Approximate BDD Optimization

Rolf Drechsler[1,2]

drechsler@uni-bremen.de

[1] Deutsches Forschungszentrum
für Künstliche Intelligenz (DFKI)
FB Cyber-Physical Systems

[2] University of Bremen
Group of Computer Architecture

@Rolf_Drechsler

Joint Work with

- Arun Chandrasekharan
- Saman Fröhlich
- Daniel Große
- Oliver Keszöcze
- Saeideh Shirinzadeh
- Mathias Soeken

Deutsche Forschungsgemeinschaft

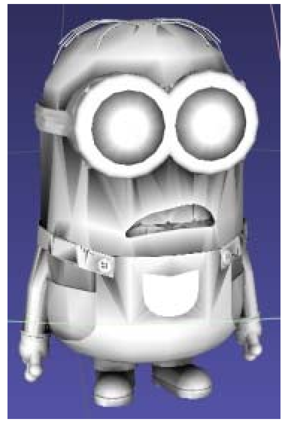
DFG

Introduction

- Approximate Computing (AC): consider applications like

robotics

search engines

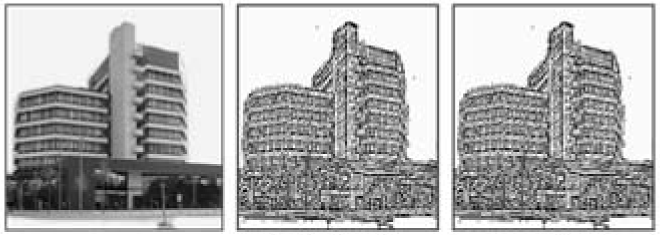


digital image processing

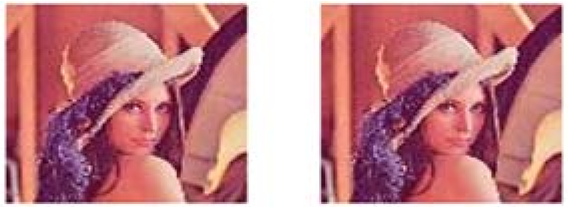
- What means "correct"?

Introduction

- Many real world applications tolerate inaccuracies
  - Media processing, recognition, data mining, etc.



Original Approximated



Original Approximated

  - Usually due to *timing induced errors* (e.g. voltage-scaling) or *functional approximation*

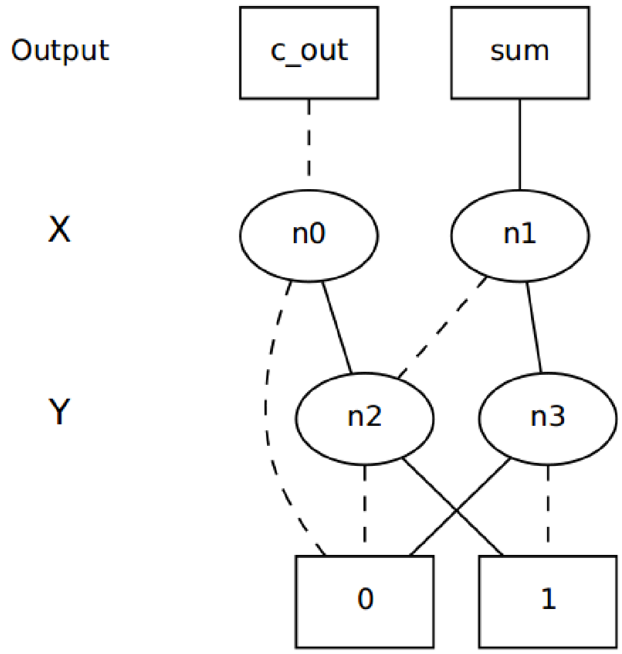
- AC **trades off** inaccuracies for performance

Why BDDs?

- Successfully applied in many fields
  - Test, verification, logic synthesis
- BDDs are well understood
  - Efficient algorithms
  - Lower/upper bounds on size of BDDs
- Allows deeper understanding of AC

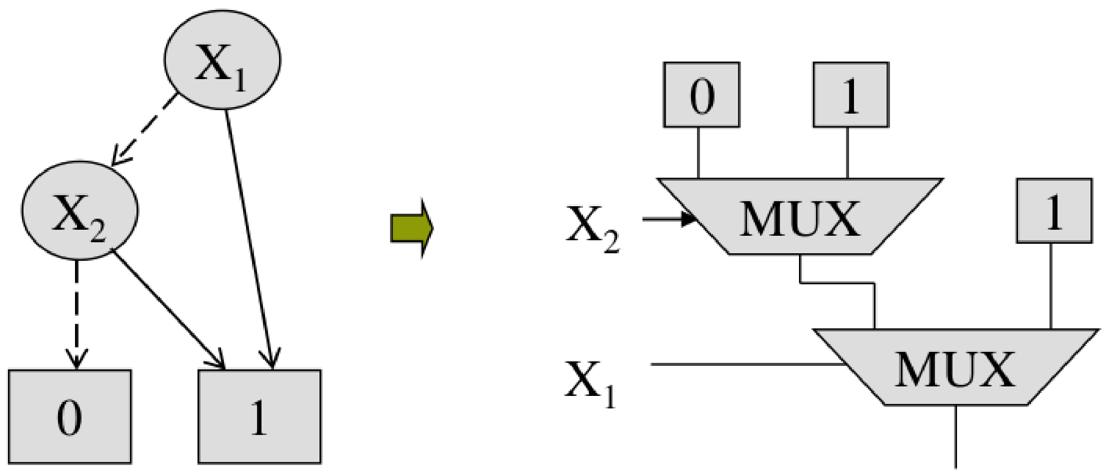
Is it a new problem?

- Optimization of BDDs
  - Variable ordering
  - Assignment of **don‘t cares** (DCs)
  - AC: dynamically changing the DC set

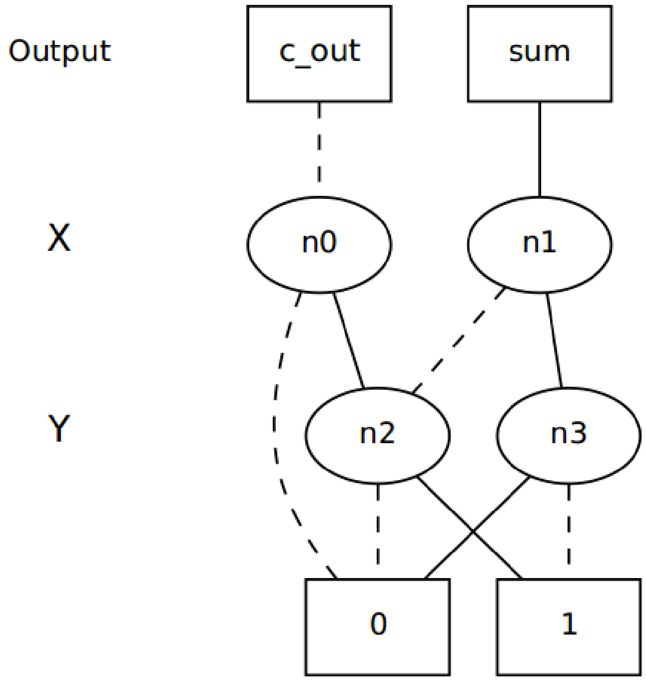


Reduction of Size

- Design styles based on direct mapping



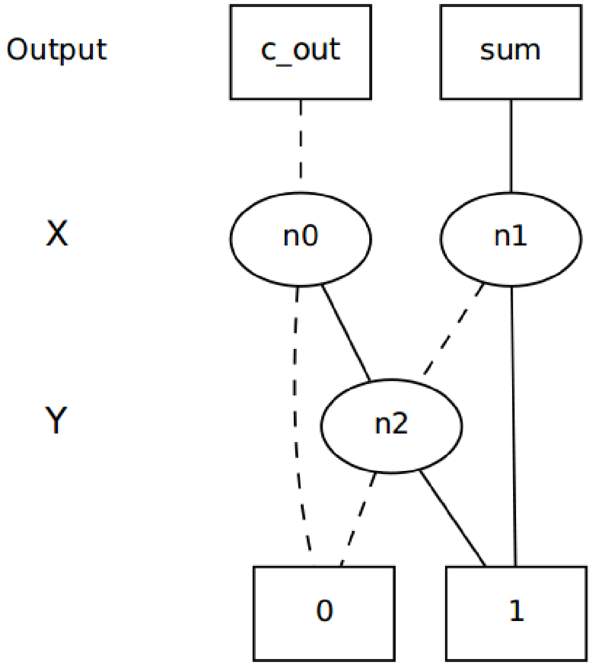
BDD for Half Adder



X	Y	Sum	C_out
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

- Remove n3

Compacted BDD for Half Adder



X	Y	Sum	C_out
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

- Smaller hardware representation
- Error in the result needs to be tolerated

Important Questions

- How good is my approximation?

- How do I generate an approximation wrt. error-metrics?

Error Metrics

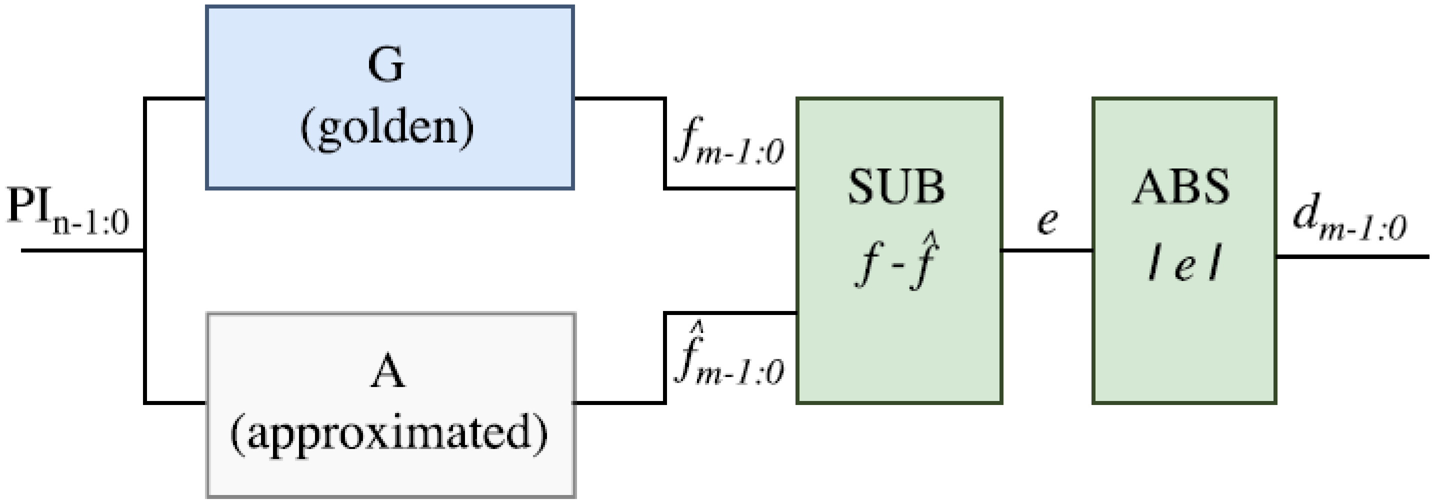
- Worst-Case:
  - $wc(f,\hat{f}) = \max_x\{|\mathrm{int}(f(x)) - \mathrm{int}(\hat{f}(x))|\}$
- Average-Case:
  - $ac(f,\hat{f}) = \frac{\sum_x |\mathrm{int}(f(x)) - \mathrm{int}(\hat{f}(x))|}{2^n}$
- Error-Rate:
  - $er(f,\hat{f}) = \frac{\sum_x |f(x) \neq \hat{f}(x)|}{2^n}$
- …

Evaluating Error Metrics

- $wc(f,\hat{f})$ is $FP^{NP} - complete$
  - Proof: $wc(f,\hat{f})$ is equivalent to LEXSAT
- $ac(f,\hat{f})$ is $\#\mathrm{P} - complete$
  - Proof: $ac(f,\hat{f})$ is equivalent to #SAT
- $er(f,\hat{f})$ is $\#\mathrm{P} - complete$
  - Proof: ec$(f,\hat{f})$ is equivalent to #SAT
- Similar complexities for other error-metrics

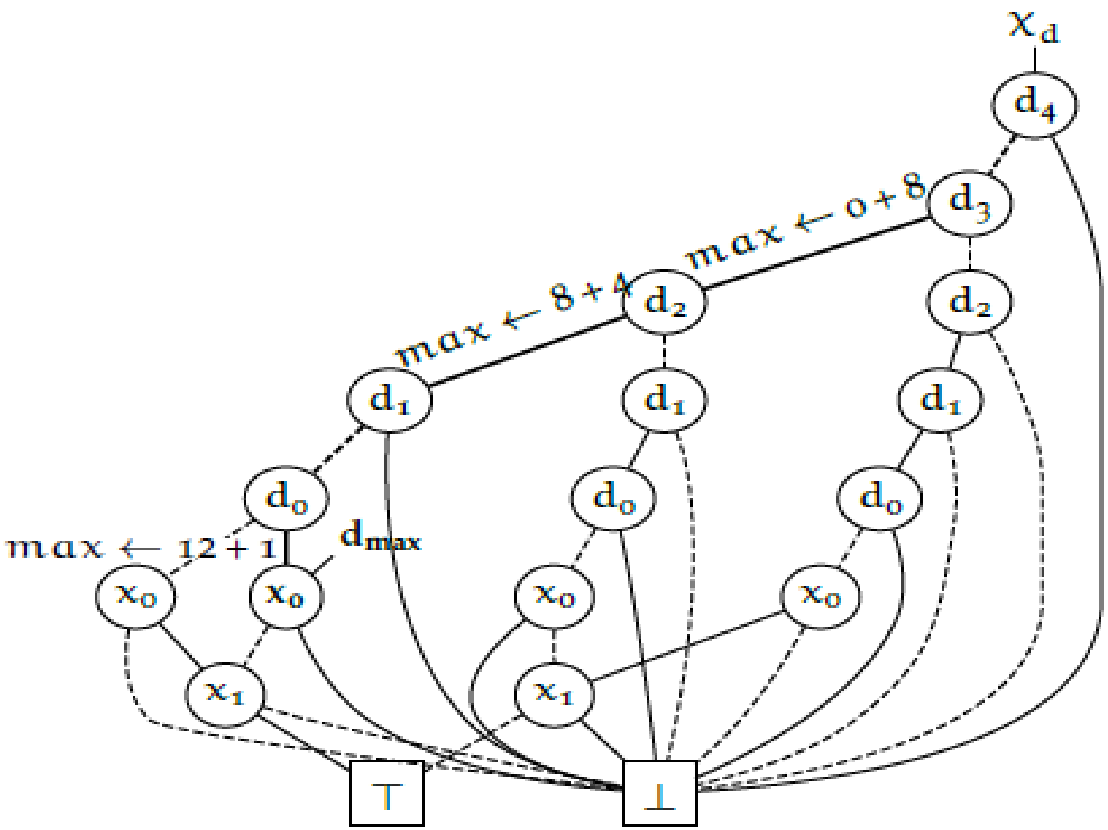
Evaluation using Miters

- Combinatorial circuits:



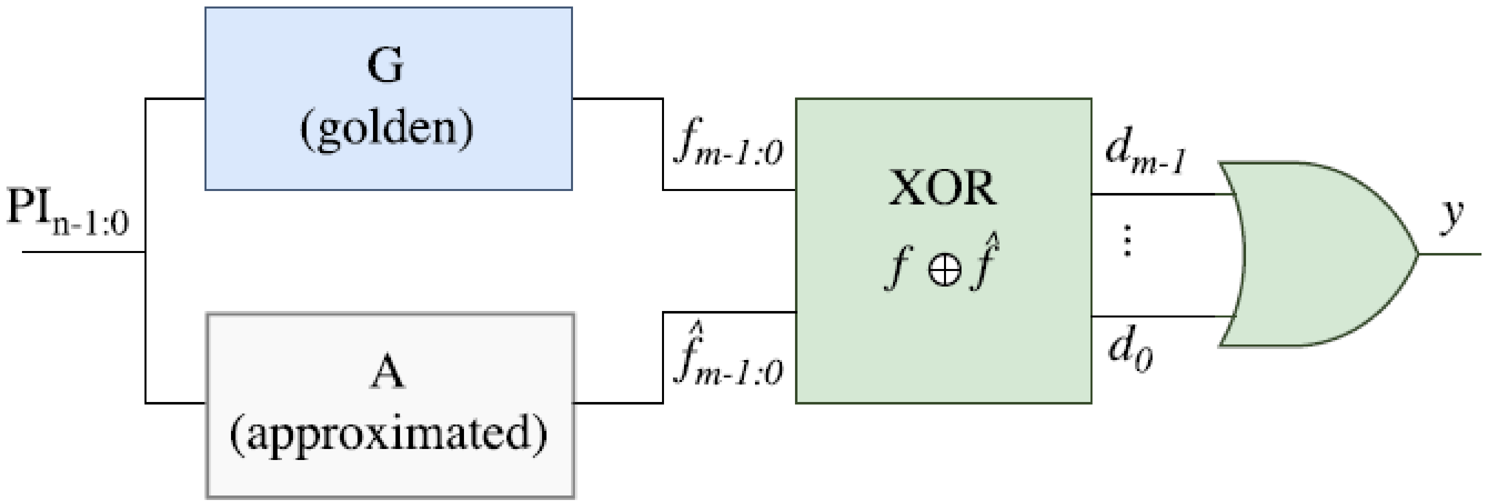
- Methodology:
  - Represent Miter as a BDD
  - Evaluate BDD in terms of the given error-metric

Worst-case Error



- similar algorithm exists for the average-case error

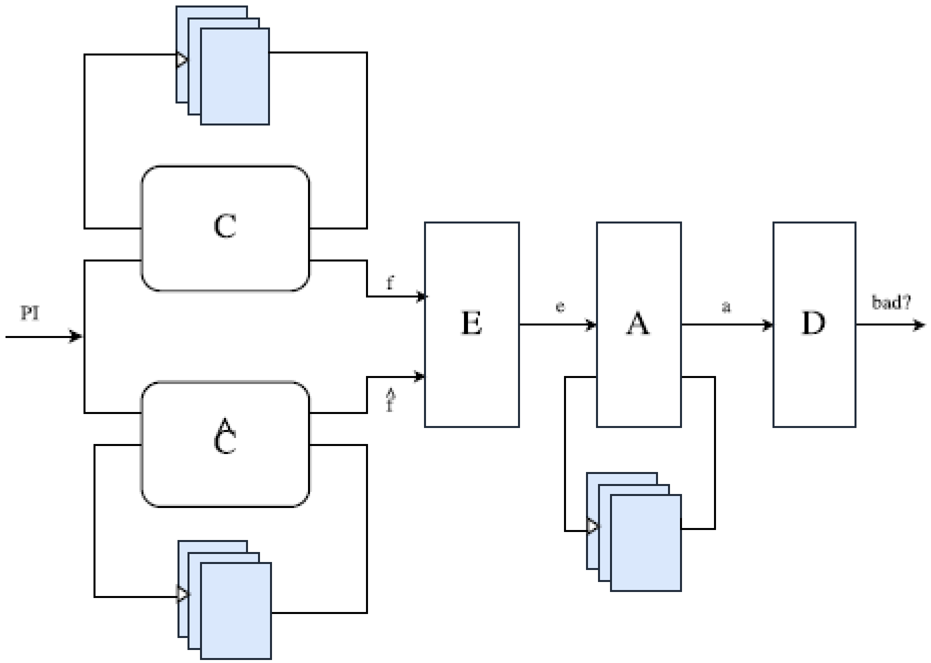
Error Rate



- Calculate ON-set of BDD-representation

Sequential Circuits

- E: error computation, A: accumulator, D: decision



  - Use Bounded Model Checking (BMC) and Property Directed Reachability (PDR) techniques to determine error-metrics

Important Questions

- How good is my approximation?

- How do I generate an approximation wrt. error-metrics?

AC Approaches for BDDs

- Directly on BDD:
  - Exact Minimization
  - Heuristic
- Evolutionary algorithms

Exact BDD Minimization

Error Bounded Exact BDD Minization

Given:

$$f : \mathbb{B}^n \to \mathbb{B}$$
$$e \in \mathbb{N}$$

Task:

$$\min B(\hat{f}), \hat{f} : \mathbb{B}^n \to \mathbb{B},\ s.t.$$
$$\sum_{x \in \mathbb{B}^n} \left[\hat{f}(x) \neq f(x)\right] \leq e$$

Idea of Exact BDD Minimization

- Flip at most e bits in truthtable of f
- Build a BDD representing all feasible functions
- Traverse BDD to find result

Experiment: Exact BDD Minimization

TABLE I

COMPUTATION TIME FOR BDD-SOLVER FOR FUNCTIONS WITH 3 VARIABLES

e	total BDD [s]	at BDD [s]	wt BDD [s]
1	0.018062	0.00007	0.00012
2	0.030604	0.00012	0.00018
3	0.021506	0.00008	0.00032

TABLE II

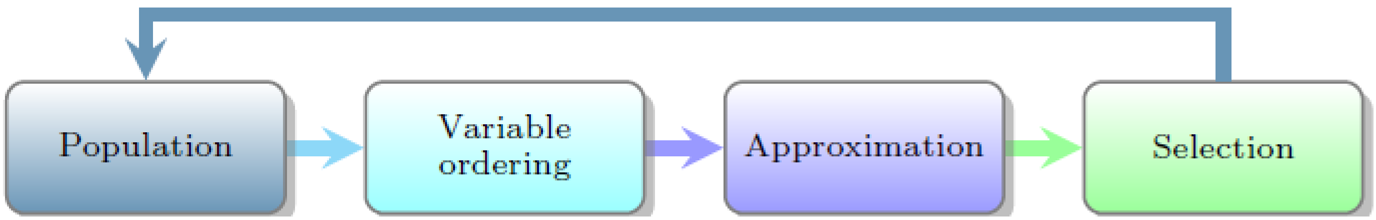
COMPUTATION TIME FOR BDD-SOLVER FOR FUNCTIONS WITH 4 VARIABLES

e	total BDD [s]	at BDD [s]	wt BDD [s]
1	21.28	0.00032	0.00102
2	100.07	0.00153	0.00260
3	444.73	0.00679	0.01300
4	1,289.70	0.01968	0.03000
5	2,798.10	0.04270	0.10200
6	3,948.30	0.06025	0.15000
7	2,392.00	0.03650	0.22000

- Does not scale well enough to be practical for large functions
  - Can be used to benchmark the quality of heuristics

Evolutionary Approach

- Based on evolutionary multi-objective optimization (MOO)
  - Variable reordering together with applying approximation operators



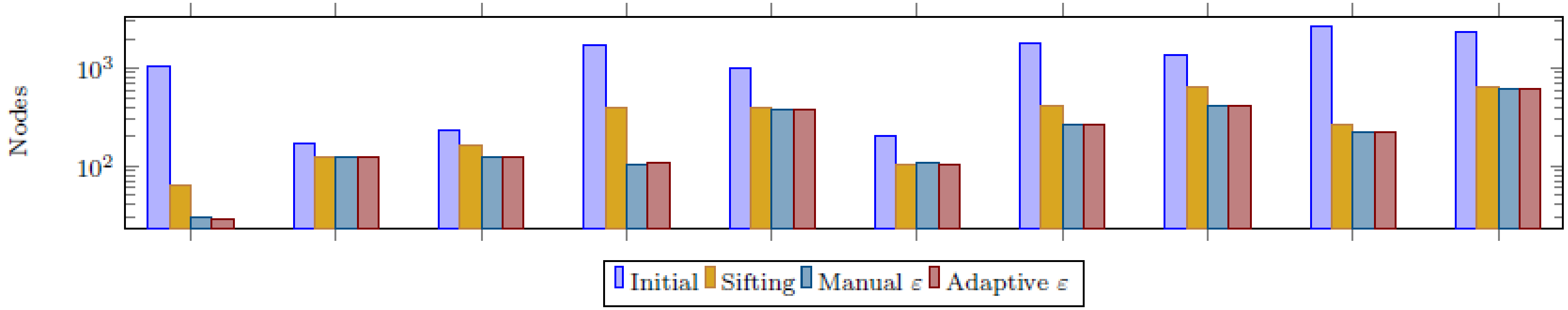
- Algorithm requires
  - Higher priority of BDD size
  - Error constraint satisfaction

Evolutionary Approach

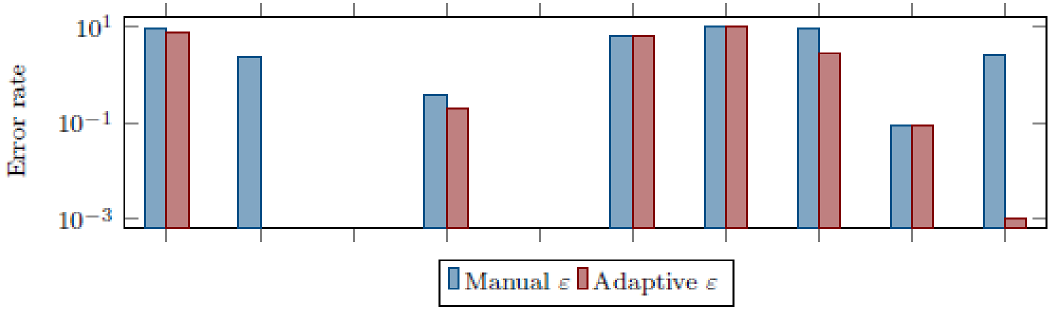
- Evolutionary Multi-objective Optimization
  - $\min_{x} f(x) = (f_1(x), f_2(x), \ldots, f_n(x))$
  - Pareto-front, Pareto-dominance
- Optimization goals are:
  - BDD size
  - Error

Evolutionary Approach: Experiments

- 68.02% size reduction at a cost of 2.12% inaccuracy on ISCAS89
- 23.51% size reduction in comparison with sifting



- 25.19% error rate reduction by adaptive approach



Universität Bremen

References

- M. Soeken, D. Große, A. Chandrasekharan and R. Drechsler: BDD Minimization for Approximate Computing - ASP-DAC 2016
- A. Chandrasekharan, M. Soeken, D. Große and R. Drechsler: Precise Error Determination of Approximated Components in Sequential Circuits with Model Checking – DAC 2016
- A. Chandrasekharan, M. Soeken, D. Große and R. Drechsler: Approximation-aware Rewriting of AIGs for Error Tolerant Applications – ICCAD 2016
- S. Froehlich, D. Große and R. Drechsler: Error Bounded Exact BDD Minimization in Approximate Computing – ISMVL 2017
- S. Shirinzadeh, M. Soeken, D. Große and R. Drechsler: An Adaptive Prioritized ε-Preferred Evolutionary Algorithm for Approximate BDD Optimization – GECCO 2017
- O. Keszöcze, M. Soeken and R. Drechsler: On the computational complexity of error metrics in approximate computing – International Workshop on Boolean Problems 2016

Conclusions

- BDDs for AC
- BDDs are intensively studied data structure
  - Theoretical analysis
  - Implementation and experiments
- Future work: more general structures

Approximate BDD Optimization

Rolf Drechsler[1,2]

drechsler@uni-bremen.de

[1] Deutsches Forschungszentrum
für Künstliche Intelligenz (DFKI)
FB Cyber-Physical Systems

[2] University of Bremen
Group of Computer Architecture

@Rolf_Drechsler