Evolutionary Multiobjective Bayesian Optimization Algorithm: Experimental Study

Josef Schwarz*
schwarz@dcse.fee.vutbr.cz
Jiří Očenášek*
ocenasek@dcse.fee.vutbr.cz

Abstract: This paper deals with the utilizing of the Bayesian optimization algorithm (BOA) for multiobjective optimization of hypergraph partitioning. The main attention is focused on the incorporation of the Pareto optimality concept. We have modified the standard algorithm BOA for one criterion optimization according to well known niching techniques to find the Pareto optimal set. This approach was compared with standard weighting techniques and the single optimization approach with the constraint. The experiments are focused mainly on the bi-objective optimization because of the visualization simplicity.

Key Words: Multiobjective optimization, evolutionary algorithms, Bayesian optimization algorithm, Pareto set, niching techniques, hypergraph bisectioning.

1 Introduction

Many real-world problems have multiple often competing objectives. While in the case of single-objective optimization the optimal solution is simply distinguishable, this is not true for multiobjective optimization. Historically, multiple objectives have been combined to form a scalar objective function through weighted sum of individual objectives or by turning objectives into constraints. But setting of weights and specification of penalty functions is not a simple task and these values can be found only experimentally. The better approach lies in finding all possible trade-offs among the multiple, competing objectives. These solutions are optimal, nondominated, in that there are no other solutions superior in all objectives. These so called Pareto optimal solutions lie on the Pareto optimal front. There are many papers that present various approaches to find of Pareto optimal front almost based on the evolutionary algorithms. Let us mention here the well known niched Pareto genetic algorithm NPGA [1]. A wide review of basic approaches and the specification of original Pareto evolutionary algorithms includes the dissertation [2], [3], [4] where the last one describes the original strength Pareto optimization algorithm SPEA. From the last period let us mention an interesting Pareto-Envelope based Selection Algorithm PESA [5] which might outperform the very good algorithm SPEA.

All of these capable algorithms based on evolutionary algorithms (EA) have the common disadvantage - the necessity of ad hoc setting of parameters like crossover, mutation and selection rate. That is why we have analyzed and used one of the Estimation of Distribution Algorithms (EDAs). These algorithms also called probabilistic model-building genetic algorithms have attached a growing interest during the last few years because crossover and mutation operators used in standard GA are replaced by probability estimation and sampling

* Department of Computer Science and Engineering, Božetěchova 2, CZ-612 66 Brno

techniques to avoid the necessity to specify the set of GA parameters. We will focus on one of them - the Bayesian optimization algorithm [6], [7]. Recently we have published our experience with this algorithm in [8] where single criterion optimization of hypergraph bisectioning was described. In this paper we have focused on the bi-objective optimization of hypergraph bisectioning.

2 Problem specification

Hypergraph partitioning is a well known problem of graph theory. We have investigated a special case of k-way partitioning for $k=2$ called bisectioning. If necessary the k-way partition can be found by recursive 2-way bisectioning. The hypergraph model can be used for many application problems e.g. for system segmentation, network partitioning and VLSI layout.
The particular bisectioning problem is 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 bisection $(V1,V2)$ of V that minimizes the number of hyperedges that have nodes in different set $V1, V2$ (1) and the difference/balance of the partition sizes (2). The set of external hyperedges can be labelled as $E_{cut}(V1,V2)$ and the following cost functions are defined:

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

$$C2(V_1,V_2\ ]=\ |\ |V_1|-|V_2|\ | \qquad (2)$$

For more formal specification of the problem, the following notation is used:

$P=(X_1, X_2,..,X_N)$ with $X_j\in P$, is the population of the solutions/string/individuals
X is a string/individual of the population P the length of which is n
$X=(x_0, x_1,..,x_{n-1})$ is a string/individual with $x_i\in\{0,1\}$
$C(X)$ is the cost function of the string X

Each solution of the bisection is represented by a binary string $X=(x_0,\ x_1,\ ...,\ x_{n-1}\ )$. The variable x_i represents the partition number, the index specifies the node in the hypergraph. For the case of simple graph $G(V,E,R)$ bisectioning we have derived the following two cost functions on the binary string $X=(x_0\ , x_1\ ,.., x_{n-1})$ to be minimized:

$$C1=\sum_{\substack{i=0\\ j>i}}^{n-1} r_{ij}(x_i+x_j-2x_ix_j) \qquad (3)$$

$$C2=|\sum_{i=0}^{n-1}x_i-\sum_{i=0}^{n-1}(1-x_i)\ |\ , \qquad (4)$$

where the coefficient $r_{ij}\in R$ equals to one in case the net/edge of the graph G exists between node i and j, else $r_{ij}=0$. The cost $C1$ represents the cut value of the bisection and the cost $C2$ expresses the balance/difference of the partition sizes. There are three approaches how to solve this 2-objective optimization problem that will be described in the next chapters.

3 The BOA algorithm

The BOA algorithm is a population based evolutionary algorithm but the reproduction process of individuals is replaced by probability estimation and sampling techniques. It uses statistical information contained in the current population to detect multivariate parameter dependencies. The learned Bayesian network BN encodes a joint probability distribution based on the conditional probabilities; the BN quality is estimated by Bayesian-Dirichlet metrics. The estimated probability model is then used to generate new promising solutions according to this distribution using the sampling process. The BOA algorithm can be described as follows [7]:

Generate initial population of size N (randomly);
While *termination criteria is false* **do**
begin
  Select parent population of M individuals according to fitness function f(X) (M<N);
  Estimate the distribution of the selected parents and construct the Bayesian network BN;
  Generate new offspring according to the estimated model and BN network;
  Replace some individuals in current population by generated offspring;
end

4 Multiobjective BOA algorithm

A general multiobjective optimization/maximization problem MOP can be described as a vector function $\boldsymbol{f}$ that maps a tuple of n parameters to a tuple of m objectives [4]:

$$\max\ \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \quad (5)$$
$$\text{subject to } \boldsymbol{x} = (x_0, x_1, \ldots, x_{n-1}) \in \boldsymbol{X}$$
$$\boldsymbol{y} = (y_1, y_2, \ldots, y_m) \in \boldsymbol{Y},$$

where $\boldsymbol{x}$ is called decision vector, $\boldsymbol{X}$ is the parameter space, $\boldsymbol{y}$ is the objective vector, and $\boldsymbol{Y}$ is the objective space.

The set of solution of MOP includes all decision vectors for which the corresponding objective vectors cannot be improved in any dimension without degradation in another - these vectors are called Pareto optimal set. The idea of Pareto optimality is based on the Pareto dominance. A decision vector $\boldsymbol{a}$ dominates decision vector $\boldsymbol{b}$ iff $f_i(\boldsymbol{a}) \geq f_i(\boldsymbol{b})$ for $i=1,2,.., m$ with $f_i(\boldsymbol{a}) > f_i(\boldsymbol{b})$ for at least one i. The vector $\boldsymbol{a}$ is called Pareto optimal if there is no vector $\boldsymbol{b}$ which dominates vector $\boldsymbol{a}$ in parameter space $\boldsymbol{X}$.

In objective space the set of nondominated solutions lie on a surface known as Pareto optimal front. The goal of the optimization is to find a representative sampling of solutions along the Pareto optimal front. From the theory of Pareto optimal set it is evident that the optimization algorithms should be able to find as many Pareto optimal solutions as possible. The techniques how to do it lies in keeping the diversity using some of the niching techniques. Standard BOA is able to find mostly one optimal solution at the end of the optimization process, when the whole population is saturated by phenotype-identical individuals.

We have implemented one variant of Pareto BOA algorithms (Pareto BOA), one variant of non-Pareto weighted sum BOA (WSO) and non-Pareto single BOA (SOP).

4.1 Single BOA with the normalization (SOP)

In this approach only one objective function $f_1(X)=1/(C1(X)+1)$ is used and the second objective function $f_2(X)=1/(C2(X)+1$ is replaced by normalization operator which modifies each individual to keep its balance in the considered bound. This operation can naturally change a partly the objective function f_1 of each individual. This effect may cause an extra genetic drift of the population.

4.2 Weighted-sum BOA (WSO)

In this approach the original vector-valued objective function is replaced by a scalar-valued objective function. The objective function of the individual X is computed as a weighted sum of all objective functions:

$$f(X) = w_1 f_1(X) + w_2 f_2(X), \quad (6)$$

where w_1, w_2 are weight coefficients. It is well known the sensitivity of the optimization process to these values. We have tested two sets of these coefficients. In WSO1 variant we have chosen $w_1=0.5$, $w_2=0.5$, in WSO2 couple of $w_1=0.005$, $w_2=0.995$ was used. These algorithms do not preserve Pareto-optimal solutions but provide mostly solutions from extremes of the Pareto front.

4.3 Pareto optimal BOA

The multiobjective optimization represents the difficult multimodal optimization problem which is mostly solved with niching methods that allow to preserve the diversity in the population of individuals/solutions. Our Pareto BOA algorithm is a modification of single BOA where we applied a promising niching techniques published in [4].
Although we solved bi-objective optimization, our algorithm is able to solve *m*-objective optimization problems. Our Pareto BOA algorithm can be described by the following steps:

Step 1: ***Initialization:*** Generate an initial population P_0 of size N randomly.

Step 2: ***Fitness assignment:*** Evaluate the initial population.

Step 3: ***Selection:*** Select the parent population as the best part of current population by 50% truncation selection.

Step 4: ***Model construction:*** Estimate the distribution of the selected parents using Bayesian network construction.

Step 5: ***Offspring generation:*** Generate new offspring (according to the distribution associated to the Bayesian network).

Step 6: ***Nondominated set detection and fitness assignment:*** Current population and offspring are joined, nondominated solutions are found, evaluated and stored at the top of the new population. Then dominated offspring and parents are evaluated separately.

Step 7: ***Replacement*:** The new population is completed by offspring and the best part of current population, so the worst individuals from current population are canceled to keep the size of the population constant.

Step 8: ***Termination:*** If maximum number of generations N_g is reached or stopping criterion is satisfied then the last Pareto front is presented, else go to Step 3.

The most important part of our Pareto algorithm is the procedure for detection of nondominated solution (current Pareto front) and sophisticated fitness calculation. The procedure for current nondominated and dominated set detection is described in following steps:

1. For each individual X in the population P compute vector of the objective functions
$$\overline{f}(X) = (f_1(X), f_2(X), \ldots, f_m(X)) \tag{7}$$
2. Detect subset of nondominated solutions
$$\overline{P} = \left\{X_j \mid X_j \in P, \,\overline{\exists} X_k \in P : \forall l \in \{1 \ldots m\} : f_l(X_k) > f_l(X_j)\right\} \tag{8}$$
Note: If two or more individuals have the same fitness vector $\overline{f}(X)$, then only one of them is accepted.
3. For each nondominated solution X_j compute its strength value as
$$s(X_j) = \frac{\left|\left\{X_k \mid X_k \in P, \,\forall l \in \{1 \ldots m\} : f_l(X_j) > f_l(X_k)\right\}\right|}{|P| + 1} \tag{9}$$
The fitness for nondominated solutions is equal to the reverse of the strength value $f'(X_j) = 1/s(X_j)$.

4. For each dominated solution X_i determine the fitness as

$$f'(X_i) = 1 \Big/ \left(1 + \sum_{X_j} s(X_j)\right), \tag{10}$$

where $X_j \in \overline{P}, \forall l \in \{1 \ldots m\}: f_l(X_j) > f_l(X_i)$. In the original approach [4] all individuals dominated by the same nondominated individuals have equal fitness. We proposed an extension by adding a term $c.r(X_i) \big/ (|P| + 1)$ into the denominator (10), where $r(X_i)$ is the number of individuals from P (not only from nondominated solutions) which dominate X_i and coefficient c is set to very small number, for example 0.0001. This term is used to distinguish the importance of individuals in the same "niche" (being dominated by the same nondominated solutions).

This type of fitness evaluation has the following advantages:

- For all nondominated individuals $f'(X_i) \geq 1$, for dominated individuals holds $f'(X_i) < 1$. If we use the replace-worst strategy, implicit Pareto elitism is included.
- Individuals from Pareto front dominated smaller set of individuals receive higher fitness, so the evolution is guided towards the less-explored search space.
- Individuals having more neighbours in their „niche" are more penalised due to the higher $s(X_j)$ value of associated nondominated solution.
- Individuals dominated by smaller number of nondominated individuals are more preferred.

5 Experimental results

5.1 Test graphs

The three types of graph structures are used [8]:

1. Hypergraphs representing real circuits labelled by *ICn.* The global optima is not known. The structure of circuits can be characterized as a random logic. The hypergraph IC67 consists of 67 nodes and 134 edges/nets, the IC116 consists of 116 nodes and 329 edges/nets.
2. Random geometric graph *Un.d.* on *n* vertices is placed in the unit square and its nodes coordinates are chosen uniformly. 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,* see Fig.1a.
3. Caterpillar graphs *CATk_n*, with *k* articulations, *(n-k)/k* legs for each articulation and *n* nodes, see Fig. 1b with *k=3* and *n=21.*

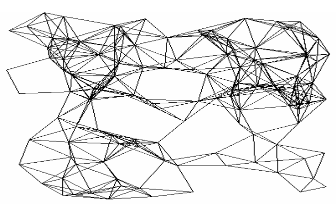

Fig.1a Geometric random graph

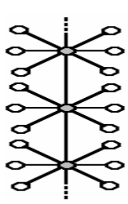

Fig.1b Caterpillar graph

5.2 Results of experiments

Let us notice that the objective space is visualized using the original cost function *C1, C2* instead of the objective functions f_1, f_2 (let us notice the cost functions *C1, C2* are minimized). Fig.2 shows the dynamics of the *IC67* bisection optimization. For the case of weighted sum algorithm the result fetched in 17-th generation of one run is shown. Population size is set to *N=2500*, the first population is generated randomly to keep the balance uniformly distributed in the range from 0 to 20% of *n*, where *n* is number of hypergraph nodes. The size of each point in the graph is proportional to the number of phenotypic equal solutions found in the current population. The current Pareto front are enlarged and pointed up.

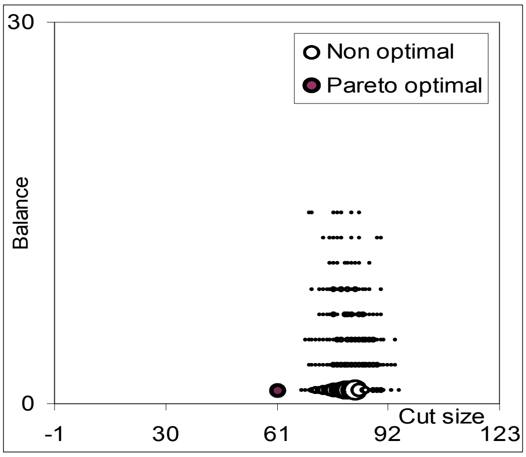

Fig.2a WSO1 algorithm

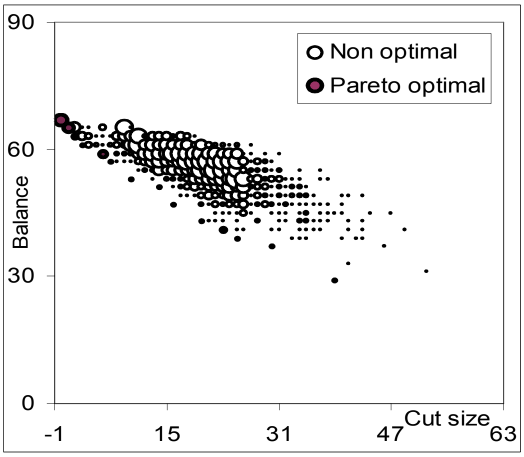

Fig 2b WSO2 algorithm

In Fig. 2a weighted sum algorithm WSO1 with w_1=0.5, w_2=0.5 is used, population is distributed with slight variability of balance. In Fig. 2b weighted sum algorithm WSO2 with w_1=0.995, w_2=0.005 is used, high balance of individuals is evident, the algorithm prefers more trivial solutions with minimum cut size and large balance. We used such values of w_1 and w_2, because even for w_1=0.99 and w_2=0.01 the algorithm still provides solution shown in Fig. 2a. In Fig. 3 the performance of Pareto, SOP and WSO algorithms for 3 types of graphs is shown.

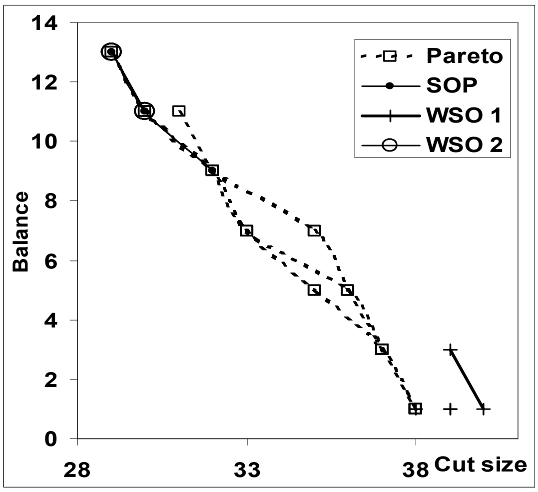

Fig.3a Bisection of *IC116*, *N=4000*

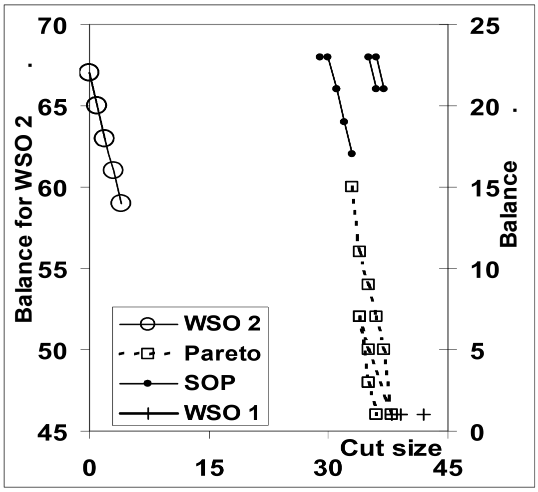

Fig.3b Bisection of *IC67*, *N=2500*

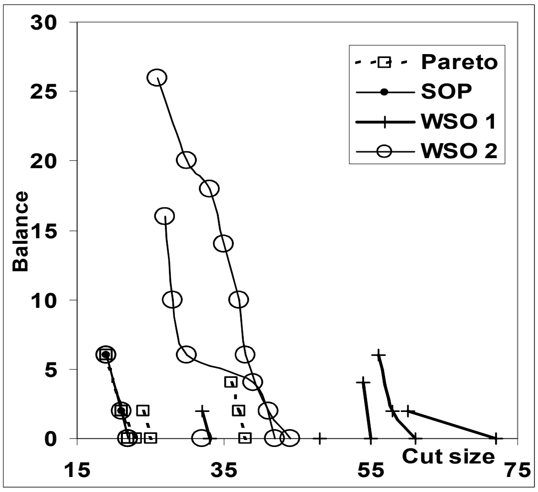

Fig.3c Bisection of *U120.5*, *N=4000*

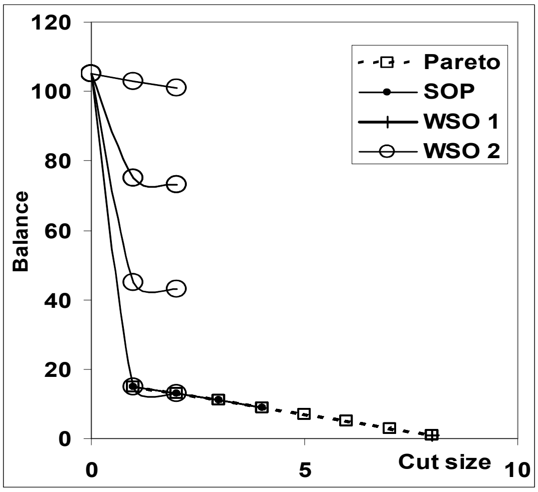

Fig.3d Bisection of *CAT7_105*, *N=3500*

The five independent runs of each algorithm were performed and five Pareto fronts from final populations are shown. For better visualization of the fronts from each run the points are connected by lines. Balance of the individuals in the initial population was uniformly distributed between zero and *0.2*n*, population size N was set proportional to *n*, the limit of the balance for normalization in SOP was set to maximum value *20%* of *n*. Maximum number of generations is set to $N_g = 200$.

From Fig. 3a it is evident that Pareto algorithm usually produces the largest Pareto set with good quality solutions, whereas SOP and WSO2 produce only solutions with low cut size but high balance and WSO1 produces solutions with low balance and higher cut value. From Fig. 3b the difficulty with WSO2 is evident - WSO2 produces trivial solutions with high balance. The SOP algorithm provides solution with small cut but high balance only. For example, solution with the cut-size=35 and balance=23 was obtained even if solution with lower balance for the same cut-size exists. The geometric graph *U120.5* seems to have many local optima. It is a hard benchmark as it is seen in Fig. 3c. Both WSO1 and WSO2 provide solutions far from optima in 4 runs from five runs. Only in one run the Pareto optimal solution was found. The SOP algorithm and Pareto BOA provide optimal fronts in most runs. The *CAT7_105* graph is an artificial graph known as a hard benchmark, the results are shown in Fig. 3d. The WSO2 produces mostly a trivial solution with high balance, the WSO1 only one solution with minimal balance and maximal cut. The Pareto BOA provides the whole Pareto front, SOP produces only individuals from the upper part of this front.

6 Parallel Pareto BOA

The establishment of current Pareto front in each generation for bi-criterial optimization takes *O(N* log N)* comparisons. The asymptotic time complexity of the proposed Pareto algorithm does not exceed the complexity of conventional BOA. The execution time of one generation for Pareto algorithm is nearly the same as for SOP or WSO, but the difference is in the number of generations used. In SOP and WSO algorithm there is an implicit detector of population saturation used to stop the evolution. In Pareto BOA the population is implicitly "split" into several niches and each of them converges to different solution which results in slower convergence. Because it is not simple to specify the stopping criterion, we often must specify maximum number of generations. The Pareto BOA wasted in our experiments five-times more generations than SOP and WSO. To decrease the wasting time, we suggest a parallel construction of Bayesian network as described in [9] for single criterion optimization. The next approach for the future work is the decomposition of the Pareto front into segments which can be constructed in separate but cooperating subpopulations.

7 Conclusions

We have implemented multiobjective Pareto BOA algorithm for the hypergraph bisectioning. The Pareto BOA performance was compared to single BOA with relaxed balance and weighted sum algorithms WSO. The WSO is very sensitive to type of problem see fig. 3a, 3b. The SOP algorithm provides mostly an upper part of Pareto front towards higher balance values. In the case of real hypergraphs IC67, IC116, the Pareto set is uniformly distributed along the Pareto front only in case of Pareto BOA. The main problem which remains to be solved is the large computation complexity and large population size. The parallelization of the Pareto BOA is necessary. The next possible improvement lies in more sophisticated niching technique, modification of replacement phase of the algorithm and introduction of problem knowledge into optimization process. The future work will be mainly directed towards the parallelization of BOA algorithm on the platform of SUN workstations, which will include the parallelization of Bayesian network construction and the decomposition of the Pareto front detection. Separate but cooperating subpopulations using the migration operator will be used.

This research has been carried out under the financial support of the Czech Ministry of Education – FRVŠ No. 0171/2001 "The parallelization of Bayesian Optimization Algorithm" and the Research intention No. CEZ: J22/98: 262200012 – "Research in information and control systems".

References

1. Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. *A Niched Pareto Genetic Algorithm for Multiobjective Optimization,* In Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, volume 1, pages 82-87, Piscataway, New Jersey, June 1994. IEEE Service Center.
2. Carlos Artemio Coello Coello. *An Empirical Study of Evolutionary Techniques for Multiobjective Optimization in Engineering Design.* PhD thesis, Department of Computer Science, Tulane University, New Orleans, LA, April 1996.
3. David S. Todd. *Multiple Criteria Genetic Algorithms in Engineering Design and Operation. PhD thesis,* University of Newcastle, Newcastle-upon-Tyne, UK, October 1997.
4. Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization*: Methods and Applications PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.
5. Joshua D. Knowles, David W. Corne, and Martin J. Oates. *The Pareto-Envelope based Selection Algorithm for Multiobjective Optimization,* In Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSNVI), pages 839-848, Berlin, September 2000. Springer.
6. Pelikan, M. *A Simple Implementation of Bayesian Optimization Algorithm in C++(Version1.0).* Illigal Report 99011, February 1999, pp. 1-16.
7. Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. *Linkage Problem, Distribution Estimation, and Bayesian Networks.* IlliGal Report No. 98013, November 1998, pp. 1-25.
8. Schwarz, Josef., Očenášek, Jiří. *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, Faculty of Mechanical Engineering, Brno, 1999, pp. 124-130, ISBN 80-214-1131-7.
9. Schwarz, Josef., Očenášek, Jiří. *The Parallel Bayesian Optimization Algorithm.* Proceedings of the European Symposium on Comutational Inteligence, Physica-Verlag, Košice, Slovak Republic, 2000, pp. 61-67, ISBN 3-7908-1322-2, ISSN 1615-3871.