Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Priyanka Golia[1,2]

Joint work with : Kuldeep S. Meel[1] and Subhajit Roy[2]

[1]National University of Singapore

[2]Indian Institute of Technology, Kanpur

Boolean Functional Synthesis

- Given:
  A Boolean relation $F(X, Y)$, with inputs $X = \{x_1, \ldots, x_n\}$, and outputs $Y = \{y_1, \ldots, y_m\}$

- Problem Statement:
  Synthesise a function vector $\Psi = \langle \psi_1 \ldots \psi_m \rangle$, where ψ_i is a function for variable y_i,

$$y_i = \psi_i(x_1, \ldots, x_n)$$

  such that

$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

- Each ψ_i is called Skolem function, and Ψ is called Skolem function vector.

Boolean Functional Synthesis

- Given:
  A Boolean relation $F(X, Y)$, with inputs $X = \{x_1, \ldots, x_n\}$, and outputs $Y = \{y_1, \ldots, y_m\}$

- Problem Statement:
  Synthesise a function vector $\Psi = \langle \psi_1 \ldots \psi_m \rangle$, where ψ_i is a function for variable y_i,

  $$y_i = \psi_i(x_1, \ldots, x_n)$$

  such that

  $$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

  Our objective is to synthesis Ψ

- Each ψ_i is called Skolem function, and Ψ is called Skolem function vector.

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$$x_1 \vee x_2 \vee y_1$$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$x_1 \vee x_2 \vee y_1$ $\longrightarrow$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$$x_1 \vee x_2 \vee y_1 \quad \longrightarrow \quad \neg(x_1 \vee x_2) \rightarrow y_1$$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$$x_1 \vee x_2 \vee y_1 \longrightarrow \neg(x_1 \vee x_2) \rightarrow y_1$$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$$x_1 \vee x_2 \vee y_1 \longrightarrow \neg(x_1 \vee x_2) \rightarrow y_1$$

$$\downarrow$$

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$$x_1 \vee x_2 \vee y_1 \longrightarrow \neg(x_1 \vee x_2) \rightarrow y_1$$

$$\downarrow$$

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

$F(X, \Psi(X)) = x_1 \vee x_2 \vee \neg(x_1 \vee x_2)$

Skolem Function: Example

Let $X = \{x_1, x_2\}$ and $Y = \{y_1\}$

$F(X, Y) : x_1 \vee x_2 \vee y_1$

Skolem functions:

$$x_1 \vee x_2 \vee y_1 \longrightarrow \neg(x_1 \vee x_2) \rightarrow y_1$$

$$\downarrow$$

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

$$F(X, \Psi(X)) = x_1 \vee x_2 \vee \neg(x_1 \vee x_2)$$

$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

Skolem Function: Example

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$
- Possible Skolem functions:

Skolem Function: Example

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$
- Possible Skolem functions:

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

Skolem Function: Example

- $F(x_1, x_2, y_1) : \ x_1 \vee x_2 \vee y_1$
- Possible Skolem functions:

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

$$\psi_1(x_1, x_2) = \neg x_1$$

Skolem Function: Example

- $F(x_1, x_2, y_1) : \ x_1 \vee x_2 \vee y_1$
- Possible Skolem functions:

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

$$\psi_1(x_1, x_2) = \neg x_1$$

$$\psi_1(x_1, x_2) = \neg x_2$$

Skolem Function: Example

- $F(x_1, x_2, y_1): \ x_1 \vee x_2 \vee y_1$
- Possible Skolem functions:

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$$

$$\psi_1(x_1, x_2) = \neg x_1$$

$$\psi_1(x_1, x_2) = \neg x_2$$

$$\psi_1(x_1, x_2) = 1$$

Applications

- Application in a wide variety of domains:
    - Certified QBF solving
    - Program synthesis
    - Cryptography

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F

Ψ

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

X

Boolean relation F

Ψ

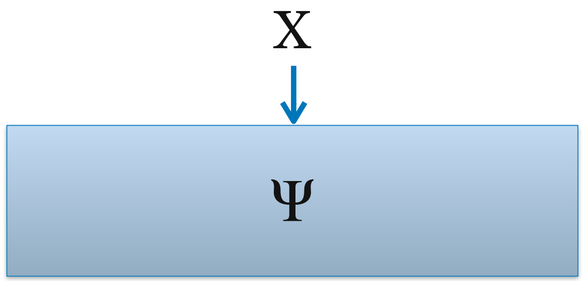
Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F



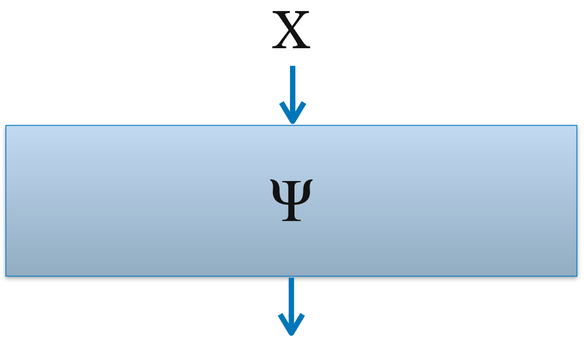
Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F



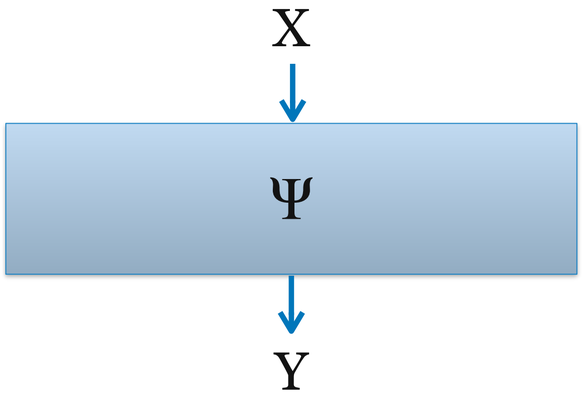
Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F



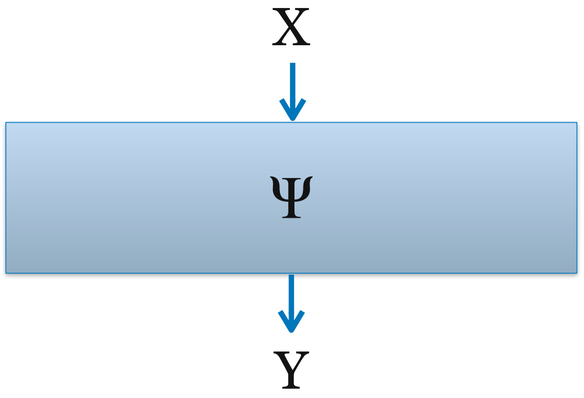
Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F



$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

Factorization Problem

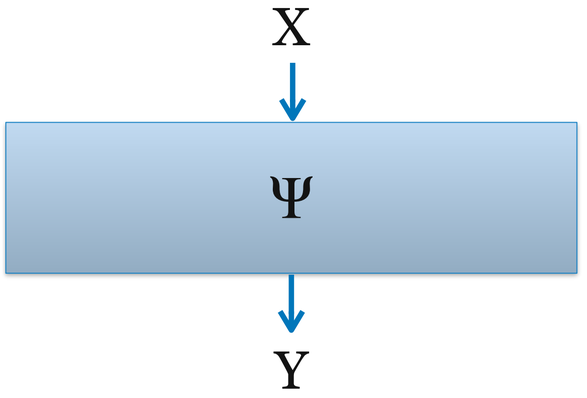
- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$

Boolean relation F

Boolean functional synthesis



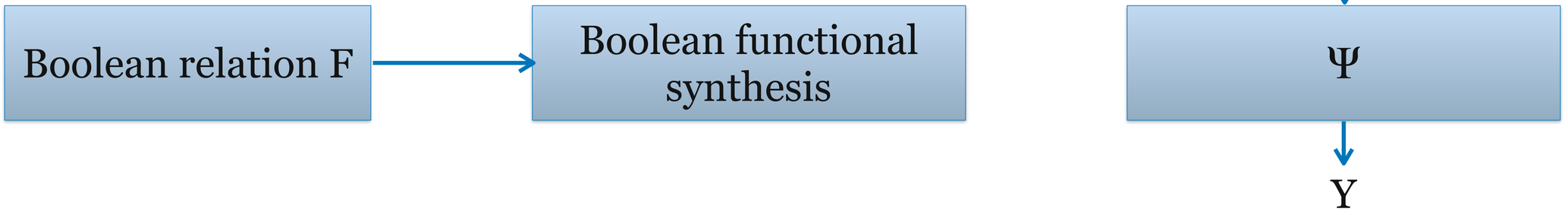
$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$



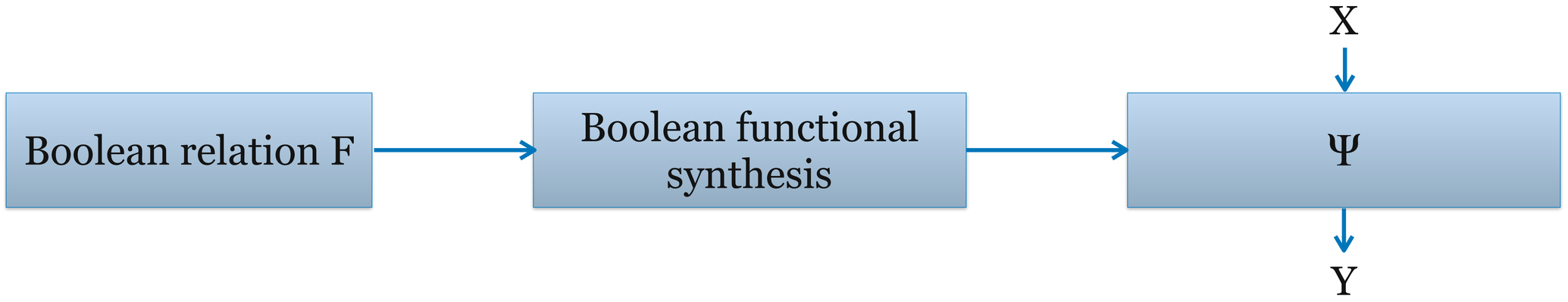
$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

Factorization Problem

- Given a n-bit number X, find m-bit number Y such that Y divides X and $Y \notin \{1, X\}$.

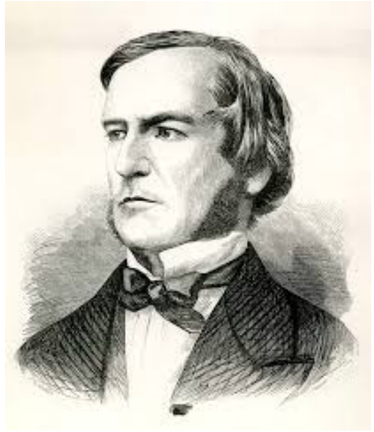
- As a relation between input and output values

$$F(X, Y) : \frac{X}{Y} \in \mathbb{Z} \text{ and } Y \notin \{1, X\}$$



$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

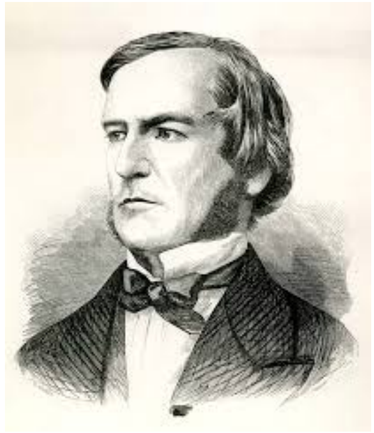
State of the Art



George Boole

Thoralf Albert Skolem

State of the Art



George Boole

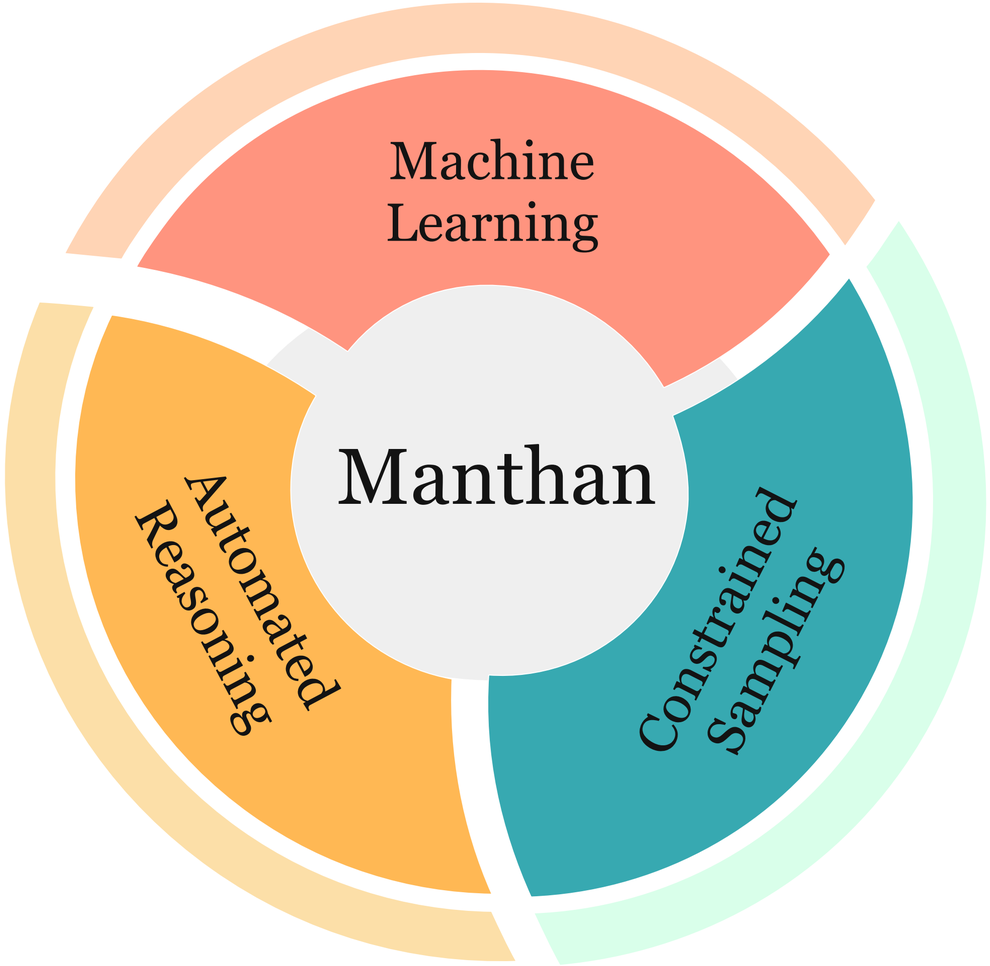
Thoralf Albert Skolem

- Theoretically

  Unless P = NP, there exist problem instance where Boolean function synthesis must take super-polynomial time. (Akshay *et al.* ,2018)

State of the Art

- From the proof of validity of $\forall X \exists Y F(X, Y)$
  - Bendetti et al, 2005
  - Jussilla et al, 2007
  - Heule et al, 2014
- Quantifier instantiation in SMT solvers
  - Barrett et al, 2015
  - Bierre et al,2017
- Input-Output separation:
  - Chakraborty et al., 2018
- Knowledge representation:
  - Kukula et al, 2000
  - Trivedi et al, 2003
  - Jiang, 2009
  - Kuncak et al., 2010
  - Balabanov and Jiang, 2011
  - John et al., 2015
  - Fried, Tabajara, Vardi, 2016,2017
  - Akshay et al., 2017,2018
  - Chakraborty et al., 2019
- Incremental determinization:
  - Rabe et. al, 2015, 2018, 2019

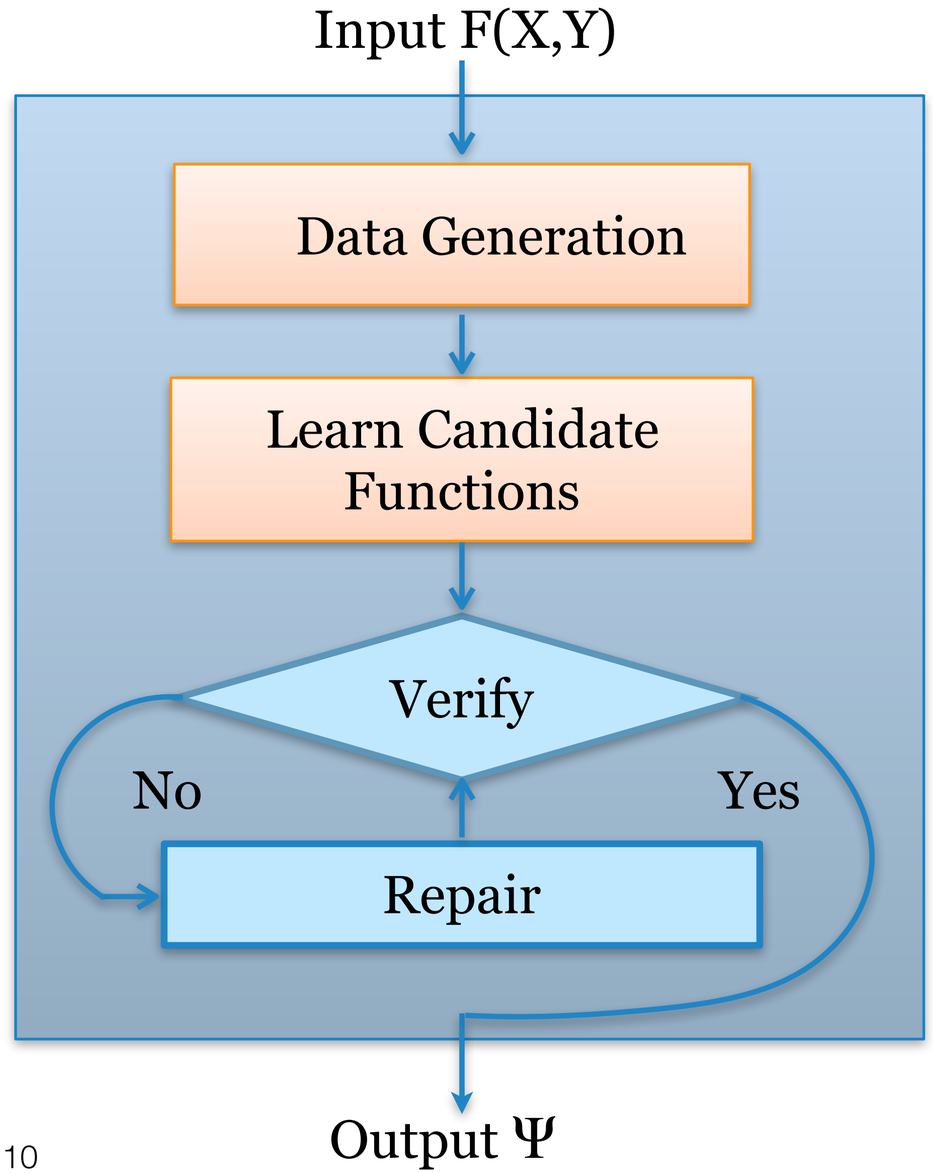


A Data-Driven Approach for Boolean Functional Synthesis

Manthan

Learn Candidate Functions

Repair Candidate Functions



Data Generation

Data Generation

- Satisfying assignments of $F(X, Y)$: the valuations of X and Y that satisfy the $F(X, Y)$.

Data Generation

- Satisfying assignments of $F(X, Y)$: the valuations of X and Y that satisfy the $F(X, Y)$.

But, the possible solution space is $2^{|X|+|Y|}$

Data Generation

- Satisfying assignments of $F(X, Y)$: the valuations of X and Y that satisfy the $F(X, Y)$.

But, the possible solution space is $2^{|X|+|Y|}$

- Sample uniformly at random from the solution space of satisfying assignments.

Data Generation

- We want to capture the relation between X and Y.

$$F(x_1, x_2, y_1, y_2): \ (x_1 \vee x_2 \vee y_1) \wedge (\neg x_1 \vee \neg x_2 \vee \neg y_2)$$

x_1	x_2	y_1	y_2
0	0	1	0/1
0	1	0/1	0/1
1	0	0/1	0/1
1	1	0/1	0

Data Generation

- We want to capture the relation between X and Y.

$$F(x_1, x_2, y_1, y_2): \ (x_1 \vee x_2 \vee y_1) \wedge (\neg x_1 \vee \neg x_2 \vee \neg y_2)$$

x_1	x_2	y_1	y_2
0	0	1	0/1
0	1	0/1	0/1
1	0	0/1	0/1
1	1	0/1	0

Unlike classical machine leaning
for the same valuation of x_1, x_2 : different y_1

Data Generation

- $F(x_1, x_2, y_1, y_2)$: $(x_1 \vee x_2 \vee y_1) \wedge (\neg x_1 \vee \neg x_2 \vee \neg y_2)$

x_1	x_2	y_1	y_2
0	0	1	0/1
0	1	0/1	0/1
1	0	0/1	0/1
1	1	0/1	0

Sample 4 data points uniformly at random

x_1	x_2	y_1	y_2
0	0	1	0
0	1	0	1
1	0	1	1
1	1	0	0

Data Generation

- $F(x_1, x_2, y_1, y_2): \ (x_1 \vee x_2 \vee y_1) \wedge (\neg x_1 \vee \neg x_2 \vee \neg y_2)$

x_1	x_2	y_1	y_2
0	0	1	0/1
0	1	0/1	0/1
1	0	0/1	0/1
1	1	0/1	0

Sample 4 data points uniformly at random

x_1	x_2	y_1	y_2
0	0	1	0
0	1	0	1
1	0	1	1
1	1	0	0

- Possible Skolem function

$$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2) \quad \psi_1(x_1, x_2) = \neg x_1 \quad \psi_1(x_1, x_2) = \neg x_2 \quad \psi_1(x_1, x_2) = 1$$
$$\psi_2(x_1, x_2) = \neg(x_1 \vee x_2) \quad \psi_2(x_1, x_2) = \neg x_1 \quad \psi_2(x_1, x_2) = \neg x_2 \quad \psi_2(x_1, x_2) = 0$$

Data Generation

- $F(x_1, x_2, y_1, y_2)$: $(x_1 \vee x_2 \vee y_1) \wedge (\neg x_1 \vee \neg x_2 \vee \neg y_2)$

x_1	x_2	y_1	y_2
0	0	1	0/1
0	1	0/1	0/1
1	0	0/1	0/1
1	1	0/1	0

Sample 4 data points

x_1	x_2	y_1	y_2
0	0	1	0
0	1	1	0
1	0	1	0
1	1	1	0

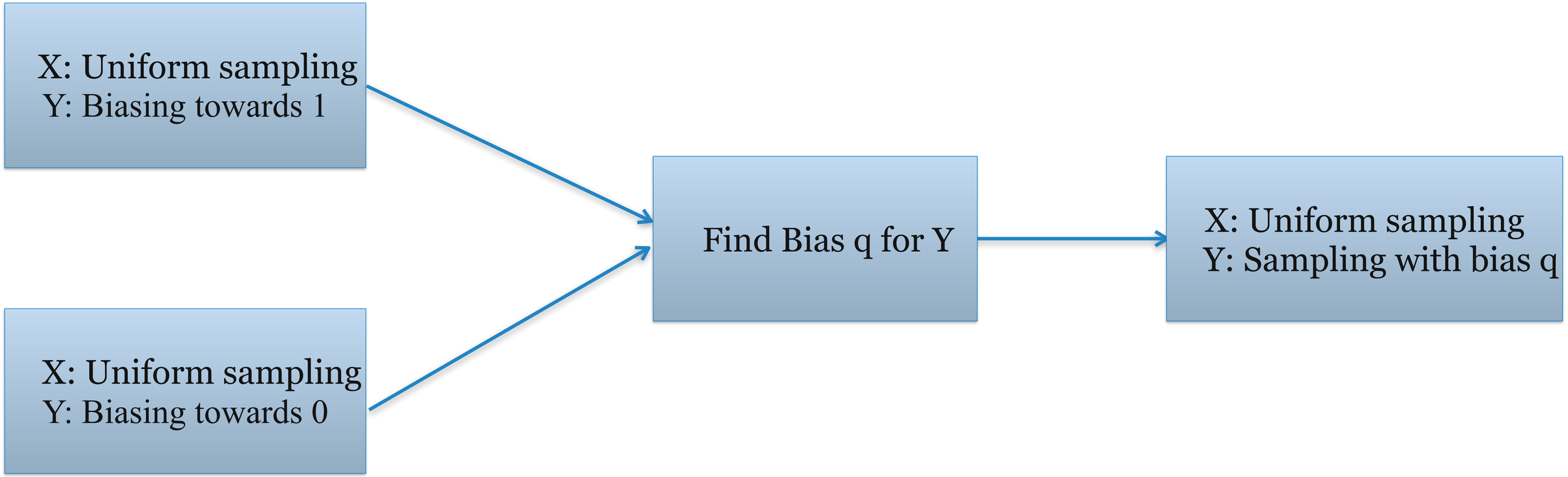
- Possible Skolem function

$\psi_1(x_1, x_2) = \neg(x_1 \vee x_2)$ $\quad \psi_1(x_1, x_2) = \neg x_1$ $\quad \psi_1(x_1, x_2) = \neg x_2$ $\quad \psi_1(x_1, x_2) = 1$

$\psi_2(x_1, x_2) = \neg(x_1 \vee x_2)$ $\quad \psi_2(x_1, x_2) = \neg x_1$ $\quad \psi_2(x_1, x_2) = \neg x_2$ $\quad \psi_2(x_1, x_2) = 0$

Data Generation

- We design a weighted sampling strategy that seeks to uniformly sample X, while biasing the valuation of Y.



Decision Tree Classifier

Decision Tree Classifier

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$

Decision Tree Classifier

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$

x_1	x_2	y_1
0	0	1
0	1	1
1	0	0
1	1	1

Decision Tree Classifier

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$

x_1	x_2	y_1
0	0	1
0	1	1
1	0	0
1	1	1

Binary classification problem.

Decision Tree Classifier

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$
  - Feature set: valuation of x_1, x_2
  - Label: valuation of y_1
  - Learn decision tree to represent y_1 in terms of x_1, x_2.

x_1	x_2	y_1
0	0	1
0	1	1
1	0	0
1	1	1

Binary classification problem.

Decision Tree Classifier

- $F(x_1, x_2, y_1)$: $x_1 \vee x_2 \vee y_1$
  - Feature set: valuation of x_1, x_2
  - Label: valuation of y_1
  - Learn decision tree to represent y_1 in terms of x_1, x_2.

x_1	x_2	y_1
0	0	1
0	1	1
1	0	0
1	1	1

Binary classification problem.

- To learn ψ_i :
  - Feature set: valuation of X in data
  - Label: valuation of y_i in data
  - Learn decision tree classifier

Verification of Candidate Functions

Verification of Candidate Functions

- When Candidate functions are not Skolem functions:

$$\exists Y F(X, Y) \not\equiv F(X, \Psi(X))$$

There exists at least one valuation of X where $\exists Y F(X, Y)$ evaluates to True, and $F(X, \Psi(X))$ also evaluates to False.

Verification of Candidate Functions

- When Candidate functions are not Skolem functions:

$$\exists Y F(X, Y) \not\equiv F(X, \Psi(X))$$

  There exists at least one valuation of X where $\exists Y F(X, Y)$ evaluates to True, and $F(X, \Psi(X))$ also evaluates to False.

- When Candidate functions are Skolem functions:

$$\exists Y F(X, Y) \equiv F(X, \Psi(X))$$

  For all the valuation of X where $\exists Y F(X, Y)$ evaluates to True, $F(X, \Psi(X))$ also evaluates to True.

Verification of Candidate Functions

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

Verification of Candidate Functions

Y and Y' are different, but same X

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

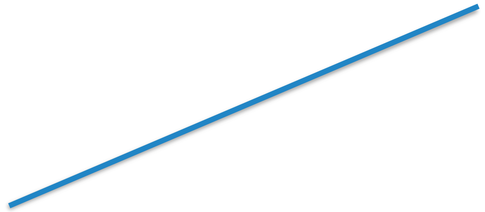
$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$



Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

Verification of Candidate Functions

Y and Y' are different, but same X

Every y'_i is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

Verification of Candidate Functions

Y and Y' are different, but same X

Every y'_i is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

$E(X, Y, Y')$ is SAT

Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

$E(X, Y, Y')$ is SAT

Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

If for all the valuation of X where $F(X, Y)$ evaluates to True, $F(X, Y')$ also evaluates to True

$E(X, Y, Y')$ is SAT

Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

If for all the valuation of X where $F(X, Y)$ evaluates to True, $F(X, Y')$ also evaluates to True

$E(X, Y, Y')$ is SAT

Verification of Candidate Functions

Y and Y' are different, but same X

Every y_i' is same as ψ_i

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

If there exists a valuation of X, s.t. $F(X, Y)$ evaluates to True, and $F(X, Y')$ evaluates to False

$E(X, Y, Y')$ is SAT

If for all the valuation of X where $F(X, Y)$ evaluates to True, $F(X, Y')$ also evaluates to True

$E(X, Y, Y')$ is UNSAT

Verification of Candidate Functions

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

- Check satisfiability of $E(X, Y, Y')$.
- If $E(X, Y, Y')$ is UNSAT: return the Skolem function Ψ.
- If $E(X, Y, Y')$ is SAT: let $\sigma \vDash E(X, Y, Y')$ be a counter example to fix.

Repairing Candidate Functions

$$E(X, Y, Y') = F(X, Y) \wedge \neg F(X, Y') \wedge (Y' \leftrightarrow \Psi)$$

- $E(X, Y, Y')$ is SAT: $\sigma \vDash E(X, Y, Y')$

$F(X, Y) \wedge (X \leftrightarrow \sigma[X]) \wedge (Y \leftrightarrow \sigma[Y])$ → True

$F(X, Y') \wedge (X \leftrightarrow \sigma[X]) \wedge (Y' \leftrightarrow \sigma[Y'])$ → False

- The potential candidates to repair : functions corresponding to y_i, if $\sigma[y_i] \neq \sigma[y_i']$

Repairing Candidate Functions

- The aim of a repair iteration is to make $F(X, Y') \wedge (Y' \leftrightarrow \Psi)$ evaluates to True with $(X \leftrightarrow \sigma[X])$.

- Let candidate function $\psi_i(X)$ corresponding to y_i needs to repair.

  With $X \leftrightarrow \sigma[X]$

Before repair	After repair
$\psi_i(X) \mapsto 0$	$\psi_i(X) \mapsto 1$
$\psi_i(X) \mapsto 1$	$\psi_i(X) \mapsto 0$

Repairing Candidate Functions

- Find UNSAT core of $G_i(X, Y) = F(X, Y) \wedge (X \leftrightarrow \sigma[X]) \wedge (y_i \leftrightarrow \sigma[y_i'])$
- Use the UNSAT core to construct β repair formula

  With $X \leftrightarrow \sigma[X]$

Before repair	Repair	After repair
$\psi_i(X) \mapsto 0$	$\psi_i(X) \leftarrow \psi_i(X) \vee \beta$	$\psi_i(X) \mapsto 1$
$\psi_i(X) \mapsto 1$	$\psi_i(X) \leftarrow \psi_i(X) \wedge \neg\beta$	$\psi_i(X) \mapsto 0$

Manthan

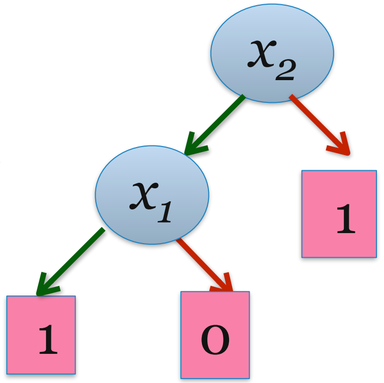
$F(X, Y)$

$X = \{x_1, x_2\}, Y = \{y_1\}$

Data Generation

x_1	x_2	y_1
0	0	1
0	1	1
1	0	0
1	1	1

Learn Candidates Ψ



$\psi_1(x_1, x_2) = x_2 \vee (\neg x_1 \wedge \neg x_2)$

Verify Candidates

Check Satisfiability of $E(X, Y, Y')$

SAT

Find UNSAT Core of $G(X, Y)$

Construct β

Repair Candidates

Repair Cycle

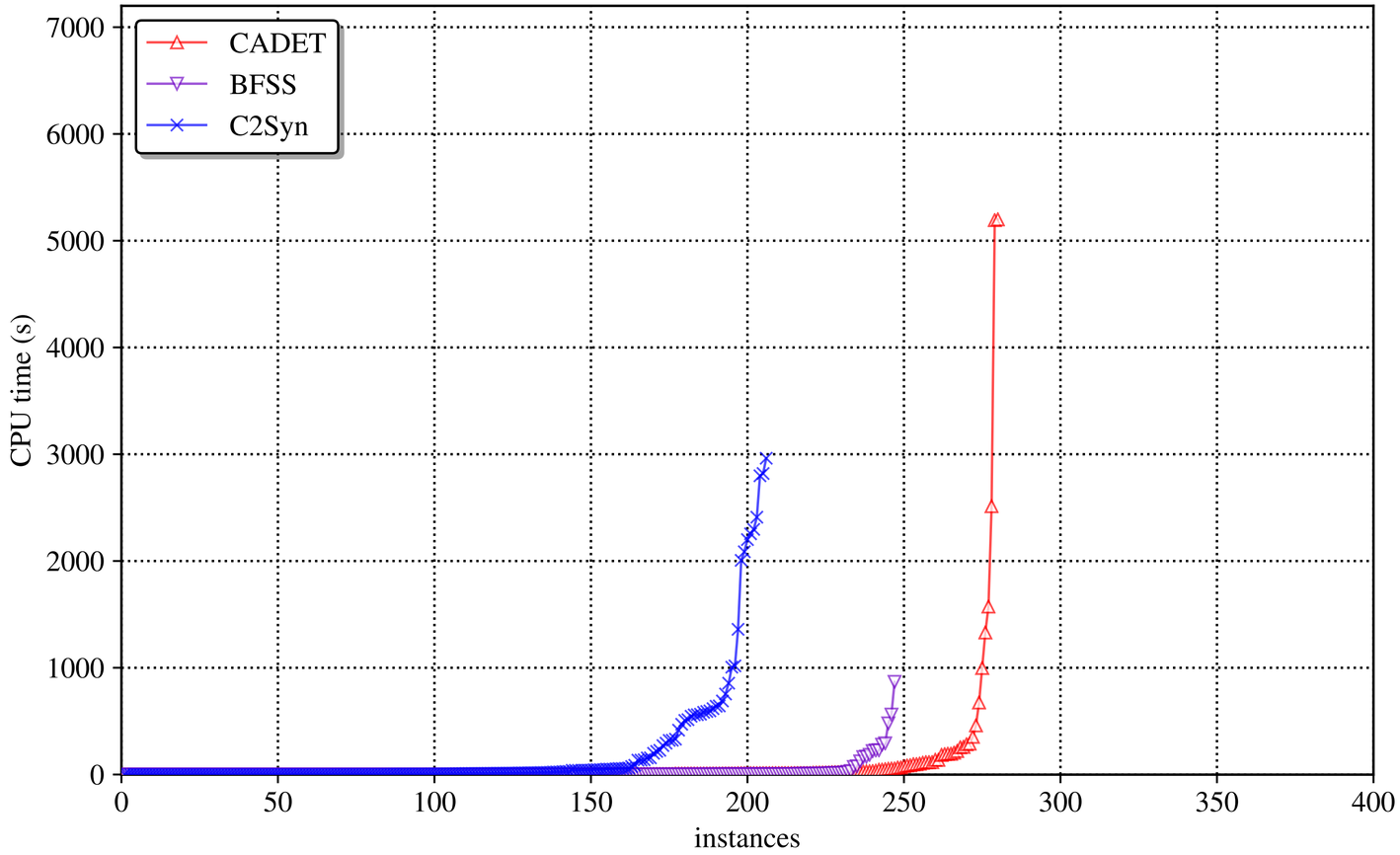
UNSAT

Return Ψ

Experimental Evaluations

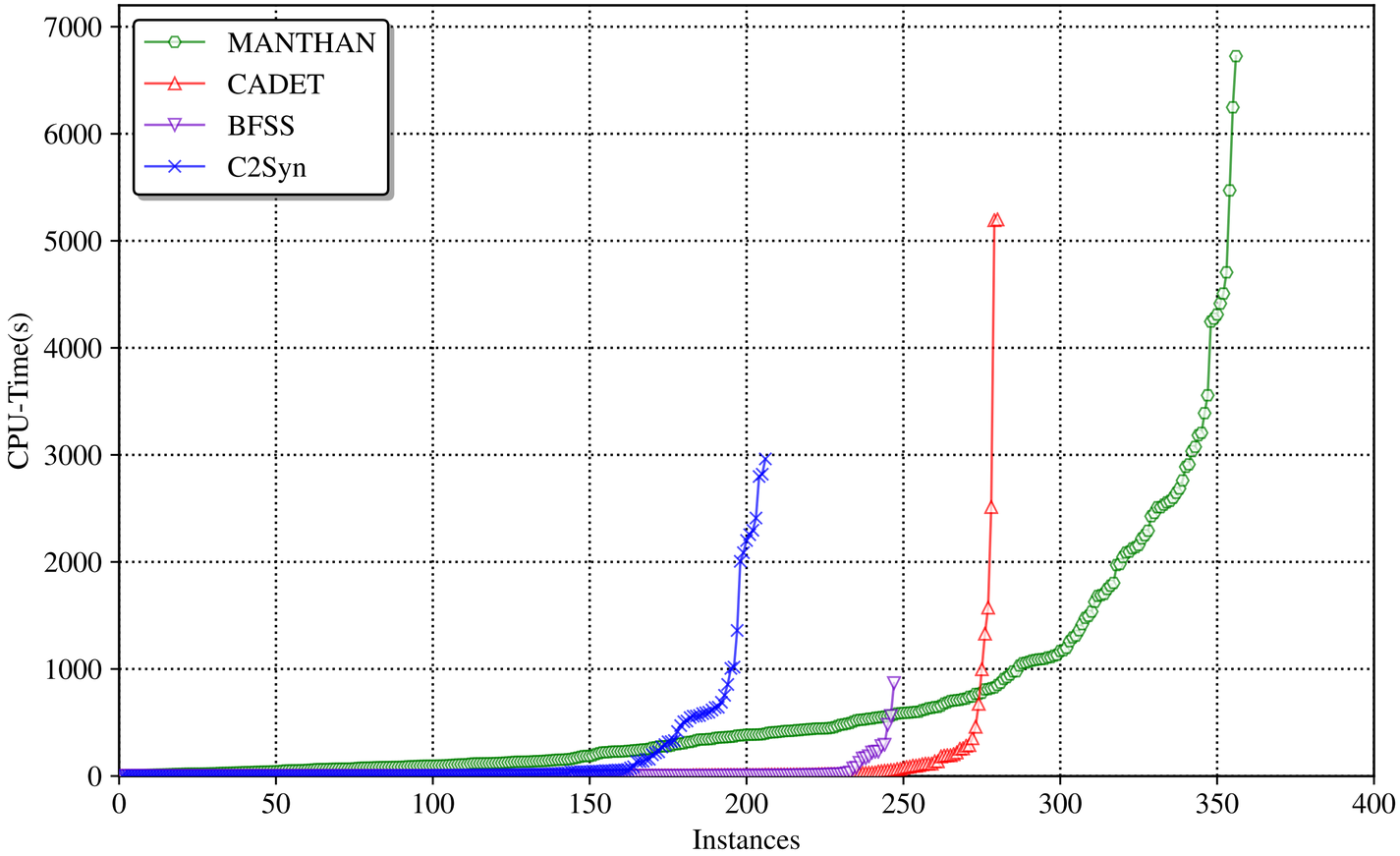
- 609 Benchmarks from:
    - QBFEval competition 2-QBF track
    - Arithmetic set (Fried, Tabajara, Vardi, 2016)
    - Disjunctive decomposition set (Akshay et al., 2017)
    - Factorization set (Akshay et al., 2017)
- Compared Manthan with State-of-the-art tools: CADET (Rabe et. al, 2019), BFSS (Akshay et al. ,2018), C2Syn (Chakraborty et al., 2019).
- Timeout: 7200s

Experimental Evaluations



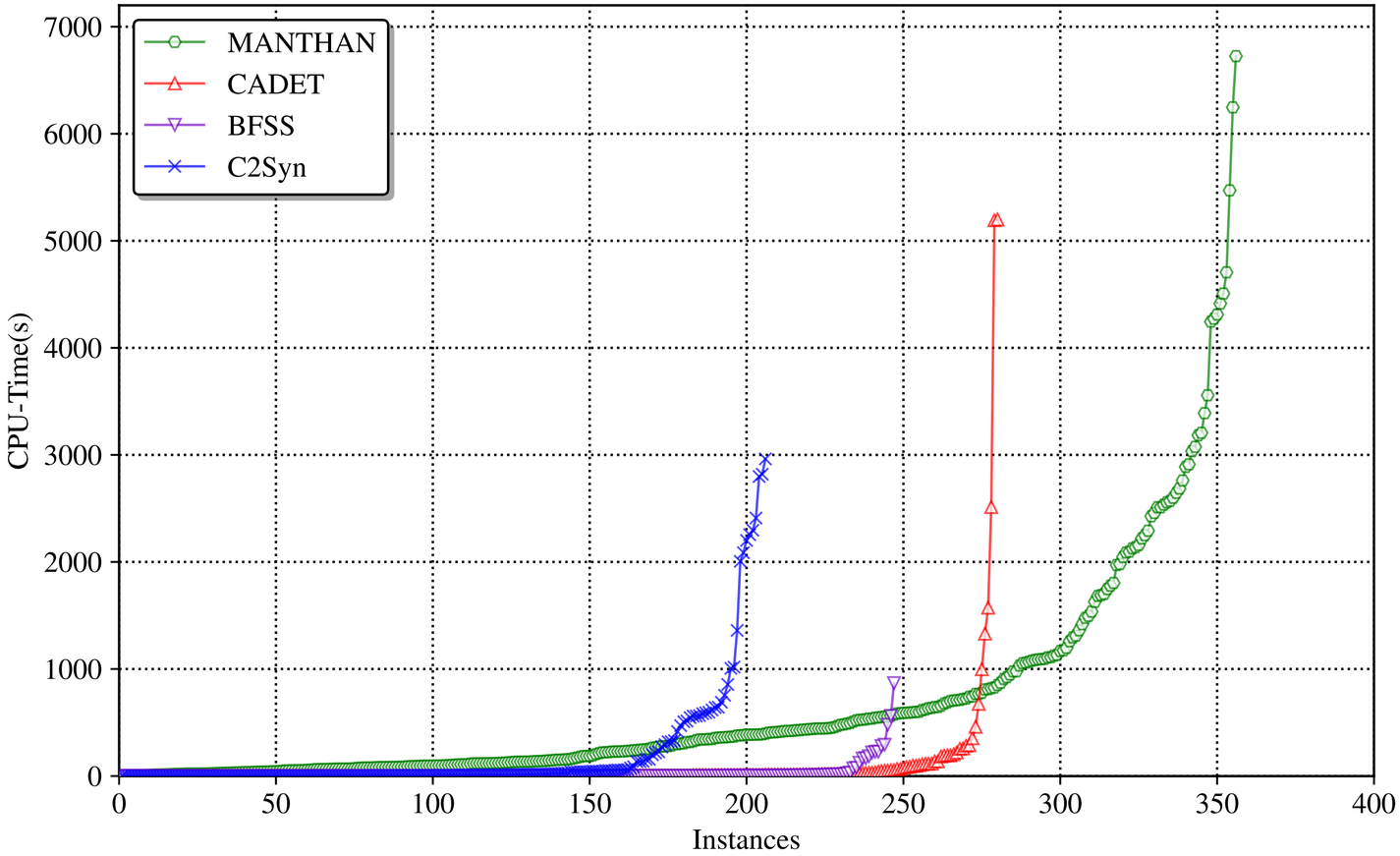
C2Syn	BFSS	CADET
206	247	280

Experimental Evaluations



C2Syn	BFSS	CADET	Manthan
206	247	280	**356**

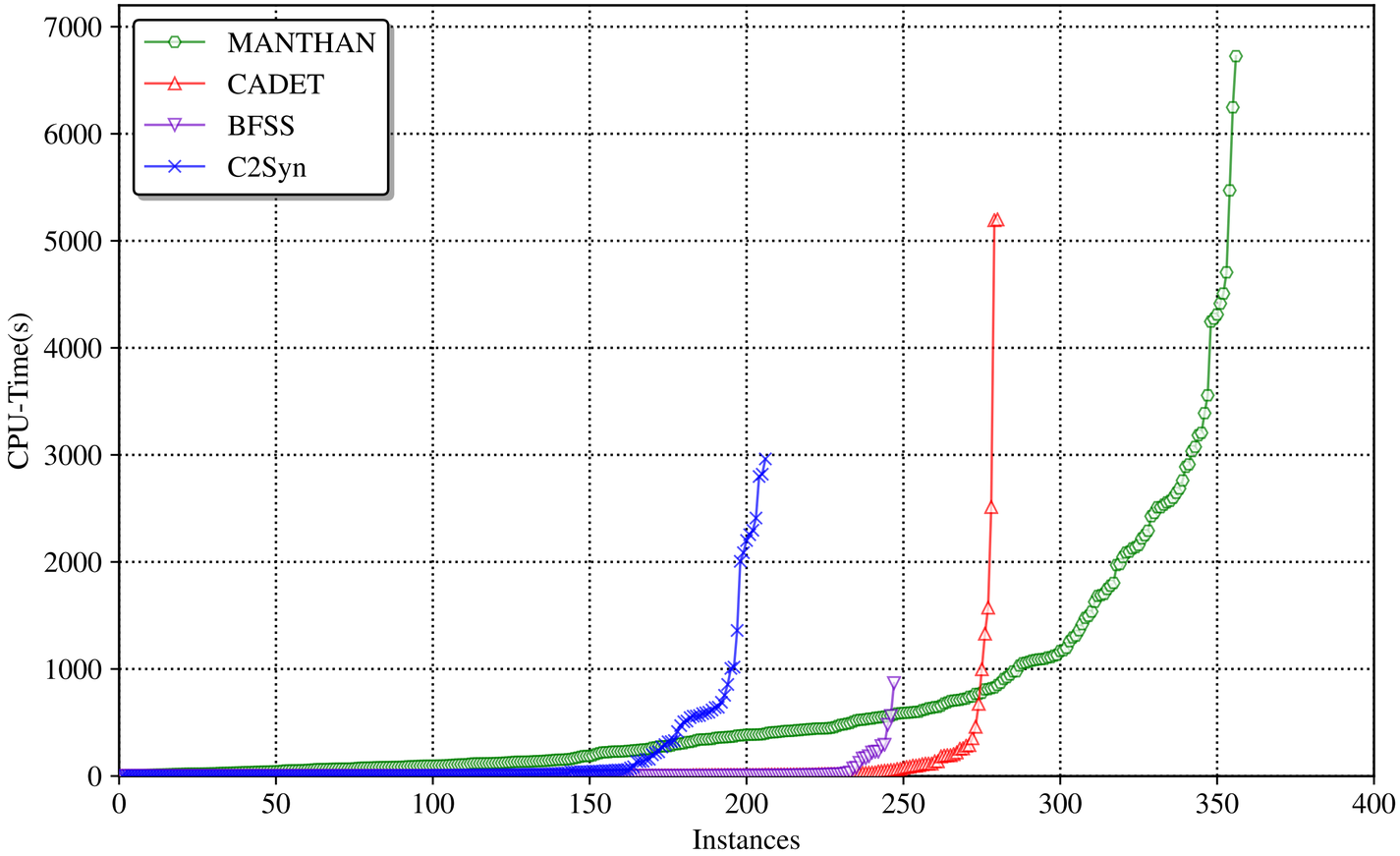
Experimental Evaluations



An increase of 76 benchmarks

C2Syn	BFSS	CADET	Manthan
206	247	280	**356**

Experimental Evaluations



An increase of 76 benchmarks

Manthan \ All tools: 60

C2Syn	BFSS	CADET	Manthan
206	247	280	**356**

Future work: interesting questions

- What is the ideal distribution to generate the data?
- How good are the candidate functions generated by data?
- Can we use similar approach for program synthesis, program repair ?

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Conclusion

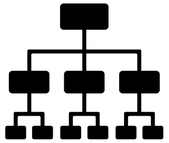
Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

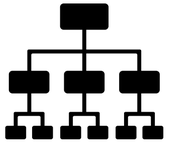
Constrained Sampling



Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling

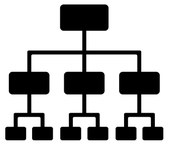


Decision Tree Classifier

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling

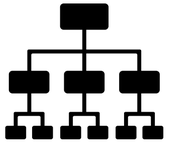


Decision Tree Classifier

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



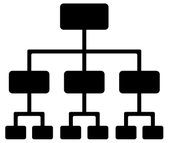
Decision Tree Classifier

Automated Reasoning

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



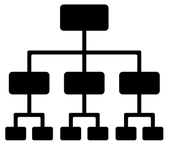
Decision Tree Classifier

Automated Reasoning

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



Decision Tree Classifier

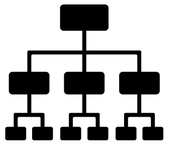
Automated Reasoning

Solves 356 benchmarks — state of the art could solve 280.

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



Decision Tree Classifier

Automated Reasoning

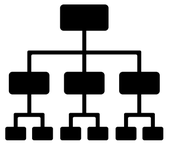
Solves 356 benchmarks — state of the art could solve 280.

Opens up several interesting directions

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



Decision Tree Classifier

Automated Reasoning

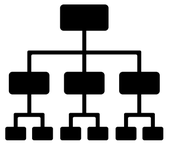
Solves 356 benchmarks — state of the art could solve 280.

Opens up several interesting directions

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



Decision Tree Classifier

Automated Reasoning

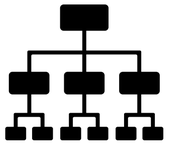
Solves 356 benchmarks — state of the art could solve 280.

Opens up several interesting directions

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



Decision Tree Classifier

Automated Reasoning

Solves 356 benchmarks — state of the art could solve 280.

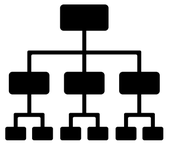
Opens up several interesting directions

https://github.com/meelgroup/manthan

Conclusion

Manthan: A Data-Driven Approach for Boolean Functional Synthesis

Constrained Sampling



Decision Tree Classifier

Automated Reasoning

Solves 356 benchmarks — state of the art could solve 280.

Opens up several interesting directions

https://github.com/meelgroup/manthan

Thanks !