RATIO CUT HYPERGRAPH PARTITIONING USING BDD BASED MBOA OPTIMIZATION ALGORITHM

Josef Schwarz
Brno University of Technology
Faculty of Information Technology
Bozetechova 2, 612 66 Brno, CZ
schwarz@fit.vutbr.cz

Jiri Ocenasek
Brno University of Technology
Faculty of Information Technology
Bozetechova 2, 612 66 Brno, CZ
ocenasek@fit.vutbr.cz

Abstract. *This paper deals with the k-way ratio cut hypergraph partitioning utilizing the Mixed discrete continuous variant of the Bayesian Optimization Algorithm (mBOA). We have tested our algorithm on three partitioning taxonomies: recursive minimum ratio cut, multi-way minimum ratio cut and recursive minimum cut bisection. We have also derived a new approach for modeling of Boolean functions using binary decision diagrams (BDDs) which are primarily used as a probabilistic model of the mBOA algorithm.*

1 Introduction

Partitioning problem is investigated in many research papers. A general survey of partitioning methodologies is presented in [1] and a comparison of several partitioning approaches was presented in [2] and [3]. Their common conclusion pointed out that the ratio cut partitioning gives the best results. The frequently mentioned reference involving the ratio cut criterion is that by Wei and Cheng [4]. They used move-based heuristic algorithm where graph node movement from one partition to the other is controlled by current ratio cut.

Recently, a generalization of the ratio cut implementation was published in [5] where the pin count estimated by Rent's rule was used. Problems of recursive balanced bisection are solved in [6]. A new enhancement of min-cut partitioning (useful for placement) was published in [7]. It validates the multilevel partitioning paradigm for hypergraphs with efficient implementation using benchmarks published in [8].

A separate class of optimizers involves genetic algorithms (GAs). An interesting study on graph partitioning using hard theoretical benchmarks was published in [9]. The newest research based on memetic algorithms and analysis of the fitness landscape was presented in [10]. Estimation distribution algorithms (EDAs) represent a new class of efficient optimizers. Unlike the standard GAs the crossover and mutation operators are replaced by probability estimation and sampling techniques. In other words, statistics about the search space is explicitly maintained by creating probabilistic models of the good solutions found. We have focused especially on the Bayesian Optimization Algorithm (BOA). Our first experience with these techniques was presented in [11] and [12] where minimum-cut bisection was tested and the ability to find global optima was presented. The multi-objective partitioning problem was published in [13] and [14]. The application of recursive partitioning for placement problem was presented in [15].

During our research we have implemented and tested a new Mixed discrete continuous variant of the Bayesian Optimization Algorithm (mBOA) to test its ability for multi-way partitioning. It follows the basic theory and implementation of the BOA initially published in [16], [17] and [18]. We used the concept of binary decision diagrams (BDDs) [19] as

graphical probabilistic model approach, including our reformulated standard formulae of the Bayes - Dirichlet metric [20]. In addition, we extended the concept of BDDs to be able to process discrete and continuous domain, that is necessary for the direct (parallel) multi-way hypergraph partitioning.

The remainder of our paper is organized as follows. The specification of partitioning problem is done in the next section. Probabilistic model and mBOA algorithm are described in the third and fourth sections. An extra application of binary decision diagram for Boolean function modeling is presented in the fifth section. The experimental results for hypergraph partitioning are presented in the sixth section.

2 Problem specification

Hypergraph partitioning is a well known problem of graph theory. It means dividing hypergraph into disjoints subhypergraphs/modules each containing a subset of nodes. The partitioning is done using various criterions. Generally, the criterion is to minimize the number of hyperedges that have nodes in different partitions or alternatively the minimization of pin count is used.

The particular bi-partitioning problem can be defined as follows: Let us assume a hypergraph $H=(V,E)$, with $n=|V|$ nodes and $m=|E|$ edges. The goal is to find such a partition $(V1,V2)$ of V that minimizes the number of hyperedges that have nodes in different set $V1, V2$ (see (1)) under the defined constraint of the partition size. It can be also expressed by the balance/unbalance of the partition size (see (2)). The set of external hyperedges can be labeled as $E_{cut}\ (V1,V2)$ and the following cost function is defined ($\bullet$ symbol represents the multiplication operation):

$$C1\ (V_1,V_2) = |E_{cut}\ (V_1,V_2)| = |\ \{e \in E\ |\ e \cap V_1 \neq \emptyset\ ,\ e \cap V_2 \neq \emptyset\}\ | \qquad (1)$$

with constraint

$$C2(V_1,V_2) = //V_1/-/V_2// <= \alpha V,\ \ \alpha \in <0,1) \qquad (2)$$

Another form of balance used in ratio cut metric is expressed in product form:

$$C3(V_1,V_2) = /V_1/\bullet/V_2/ = /V_1/\bullet/V-V_1/ \qquad (3)$$

Therefore the ratio cut partitioning that can be specified by the criterion (4) was introduced:

$$RC = C1 / (/V1/\bullet/V2/) = C1/C3 \qquad (4)$$

The ratio cut formulation allows the tradeoff between nets cut and the balance value during the partitioning. The numerator represents the minimum-cut criterion while denominator favours near-bisection. Our goal is to test the performance of the newly designed advanced mBOA algorithm for three main partition taxonomies: recursive minimum ratio cut, multi-way (parallel) minimum ratio cut and recursive minimum cut bisection. We used mainly hard artificial benchmarks with known global optimum and high nonlinearity/epistasis of instances.

3 Solution encoding in evolutionary algorithms

Using the population based evolution algorithm the solution of bi-partitioning is represented by binary string:

$X = (X_0, X_1, .., X_{n-1})$ is a string/solution of length n with X_i as a variable,
$x = (x_0, x_1, \ldots, x_{n-1})$ is a string/solution with $x_i \in \{0,1\}$ as a possible instantiation of variable X_i.

In case of direct *k*-way partitioning the encoding of solution uses an alphabetic string. Each variable can acquire *k* distinct alleles. For the simplest case of *2* - way partitioning/bisection of a simple graph *G(V,E,W)* we derived on the binary string $X=(x_0, x_1, .., x_{n-1})$ the following quadratic cost function:

$$C1 = Ecut(V1, V2) = \sum_{\substack{i=0 \\ j>i}}^{n-1} w_{ij}(x_i + x_j - 2x_i x_j) \tag{5}$$

with the balance value $C2 = \left| \sum_{i=0}^{n-1} x_i - \sum_{i=0}^{n-1} (1 - x_i) \right|$, (6)

where the coefficient $w_{ij}=1$ in case that net/edge exists between node *i* and *j*, else $w_{ij}=0$. The balance *C3* can be expressed by term:

$$C3 = \sum_{i=0}^{n-1} x_i \bullet \sum_{i=0}^{n-1} (1 - x_i) \tag{7}$$

4 Standard Bayesian optimization algorithm (BOA)

The principle of BOA algorithms that work on the basis of probabilistic models can be specified on the following framework:

Generate initial population of individuals of size M (randomly);
While *termination criteria is false* ***do***
begin
Select parent population of N individuals according to a selection method;
Estimate the probability distribution of the selected parents;
Generate new offspring according to the estimated probabilistic model;
Replace some individuals in current population with generated offspring;
end

4.1 Bayesian network (BN)

The original Bayesian Optimization Algorithm [17] operates on the population of strings/chromozomes of *n* binary variables/genes. The Bayesian Network is learned in each generation how to encode the structure of promising solutions. The following step includes a sampling process to discover the promising areas of the search space. In BN for each variable X_i a set of parent variables Π_{X_i} is defined which it depends on, so the distribution of individuals is expressed by the conditional probabilities:

$$p(X) = \prod_{i=0}^{n-1} p\left(X_i \mid \Pi_{X_i}\right) \tag{8}$$

Generally, the existence of a directed edge from X_j to X_i in the network implies the belonging of the variable X_j to the set Π_{X_i}.

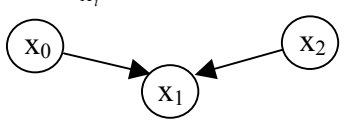

Figure 1: Example of BN with 3 nodes, where X_0 and X_2 are independent and X_1 depends on X_0 and X_2

4.2 Binary decision diagram (BDD)

Additional accuracy and efficiency can be achieved by utilizing decision trees or diagrams in Bayesian network, see [19]. For the BN in Fig. 1, the corresponding contingency table for X_1 would have 4 rows, one row for each possible instance of the substring (X_0,X_2). Now let us suppose that in some case $p(X_1|10) = p(X_1|11)$ and $p(X_1|00) \neq p(X_1|01)$. It is evident, that if $X_0=1$ the value of X_1 does not depend on X_2. This allows to reduce the number of table rows (* is a don't-care symbol).

Table 1. Simplified contingency table by don't-care symbols.

X_0	X_2	consequent $p(X_1)$
0	0	$p(X_1\|00)$
0	1	$p(X_1\|01)$
1	*	$p(X_1\|1*)$

This situation can be expressed by a decision tree (see Fig. 2). Each variable, which determines the X_1 value corresponds to one or more split nodes in the tree. Each row in the Tab.1 corresponds to one leaf of the tree and each leaf determines $p(X_1)$ among the individuals fulfilling the split conditions on the path from the root:

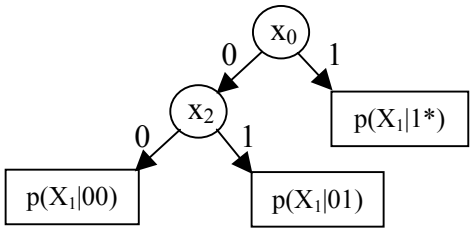

Figure 2: Binary decision tree for the determination of X_1

Next advantage of decision trees lies in low complexity of their building – the step of adding new split node is easy to evaluate by the metric (9) – it splits only one row in the contingency table. From the Bayes-Dirichlet metrics (BD) we derived the incremental equation for adding one new binary split:

$$Gain(X_i,X_j)=\frac{\prod_{r\in\{0,1\}}\prod_{s\in\{0,1\}}\Gamma(m_{r,s}+1) \cdot \Gamma(\sum_{r\in\{0,1\}}\sum_{s\in\{0,1\}}(m_{r,s}+1))}{\prod_{r\in\{0,1\}}\Gamma(\sum_{s\in\{0,1\}}(m_{r,s}+1)) \cdot \prod_{r\in\{0,1\}}\Gamma(\sum_{s\in\{0,1\}}(m_{r,s}+1))}, \qquad (9)$$

where X_i is the child variable, X_j is the parent variable - possible split, and $m_{r,s}$ is the number of individuals having $X_j=r$ and $X_i=s$. Note that the splitting is performed recursively, so $m_{r,s}$ is determined only from the subpopulation being split.

Table 2. Notation of symbols $m_{r,s}$

	$\mathbf{X_i = 0}$	$\mathbf{X_i = 1}$
$\mathbf{X_j = 0}$	$m_{0,0}$	$m_{0,1}$
$\mathbf{X_j = 1}$	$m_{1,0}$	$m_{1,1}$

Moreover, the gain of each split operation can be penalized by the model complexity (e.g. the depth of the tree), which removes the need of the limitation of the number of parents.

The algorithm for building the binary decision tree from the population can be illustrated in the following figure:

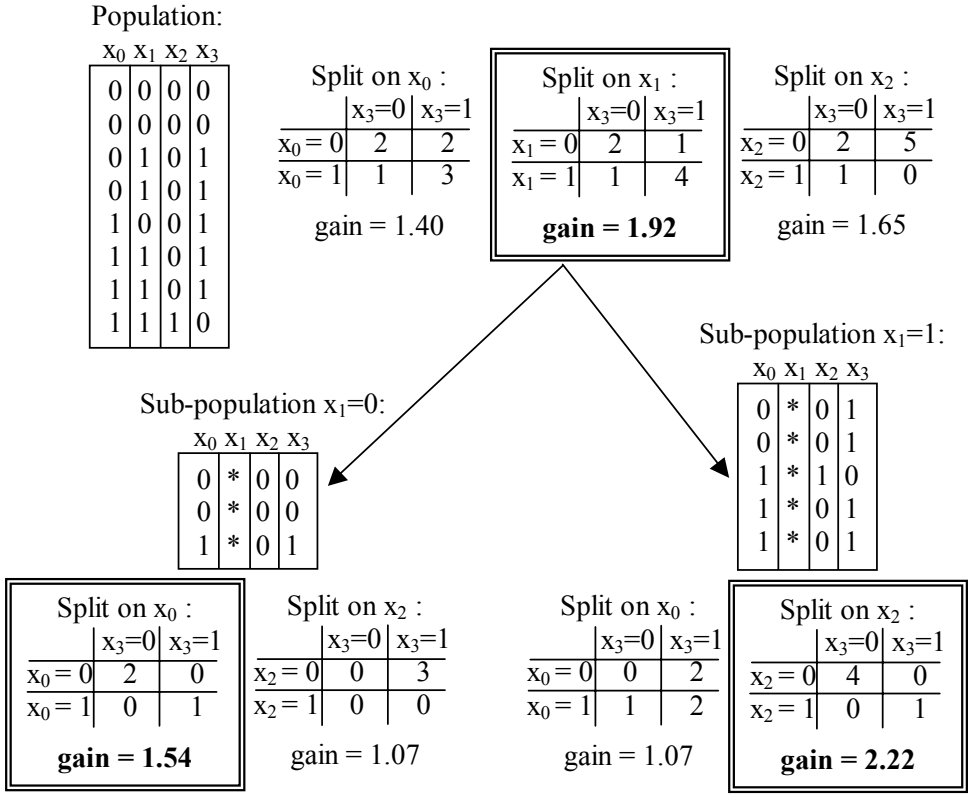

Figure 3: Building binary decision tree for x_3

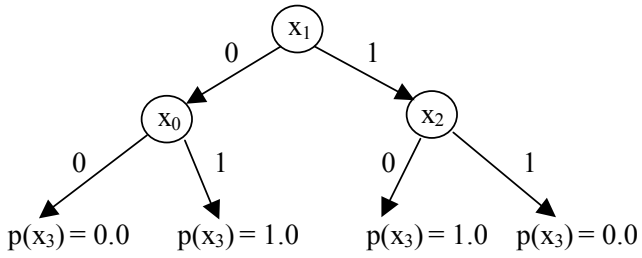

Figure 4: Final binary decision tree for x_3

4.3 BDD for mixed continuous discrete mBOA algorithm

The idea of the utilizing BDD in BOA algorithm was mentioned for the first time in [19]. In our Mixed Bayesian Optimization Algorithm (mBOA) we extended the idea of decision diagram to continuous and integer domains. Our mBOA is the only one EDA which is able to solve problems with mixed real-discrete parameters (alleles) without conversion to binary representation. In Fig. 5 an example with continuous child variable X_i and continuous parent variable X_j is shown. Our algorithm tries to find a X_j and X_i boundaries such that the numbers of individuals in each quadrant maximize (9). In the case of integer domain (see Fig. 6) we use hill-climbing algorithm to split the set of possible X_j values into left and right subset. The variable s in (9) goes through all possible X_i values instead of only two values *{0,1}*.

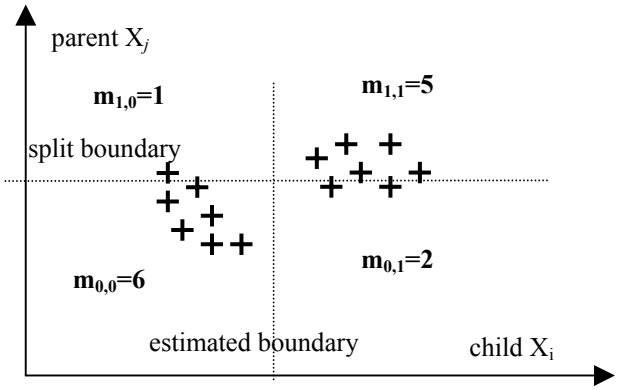

Figure 5: An example of real domain split

	$X_i = 0$	$X_i = 1$	$X_i = 2$	$X_i = 3$
$X_j \in \{0,2\}$	$m_{0,0}$	$m_{0,1}$	$m_{0,2}$	$m_{0,3}$
$X_i \in \{1,3\}$	$m_{1,0}$	$m_{1,1}$	$m_{1,2}$	$m_{1,3}$

Figure 6: An example of integer domain split

4.4 Probabilistic model sampling

As a result of probabilistic model construction we obtain a set of decision trees, one tree for each variable. This set of trees is used for generation of the offspring (new population) during the Probabilistic Logic Sampling (PLS). During this PLS process the variables (whose parents are already determined) are generated by traversing their decision trees. This is repeated for each offspring until all its variables are generated.

In our example in Fig. 1 the variables X_0 and X_2 are generated as first. Then the concrete value of $p(X_1)$ is determined according to concrete values of X_0 and X_2 - using decision tree from Fig. 2.

4.5 Parallel BDD construction

The probabilistic model construction is the most time consuming task in mBOA. We are currently working on the distributed mBOA using Message Passing Interface (MPI). The goal is to utilize more processors when searching for a good model. Our consolation is that the BD metric is separable and can be written as a product of n factors, where i-th factor expresses the quality of decision tree for variable X_i. It is possible to use up to n processors, each processor has its own local copy of parent population and it builds tree for different variable. The addition of splits/parents to the trees is parallel, so we need an additional mechanism to keep the mutual dependencies acyclic. In [21] we proposed the concept of restricted set of parents in BN. We are going to extend this concept for BDD in mBOA. In each generation, variables will be ordered in advance, according to a random permutation vector $\boldsymbol{q} = (q_0, q_{1,..,}q_{n-1})$. Each decision tree of variable X_i may contain only such parental splits X_j having $q_j < q_i$. This approach ensures linear scalability, because no communication overhead is required. In addition, scalable methods for overlapping the communication latency during generation, evaluation and broadcasting of new population among the processes will be implemented using the farmer-workers architecture.

5 BDD diagram as a model of Boolean function

Binary Decision Diagrams are commonly used for representation of Boolean functions because of their efficiency in terms of time and space. The BDDs are capable to improve many conventional algorithms significantly. Besides Boolean function, BDDs can be also used for representation of other types of discrete functions, such as multi-valued functions, cube sets and arithmetic formulas [22]. BDDs or oriented BDDs (OBDDs) can be constructed from Binary Decision Tree (BDT) using two basic reduction rules: 1) reduction

of nodes with unique ancestor nodes and 2) sharing all equivalent sub-graphs. The previous two-phase approach suffers from the necessity of a specification of the node ordering in the first phase. The node ordering is often provided by genetic algorithms [23]. In another approaches the ordering is found dynamically during the process of BDD building. We suggest a new approach to build BDDs for a Boolean function (see Fig. 7 and Fig. 8). This approach using BD metric is similar to the previous one presented in Fig. 4 in case that we interpret the variable x_3 as a Boolean function. The function must be represented by truth table. But this can be considered as a bottleneck of the method. The advantage of suggested approach lies in the implicit ordering of BDD nodes and lower BDT complexity.

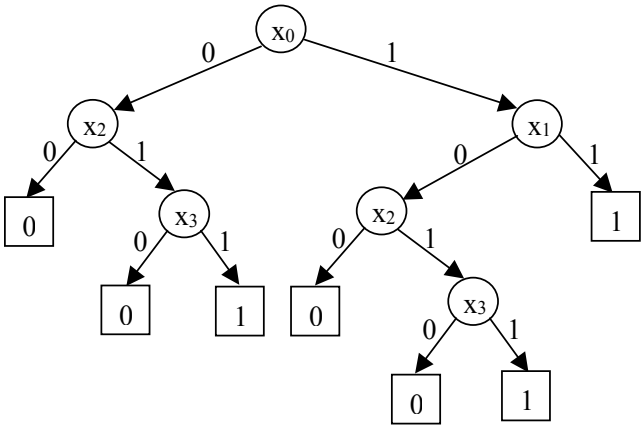

Figure 7: BDT-model of Boolean function $F=x_0x_1+x_2x_3$

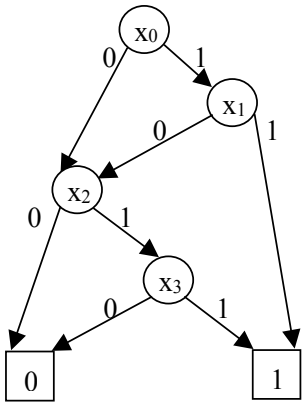

Figure 8: BDD after removing duplicate subgraphs

6 Experimental results

6.1 Test benchmarks

The complexity of the partitioning problem is determined by the type and complexity of the instances – benchmark graphs. We used four types of benchmarks - three artificial ones (like in the excellent experimental studies [9], [10] and in [11]) with known global optimum (see Fig. 9) and one benchmark consisting of two real circuits (random logic) from benchmark package [8]:

1. Regular graphs *Grid_n* with square grid structure, where the notion *n* specifies the number of nodes. Graphs *GridH_n* have a 2-edge asymmetrical horizontal bottle-neck and *GridHV_n* have an extra 2-edge asymmetrical vertical bottle-neck. As an example a quadrisection of asymetric graph *GridH_100* is represented in Fig. 9.
2. Random geometric graph *U_n.d* with *n* vertices placed in the unit square. The coordinates of vertices are chosen randomly with uniform distribution. An edge exists between two vertices if their Euclidean distance is *l* or less, where the expected vertex degree is specified by $d = n\pi l^2$. We have chosen *n=120, d=5*.
3. Caterpillar graphs *CAT_n.k*, with *k* articulations, *(n-k)/k* legs for each articulation and *n* nodes.
4. Real circuits labeled by *IC_n.* The hypergraph IC_67 consists of 67 nodes and 138 edges/nets, the IC_116 consists of 116 nodes and 329 edges/nets.

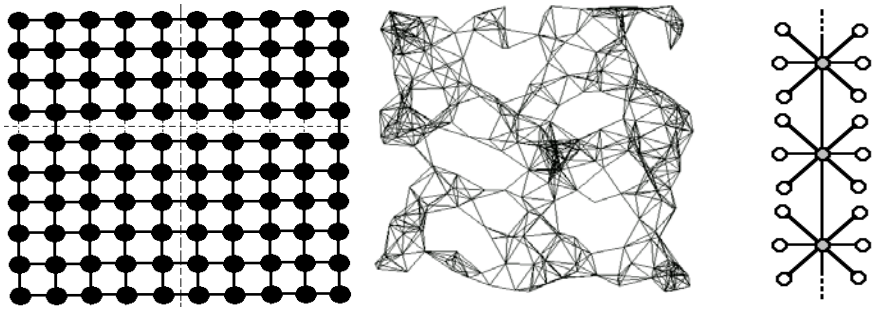

Figure 9: Graph structure of *GridH_100, U_120.5* and a segment of the *CAT_21.3 graph*.

6.2 Summary of experimental results

Experimental results are summarized in Tab. 3 and partly in Fig. 10. Our algorithm is capable to solve the top-down *k*-way hypergraph partitioning but for the better preview we present results for *4*-way partitioning only. We have arranged 3 types of experiments:

- Ratio cut partitioning by recursive bi-partitioning RRC
- Ratio cut partitioning by non recursive multi-way partitioning MRC
- Partitioning by recursive bisection RB

The ratio cut value is presented in the inverse mode, because this mode for fitness representation is used. The 1/RC values printed in bold represent the global optimum.

Table 3. Partitioning results for four types of graphs and three algorithms. The population size was set to N=40*n for RRC and RB, N=100*n for MRC, computation time is expressed in seconds.

			RRC		MRC		RB	
Benchmark	n	1/RC optimum	1/RC cut size	Time Eval.	1/RC cut size	Time Eval.	1/RC cut size	Time Eval.
Grid_64	64	**3855.1**	**3855.1** 16	179 35939	3626.6 17	2574 233600	**3855.1** 16	479 80316
GridH_64	64	**5236.4**	**5236.4** 10	136 31864	**5236.4** 10	2392 214400	3640.9 17	479 87819
GridHV_64	64	**10125.0**	**10125.0** 4	134 31830	**10125.0** 4	1979 168600	4369,1 14	483 81253
Grid_100	100	**18601.2**	**18601.2** 20	1041 78000	17857.1 20	15258 495000	**18601.2** 20	3406 219000
GridH_100	100	**27692.3**	**27692.3** 12	867 75400	26584.6 12	12617 385000	18601.2 20	3405 221000
GridHV_100	100	**66355.2**	**66355.2** 4	850 67440	**66355.2** 4	10931 330000	22977.9 16	3491 220000
IC_67	67	-	1006.9 64	387 88521	**1165.9** 65	2647 219000	1091.8 71	932 167232
IC_116	116	-	7253.33 62	1554 93344	**8250.7** 80	17977 435000	7113.5 95	3956 204477
U_120.5	120	-	**15669.5** 31	2613 129047	14168.0 44	30111 594000	12272.7 65	5771 247200
CAT_105.7	107	-	**101250.0** 3	933 66750	**101250.0** 3	12358 336000	29659.5 15	5292 295679

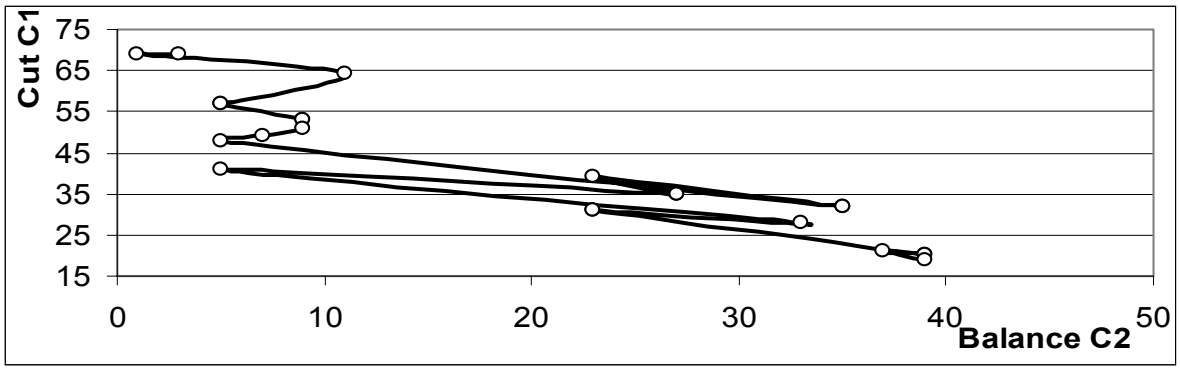

Figure 10: Relation between criterion *C1* and *C2* during the RRC minimization of the *IC67* hypergraph.

7 Conclusions

We have implemented a new advanced evolutionary algorithm mBOA for multi-way partitioning of hypergraphs. Its main advantage against the mostly used move-based heuristic methods lies in the ability to discover and determine the amount of epistasis in a given problem instance and to find the optimal solution. The Bayesian statistics and BDD diagrams used in probabilistic model cause high performance of mBOA. It is evident from Tab. 3 that the recursive ratio cut algorithm RRC is always able to find the known global optimum in case of artificial graphs. The non recursive multi-way ratio cut algorithm MRC provides almost comparable results to the RRC but the time complexity is very high – approximately 14 times greater than for the RRC. The recursive bisection algorithm RB is worse in more cases in comparison with the previous two algorithms. An example of the optimization process for RRC algorithm is shown in Fig. 10. The white nodes on the optimization curve represent the best solution in each generation. It can be recognized how the algorithm gradually searches for the minimum cut size value resulting in greater balance value *C2*. The future activity will be focused on the sophisticated RRC algorithm including the external pin count, Rent's rule and the massive parallelization of the mBOA algorithm. The research will be also directed towards the enhancement of the suggested approach to the modeling of Boolean functions using BDDs with BD metric.

Acknowledgments

This research has been carried out under the financial support of the Research intention no. CEZ: J22/98: 262200012-"Research in information and control systems" (Ministry of Education, CZ) and the research grant GA 102/02/0503 "Parallel system performance prediction and tuning" (Grant Agency of Czech Republic).

References

[1] Alpert, C. J., Kahng, A. B.: Recent Directions in Netlist Partitioning: A Survey, Integration: The VLSI Journal 19 (1995), pp. 1-81.

[2] Agrawal, B., Narendran ,N., Shivakumar, N.: Multi-Way VLSI Circuit Partitioning. Proceedings of 9th International Conference on VLSI Design, Bangalore, India, Jan'96, pp.1-7.

[3] Hagen, L., Kahng A. B., Kurdahi F.: "On the Intrinsic Rent Parameter and New Spectra-Based Methods for Wireability Estimation". IEEE Transaction on CAD 13(1), January 1994, pp. 27-37.

[4] Wei, Y.- C., Cheng, C.-K: Ratio Cut Partitioning for Hierarchical design, IEEE Trans. Computer Aided Design Integrated Circuit & System, Vol.10 (No.7), pp. 911-921, July 1991.

[5] Stroobandt, D.: Pin Count Prediction in Ratio Cut Partitioning for VLSI and ULSI. In M. A. Bayoumi, Editor, Proceedings of the IEEE International Symposium on Circuits and Systems, Pages VI/262-VI/265, May 1999.

[6] Simon, H. D., Teng, S.-H.: How Good is Recursive Bisection?, SIAM J. Scientific Computing 18 (5) (1997), pp. 1436-1445.

[7] Karypis, G., Kumar, V.: Hmetis - A Hypergraph Partitioning Package, Version 1.5.3, University of Minnesota, Department of Computer Science & ENGINEERING, Army HPC Research Center Minneapolis. http://www-users.cs.umn.edu/~karypis/metis/hmetis/download.shtml.

[8] A Benchmark Set, University of California, Los Angeles, VLSI CAD Laboratory, http://vlsicad.cs.ucla.edu/~cheese/benchmarks.html.

[9] Than Nguen Bui, Byung Ro Moon: Genetic Algorithm and Graph Partitioning. IEEE Transactions on Computers, Vol. 45, No.7, July 1996, pp. 841-855.

[10] Merz P., Freisleben, B.: Fitness Landscapes, Memetic Algorithms and Greedy Operators for Graph Bi-Partitioning. Evolutionary Computation, Vol. 8, No. 1, pp. 61-91, 2000. Preprint Available as Technical Report No. 98-01 (Informatik-Berichte).

[11] Schwarz, J., Očenášek, J.: Experimental Study: Hypergraph Partitioning Based on the Simple and Advanced Genetic Algorithm BMDA and BOA, Proceedings of the Mendel' 99 Conference, Brno University of Technology, 1999, pp. 124-130, ISBN 80-214-1131-7.

[12] Schwarz, J., Očenášek, J.: The Knowledge-based Evolutionary Algorithm KBOA for Hypergraph Bisectioning. Proceedings of the Fourth Joint Conference on Knowledge-based Software Engineering Brno, Czech Republic, 2000, pp. 51-58, ISBN 1 58603 060 4 (IOS Press).

[13] Schwarz, J., Očenášek, J.: Multiobjective Bayesian Optimization Algorithm for Combinatorial Problems: Theory and Practice, Neural Network World, Vol.11, No.5, 2001, Published by Academy of Science Czech Republic, pp.423-441, ISSN 1210-0552.

[14] Schwarz, J., Očenášek, J.: Evolutionary Multiobjective Bayesian Optimization Algorithm: Experimental Study. Proceedings of the 35th Spring International Conference MOSIS'01, Vol. 1, MARQ Ostrava, Hradec nad Moravicí, 2001, p. 101-108, ISBN 80-85988-57-7.

[15] Schwarz, J., Očenášek, J.: Partitioning-oriented Placement Based on Advanced Genetic Algorithm BOA, Proceedings of the 6th International Mendel Conference on Soft Computing, Brno University of Technology, Faculty of Mechanical Engineering, Brno, 2000, pp . 145-150.

[16] Pelikan, M., Goldberg, D. E., & Cantú-Paz, E.: Linkage Problem, Distribution Estimation, and Bayesian Networks. IlliGal Report No. 98013, November 1998, pp. 1-25.

[17] Pelikan, M.: A Simple Implementation of Bayesian Optimization Algorithm in C++ (Version 1.0). Illigal Report 99011, February 1999, pp. 1-16.

[18] Pelikan, M., Goldberg, D. E., & Lobo, F.: A Survey of Optimization by Building and Using Probabilistic Model, Illigal Report 99018, September 1999, pp. 1-12.

[19] Pelikan, M., Goldberg, E., Sastry, K.: Bayesian Optimization Algorithm, Decision Graphs, and Occams Razor. Illigal Report No. 2000020, May 2000, pp.1-24.

[20] Heckerman, D., Geiger, D., & Chickering, M. (1994). Learning Bayesian Networks: The combination of Knowledge and Statistical Data (Technical Report MSR-TR-94-09), Redmont, WA: Microsoft Research, 1995, pp. 1-53.

[21] Očenášek, J., Schwarz, J.: The Distributed Bayesian Optimization Algorithm, Proceeedings of the Eurogen 2001 - Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, The National Technical University of Athens, Greece, 19-21 September 2001, in print.

[22] Sasao T., Fujita, M., editors: Representations of Discrete Functions. Kluwer Academic Publisher, London 1996.

[23] Drechsler R.: Evolutionary algorithms for VLSI CAD. Kluwer Academic Publishers, London 1998, ISBN 0-7923-8168-8.