Approximate BDD Optimization with Prioritized ε-Preferred Evolutionary Algorithm

Saeideh Shirinzadeh[1] Mathias Soeken[2] Daniel Große[1,3] Rolf Drechsler[1,3]

[1] Department of Mathematics and Computer Science, University of Bremen, Germany
[2] Integrated Systems Laboratory, EPFL, Lausanne, Switzerland
[3] Cyber-Physical Systems, DFKI GmbH, Bremen, Germany
{saeideh,grosse,drechsle}@cs.uni-bremen.de, mathias.soeken@epfl.ch

ABSTRACT

Approximate computing has gained high attention in various applications that can benefit from a reduction in costs by lowering the accuracy. In this paper we present an optimization approach for functional approximation of *Binary Decision Diagrams* (BDDs) which are known for their widespread applications in electronic design automation and formal verification. We propose a three-objective ε-preferred evolutionary algorithm with the first objective set to the BDD size which is given higher priority to the two other objectives set to errors caused by approximation. This is highly demanded by the application to ensure that the minimum size for the approximated BDD is accessible when the error metrics meet certain threshold values. While BDD size minimization is guaranteed by incorporating priority, the use of ε in the proposed approach ensures to guide the search towards desired error values in parallel. Experiments confirm the efficiency of the proposed approach by a size improvement of 64.24% at a fair cost of 3.86% inaccuracy on average.

Keywords

BDD approximation; ε-preferred evolutionary algorithm

1. INTRODUCTION

Approximate computing is an emerging methodology that provides higher efficiency with a loss of quality for applications which can tolerate acceptable error rates.

Minimization of *Binary Decision Diagrams* (BDDs) for approximate computing applications has recently been proposed [4]. BDDs are a graph based data structure for efficient representation and manipulation of Boolean functions. BDDs are widely used in CAD applications such as formal verification, logic synthesis, and test generation [2]. All these applications exploit BDD optimization that aims at finding the BDD with the minimum number of nodes which is denoted by a permutation vector representing an order of input variable indices.

GECCO '16 July 20-24, 2016, Denver, CO, USA

ACM ISBN 978-1-4503-4323-7/16/07.

DOI: http://dx.doi.org/10.1145/2908961.2908987

In this work we contribute to both approximation and variable reordering to find approximated BDDs meeting user defined error thresholds. We map the approximate BDD optimization problem to a multi-criteria problem[1] with three objectives including BDD size and two error metrics which are error rate and the worst-case error. A solution to this problem, i.e., an optimal BDD, is represented by a variable ordering and an approximation vector designating the type of operators and the level indices to which they are applied. We propose an ε-preferred evolutionary algorithm with priority to the size of BDDs which highly corresponds to the specific requirements of the problem, i.e., the size minimization and satisfying error thresholds.

2. BDD APPROXIMATION

Approximate BDD Minimization [4] asks for a given function f, an error metric e, a threshold t, and a bound b, whether there exists a function $\hat{f}$ such that $e(f, \hat{f}) \leq t$ and the number of nodes in the BDD for $\hat{f}$ is at most b.

In this paper, we consider the frequently used worst-case error and error rate as error metrics for e. The *worst-case error*

$$\mathrm{wc}(f, \hat{f}) = \max\{|\operatorname{int}(f(x)) - \operatorname{int}(\hat{f}(x))| \; \forall x \in \mathbb{B}^n\}, \quad (1)$$

where 'int' returns the integer representation of a bit vector, is the maximum difference between the approximate output value and a correct version for all possible inputs. The *error rate*

$$\mathrm{er}(f, \hat{f}) = \frac{\sum_{x \in \mathbb{B}^n} [f(x) \neq \hat{f}(x)]}{2^n} \quad (2)$$

is the ratio of the errors observed in the output value as a result of approximation to the total number of input combinations. The product of error rate and the total number of input combinations corresponds to the Hamming distance when applied to single-output functions.

In order to approximate a BDD we use the operators that have been presented in [4]. They can be efficiently used for BDDs and reduce the size of the BDD while trying to keep the error small.

3. PROPOSED APPROACH

When considering approximate BDD optimization we only care about satisfying the error thresholds. In fact, we ensure not to lose the minimum BDD size at a cost of finding error values smaller than required which is realized by giving

[1] It can also be considered as a lexicographic approach.

higher priority to the size of BDDs. For the error threshold setting we use ε to allow large threshold values. This guarantees that the algorithm will not get stuck in finding invalid solutions violating the error thresholds, and at the same time, the search is directed toward desired error values considering limits for each error metric.

To fulfill the requirements of the approximate BDD optimization problem we use the prioritized ε-preferred relation proposed in [3]. Given two solutions x and y, x is ε-preferred to y if the number of objective functions of x violating ε are smaller than that of y. If both solutions violate ε for the same number of times, the number of different objective functions between both solutions is considered. In this case, x is preferred to y if the number of objective functions of x which are smaller than their corresponding values of y is larger than the number of objective functions of y smaller than the same components of x. Let $p = \{p_1, p_2, \ldots, p_m\}$ be a priority vector determining priorities assigned to an m-objective problem. Each component p_i, $i \in \{1, 2, \ldots, m\}$ can adopt values from the set $\{1, 2, \ldots, n\}$, $n \leq m$, where n is equal to m in case that all objectives have different priorities. Considering a minimization problem, we assume that a lower value of p_i means objective i is of higher priority. Assuming $x|_j$ and $y|_j$ represent subvectors of x and y only including objective functions with priority of j, prioritized ε-preferred is defined as

$$\mathbf{x} \prec_{\text{prio-}\varepsilon\text{-pref}} \mathbf{y} :\Leftrightarrow \exists j \in \{1, 2, \ldots, n\} : \mathbf{x}|_j \prec_{\varepsilon\text{-preferred}} \mathbf{y}|_j \wedge \forall k < j : \mathbf{y}|_k \not\prec_{\varepsilon\text{-preferred}} \mathbf{x}|_k. \quad (3)$$

The relation defined above compares subvectors of objective functions with equal priorities. x is prioritized ε-preferred to y if there is a subvector of the objective functions in x with priority value j that is ε-preferred to the corresponding subvector in y, and at the same time $x|_j$ is not ε-preferred by any subvector of priority higher than j in y.

Our proposed evolutionary algorithm uses relation prioritized ε-preferred to find the smallest approximated BDDs. Each solution inside the population is represented by a permutation of input variables of the Boolean function designating the exact BDD together with a vector consisting of approximation operators and the BDD level indices where each operator should be applied. For each solution, first the exact BDDs are created according to the variable orderings, and then the approximation vectors are applied to the corresponding BDDs to create the approximated ones.

4. RESULTS AND CONCLUSION

We have assessed the performance of our proposed algorithm on 20 multiple-output benchmark set functions from ISCAS89 [1]. For each benchmark, the population size is set to three times the number of input variables but not larger than 120. The algorithm terminates after 200 generations. The maximum number of times that approximation operators are applied to any solution in the population is set to 3 during all experiments. The threshold values and ε values for both error metrics are set to 25% and 10%, respectively. For BDD representation of the benchmark functions and for the implementation of the approximation operators we have used the CUDD package [5].

Table 1 shows the results of the proposed approximate BDD optimization approach. We have compared the BDD sizes obtained by our approach with the non-optimized initially ordered BDDs indicated by $\#N$-initial. The experimental evaluations show a noticeable size reduction at a small cost of error. More precisely, an average size improvement of 64.24% has been achieved while the total average inaccuracy of both error metrics is just 3.86% that is insignificant compared to the achieved size improvement.

Table 1: Experimental Evaluation

Benchmark-$\#I/O$	$\#N$-initial	Prioritized ε-preferred			
		$\#N$	ER	WC	impr.
s208-18/9	1033	63	12.54%	0.19%	93.90%
s298-17/20	125	74	<0.001%	<0.001%	40.80%
s344-24/26	206	127	6.25%	6.24%	38.35%
s349-24/26	206	112	6.25%	0.78%	45.63%
s382-24/27	168	122	23.52%	0.21%	27.38%
s386-13/13	281	117	0.39%	0.19%	58.36%
s400-24/27	168	130	<0.001%	<0.001%	22.62%
s420-34/17	262227	137	3.12%	<0.001%	99.95%
s444-24/27	226	127	<0.001%	<0.001%	43.81%
s510-25/13	19076	170	0.39%	12.50%	99.11%
s526-24/27	232	128	<0.001%	<0.001%	44.83%
s641-54/42	1352	537	7.59%	<0.001%	60.28%
s713-54/42	1352	540	12.50%	<0.001%	60.06%
s820-23/24	2651	241	0.09%	9.76%	90.91%
s832-23/24	2651	236	0.12%	15.62%	91.10%
s953-45/52	1723	383	<0.001%	<0.001%	77.77%
s967-45/52	1755	395	5.07%	<0.001%	77.49%
s1196-32/32	2295	605	1.22%	<0.001%	73.64%
s1238-32/32	2295	654	2.92%	<0.001%	71.50%
s1488-14/25	1016	325	1.95%	>25%	68.01%
AVG	15051.9	261.15	4.20%	3.52%	64.24%

$\#I/O$: number of inputs/outputs, $\#N$-initial: exact BDD size, $\#N$: BDD size after approximation, ER: error rate, WC: worst case error, improvement is calculated compared to the initially ordered exact BDD

Acknowledgments

This research was supported by the University of Bremen's graduate school SyDe funded by the German Excellence Initiative, by the German Research Foundation within projects MANIAC (DFG) (DR 287/29-1) and Reinhart Koselleck (DFG) (DR 287/23-1), and by H2020-ERC-2014-ADG 669354 CyberCare.

5. REFERENCES

[1] F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In *ISCAS*, pages 1929–1934, 1989.

[2] R. E. Bryant. Binary decision diagrams and beyond: enabling technologies for formal verification. In *ICCAD*, pages 236–243, 1995.

[3] N. Drechsler, A. Sülflow, and R. Drechsler. Incorporating user preferences in many-objective optimization using relation ε-preferred. *Natural Computing*, 14(3):469–483, 2015.

[4] M. Soeken, D. Große, A. Chandrasekharan, and R. Drechsler. BDD minimization for approximate computing. In *ASP-DAC*, pages 474–479, 2016.

[5] F. Somenzi. CUDD: CU Decision Diagram package release 2.5.0. University of Colorado at Boulder, 2012.