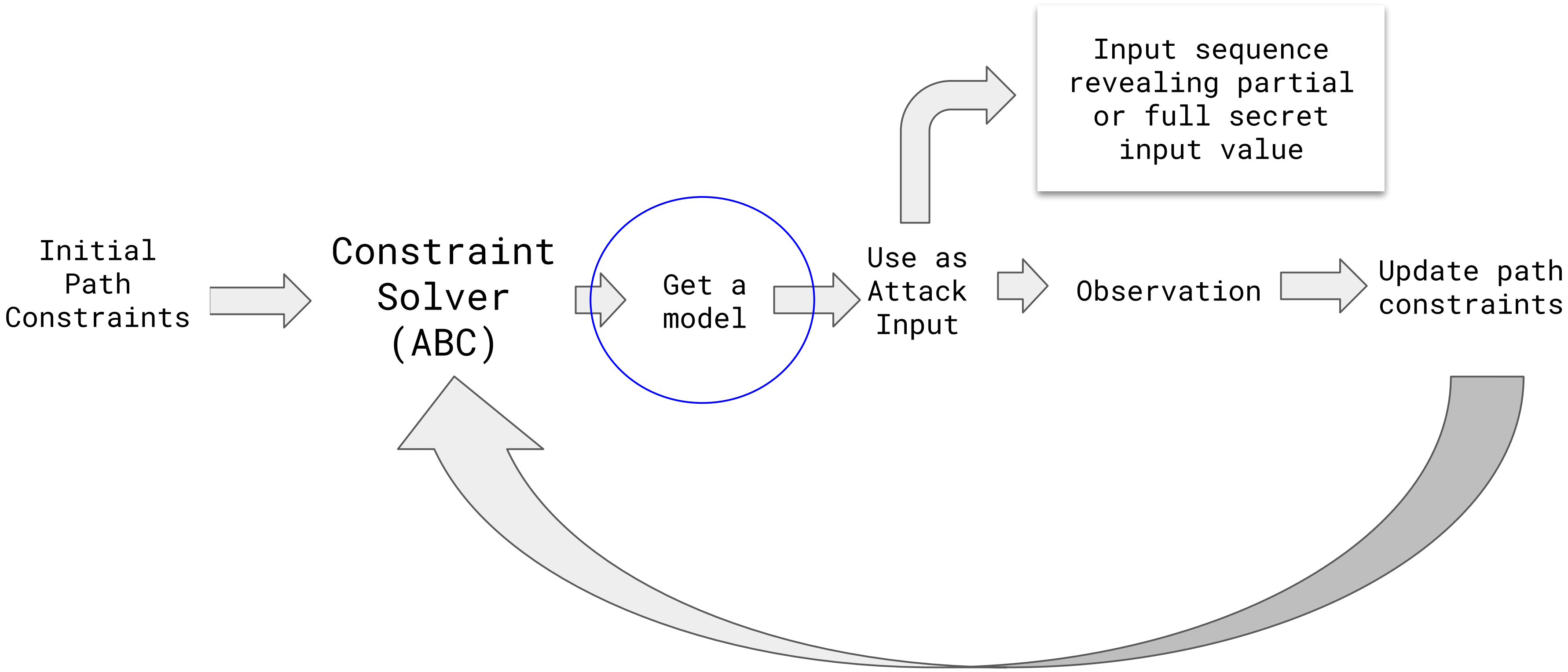
i	Observation Constraint, ψ_i	o
1	$charat(l,0) \neq charat(h,0)$	63
2	$charat(l,0) = charat(h,0) \wedge charat(l,1) \neq charat(h,1)$	78
3	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) \neq charat(h,2)$	93
4	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) \neq charat(h,3)$	108
5	$charat(l,0) = charat(h,0) \wedge charat(l,1) = charat(h,1) \wedge charat(l,2) = charat(h,2) \wedge charat(l,3) = charat(h,3)$	123



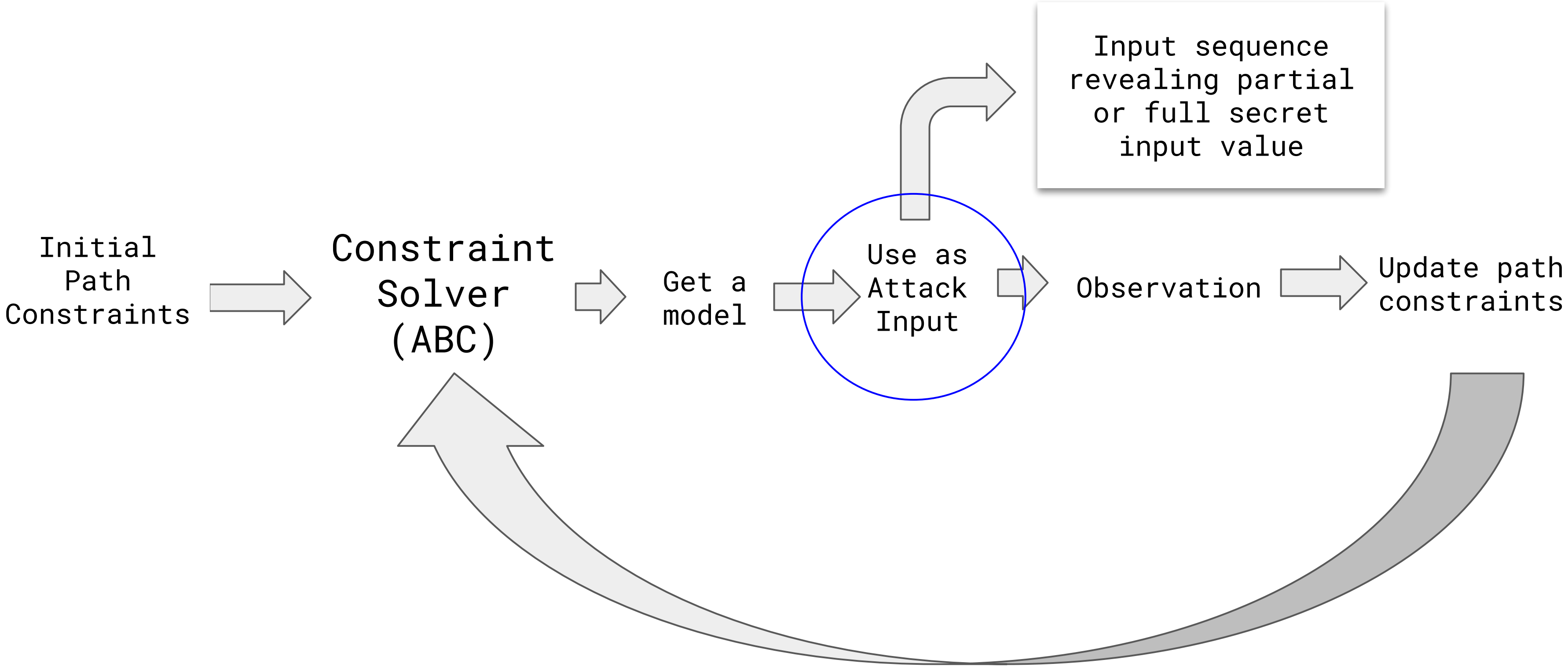
Attack Synthesis Phase



Attack Synthesis Phase



Attack Synthesis Phase



- We can automatically generate an attack using
  - Program path constraints
  - Observation from program execution
  - Generating constraints from observation
  - Updating constraints on secret value
  - Solving constraints to get attack input

We call this Model-Based Attack Synthesis (M)

We can synthesize attacks using Model-Based (M) Attack Synthesis

Why do we need meta-heuristics?

String inequality Function

```
public String inequality(string i) {

    if(s <= i)
        do something simple; // 2 seconds
    else
        do something complex; // 5 seconds

    return 0;
}
```

String inequality Function

```
public String inequality(string i) {

    if(s <= i)
        do something simple; // 2 seconds
    else
        do something complex; // 5 seconds

    return 0;
}
```

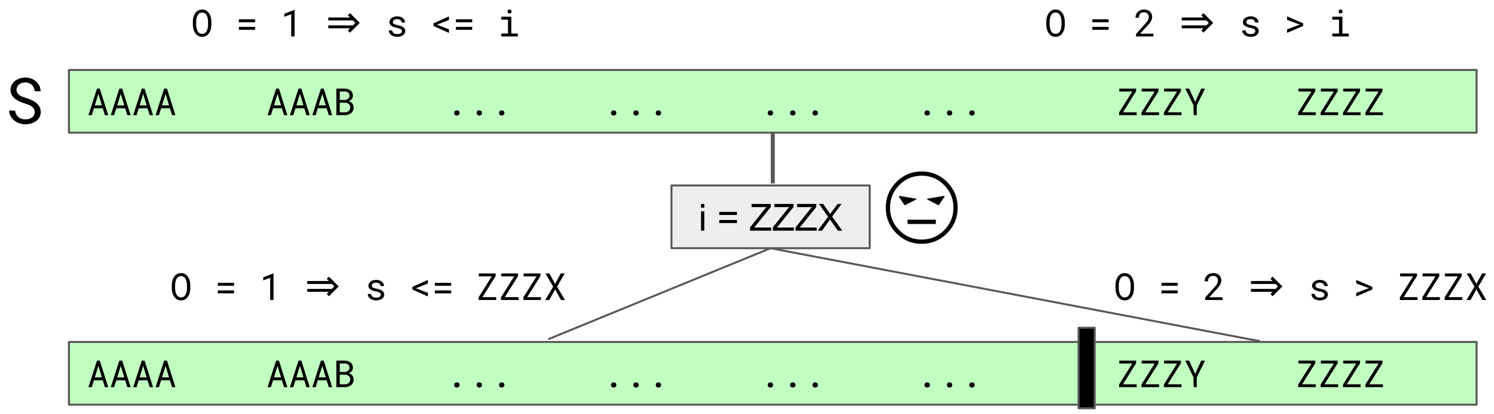
0 = 1 ⇒ s <= i

0 = 2 ⇒ s > i

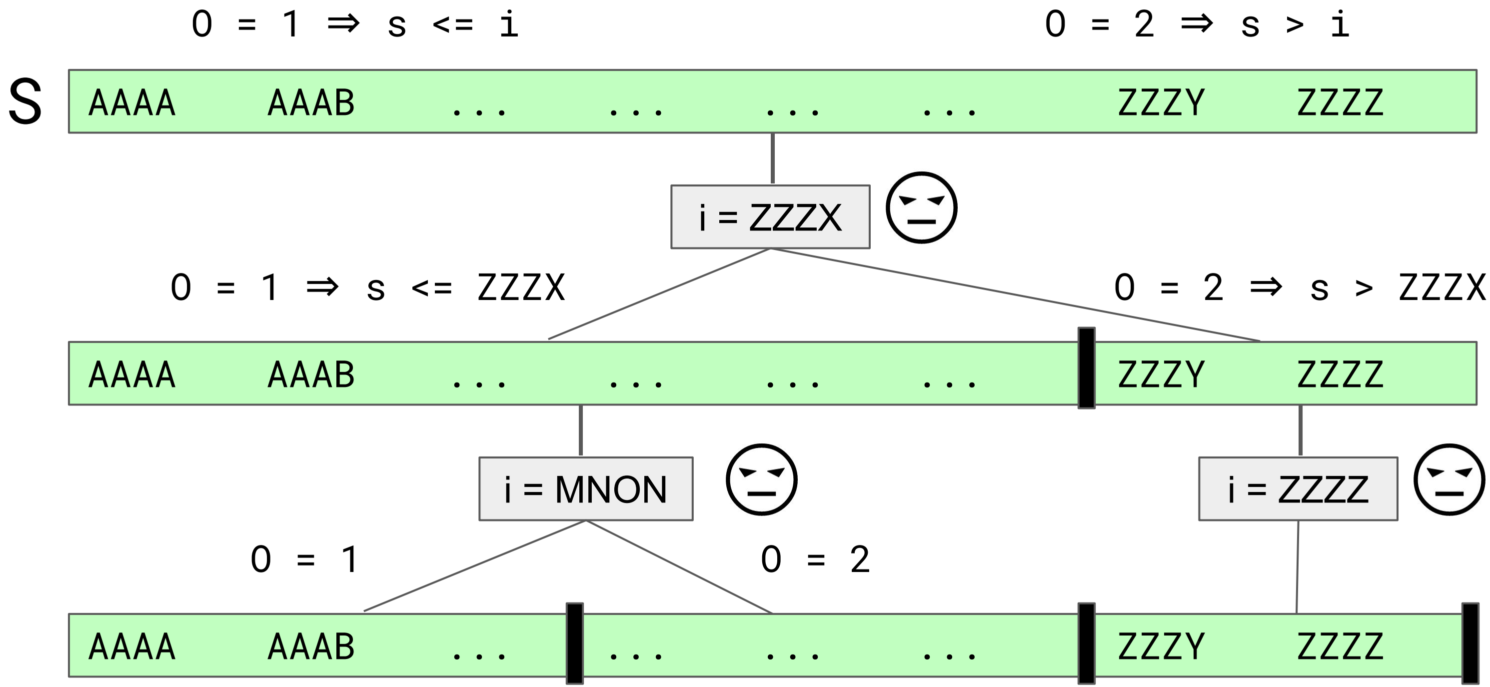
O = 1 ⇒ s <= i   O = 2 ⇒ s > i

S

AAAA	AAAB	...	...	...	...	ZZZY	ZZZZ



Attacker's input and observation **partitions** domain of S

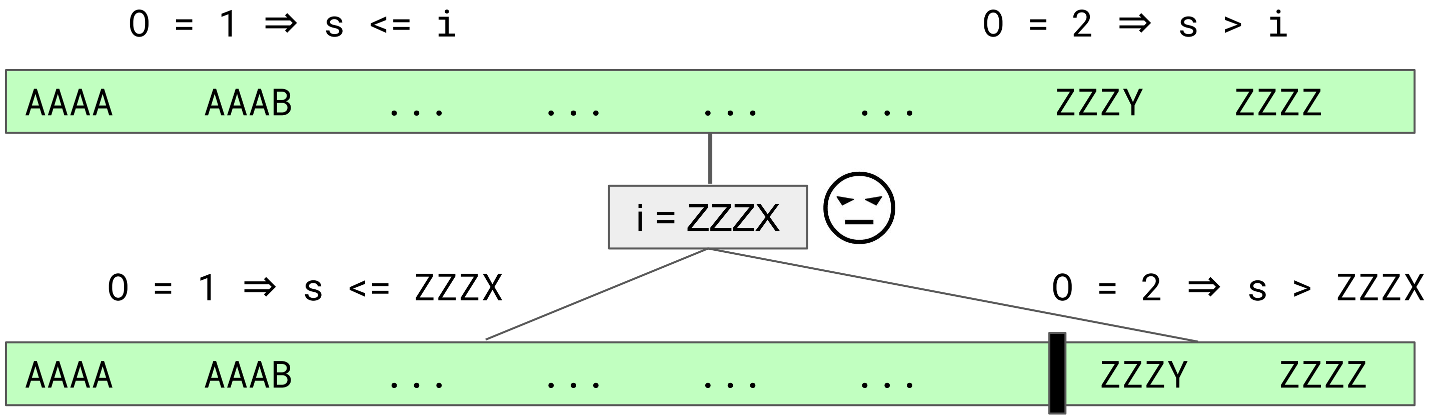


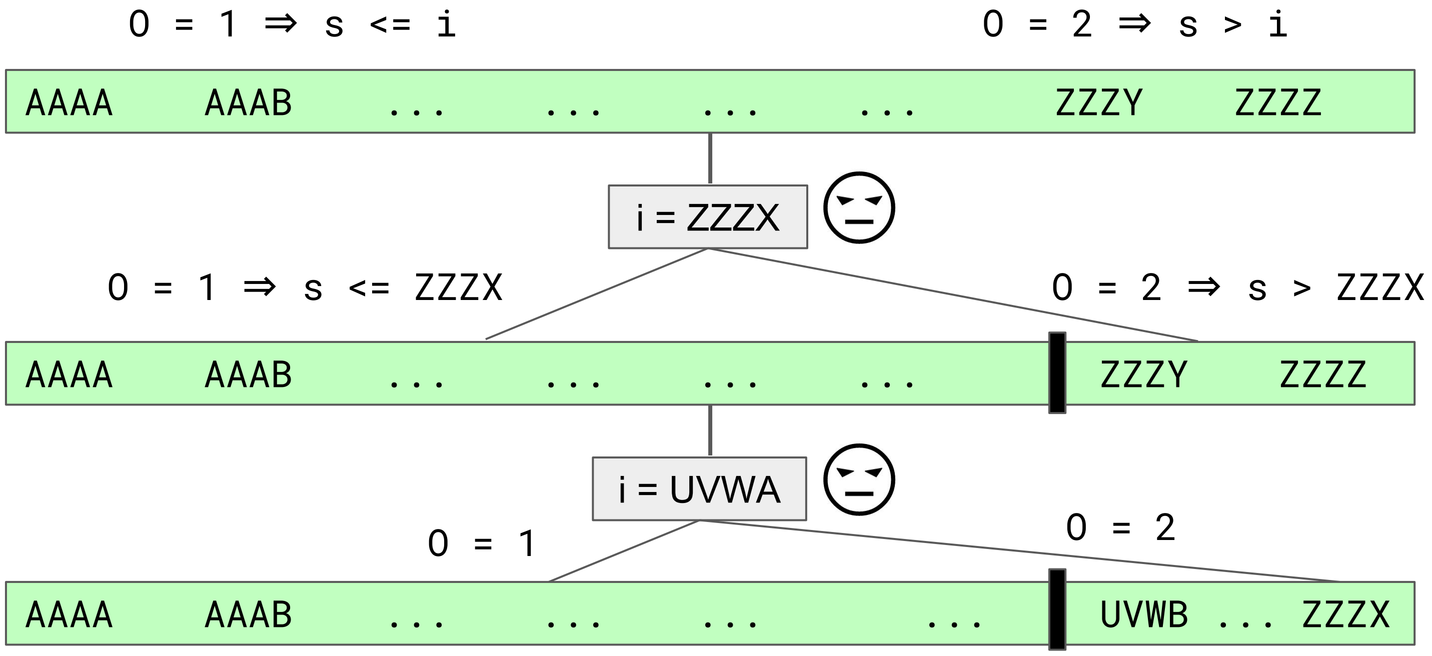
Attacker’s input and observation **sequences** partitions domain of S

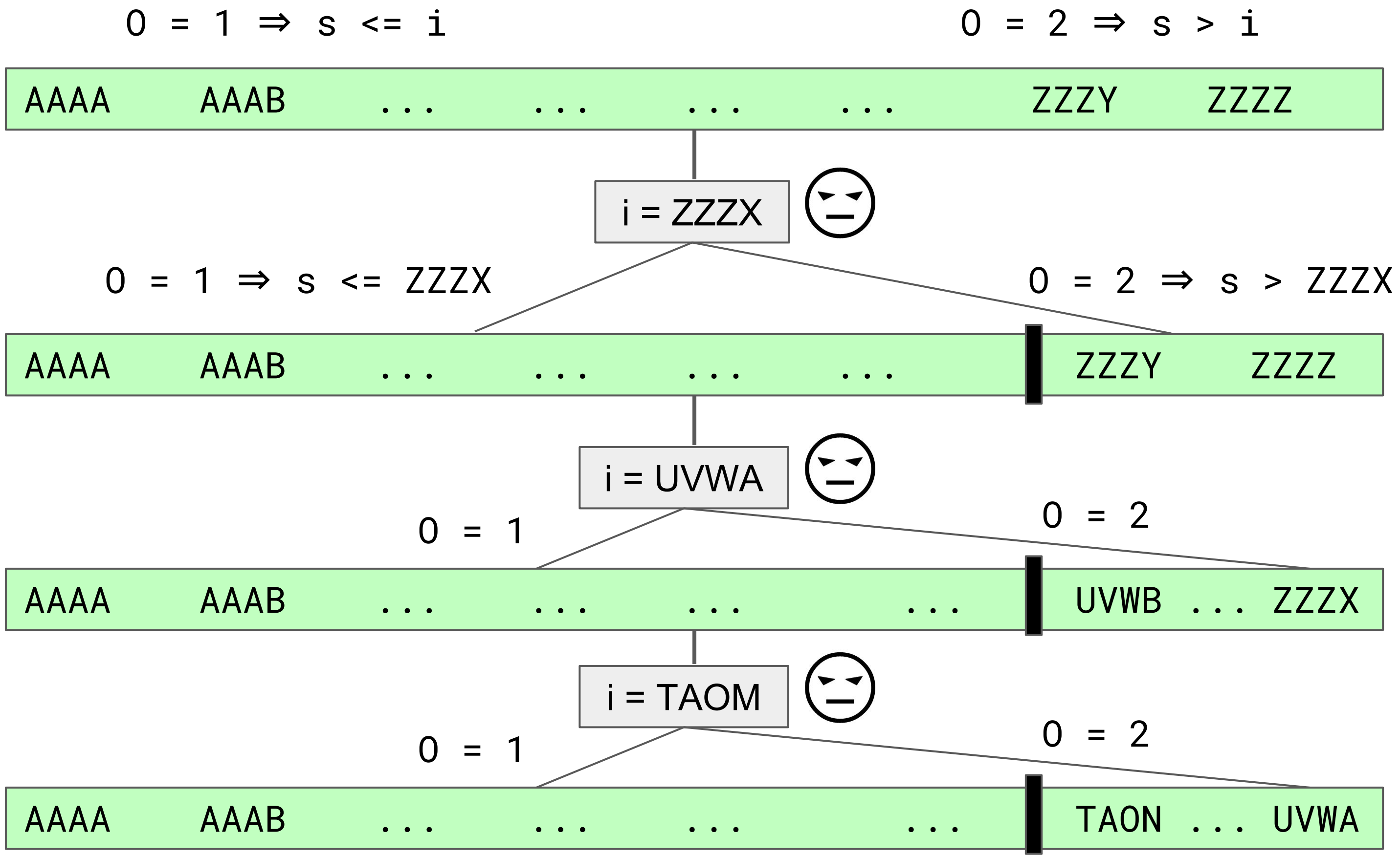
How input and observation affects partitioning?

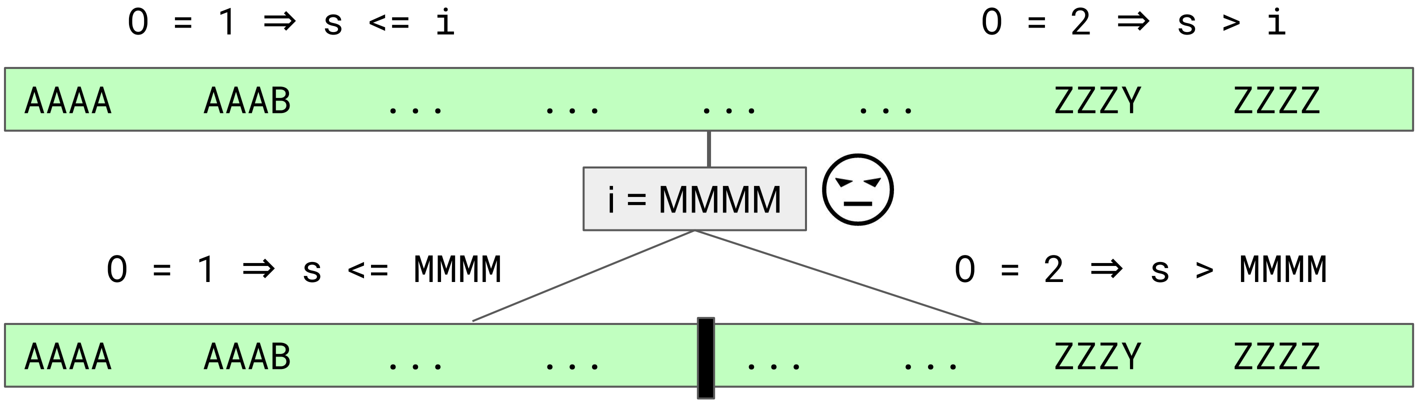
O = 1 ⇒ s <= i   O = 2 ⇒ s > i

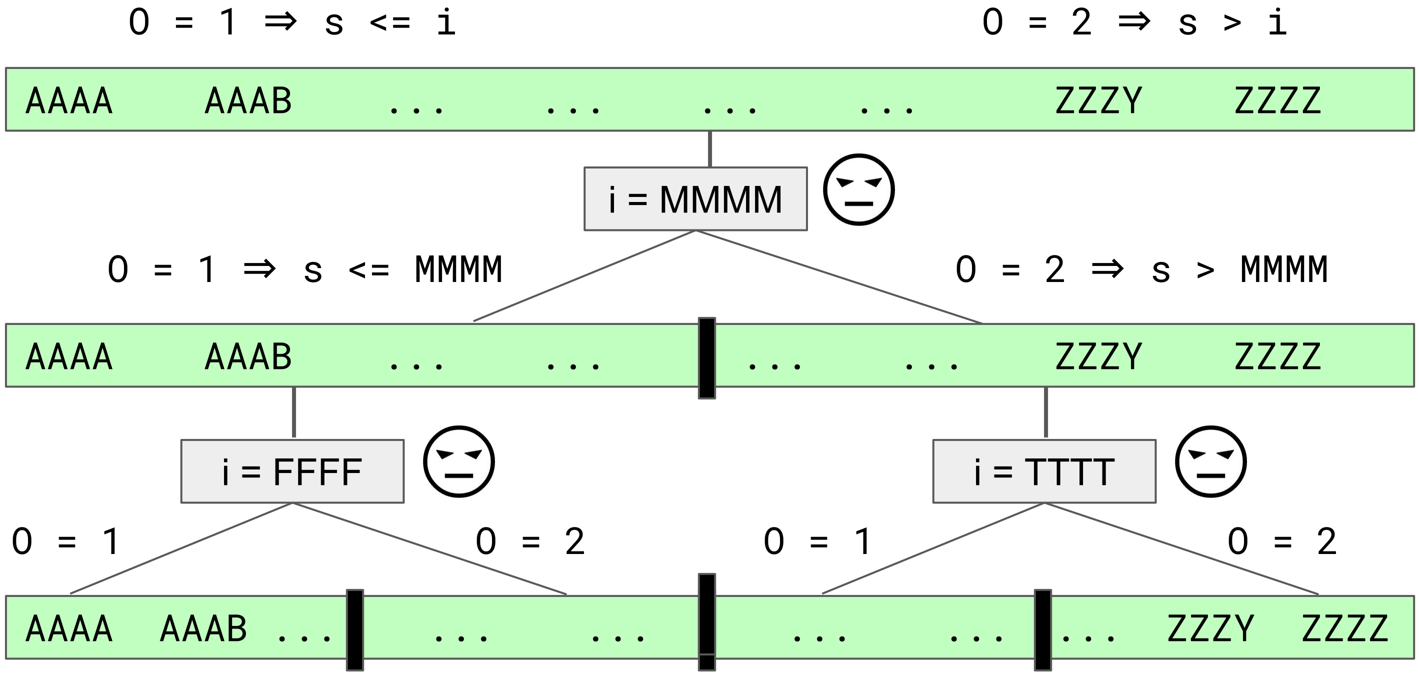
AAAA   AAAB   ...   MNOO   MNOP   ...   ZZZY   ZZZZ

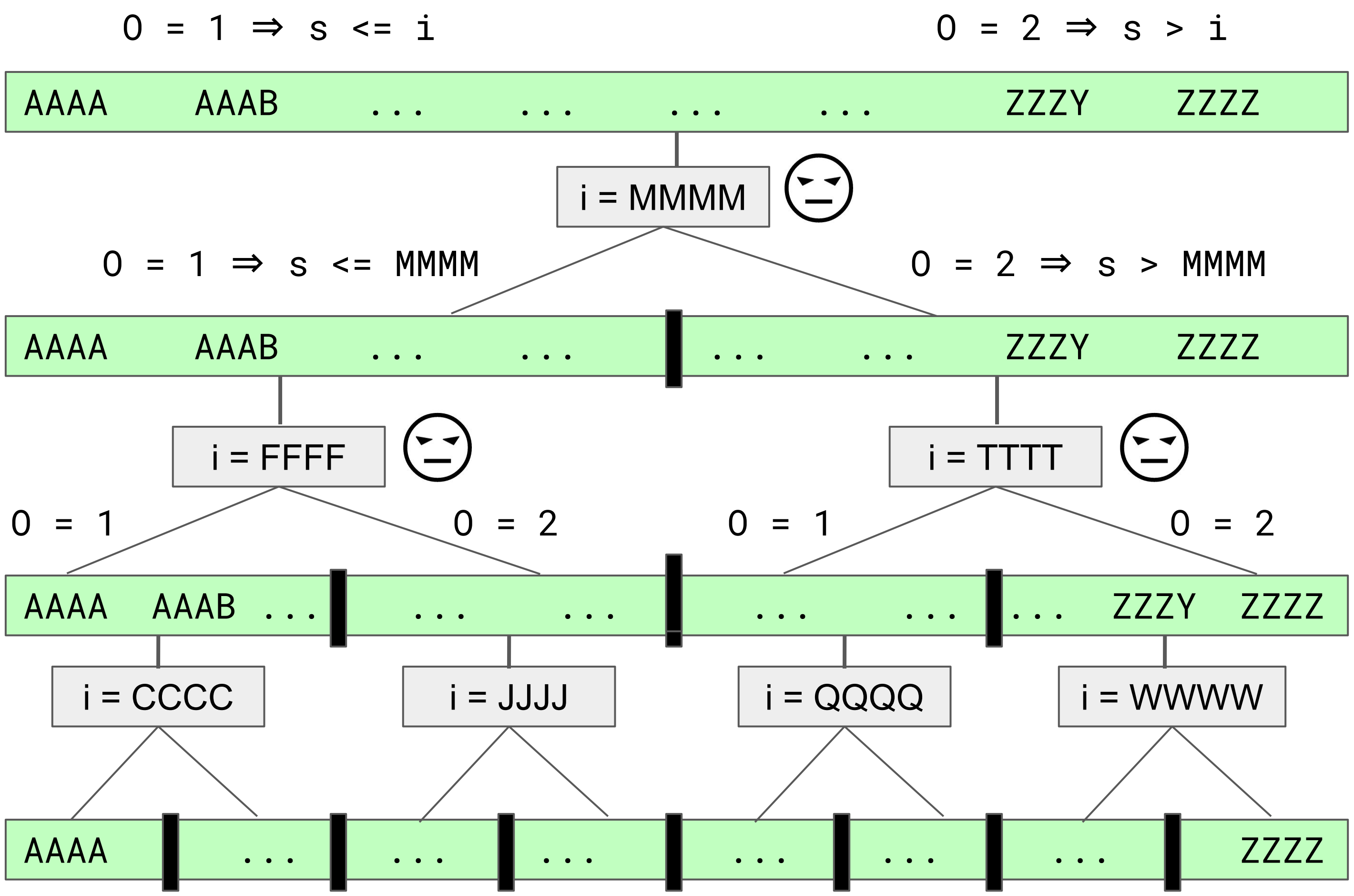

O = 1 ⇒ s <= i
O = 2 ⇒ s > i
AAAA AAAB ... ... ZZZY ZZZZ
i = ZZZX
O = 1 ⇒ s <= ZZZX
O = 2 ⇒ s > ZZZX
AAAA AAAB ... ... ZZZY ZZZZ


O = 1 ⇒ s <= i
O = 2 ⇒ s > i
AAAA AAAB ... ... ZZZY ZZZZ
i = ZZZX
O = 1 ⇒ s <= ZZZX
O = 2 ⇒ s > ZZZX
AAAA AAAB ... ... ZZZY ZZZZ
i = UVWA
O = 1
O = 2
AAAA AAAB ... ... UVWB ... ZZZX


O = 1 ⇒ s <= i
O = 2 ⇒ s > i
AAAA AAAB ... ... ZZZY ZZZZ
i = ZZZX
O = 1 ⇒ s <= ZZZX
O = 2 ⇒ s > ZZZX
AAAA AAAB ... ... ZZZY ZZZZ
i = UVWA
O = 1
O = 2
AAAA AAAB ... ... UVWB ... ZZZX
i = TAOM
O = 1
O = 2
AAAA AAAB ... ... TAON ... UVWA


O = 1 ⇒ s <= i
O = 2 ⇒ s > i
AAAA AAAB ... ... ZZZY ZZZZ
i = MMMM
O = 1 ⇒ s <= MMMM
O = 2 ⇒ s > MMMM
AAAA AAAB ... ... ZZZY ZZZZ


O = 1 ⇒ s <= i
O = 2 ⇒ s > i
AAAA AAAB ... ... ZZZY ZZZZ
i = MMMM
O = 1 ⇒ s <= MMMM
O = 2 ⇒ s > MMMM
AAAA AAAB ... ... ZZZY ZZZZ
i = FFFF
i = TTTT
O = 1
O = 2
O = 1
O = 2
AAAA AAAB ... ... ZZZY ZZZZ


O = 1 ⇒ s <= i
O = 2 ⇒ s > i
AAAA AAAB ... ... ZZZY ZZZZ
i = MMMM
O = 1 ⇒ s <= MMMM
O = 2 ⇒ s > MMMM
AAAA AAAB ... ... ZZZY ZZZZ
i = FFFF
i = TTTT
O = 1
O = 2
O = 1
O = 2
AAAA AAAB ... ... ZZZY ZZZZ
i = CCCC
i = JJJJ
i = QQQQ
i = WWWW
AAAA ... ... ZZZZ

Imbalanced partitions

$\Downarrow$

Worst case :

number of inputs = domain size = 26^4 = 456976

[number of alphabets = 26, length =4]

Balanced partitions

$\Downarrow$

Worst case :

number of inputs = $\log_2(456976) = 18.8$

[number of alphabets = 26, length =4]

Objective Function

Balanced partitions

$$\Downarrow$$

Maximizes information gain

Objective Function

O = 1 ⇒ s <= i   O = 2 ⇒ s > i

Maximize information gain ⇒ Binary Search

Objective Function

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

Maximize information gain ⇒ Binary Search

Programs in general

Maximize information gain ⇒ Optimal Search

Objective Function

information gain

$\Downarrow$

Shannon Entropy Formula

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

Shannon Entropy Formula

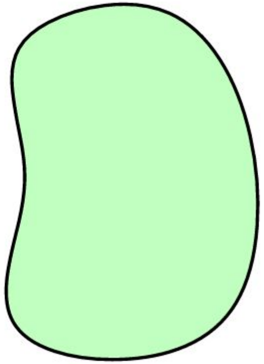
$$\mathcal{H} = \sum_{j=1}^{n} p_j \log_2 \frac{1}{p_j}$$

What is P_j?

How to calculate P_j?

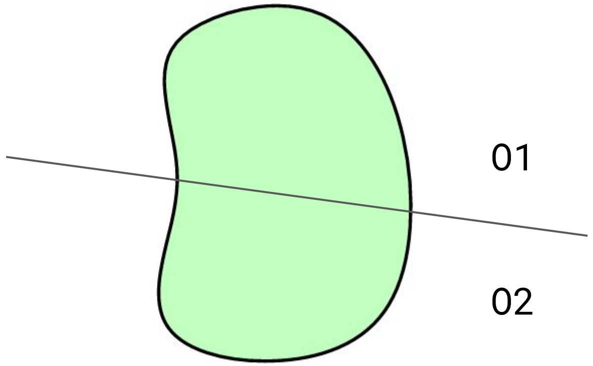
$i_0 \in I$

secret $s \in S$



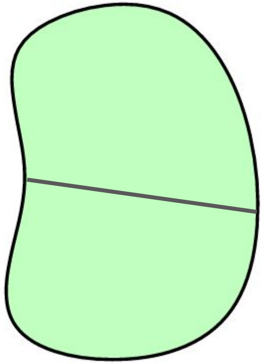
$i_0 \in I$

secret $s \in S$



$i_0 \in I$

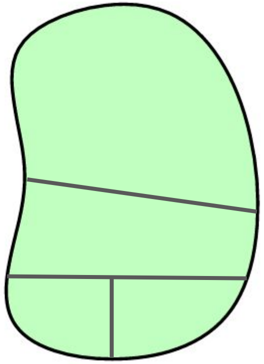
secret $s \in S$



$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



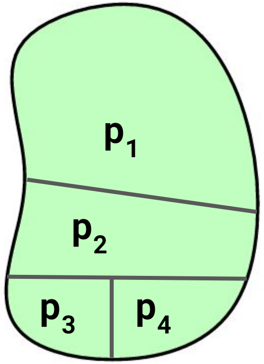
secret $s \in S$

secret $s \in S$

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$



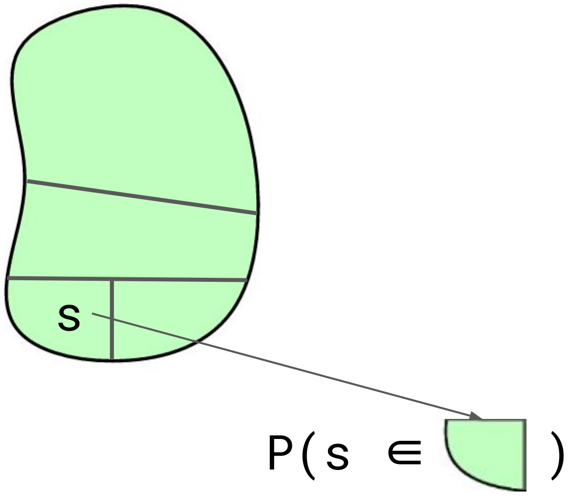
$$\mathscr{H} = \sum_{j=1}^{n} p_j \log_2 \frac{1}{p_j}$$

$i_0 \in I$

$i_1 \in I$

$i_2 \in I$

secret $s \in S$

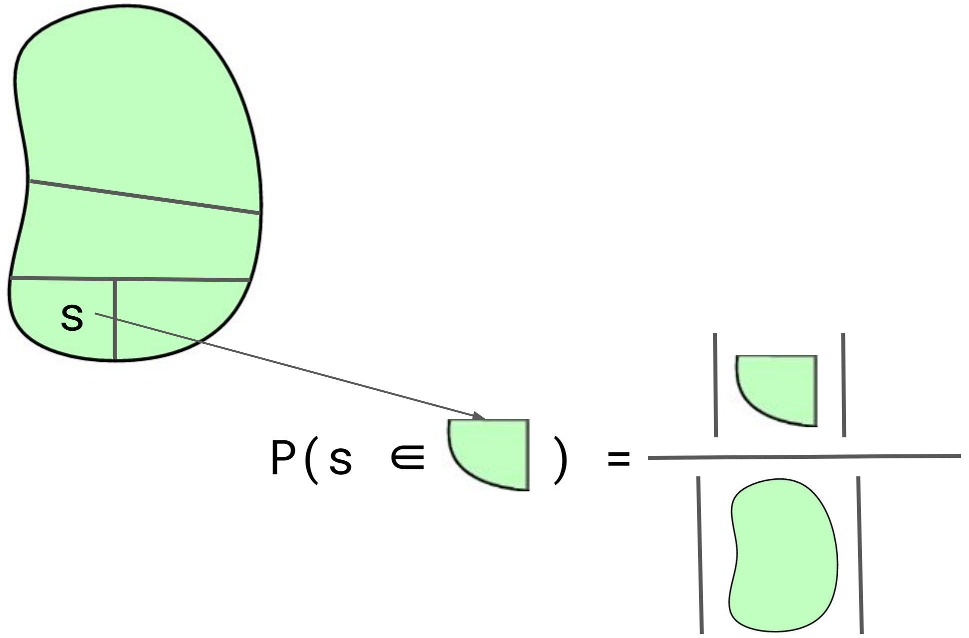


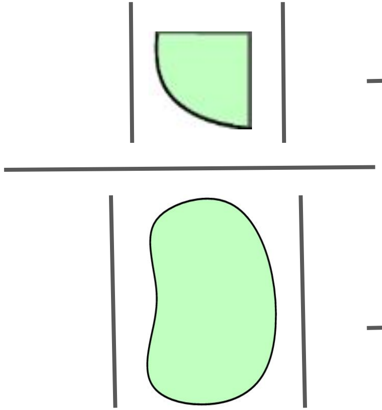
$i_0 \in I$

$i_1 \in I$

$i_2 \in I$

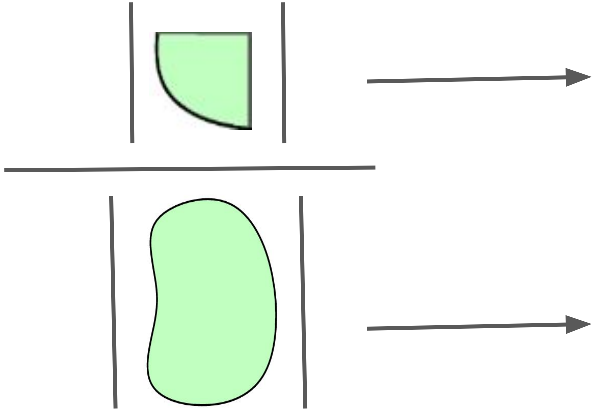
secret $s \in S$





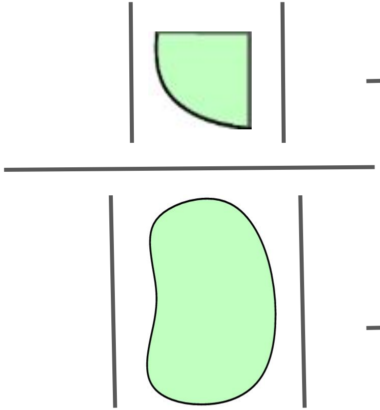
Number of secret inputs belong to this partition

Domain size



Number of secret inputs consistent with partition’s path constraint

Domain size



Count number of models for this constraint

Domain size

Model Counting Problem

Automata Based Model Counting (ABC)

Automata Based Model Counting (ABC)

Count the number of strings consistent with PC

Automata Based Model Counting (ABC)

Count the number of strings consistent with PC

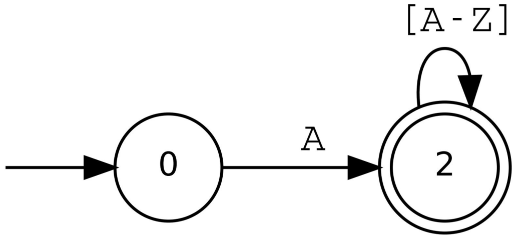
ABC constructs an automaton recognizing solution to PC

Automata Based Model Counting (ABC)

Count the number of strings consistent with PC

ABC constructs an automaton recognizing solution to PC

x in [A-Z]+ ^ charat(x,0)='A'

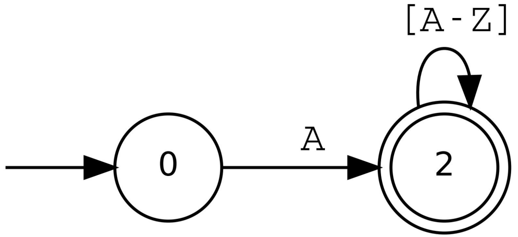


Automata Based Model Counting (ABC)

Count the number of strings consistent with PC

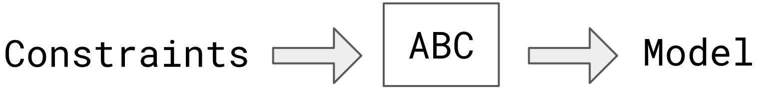
ABC constructs an automaton recognizing solution to PC

```
x in [A-Z]*  ^  charat(x,0)='A'
```

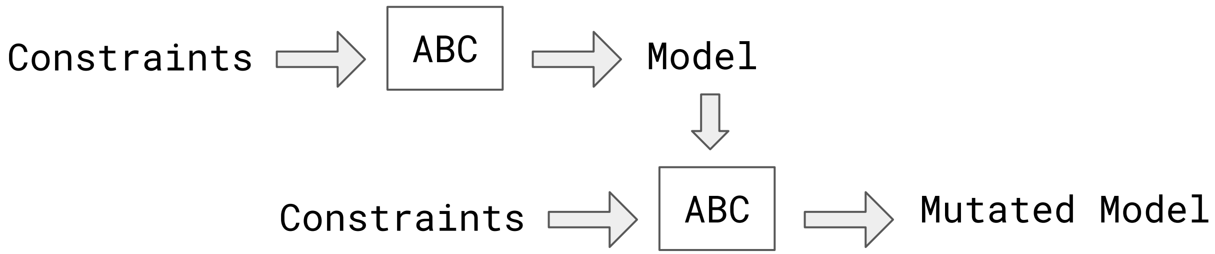


Model count (|PC|) is the number of accepting paths in automaton

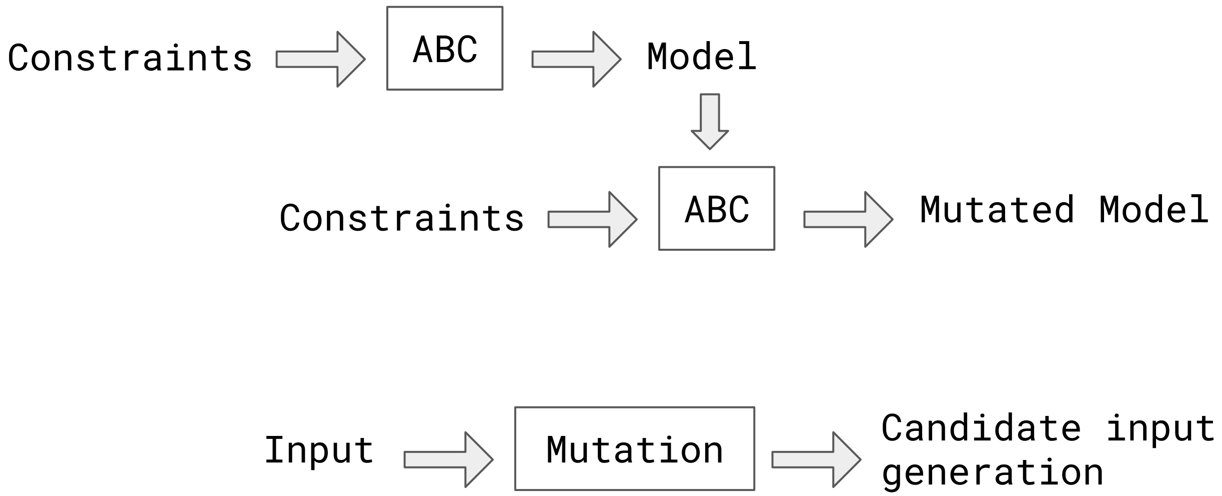
Constraint-based Model Generation



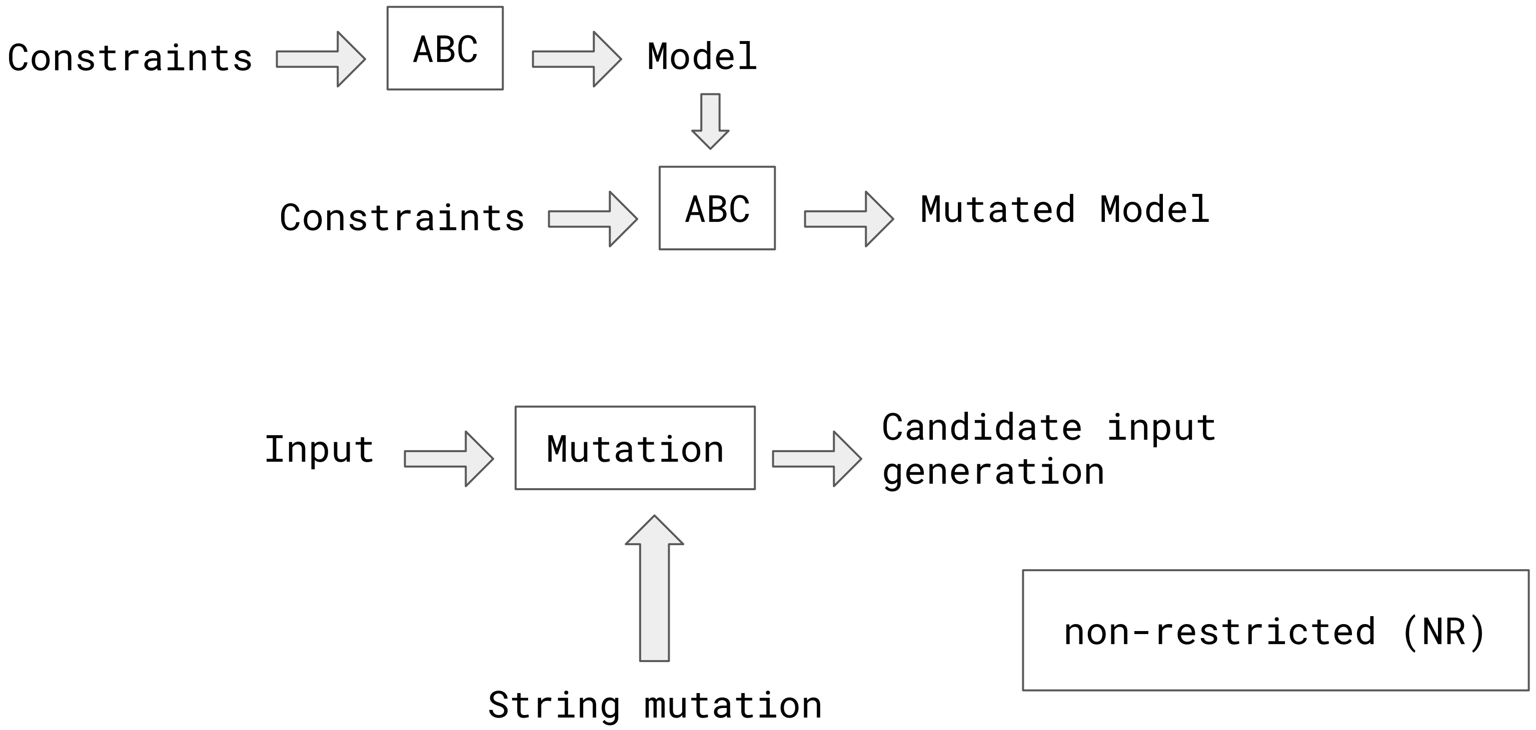
Constraint-based Model Generation



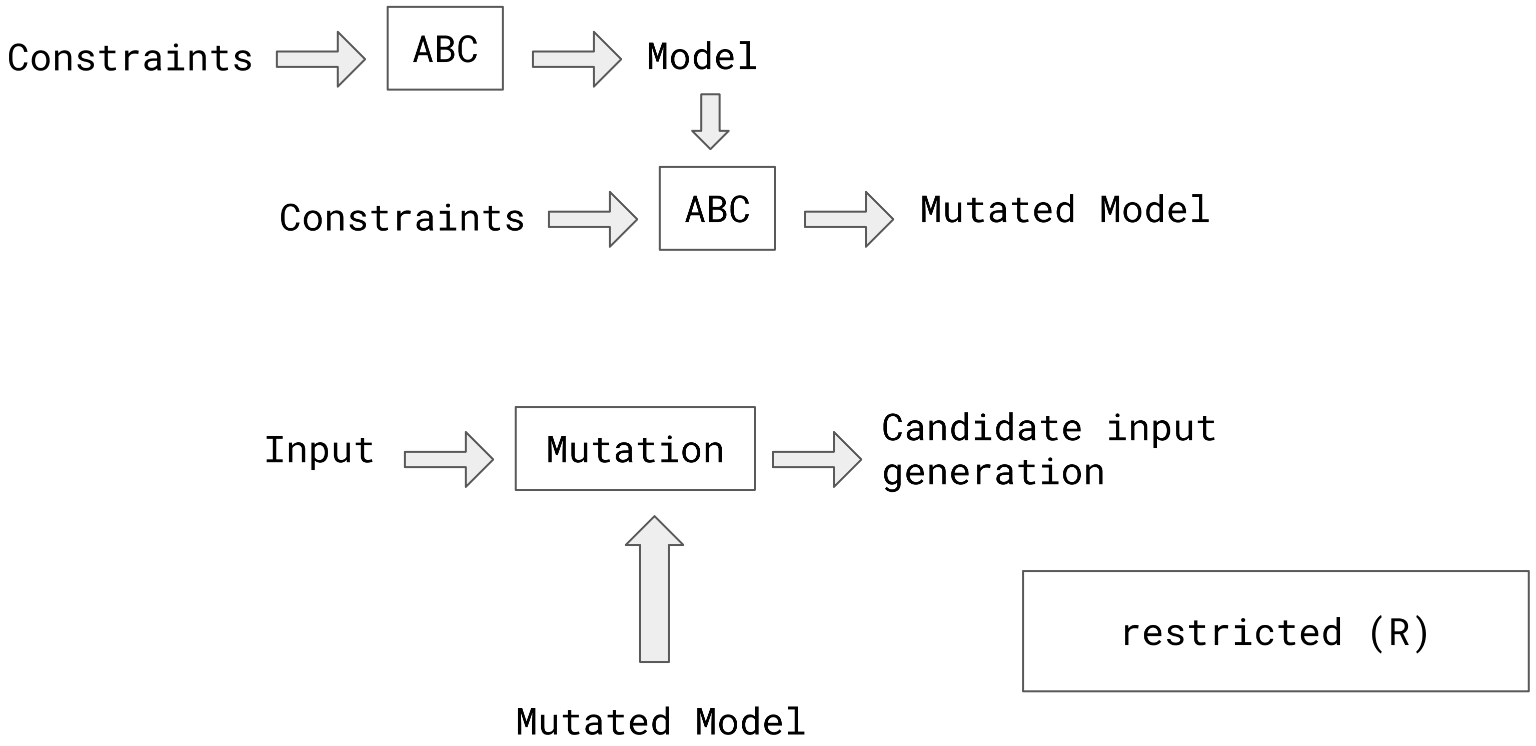
Constraint-based Model Generation



Constraint-based Model Generation



Constraint-based Model Generation



Maximize information gain ⇒ Optimal Search

Meta-heuristics Techniques

Random Search

Simulated Annealing

Genetic Algorithm

Meta-heuristics Techniques

Random Search

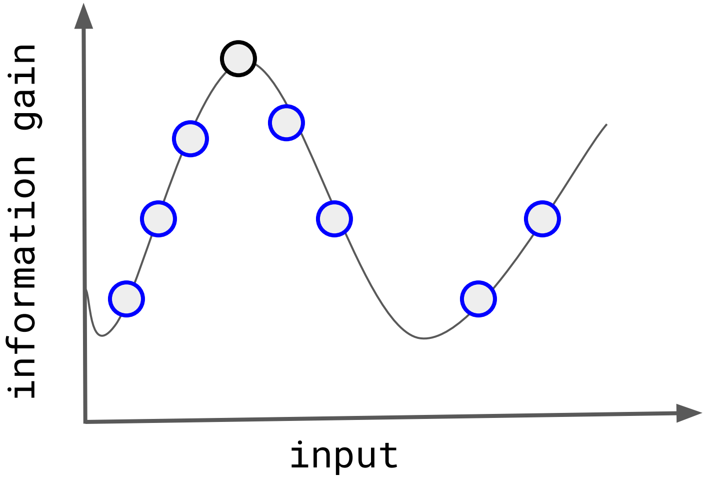
Simulated Annealing

Genetic Algorithm

- We implement and experiment these popular meta-heuristics techniques as
  - black box optimization procedures that
    - make repeated calls to ABC
    - to evaluate the information gain objective function

Random Search

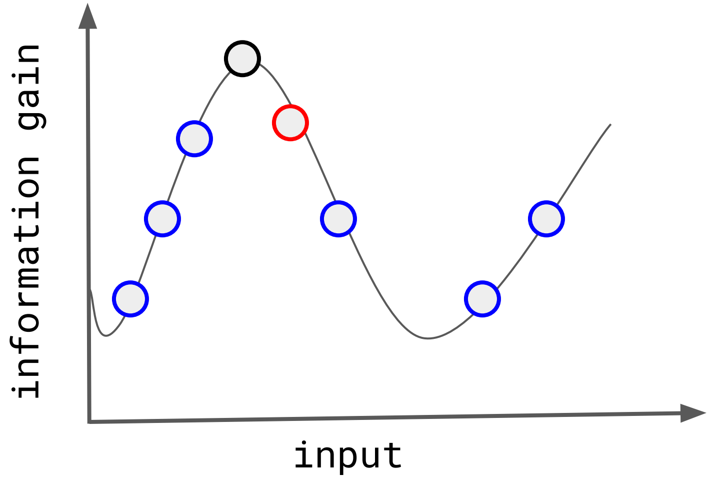
Calculate information gain for random candidate inputs



Random Search

Calculate information gain for random candidate inputs

Select candidate input with maximum information gain

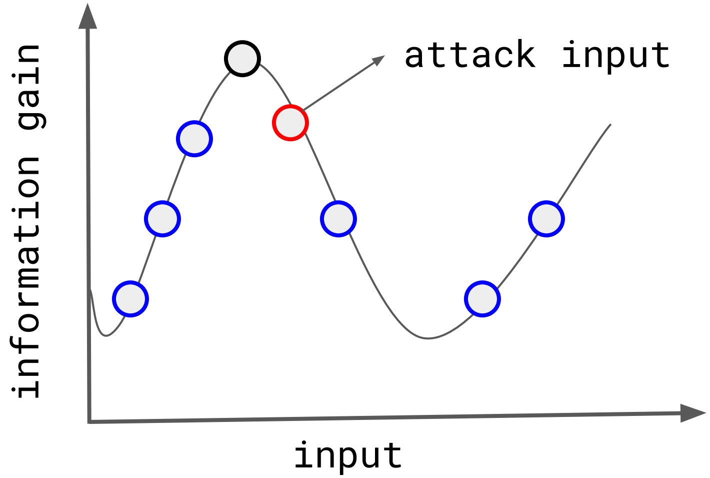


Random Search

Calculate information gain for random candidate inputs

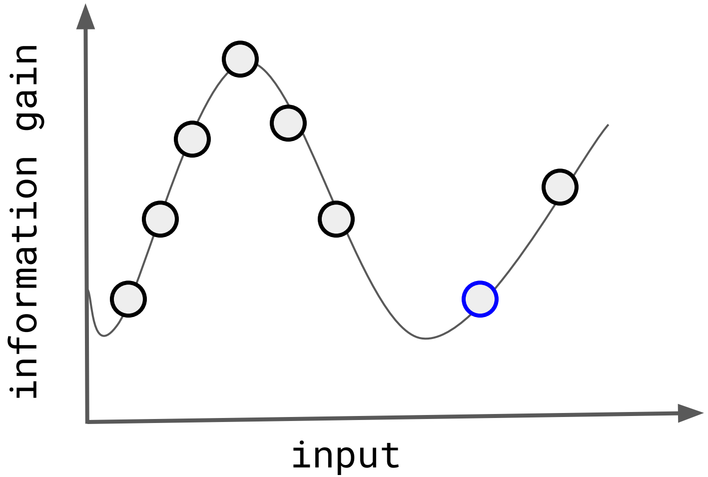
Select candidate input with maximum information gain

Use the candidate as next attack input



Simulated Annealing

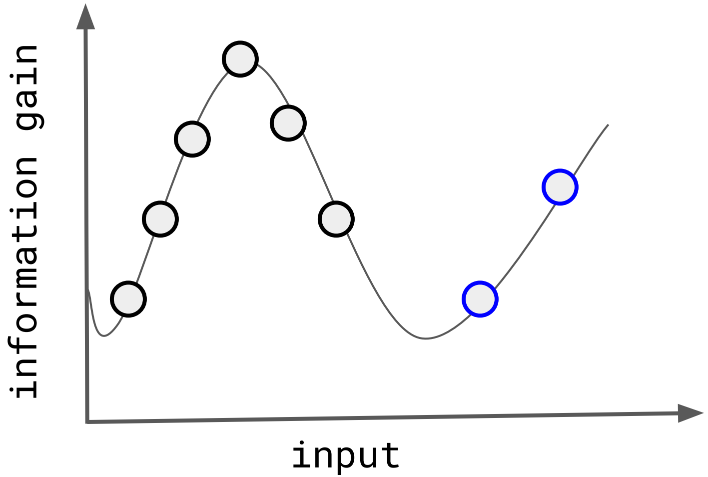
information gain for first candidate input



Simulated Annealing

information gain for first candidate input

information gain for new candidate input

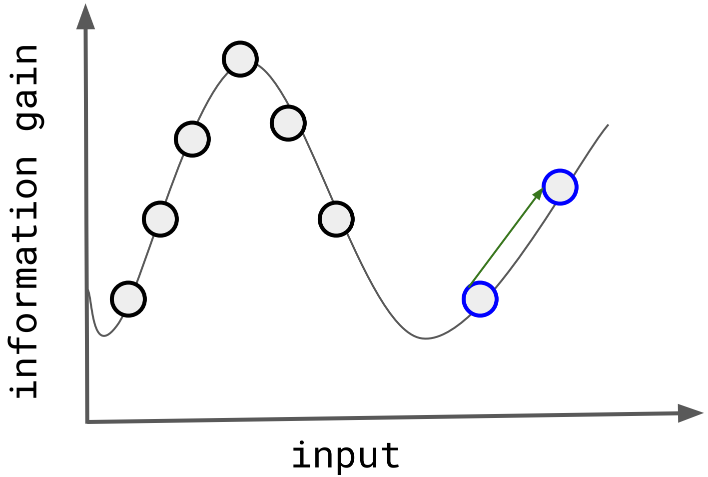


Simulated Annealing

information gain for first candidate input

information gain for new candidate input

better information gain ⇒ select as attack input



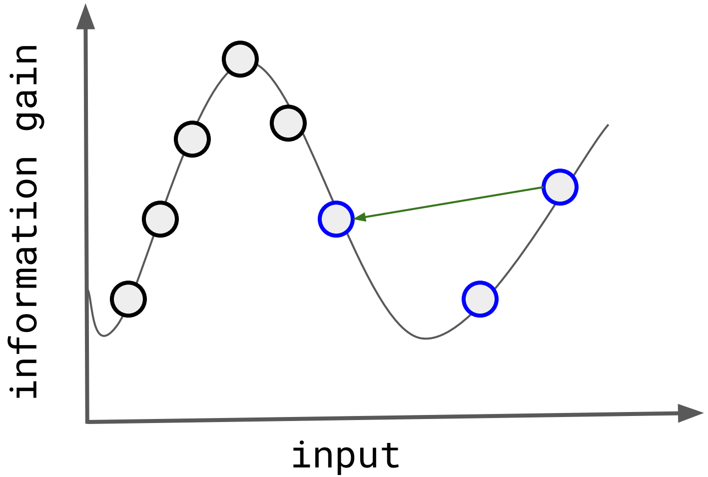
Simulated Annealing

information gain for first candidate input

information gain for new candidate input

better information gain ⇒ select as attack input

less information gain ⇒ select with an acceptance probability



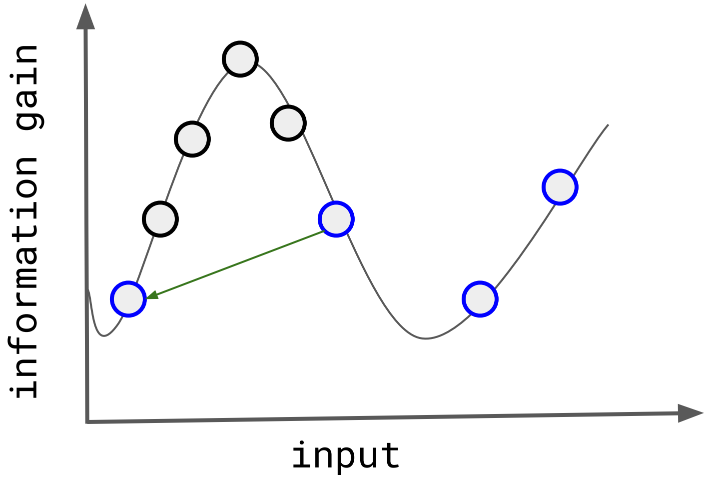
Simulated Annealing

information gain for first candidate input

information gain for new candidate input

better information gain ⇒ select as attack input

less information gain ⇒ select with an acceptance probability



Simulated Annealing

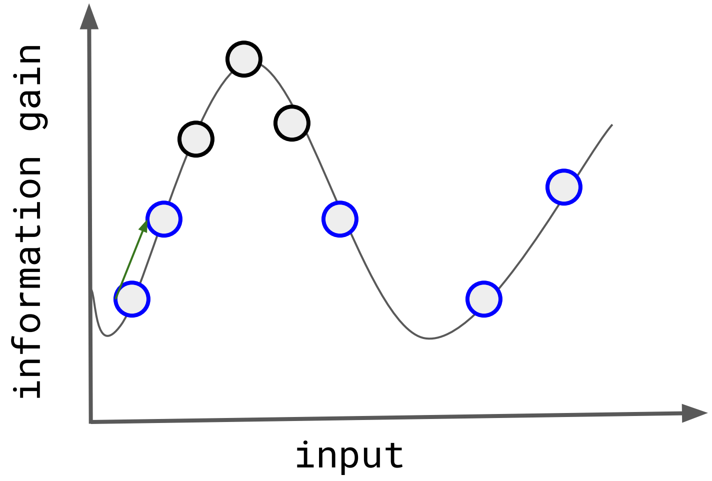
information gain for first candidate input

information gain for new candidate input

better information gain ⇒ select as attack input

less information gain ⇒ select with an acceptance probability

reduce acceptance probability as temperature cools down



Simulated Annealing

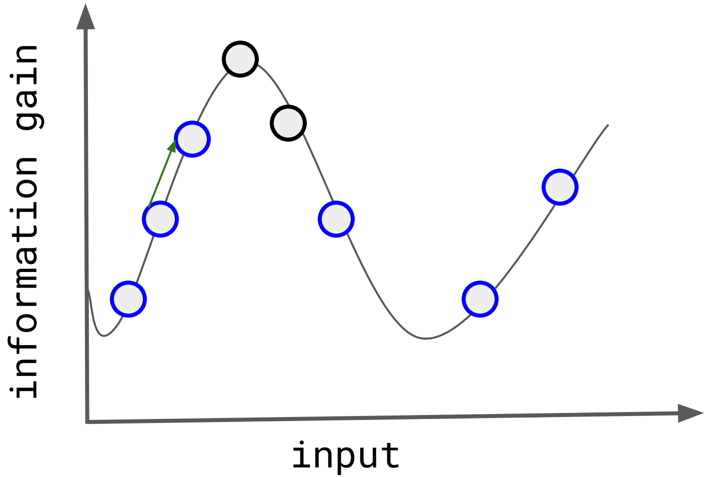
information gain for first candidate input

information gain for new candidate input

better information gain ⇒ select as attack input

less information gain ⇒ select with an acceptance probability

Reduce acceptance probability as temperature cools down



Simulated Annealing

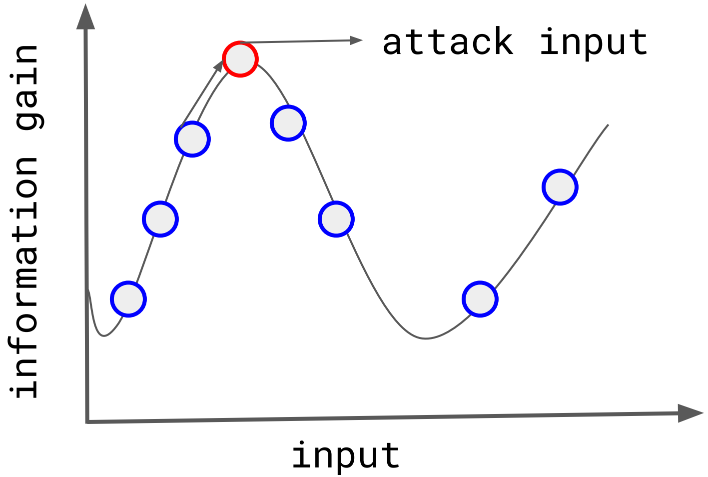
information gain for first candidate input

information gain for new candidate input

better information gain ⇒ select as attack input

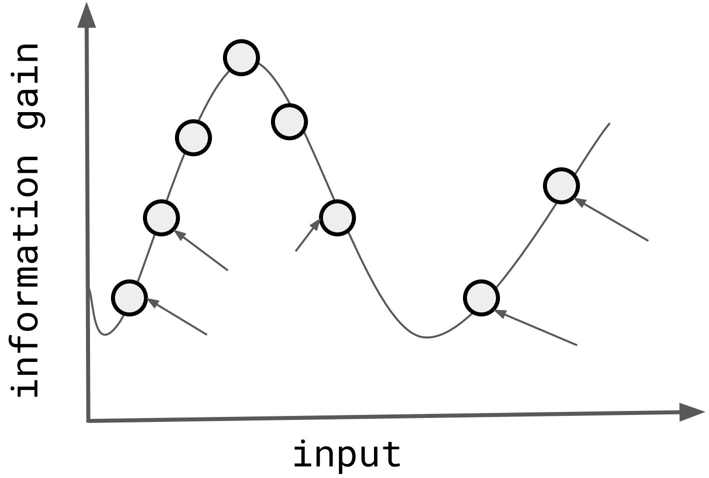
less information gain ⇒ select with an acceptance probability

Reduce acceptance probability as temperature cools down



Genetic Algorithm

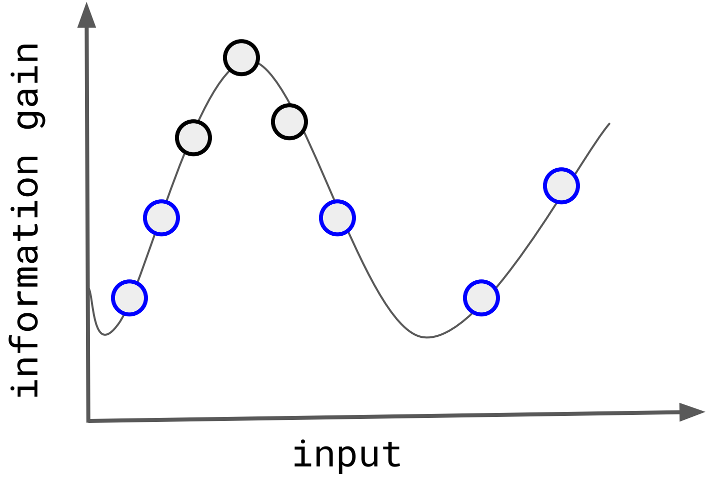
Population of candidate inputs



Genetic Algorithm

Population of candidate inputs

fitness (information gain) of these candidates

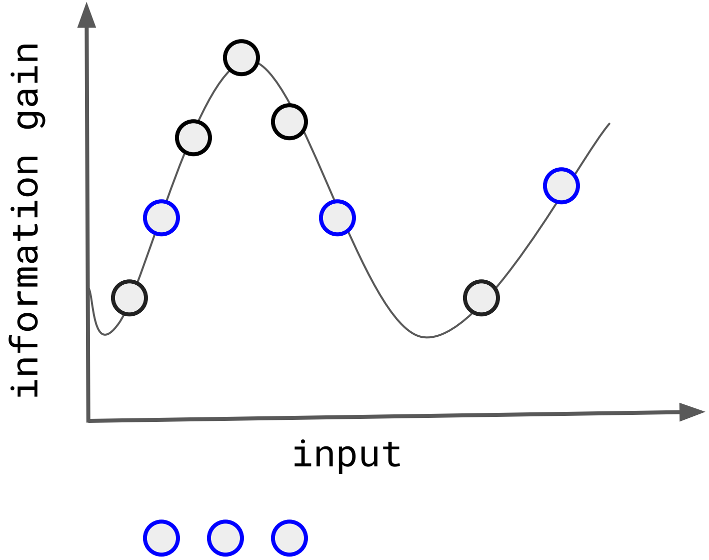


Genetic Algorithm

Population of candidate inputs

fitness (information gain) of these candidates

Select top candidates



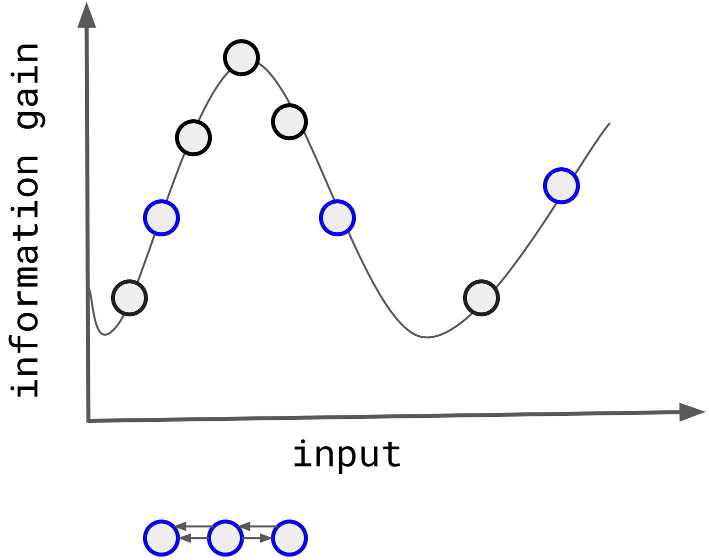
Genetic Algorithm

Population of candidate inputs

fitness (information gain) of these candidates

Select top candidates

Mutate and crossover



Genetic Algorithm

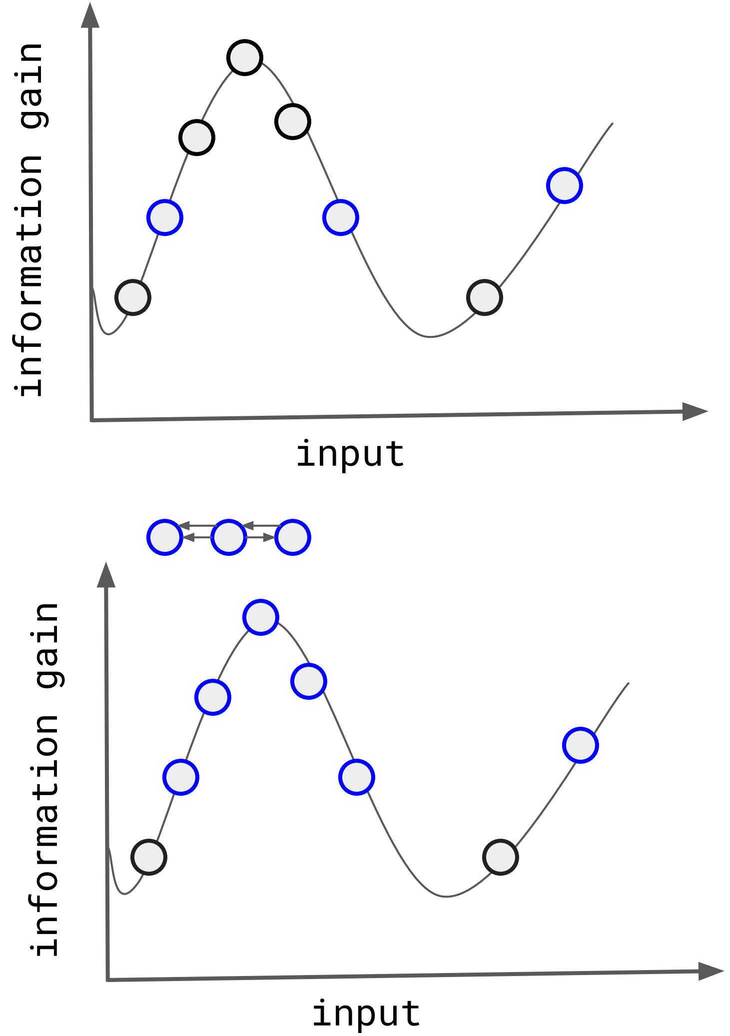
Population of candidate inputs

fitness (information gain) of these candidates

Select top candidates

Mutate and crossover

Update population



Genetic Algorithm

Population of candidate inputs

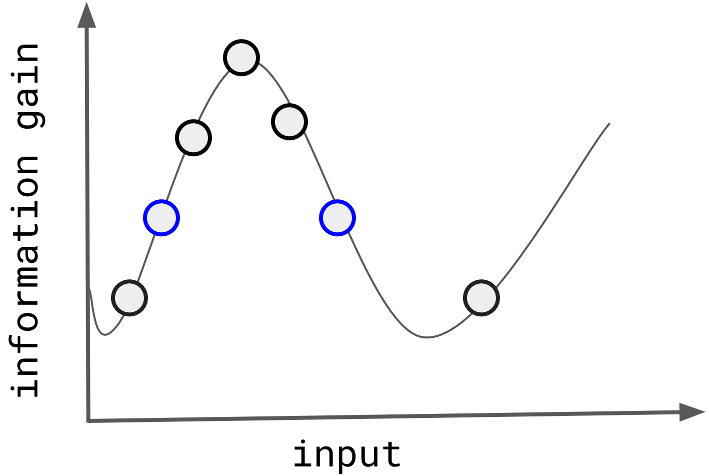
fitness (information gain) of these candidates

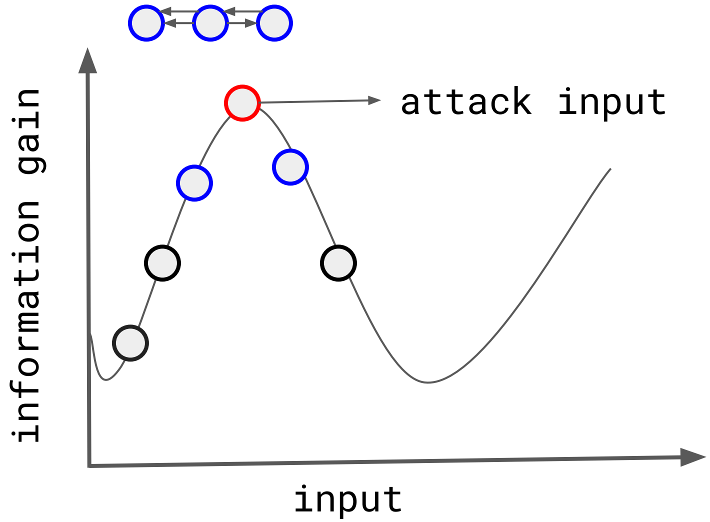
Select top candidates

Mutate and crossover

Update population

Select top candidate from population as attack input (l*)





Experimental Results

Experimental Benchmark

Benchmark	ID	Operations	Low Length	High Length	$\|\Phi\|$	$\|\Psi\|$
passCheckInsec	PCI	charAt,length	4	4	5	5
passCheckSec	PCS	charAt,length	4	4	5	1
stringEquals	SE	charAt,length	4	4	9	9
stringInequality	SI	$<, \geq$	4	4	2	2
stringCharInequality	SCI	charAt,length,$<, \geq$	4	4	80	2
indexOf	IO	charAt,length	1	8	9	9
compress	CO	begins,substring,length	4	4	5	5
editDistance	ED	charAt,length	4	4	2170	22

Experimental Benchmark

Benchmark	ID	Operations	Low Length	High Length	$\lvert\Phi\rvert$	$\lvert\Psi\rvert$
passCheckInsec	PCI	charAt,length	4	4	5	5
passCheckSec	PCS	charAt,length	4	4	5	1
stringEquals	SE	charAt,length	4	4	9	9
stringInequality	SI	$<,\geq$	4	4	2	2
stringCharInequality	SCI	charAt,length,$<,\geq$	4	4	80	2
indexOf	IO	charAt,length	1	8	9	9
compress	CO	begins,substring,length	4	4	5	5
editDistance	ED	charAt,length	4	4	2170	22

Experimental Benchmark

Benchmark	ID	Operations	Low Length	High Length	$\|\Phi\|$	$\|\Psi\|$
`passCheckInsec`	PCI	charAt,length	4	4	5	5
`passCheckSec`	PCS	charAt,length	4	4	5	1
`stringEquals`	SE	charAt,length	4	4	9	9
`stringInequality`	SI	$<, \geq$	4	4	2	2
`stringCharInequality`	SCI	charAt,length,$<, \geq$	4	4	80	2
`indexOf`	IO	charAt,length	1	8	9	9
`compress`	CO	begins,substring,length	4	4	5	5
`editDistance`	ED	charAt,length	4	4	2170	22

Experimental Benchmark

Benchmark	ID	Operations	Low Length	High Length	$\|\Phi\|$	$\|\Psi\|$
passCheckInsec	PCI	charAt,length	4	4	5	5
passCheckSec	PCS	charAt,length	4	4	5	1
stringEquals	SE	charAt,length	4	4	9	9
stringInequality	SI	$<, \geq$	4	4	2	2
stringCharInequality	SCI	charAt,length,$<, \geq$	4	4	80	2
indexOf	IO	charAt,length	1	8	9	9
compress	CO	begins,substring,length	4	4	5	5
editDistance	ED	charAt,length	4	4	2170	22

Number of path constraints

Number of merged path constraints

Experimental Benchmark

Benchmark	ID	Operations	Low Length	High Length	$\lvert\Phi\rvert$	$\lvert\Psi\rvert$
passCheckInsec	PCI	charAt,length	4	4	5	5
passCheckSec	PCS	charAt,length	4	4	5	1
stringEquals	SE	charAt,length	4	4	9	9
stringInequality	SI	$<, \geq$	4	4	2	2
stringCharInequality	SCI	charAt,length,$<, \geq$	4	4	80	2
indexOf	IO	charAt,length	1	8	9	9
compress	CO	begins,substring,length	4	4	5	5
editDistance	ED	charAt,length	4	4	2170	22

Number of path constraints

Number of merged path constraints

Experimental Benchmark

Benchmark	ID	Operations	Low Length	High Length	$\|\Phi\|$	$\|\Psi\|$
`passCheckInsec`	PCI	charAt,length	4	4	5	5
`passCheckSec`	PCS	charAt,length	4	4	5	1
`stringEquals`	SE	charAt,length	4	4	9	9
`stringInequality`	SI	$<, \geq$	4	4	2	2
`stringCharInequality`	SCI	charAt,length,$<, \geq$	4	4	80	2
`indexOf`	IO	charAt,length	1	8	9	9
`compress`	CO	begins,substring,length	4	4	5	5
`editDistance`	ED	charAt,length	4	4	2170	22

Number of path constraints

Number of merged path constraints

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Initial uncertainty of secret input (in bits)

Number of alphabets = 26

Length of secret = 4

Domain size of h = 26^4 = 456976

Initial uncertainty = $\log_2(456976)$ = 18.8

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Metrics:

- Time (in seconds)
- Number of attack steps
- Remaining Uncertainty

Remaining Uncertainty =
Initial uncertainty - information gain

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Techniques:

- Model Based
- Random search
- Simulated Annealing
- Genetic Algorithm

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Model Based:

- Shorter execution time per attack step
- More attack steps

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Simulated Annealing:

- Longer execution time per attack step
- Less attack steps

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Password Check Insecure:

- 1 hour timeout
- 5 observationally distinguishable path
- Better information leakage

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

Password Check Secure:

- 1 hour timeout
- 1 observationally distinguishable path
- Hardly leaks information
- Attack becomes exhaustive

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

String Char Inequality:
- 1 hour timeout
- 80 path constraints
- 2 observationally distinguishable path
- Information leakage rate is slower

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

String Edit Distance:
- 1 hour timeout
- 2170 path constraints
- 22 observationally distinguishable path
- Information leakage rate is slower

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

- Faster execution time per attack step than Simulated Annealing
- Need more attack steps than Simulated annealing

Reason:
Random search leads to less optimal input

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

- Faster than Simulated Annealing but
- Need more attack steps than Simulated annealing

Reason:
Mutation and crossover leads to non-restricted model

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

- Meta-heuristics does not perform better than model based when
    - Every input in a particular attack step leaks same amount of information

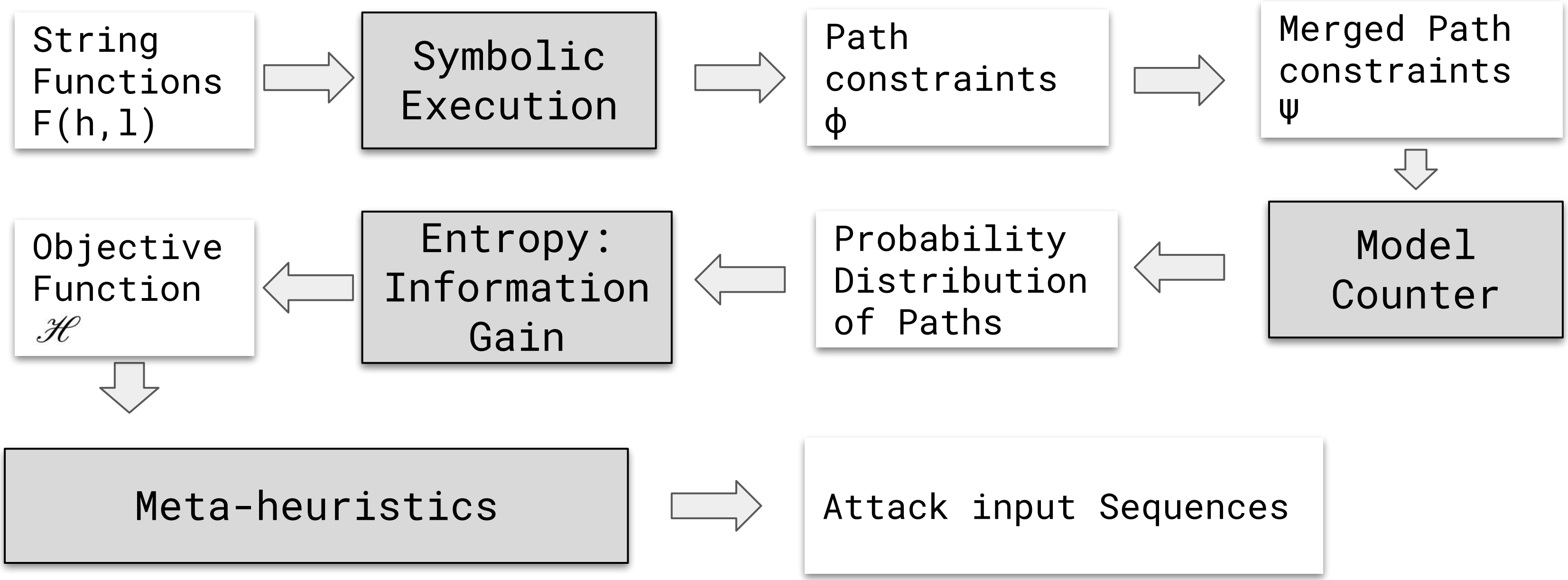
      For example: Password Check Insecure

Experimental Results

ID	$\mathcal{H}_{init}$	Metrics	M	RA	SA	GA
PCI	18.8	Time (s)	15.9	3600.0	3600.0	3600.0
		Steps	54.2	39.4	34.5	41.5
		$\mathcal{H}_{final}$	0.0	5.7	8.4	8.5
PCS	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	118.0	41.4	33.2	38.0
		$\mathcal{H}_{final}$	18.8	18.8	18.8	18.8
SE	18.8	Time (s)	22.0	3600.0	3600.0	3600.0
		Steps	62.2	42.6	25.3	30.8
		$\mathcal{H}_{final}$	0.0	6.1	11.1	8.4
SI	18.8	Time (s)	6.1	78.3	268.2	218.5
		Steps	38.2	18.6	17.5	18.2
		$\mathcal{H}_{final}$	0.0	0.0	0.0	0.0
SCI	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	34.6	4.0	2.0	2.0
		$\mathcal{H}_{final}$	12.9	16.2	17.7	17.5
IO	37.6	Time (s)	29.1	3600.0	3600.0	3600.0
		Steps	26.0	18.0	9.5	11.4
		$\mathcal{H}_{final}$	1.0	8.7	16.6	20.1
CO	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	734.0	147.0	83.0	97.8
		$\mathcal{H}_{final}$	13.48	9.2	10.3	9.1
ED	18.8	Time (s)	3600.0	3600.0	3600.0	3600.0
		Steps	27.6	1.0	1.0	1.0
		$\mathcal{H}_{final}$	12.6	17.8	17.8	17.8

- Model based attack:
  - simpler and faster execution of attack step
  - needs more attack step
- Meta-heuristics technique:
  - slower
  - need less attack step
- Simulated annealing:
  - performs better to leak information per attack step

Attack Synthesis for Strings using Meta-heuristics



Thank You