Engineering an Efficient Boolean Functional Synthesis Engine

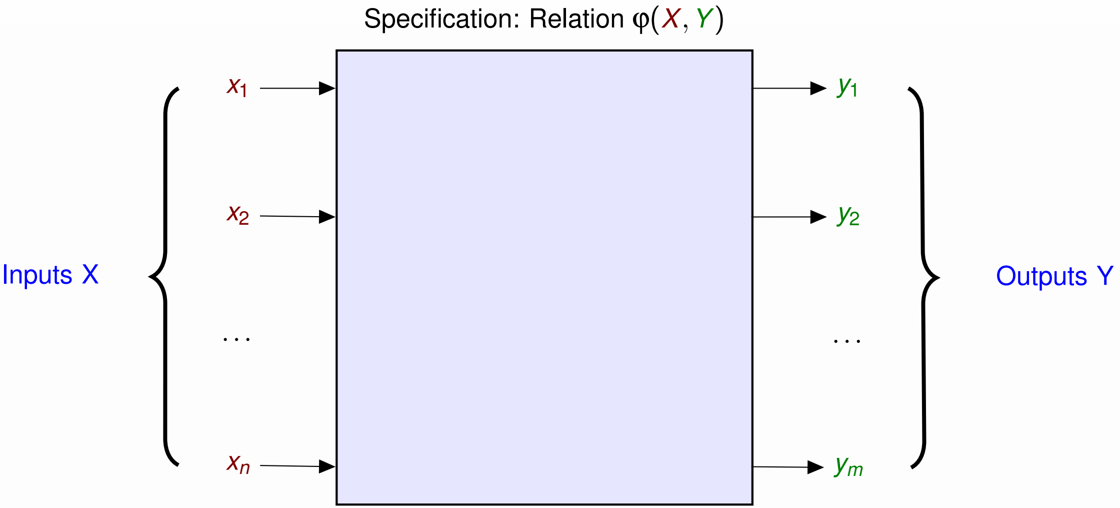
Priyanka Golia [1,2]

Joint work with: Friedrich Slivovsky[3], Subhajit Roy [1], and Kuldeep S. Meel [2]

[1]Indian Institute of Technology Kanpur
[2]National University of Singapore
[3]TU Wien

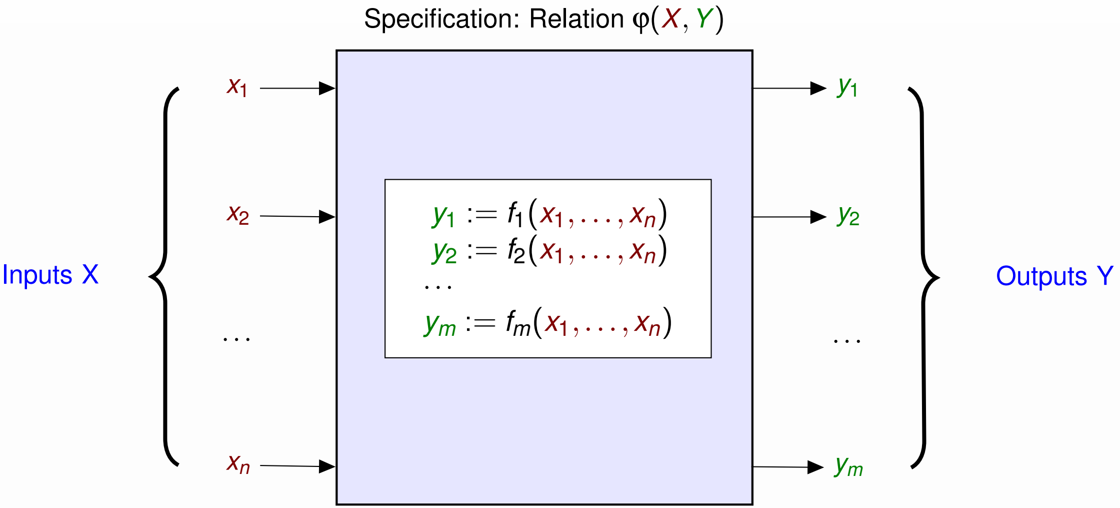
Corresponding Paper at ICCAD 2021

Synthesis



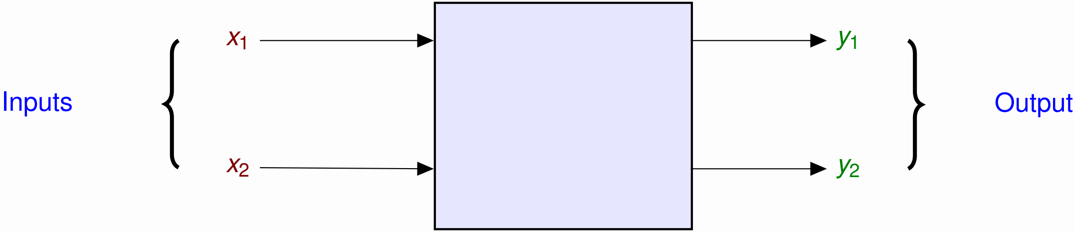
Specification: Relation φ(X, Y)
Inputs X
x_1
x_2
...
x_n
y_1
y_2
...
y_m
Outputs Y

Synthesis


Specification: Relation $\varphi(X, Y)$
Inputs X
x_1
x_2
$\dots$
x_n
$y_1 := f_1(x_1, \dots, x_n)$
$y_2 := f_2(x_1, \dots, x_n)$
$\dots$
$y_m := f_m(x_1, \dots, x_n)$
y_1
y_2
$\dots$
y_m
Outputs Y

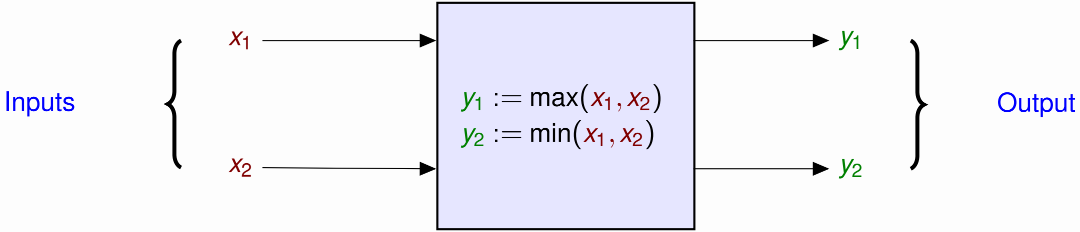
Synthesis – Example(I)

$$\varphi(X, Y) = (y_1 \geq x_1) \wedge (y_1 \geq x_2) \wedge ((y_1 = x_1) \vee (y_1 = x_2))$$
$$\wedge (y_2 \leq x_1) \wedge (y_2 \leq x_2) \wedge ((y_2 = x_1) \vee (y_2 = x_2))$$



Synthesis – Example(I)

$$\varphi(X, Y) = (y_1 \geq x_1) \wedge (y_1 \geq x_2) \wedge ((y_1 = x_1) \vee (y_1 = x_2))$$
$$\wedge (y_2 \leq x_1) \wedge (y_2 \leq x_2) \wedge ((y_2 = x_1) \vee (y_2 = x_2))$$



Functional Synthesis

Given $\varphi(X, Y)$ over inputs $X = \{x_1, x_2, \ldots, x_n\}$ and outputs $Y = \{y_1, y_2, \ldots, y_m\}$.

Synthesize A function vector $F = \{f_1, f_2, \ldots, f_m\}$, such that $y_i := f_i(x_1, \ldots, x_n)$ such that:

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

Each f_i is called Skolem function and F is called Skolem function vector.

Skolem Functions

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = x_1 \vee x_2 \vee y_1$

Skolem Functions

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = x_1 \vee x_2 \vee y_1$

Possible Skolem function: $f(x_1, x_2) := \neg(x_1 \vee x_2)$

Skolem Functions

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = x_1 \vee x_2 \vee y_1$

Possible Skolem function: $f(x_1, x_2) := \neg(x_1 \vee x_2)$

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

X	$\exists Y \varphi(X, Y)$		$\varphi(X, F(X))$
$x_1 = 0, x_2 = 0$	$y_1 = 1$	True	True
$x_1 = 0, x_2 = 1$	$y_1 = 1$	True	True
$x_1 = 1, x_2 = 0$	$y_1 = 1$	True	True
$x_1 = 1, x_2 = 1$	$y_1 = 1$	True	True

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

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = x_1 \vee x_2 \vee y_1$

Possible Skolem function: $f(x_1, x_2) := \neg(x_1 \vee x_2)$

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

X	$\exists Y \varphi(X, Y)$		$\varphi(X, F(X))$
$x_1 = 0, x_2 = 0$	$y_1 = 1$	True	True
$x_1 = 0, x_2 = 1$	$y_1 = 1$	True	True
$x_1 = 1, x_2 = 0$	$y_1 = 1$	True	True
$x_1 = 1, x_2 = 1$	$y_1 = 1$	True	True

$\Big\}$ $\exists Y \varphi(X, Y) \equiv \varphi(X, F(X))$

Other possible Skolem functions: $f_1(x_1, x_2) = \neg x_1$ $\quad f_1(x_1, x_2) = \neg x_2$ $\quad f_1(x_1, x_2) = 1$

Diverse Approaches

- From the proof of validity of $\forall X \exists Y \varphi(X, Y)$
  - (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)

Diverse Approaches

- From the proof of validity of $\forall X \exists Y \varphi(X, Y)$

  (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)

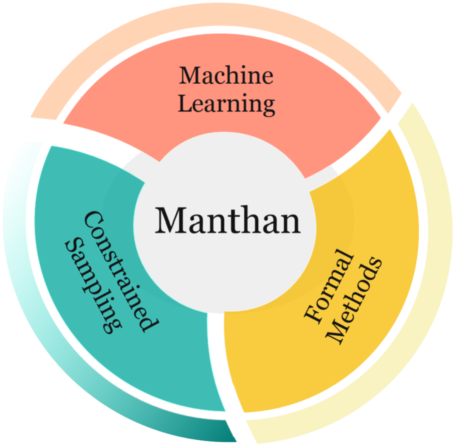
- Knowledge representation

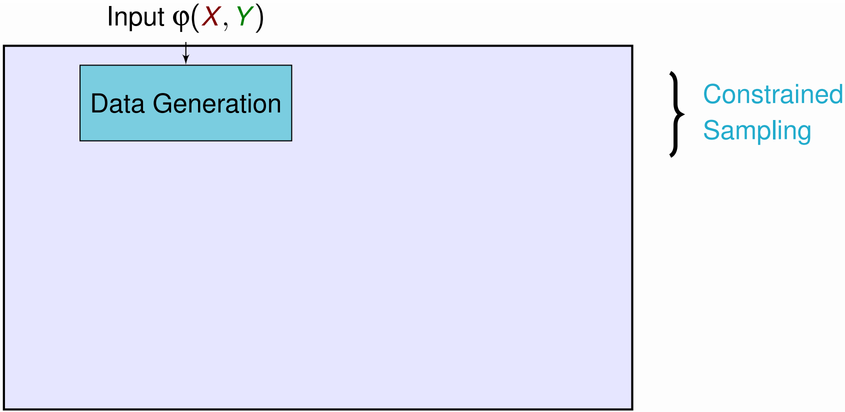
  (Kukula et al., 2000)
  (Trivedi et al., 2003)
  (Jiang, 2009)
  (Kuncak et al., 2010)
  (Balabanov and Jiang, 2011)
  (John et al., 2015)
  (Fried, Tabajara, Vardi, 2016,2017)
  (Akshay et al., 2017,2018)
  (Chakraborty et al., 2019)

- Incremental determinization

  (Rabe et al., 2015, 2018, 2019)

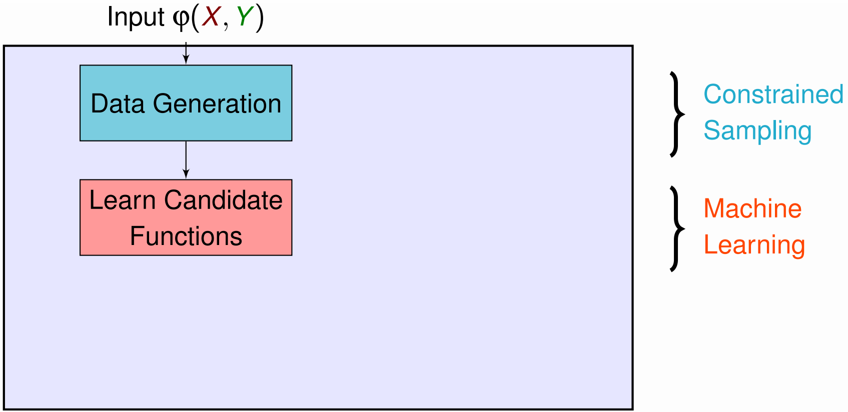
Data-Driven Approach (Golia, Roy, Meel, 2020)


Machine Learning
Manthan
Constrained Sampling
Formal Methods

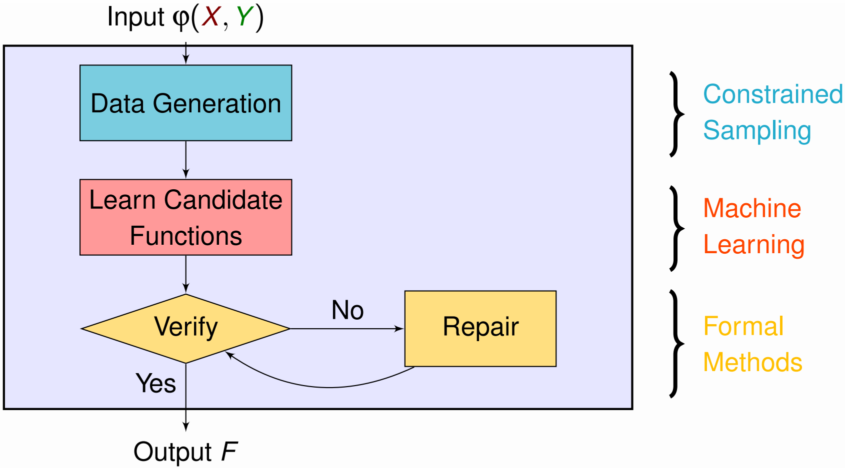


Input φ(X, Y)
Data Generation
Constrained
Sampling

Manthan



Input φ(X, Y)
Data Generation
Learn Candidate Functions
Constrained Sampling
Machine Learning

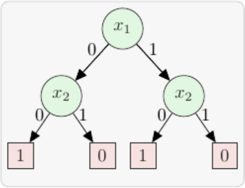

Input φ(X, Y)
Data Generation
Learn Candidate Functions
Verify
No
Repair
Yes
Output F
Constrained Sampling
Machine Learning
Formal Methods

$\varphi(x_1, x_2, y_1, y_2)$ ⟶

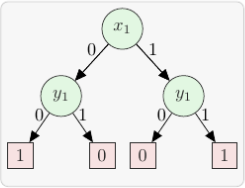
x_1	x_2	y_1	y_2
0	0	1	0
0	1	0	1
1	0	1	1
1	1	0	0

Learn Candidate Functions

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

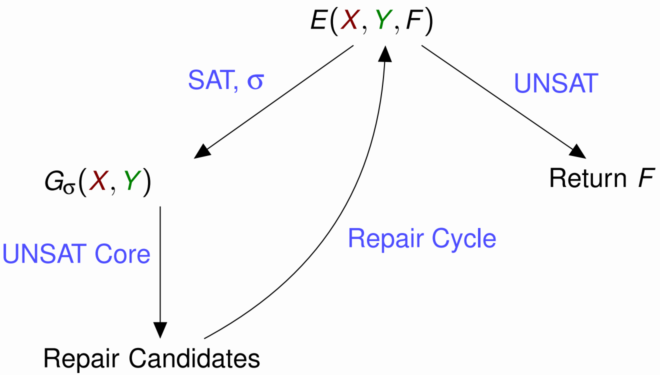


$p_1 := (\neg x_1 \wedge \neg x_2)$,
$p_2 := (x_1 \wedge \neg x_2)$
f_1 = if p_1 then 1
elif p_2 then 1
else 0



$p_1 := (\neg x_1 \wedge \neg y_1)$,
$p_2 := (x_1 \wedge y_1)$
f_2 = if p_1 then 1
elif p_2 then 1
else 0

Repair of Approximations



Manthan to Manthan2

Address scalability barriers faced by Manthan.

- Unique function extraction.
- Retention of Determined features.
- Clustering-based Multi-Classification.

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = x_1 \vee x_2 \vee y_1$

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

Unique Function Extraction

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = x_1 \vee x_2 \vee y_1$

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

y_1 is not uniquely defined.

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1))$

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

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1))$

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

y_1 is uniquely defined.

Unique Function Extraction

Let $X = \{x_1, x_2\}$, $Y = \{y_1\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1))$

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

y_1 is uniquely defined.

y_i is uniquely defined: for a fixed valuation of X, valuation of y_i is fixed.

- Extract the Skolem function f_i using interpolation-based method.

Unique function extraction reduces the number of candidate functions to learn.

Retention of Determined features

Determined features: Set of uniquely defined Y variables.

Retention of Determined features

Determined features: Set of uniquely defined Y variables.

Let $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1)) \wedge (y_1 \vee y_2)$

Retention of Determined features

Determined features: Set of uniquely defined Y variables.

Let $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1)) \wedge (y_1 \vee y_2)$

- y_1 has unique function. $f_1 = (x_1 \vee x_2)$.

Retention of Determined features

Determined features: Set of uniquely defined Y variables.

Let $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1)) \wedge (y_1 \vee y_2)$

- y_1 has unique function. $f_1 = (x_1 \vee x_2)$.
- Variable Elimination (suggested in Akshay et al., 2017,2018)
  $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow (x_1 \vee x_2)) \wedge ((x_1 \vee x_2) \vee y_2)$

$$\varphi(X, Y) = ((x_1 \vee x_2) \vee y_2)$$

- Possible Skolem function $f_2 = \neg(x_1 \vee x_2)$.

Retention of Determined features

Determined features: Set of uniquely defined Y variables.

Let $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$ and $\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1)) \wedge (y_1 \vee y_2)$

- y_1 has unique function. $f_1 = (x_1 \vee x_2)$.
- Variable Retention

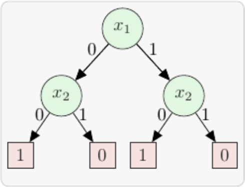
$$\varphi(X, Y) = ((x_1 \vee x_2) \leftrightarrow y_1) \wedge (y_1 \vee y_2)$$

- Possible Skolem function $f_2 = \neg(y_1)$.

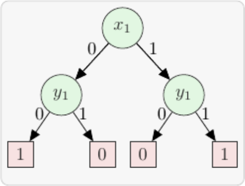
Retention of determined features helps to learn simpler candidate functions.

Clustering-based Multi-Classification

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



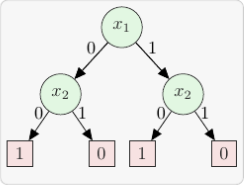
$p_1 := (\neg x_1 \wedge \neg x_2)$,
$p_2 := (x_1 \wedge \neg x_2)$
f_1 = if p_1 then 1
elif p_2 then 1
else 0



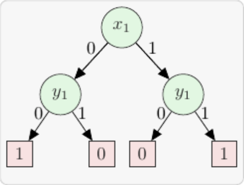
$p_1 := (\neg x_1 \wedge \neg y_1)$,
$p_2 := (x_1 \wedge y_1)$
f_2 = if p_1 then 1
elif p_2 then 1
else 0

Clustering-based Multi-Classification

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



$p_1 := (\neg x_1 \wedge \neg x_2)$,
$p_2 := (x_1 \wedge \neg x_2)$
f_1 = if p_1 then 1
elif p_2 then 1
else 0

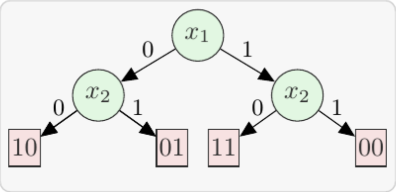


$p_1 := (\neg x_1 \wedge \neg y_1)$,
$p_2 := (x_1 \wedge y_1)$
f_2 = if p_1 then 1
elif p_2 then 1
else 0

Can we learn functions for y_1 and y_2 together to save candidate learning time?

Clustering-based Multi-Classification

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



$p_1 := (\neg x_1 \wedge \neg x_2)$,
$p_2 := (x_1 \wedge \neg x_2)$
f_1 = if p_1 then 1
elif p_2 then 1
else 0

$p_1 := (\neg x_1 \wedge x_2)$,
$p_2 := (x_1 \wedge \neg x_2)$
f_2 = if p_1 then 1
elif p_2 then 1
else 0

Clustering-based Multi-Classification (Cont.)

- Partition Y variables with disjoint subsets.
- Use a multi-classifier to learn candidates for each partition.

Clustering-based Multi-Classification (Cont.)

- Partition Y variables with disjoint subsets.
- Use a multi-classifier to learn candidates for each partition.

How should the variable partitioning be driven ?

Clustering-based Multi-Classification (Cont.)

- Partition Y variables with disjoint subsets.
- Use a multi-classifier to learn candidates for each partition.

How should the variable partitioning be driven ?

- Learn related variables together — lead to smaller decision tree.

Clustering-based Multi-Classification (Cont.)

- Partition Y variables with disjoint subsets.
- Use a multi-classifier to learn candidates for each partition.

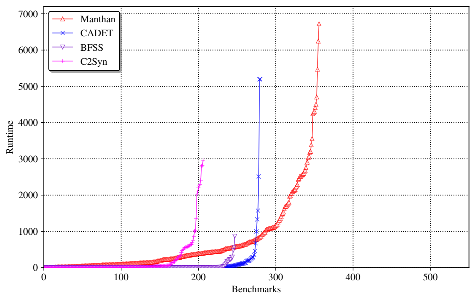
How should the variable partitioning be driven ?

- Learn related variables together — lead to smaller decision tree.
- Use edge(hop) distance in primal graph to cluster Y variables into disjoint subsets.

Experimental Evaluations

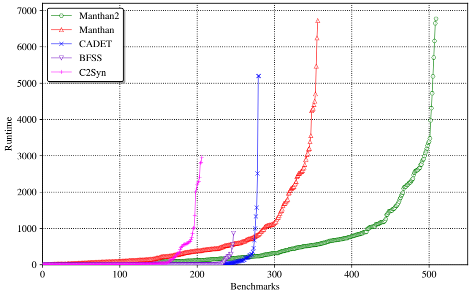
- 609 Benchmarks from:
  - QBFEval competition
  - Arithmetic
  - Disjunctive decomposition
  - Factorization
- Compared Manthan2 with State-of-the-art tools: Manthan (Golia et al., 2020), CADET (Rabe et al., 2019), BFSS (Akshay et al. ,2018), C2Syn (Chakraborty et al., 2019).
- Timeout: 7200 seconds.

Experimental Evaluations



C2Syn	BFSS	CADET	Manthan
206	247	280	356

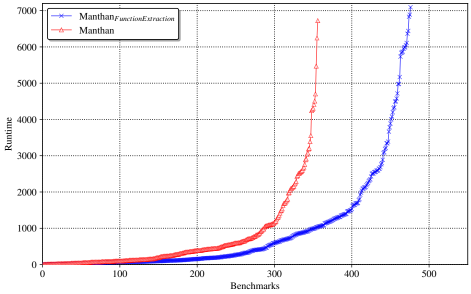
Experimental Evaluations



C2Syn	BFSS	CADET	Manthan	Manthan2
206	247	280	356	509

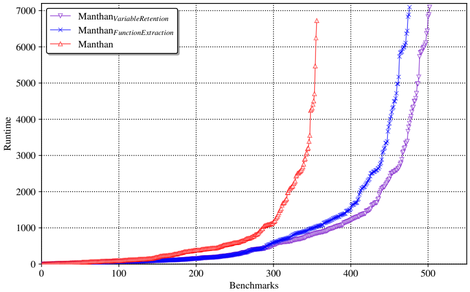
An increase of 153 benchmarks.

Impact of Individual Contribution



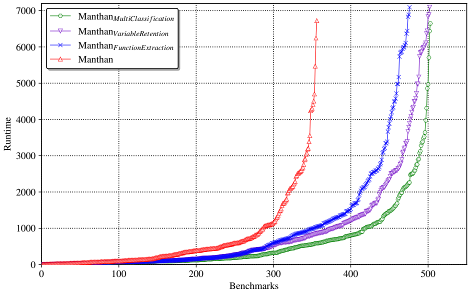
Manthan	Manthan$_{FunctionExtraction}$
356	477
6374.39	3523.28

Impact of Individual Contribution



Manthan	Manthan$_{FunctionExtraction}$	Manthan$_{VariableRetention}$
356	477	502
6374.39	3523.28	3227.11

Impact of Individual Contribution



Manthan	Manthan$_{FunctionExtraction}$	Manthan$_{VariableRetention}$	Manthan$_{MultiClassification}$
356	477	502	509
6374.39	3523.28	3227.11	2858.61

Conclusion

Engineering an Efficient Boolean Functional Synthesis Engine

Unique function extraction

+ Variable Retention.

Constrained Sampling

Multi-Class Classifier

Formal Methods

Solves 509 benchmarks — state of the art could solve 356

https://github.com/meelgroup/manthan

Thanks!