Far East Journal of Mathematical Sciences (FJMS)
Volume…, Number…, 2013, Pages …
Available online at http://pphmj.com/journals/fjms.htm
Published by Pushpa Publishing House, Allahabad, INDIA

USING CHI-SQUARE MATRIX TO STRENGTHEN MULTI-OBJECTIVE EVOLUTIONARY ALGORITHM

Jiradej Ponsawat[*], Nachol Chaiyaratana, Chatchawit Aporntewan and Prabhas Chongstitvatana[*]

[*]Department of Computer Engineering
Chulalongkorn University
Thailand
e-mail: jiradej@kku.ac.th
prabhas@chula.ac.th

Department of Electrical and Computer Engineering
King Mongkut's University of Technology North Bangkok
Thailand
e-mail: nchl@kmitnb.ac.th

Department of Mathematics
Chulalongkorn University
Thailand
e-mail: Chatchawit.a@chula.ac.th

Abstract

Many complex engineering problems have multi-objectives where each objective is conflicting with others. However, a lot research

2010 Mathematics Subject Classification: **Kindly provide.**

Keywords and phrases: multi-objective problems, Building Block Identification, Genetic Algorithm.

Submitted by K. K. Azad

Received December 4, 2012

works in optimization by Competent Genetic Algorithm are focused on single objective methods. These algorithms work very well for single objective problems but stumble when trying to discover a large number of solutions naturally occurred in multi-objective problems. There are many multi-objective problems where solutions share common characteristic, for example decomposable multi-objective problems. This characteristic can be exploited to identify and compose these common structures. This work proposes to apply the concept of Building Blocks to improve evolutionary algorithms to tackle such problems. Building Block Identification algorithm is used to guide the crossover operator in order to maintain good Building Blocks and mix them effectively. The proposed method is evaluated by using Building Block Identification guided crossover in a well-known Genetic Algorithm to solve multiple-objective problems. The result shows that the proposed method is effective. Moreover, it obtains a good spread of solutions even when the Building Blocks are loosely encoded.

1. Introduction

In [1], Goldberg gives description that Building Blocks (BBs) are short, low order and highly fit schemata and that these BBs play important role in action of GA because they are sampled, recombined, and resampled to form strings of potentially high fitness. The class of problems called GA-deceptive are designed to mislead a Simple Genetic Algorithm or any hill-climber algorithm. For this class of problems, Building Block Identification [2] is shown to be a good solver.

There are many algorithms for identifying and composing Building Blocks in single objective problems where the target is to learn the structure of solution which has one global optima [3]. For many real world problems there are many conflicting objectives and there are many solutions. Therefore algorithms which work well for single objective problems usually fail to obtain the structure of solution for all solutions. There are some classes of problems in which solutions have common characteristic such as decomposable multi-objective problems where the objectives compete in different partitions of the problem decomposition. This work employs Chi-

square matrix to identify Building Blocks. It is found that incorporating this method into evolutionary process improved solutions faster than a standard Genetic Algorithm, especially for large problems.

For multiple-objective deceptive problems, applying Genetic Algorithm with single point crossover has limited success due to the disruptive effect of the crossover operator on the Building Blocks. Building Block Identification algorithm as its name implied partitions bit-positions into groups. These groups are regarded as Building Blocks (BBs). The knowledge of BBs can be used to prevent disruption of highly fit solutions from crossover operators [4]. When performing crossover, group of bits in the same BB should not be divided in order to preserve them.

This work proposed the application of Building Block Identification algorithm to solve multiple-objective problems. We focus on finding good Building Blocks in the context of multiple-objective problems. The multiple-objective problems are solved with evolutionary algorithms. The Building Block Identification algorithm is used to guide the crossover operator which will mix Building Blocks. The aim of this work is to find out whether the claim that Building Blocks are important to multiple-objective problems can be substantiated [5, 6].

The approach taken in this work is to modify a standard MOEA to use Building Block Identification in place of its original crossover operator. NSGA-II is chosen to be the evolutionary algorithm in our work. For the test problems, we employ three different deceptive problems [7, 8], which are problems with clearly defined BBs (see Section 4). There exist many works that use the trap function as the test problem, e.g., [9-13].

The structure of this paper is as follows: The next section gives the background on NSGA-II algorithm. Section 3 explained the proposed algorithm in details. The description of Building Block Identification algorithm and its application to modify the crossover operator are given. Section 4 gives description details of the test problems, the multiple-objective deceptive function. Section 5 gives the detail of the experiments and discusses the results. Finally, the conclusion is offered in Section 6.

2. Background

Evolutionary algorithms are popular methods to solve multiple-objective problems. There are two approaches to Building Blocks finding: either it is explicit method or implicit [5]. Messy Genetic Algorithm (mGA) and Linkage Learning GA (LLGA) [14] are some examples of works that are explicit BBs finding. In these algorithms, each bit is tagged with the position numbers so that they can be moved around without losing the meaning. The messy GA is later developed to Fast Messy Genetic Algorithm (FMGA) [15] and GEne Messy Genetic Algorithm (GEMGA) [16]. The latter is an early work that can find the optimal solution for $m \times k$ trap function. Bayesian Optimization Algorithm (BOA) using Bayesian network to model a population proposed by Pelikan et al. [17] is another one of explicit BBs finding. In a later version of BOA called hierarchical BOA (hBOA) [18], the l-vertex network is represented by l decision trees/graphs in order to avoid the exponentially growth of the number of conditional probabilities in the network. As the result, the latter models are more compact and applicable for problems having higher order of variable interaction.

In the context of multiple-objective problems, the Multiple-objective Bayesian Optimization Algorithm (mBOA) [19] is identical to BOA, except that the selection procedure which is replaced by the non-dominated sorting and selection mechanism of NSGA-II [20]. The NSGA-II is well known in Multiple-objective Evolutionary Algorithm (MOEA) group and there have been much interest in improving its quality, for example, in [21] and [22]. NSGA-II is considered to be a leader of MOEA. NSGA-II is an implicit BB builder rather than an explicit one. Non-dominated Sorting Genetic Algorithm II (NSGA-II), the extended NSGA [23], is one of the most popular Genetic Algorithms. In NSGA-II, the population is sorted according to the level of non-domination. The diversity among non-dominated solutions is maintained using a measure of density of solutions in the neighbourhood. NSGA-II is able to find much better widespread solutions and better convergence near the true Pareto-optimal front in most problems.

3. Algorithm

Building Blocks can be identified by computing Chi-square matrix then perform partitioning of bit positions using Partition Algorithm proposed in [24]. Each element of Chi-square matrix represents the degree of relation between each bit of selected population. Partition Algorithm groups bits which are highly related into BBs. This knowledge of BBs is used in the design of crossover operator. When performing crossover, all bits in the same partition will be moved together.

3.1. Chi-square matrix

To identify highly-related-group of bits, it is noted that their quantities are inversely related to randomness. The Chi-square matrix [24] is chosen for measuring randomness because computing the matrix is simple and fast.

3.2. Partitioning Algorithm

Partitioning Algorithm [2] will partition each bit into suitable blocks. When performing crossover, bits in the same partition must not be separated. The motivation is to put bit *i* and bit *j* into the same partition subset if Chi-square $(i,\ j)$ is high.

3.3. Crossover method

In the experiment, the crossover step is replaced by BB-wise crossover, shown in Figure 1. The original crossover operator is a two-point crossover method.

The selected solutions will go through crossover and mutation operations to become the new population. The crossover operator can exploit the knowledge of BBs by choosing appropriate cut points. The cut point should not separate bits in the same BB. To achieve this, a crossover mask is created for each partition. When parents exchange bits to create offspring, all bits in the same partition will be moved together.

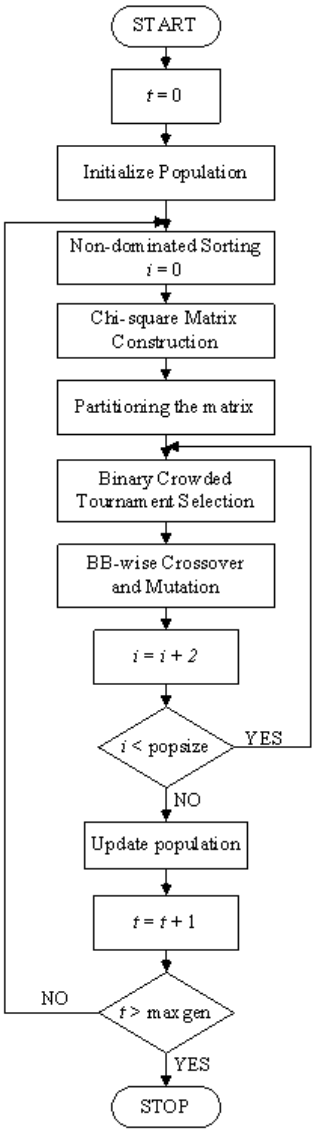

Figure 1. The steps of BB-NSGA-II.

The mask bit is generated for each partition. All bits in each partition can be either exchanged (with other individual) or remained the same. Flip coin method is used to choose whether a partition will be moved or remain unchanged. For instance, if the partition number 1 is assigned to be

exchanged (mask value 0), all bit-positions in that partition will be assigned 0 in the mask. After all partitions have been assigned, the partitions which are assigned to 0 will be swapped with their mates. Otherwise, they remain the same. The following example illustrates the operation:

If the Chi-square matrix be partitioned like this one, {{1, 5, 9, 13, 17}, {2, 6, 10, 14, 18}, {3, 7, 11, 15, 19}, {4, 8, 12, 16, 20}} and the partitions 1 and 2 are selected to be exchanged then all bits in the partitions 1 and 2 are exchanged between two parents. Here is the situation.

Partition <1234 1234 1234 1234 1234>

Mask Bits <0011 0011 0011 0011 0011>

x	x	x	x	x	x	x	x	…	x	x	x	x

Parent 1

y	y	y	y	y	y	y	y	…	y	y	y	y

Parent 2

After crossover, the two parents produce two children.

y	y	x	x	y	y	x	x	…	y	y	x	x

Child 1

x	x	y	y	x	x	y	y	…	x	x	y	y

Child 2

4. Multi-objective Deceptive Problems

The experiment is set to find out the effectiveness of the application of Building Block Identification to the multiple-objective trap problems. This class of functions usually leads the GA away from the global optimum. NSGA-II is used as the evolutionary algorithm. Its crossover operator is replaced by the crossover operator with Building Block Identification. The experiment is set to compare the result of the proposed method (named BB-

NSGA-II) with the original NSGA-II. Any difference in the results should arise mainly from the use of Building Block Identification.

4.1. Deceptive functions

The algorithm is tested on six test functions:

- T1-Multiple interleaved minimal deceptive problem
- T2-Complement of T1
- T3-Multiple interleaved 5-bit trap
- T4-Complement of T3
- T5-Multiple interleaved symmetric 5-bit trap
- T6-Complement of T5

Each of the function is described in the remainder of this section. The test functions are difficult in four aspects: deception [25], loose linkage, multimodality [26], [27] and combinatorial with large search space. These test functions were combined together to make three multi-objective problems:

(1) MOP1 (T1 and T2)

(2) MOP2 (T3 and T4)

(3) MOP3 (T5 and T6)

The multi-objective problems are listed in Table 1.

Table 1. Listing multi-objective problems

Problems	Description	Problem size	#Distinct points in Pareto optimal front	#Distinct solutions in Pareto optimal front
MOP1	T1 & T2 (MDP)	30	16	2^{15}
		60	31	2^{30}
		90	46	2^{45}

MOP2	T3 & T4 (Trap 5)	30	7	2^6
		60	13	2^{12}
		90	19	2^{18}
MOP3	T5 & T6 (Symmetric-Trap 5)	30	7	2^6
		60	13	2^{12}
		90	19	2^{18}

4.2. MOP1 - Interleaved minimal deceptive problem (T1 & T2)

The interleaved minimal deceptive problems are designed to test an algorithm's ability to discover loosely linked bits by dividing the string in to two halves and coupling one bit from each half. Figure 2 illustrates how the bits are correlated. Bits having the same pattern are rewarded, while alternating couplets are not. Additionally, equations (1) and (2) indicate the bit couplet fitness for T1 and T2,

$$f_{T1}(x) = \begin{cases} 0.9 & \text{if } u = 0, \\ 0 & \text{if } u = 1, \\ 1 & \text{if } u = 2, \end{cases} \tag{1}$$

where u is the number of 1s in the input string of two bits.

$$f_{T2}(x) = \begin{cases} 1 & \text{if } u = 0, \\ 0 & \text{if } u = 1, \\ 0.9 & \text{if } u = 2, \end{cases} \tag{2}$$

where u is the number of 1s in the input string of two bits.

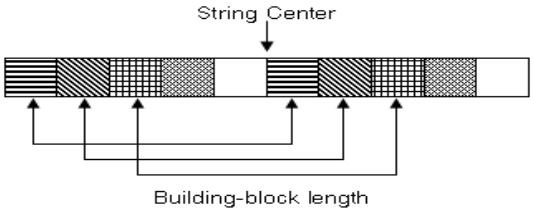

Figure 2. The linkage in T1 and T2 in problem MOP1.

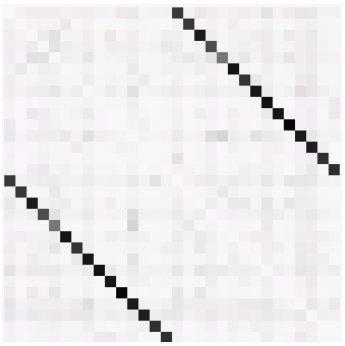

Figure 3. The example of identified linkage in MOP1.

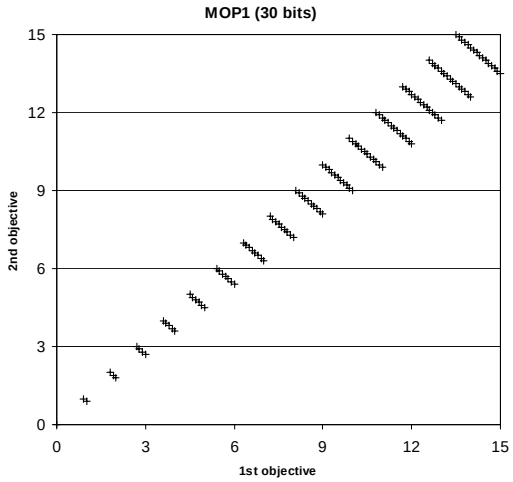

Figure 4. Objective space of MOP1 (30 bits).

4.3. MOP2 - Interleaved deceptive problem (T3 & T4)

The interleaved 5-bit trap function is devised to test an algorithm's ability to find loose linkages having non-consecutive bits. Bits in problems T3 and T4 both have correlated bits with a distance of l/5 from one another. Figure 5 illustrates how the bits in groups of five are coupled. Additionally, Figure 6 graphically illustrates how the fitness behavior varies according to the number of bits that are set in the described 5-bit linkage pattern.

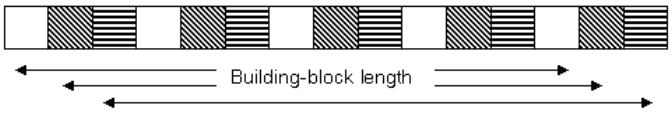

Figure 5. Coupling of the bits in MOP2 of T3 and T4.

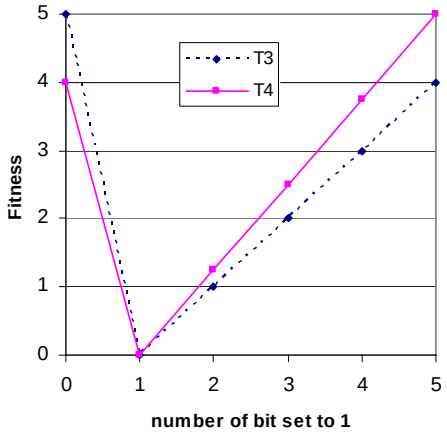

Figure 6. The fitness of deceptive problems T3 and T4.

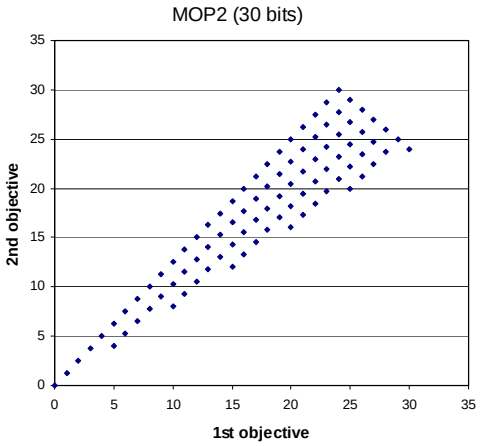

Figure 7. Objective space of MOP2 (30 bits).

4.4. MOP3 - Interleaved deceptive problem (T5 & T6)

Trap functions are difficult problems for Genetic Algorithms. Moreover, they are problems which Building Blocks are obviously defined. The

deceptive trap functions are modified to be multiple-objective style in [8]. A modified version called Shuffle trap function is also created. This function creates non-compact Building Blocks (bit positions are not contiguous) which renders a simple crossover operator ineffective. These problems are defined in this section.

This problem has 2 objectives: $m \times k$ deceptive trap, and $m \times k$ deceptive inverse trap. String positions are first divided into disjoint subsets or partitions of k bits each. The k-bit trap and inverse trap are defined as follows:

$$\text{trap}_k(u) = \begin{cases} k; & \text{if } u = k, \\ (k-d)\left[1 - \dfrac{u}{k-1}\right]; & \text{otherwise,} \end{cases} \tag{3}$$

$$\text{invtrap}_k(u) = \begin{cases} k; & \text{if } u = 0, \\ (k-d)\left[\dfrac{u-1}{k-1}\right]; & \text{otherwise,} \end{cases} \tag{4}$$

where u is the number of 1s in the input string of k bits, and d is the signal difference. Here, we use $k = 5$, and $d = 1$. The $m \times k$-trap conflicts with the inverse trap by its objective. A solution that sets the bits in its partition either to 0s or 1s is Pareto optimal and there are a total of 2^m solutions in the Pareto-optimal front.

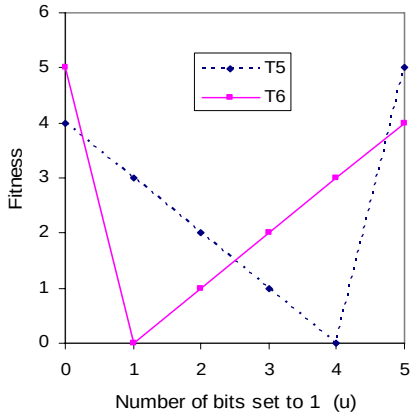

Figure 8. The fitness of deceptive problems T5 and T6.

$m \times k$ -trap function is defined as follows:

$$F_{m\times k}(B_0 \cdots B_{m-1}) = \sum_{i=0}^{m-1} F_k(B_i), \quad B_i \in \{0, 1\}^k, \tag{5}$$

where F_k is trap_k or invtrap_k function. The *m* and *k* are varied to produce many test functions. These functions are often referred to as Additively Decomposable Functions (ADFs). In this experiment, this function is modified to be multi-objective.

The shuffle trap function is constructed by separating the bit position of the same Building Blocks in order to deceive the algorithm. For instance, normal 4×5 -trap function has Building Blocks as shown.

11111 ***** ***** *****

In shuffle trap, the modulo method is used to construct one Building Block. The same Building Block is repeated in every *m* bit.

1*** 1*** 1*** 1*** 1***

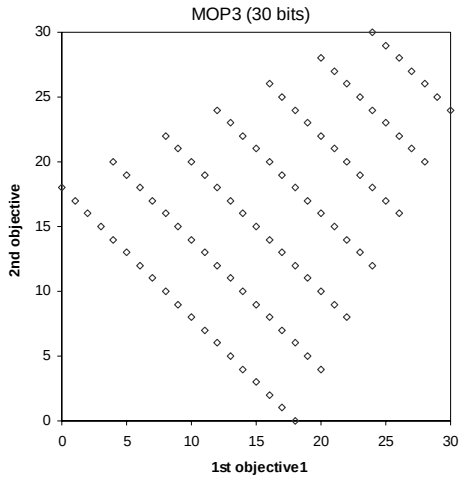

Figure 9. Objective space of MOP3 (30 bits).

5. Result and Discussion

The data in the experiments are the average from 30 independent runs. Table 2 shows the experiment settings for both NSGA-II and BB-NSGA-II. They are compared at the same effort.

Table 2. The experimental setting for both algorithms

Problem	Problem size	Population size	Number of generation	Number of evaluations
MOP1	30	300	40	12000
	60	500	60	30000
	90	700	80	56000
MOP2	30	1300	40	52000
	60	3300	60	198000
	90	5200	80	416000
MOP3	30	1300	40	52000
	60	3300	80	264000
	90	6000	120	720000

The parameters for GA in all problems are: crossover rate 0.9, mutation rate 0.01, the value “alpha” of PAR algorithm 0.95. All problems employ same set of parameters without any further tuning.

5.1. Result of MOP1 (T1 & T2)

Table 3. Results of the experiment on MOP1

Evaluation	MOP1’s size	Algorithms	
		NSGA-II	BB-NSGA-II
Unique strings found	30	10.20	9.00
	60	12.53	17.57
	90	15.93	19.97
Pareto front points found	30	7.00	6.57
	60	6.17	8.60
	90	6.40	10.03

From Table 3, for the problem of size 30 bits, NSGA-II performed better than the proposed method. However, when the problem is larger, the length

of Building Blocks is wider; Building Block Identification becomes more prominent. For the large problems such 60 and 90 bits, the 2-point crossover employed in NSGA-II is outperformed by BB-wise crossover in the proposed method.

These test functions, MOP1, are the minimum deceptive problems. They are the easiest problems in the experiment. The 2-point crossover performed moderately well in the small problems. When the distance between the bits in the same Building Block is increased, the problem becomes harder. In this situation, the BB-wise crossover performed better than the 2-point crossover.

The half lower row of Table 3 shows the average number of solutions found in the Pareto front. This number indicates how well a method discovered solutions in the Pareto front. For NSGA-II, as the problem size grows larger, the number of solutions is reduced. This is in contrast to the proposed method in which more solutions are found in the larger problems.

5.2. Result of MOP2 (T3 & T4)

Table 4. Results of the experiment on MOP2

Evaluation	MOP2's size	Algorithms	
		NSGA-II	BB-NSGA-II
Unique strings found	30	1.80	2.80
	60	1.03	3.20
	90	1.00	2.80
Pareto front points found	30	1.77	2.50
	60	1.03	2.50
	90	1.00	2.43

These problems are harder than MOP1. NSGA-II is outperformed by BB-wise-NSGA-II in all problems. For these problems, the solutions in the Pareto front have good spread over the objective space (Figure 7). This makes it hard for NSGA-II which collected all good solutions from the Pareto front before considering their distribution. NSGA-II employs a measure of distribution using crowding distance to select solutions so that they have good distribution in the objective space. However, for the discrete

domain, this measure cannot maintain the diversity of the solutions due to the limited number of solutions that can be stored.

The results of this experiment show that BB-wise crossover is better in maintaining the diverse solutions.

5.3. Result of MOP3 (T5 & T6)

These problems have similar Building Blocks to the problems in MOP2. The difference is in the symmetry of the solutions in the objective space. The distribution of solutions in the objective space is also different (compare Figure 7 to Figure 9). The distribution of solutions in the Pareto front of MOP3 is smaller in the beginning. This fact allows NSGA-II to perform well compared to the results from the previous experiments. Table 5 shows the results of the experiment of MOP3.

The results show similar trend to MOP1. That is, for the small problems NSGA-II performed better than BB-wise-NSGA-II. For the large problems the proposed method is better. This is because when the length of Building Blocks increased, the BB-wise crossover has a better chance in avoiding disruption of Building Blocks than the 2-point crossover. The problems are set up such that whenever one objective pulls the solutions towards all 1s then other objective pulls the solutions towards all 0s. In this situation, the 2-point crossover will exchange the Building Blocks 0 with the Building Blocks 1, whereas the BB-wise crossover will exchange the Building Blocks of the same type.

Table 5. Results of the experiment on MOP3

Evaluation	MOP3's size	Algorithms	
		NSGA-II	BB-NSGA-II
Unique strings found	30	4.83	2.70
	60	2.67	3.53
	90	2.73	5.17
Pareto front points found	30	3.67	2.37
	60	2.67	3.00
	90	2.70	4.00

Relying on Building Block composition alone required that there must be sufficient amount of Building Blocks in the population [28]. For the small problems, the size of population is also small. Therefore, there is a possibility that there is not enough Building Blocks existed in the population to allow convergent to the solutions.

6. Conclusion

This paper discussed the effect of using Building Block Identification (BBI) in solving decomposable multi-objective problems where the objectives compete in different partitions of the problem decomposition. The Building Block Identification composed of the calculation of Chi-square matrix and the partitioning. The partitions are used to guide the crossover operation to mix Building Blocks properly that will lead to good solutions. Compare the original NSGA-II to BB-NSGA-II which incorporated Building Block Identification, the result shows BBI method is effective in solving the multi-objective trap functions which appear to have high epistasis. From the experiment, the proposed method performs more effectively than NSGA-II, even though NSGA-II is designed for multi-objective problems. The experiment with the shuffle trap function demonstrates clearly that composing Building Blocks is highly effective.

Several interesting topics regarding the Building Block Identification require further exploration. We would apply our algorithm to other multi-objective problems such as some interesting problems multi-objective travelling salesperson problems [29] and another real-world problems, e.g., in [30, 31].

References

[1] D. E. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning, Addison Wesley, Reading, MA, 1989.

[2] C. Aporntewan and P. Chongstitvatana, Building-block identification by simultaneity matrix, Soft Computing 11 (2007), 541-548.

[3] Y.-P. Chen, T.-L. Yu, K. Sastry and D. E. Goldberg, A survey of linkage learning

techniques in genetic and evolutionary algorithms, IlliGAL Report No. 2007014, University of Illinois at Urbana-Champaign, Urbana, IL, 2007.

[4] M. Tsuji and M. Munetomo, Linkage analysis in genetic algorithms, Computational Intelligence Paradigms: Innovative Applications, Studies in Computational Intelligence, 137, L. C. Jain, M. Sato-Ilic, M. Virvou, G. A. Tsihrintzis, V. E. Balas and C. Abeynayake, eds., pp. 251-279, Springer, 2008.

[5] R. O. Day and G. B. Lamont, An effective explicit building block MOEA, the MOMGA-IIa, 2005 IEEE Congress on Evolutionary Computation (CEC'2005), Vol. 1, pp. 17-24, Edinburgh, Scotland, September 2005.

[6] C. A. Coello Coello, D. A. Van Veldhuizen and G. B. Lamont, Evolutionary Algorithms for Solving Multi-objective Problems, Kluwer Academic Publishers, New York, 2002.

[7] N. Khan, D. E. Goldberg and M. Pelikan, Multi-objective Bayesian optimization algorithm, Technical Report 2002009, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, Illinois, 2002.

[8] K. Sastry, M. Pelikan and D. E. Goldberg, Limits of scalability of multiobjective estimation of distribution algorithms, Proc. of the Congress on Evolutionary Computation, 3, 2005, pp. 2217-2224.

[9] S. Rimcharoen, D. Sutivong and P. Chongstitvatana, Real options approach to evaluating genetic algorithms, Applied Soft Computing 9(3) (2009), 896-905.

[10] C. Aporntewan and P. Chongstitvatana, A quantitative approach for validating the building block hypothesis, IEEE Congress of Evolutionary Computation, Edinburgh, September 2-5, 2005.

[11] C. Aporntewan and P. Chongstitvatana, Simultaneity matrix for solving hierarchically decomposable functions, Proc. of the Genetic and Evolutionary Computation (GECCO 2004), Lecture Notes in Computer Science, pp. 877-888, Seattle, WA, USA, June 26-30, 2004.

[12] M. Clergue and P. Collard, GA-hard functions built by combination of trap functions, Proc. of the 2002 Congress on Evolutionary Computation, May 12-17, 2002.

[13] S. Nijssen and T. Back, An analysis of the behavior of simplified evolutionary algorithms on trap functions, IEEE Trans. Evolutionary Computation 7(1) (2003), 11-22.

[14] G. R. Harik, Learning linkage, Foundation of Genetic Algorithms 4, Morgan Kaufmann, San Francisco, 1997, pp. 247-262.

[15] D. E. Goldberg, K. Deb, H. Kargupta and G. Harik, Rapid, accurate optimization of difficult problems using fast messy genetic algorithms, Proc. of the 5th International Conference on Genetic Algorithm, 1993, pp. 56-64.

[16] H. Kargupta, The gene expression messy genetic algorithm, Proc. of Congress on Evolutionary Computation, 1996, 814-819.

[17] M. Pelikan, D. E. Goldberg and E. Cantú-Paz, The Bayesian optimization algorithm, Proc. of Genetic Algorithm and Evolutionary Computation Conference, 1999.

[18] M. Pelikan and D. E. Goldberg, Escaping hierarchical traps with competent genetic algorithms, Technical Report 2001003, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana Champaign, **Year**.

[19] M. Laumanns and J. Ocenasek, Bayesian optimization algorithms for multi-objective optimization, Proc. of 7th Int. Conf. Parallel Problem Solving from Nature (PPSN VII), Granada, Spain, September 7-11, 2002.

[20] K. Deb, A. Pratap, S. Agrawal and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evolutionary Computation 6(2) (2002), 182-197.

[21] F. G. Lobo and C. F. Lima, Revisiting evolutionary algorithms with on-the-fly population size adjustment, Proc. of the Genetic and Evolutionary Computation (GECCO 2006), Seattle, Washington, USA, July 8-12, 2006, pp. 1241-1248.

[22] K. D. Tran, Elitist non-dominated sorting GA-II (NSGA-II) as a parameter-less multi-objective genetic algorithm, Proc. of the SIGCHI Conference on Human Factors, IEEE, SoutheastCon, April 8-10, 2005, pp. 359-367.

[23] N. Srinivas and K. Deb, Multiple objective optimization using nondominated sorting in genetic algorithms, Evolutionary Computation 2 (1994), 221-248.

[24] C. Aporntewan and P. Chongstitvatana, Chi-square matrix: an approach for building-block identification, Proc. of 9th Asian Computing Science Conference, December 8-10, 2004, pp. 63-77.

[25] K. Deb and D. E. Goldberg, Analyzing deception in trap functions, Foundations of Genetic Algorithms 2 (1993), 93-108.

[26] D. E. Goldberg, K. Deb and J. Horn, Massive multimodality, deception, and genetic algorithms, Parallel Problem Solving from Nature 2 (1992), 37-46.

[27] K. Deb, J. Horn and D. E. Goldberg, Multimodal deceptive functions, Technical Report, IlliGAL Report No 92003, University of Illinois, Urbana, **Year**.

[28] D. E. Goldberg, The Design of Innovation: Lessons from and for Competent Genetic Algorithms, Kluwer Academic Publishers, Boston, MA, 2002.

[29] E. Zitzler, Evolutionary algorithms for multiobjective optimization: methods applications, Ph.D. Thesis, Shaker Verlag, Aachen, Germany, 1999.

[30] Z. Karakehayov, Multiobjective design of wireless ad hoc networks: security, real-time and lifetime, J. Telecom. Inform. Tech. 2 (2009), 13-21.

[31] R. J. Olsson, Z. Kapelan and D. A. Savic, Probabilistic building block identification for optimal design and rehabilitation of water distribution systems, J. Hydroinformatics 11(2) (2009), 89-105.

Paper No. PPH-1212007-MS Kindly return the proof after correction to: *The Publication Manager* *Pushpa Publishing House* *Vijaya Niwas* 198, *Mumfordganj* *Allahabad*-211002 (*India*) along with the print charges* by the <u>fastest mail</u> ***Invoice attached**	Proof read by: …… Copyright transferred to the Pushpa Publishing House Signature: ……..…… Date: ……..…… Tel: ……..…… Fax: ……..…… e-mail: ……..……..…… Number of additional reprints required …… Cost of a set of 25 copies of additional reprints @ Euro 12.00 per page. (25 copies of reprints are provided to the corresponding author ex-gratis)