Satisfiability Solving Meets Evolutionary Optimisation in Designing Approximate Circuits

Milan Češka, Jiří Matyáš, Vojtech Mrazek, Tomáš Vojnar

Faculty of Information Technology, Brno University of Technology

imatyas@fit.vutbr.cz

Supported by the Brno PhD. Talent scholarship program

BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

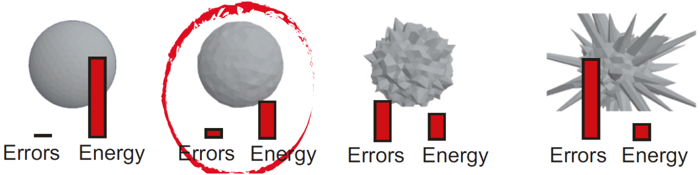
25th June 2020

Motivation for Approximate Computing

- **Emerging need for energy-efficient systems**.
- Many computer applications are inherently error resilient:
  - signal and multimedia processing,
  - data mining,
  - neural networks, ...
- **Up to 80 % of computational time is spent in computations that can be approximated** (Chippa et al., 2013).
- Approximate multipliers in HW neural networks:
  - **energy consumption reduced by 90 %,**
  - classification accuracy decreased by only 2 %,
  - Mrazek et al., ICCAD 2016.

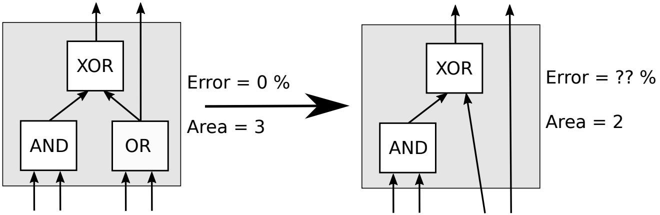
Functional Approximation

- Implementing the system with fuctionality that differs from the original one but has better non-funtional parameters.
- The approximation process iterates two basic steps:
  - generation of candidate solutions,
  - **candidate solution error evaluation.**



- We seek an ideal trade-off between the approximation error and energy savings.

- Automatic approaches are preferred.
- Simplification of the original accurate circuit.
- **Goal: reduction of circuit area and power consumption.**
- Area is estimated using a heuristic:
  - number of gates,
  - sum of gate sizes in a target technology.
- Area correlates well with power consumption.



Utilisation of satisfiability solving in circuit approximation.
Designed approaches:

1. Monolithic Approach,
2. Iterative sub-circuit Approximation,
3. Evolutionary Approximation with Satisfiability-based (StS) Optimisation.

Monolithic Approach

- **Builds a single formula encoding the synthesis problem.**
- Inputs:
  - golden circuit GC,
  - error bound T,
  - size S of currently best known approximation.
- Goal: synthesise approximate circuit AC, where

$$size(AC) < S \wedge error(AC, GC) < T$$

- Used encodings (details can be found in the paper):
  - pure SAT encoding,
  - SMT encoding,
  - SMT encoding using arrays,
  - SMT encoding using arrays and quantifiers.
- Does not scale to circuits with more than 8 gates.

Sub-circuit Approximation

Iterative approximation process:

1. Selection of a sub-circuit SC of the current approximate solution.
2. Create an optimal approximation ASC of the sub-circuit SC (using monolithic approach on SC).
3. Replace SC with ASC in the current solution.
4. Check the error of the current approximate solution.

Drawbacks:

- We need to evaluate the error of the approximate solution in each iteration.
- Each sub-circuit approximation is expensive and can be rejected.

Approximation interleaving Cartesian Genetic Programming (CGP) and StS optimisation:

1. CGP **introduces error** into the solution.
2. StS approach **optimises sub-circuits** (preserves the functionality).

Advantages:

- CGP quickly introduces modifications.
- Each successful StS optimisation is accepted.
- StS helps to introduce larger changes in circuit structure and escape local optimums.

- CGP – pure evolutionary approximation.
- SMT – iterative StS approximation.
- COMB – interleaving approach using evolutionary approximation and StS optimisation.
- Time limit 2 hours, starting from the original golden solution.
- Table shows area of final solution relative to original circuit.

Performance on small circuits:

	8-bit adders			4-bit multipliers		
Err	CGP	SMT	COMB	CGP	SMT	COMB
1 %	64.8	83.5	**54.5**	78.4	90.5	**74.6**
2 %	52.6	78.0	**44.9**	69.3	82.6	**67.1**
5 %	37.1	57.4	**32.3**	53.4	77.0	**49.7**

- CGP – pure evolutionary approximation.
- COMB – interleaving approach using evolutionary approximation and StS optimisation.
- Approximation process was seeded with best solutions found by pure CGP.
- Time limit: 10h for adders, 75h for multipliers.
- Table shows area of final solution relative to the seed circuit.

Performance on large circuits:

32-bit adders			16-bit multipliers		
Err(%)	CGP	COMB	Err(%)	CGP	COMB
10^{-5}	100.0	**81.5**	10^{-3}	97.9	**91.4**
10^{-4}	100.0	**81.3**	0.01	97.6	**91.1**
10^{-3}	100.0	**81.1**	0.1	95.0	**90.1**

- 16-bit multiplier approximation.
- Progress of solution area in time.
- Error thresholds: red=10^{-1}%, green=10^{-2}%, blue=10^{-3}%.