Normalization in Genetic Algorithms

Sung-Soon Choi and Byung-Ro Moon

School of Computer Science and Engineering
Seoul National University
Seoul, 151-742 Korea
{sschoi,moon}@soar.snu.ac.kr

Abstract. Normalization is an approach that transforms the genotype of one parent to be consistent with that of the other parent. It is a method for alleviating difficulties caused by redundant encodings in genetic algorithms. We show that normalization plays a role of reducing the search space to another one of less size. We provide insight into normalization through theoretical arguments, performance tests, and examination of fitness-distance correlations.

1 Introduction

In a genetic algorithm, solutions are encoded into chromosomes by an encoding scheme, and the chromosomes are handled by the genetic operators. A fitness function plays a role as a bridge that connects chromosomes and solutions in that it evaluates the qualities of solutions by decoding chromosomes. By the analogy with biology, chromosomes and solutions are called genotypes and phenotypes, respectively.

An encoding scheme determines the relation between genotype space and phenotype space. According to the relations between the two spaces, encoding schemes are classified into three classes. First, one phenotype is represented by one genotype. In this case, the genotype space is one of the minimal spaces that represent the phenotype space without any loss of information. This is in general considered as the most desirable [1] [2]. Representative examples are real-valued encodings in function optimization, function approximation, and so on [3] [4].

Next, several phenotypes are represented by one genotype. Such an encoding scheme can be used to reduce the size of the search space by considering only the important parameters related to a problem. It seems to be suitable for hybrid genetic algorithms in which more important parameters are treated by the genetic process and less important ones are fine-tuned by local optimization methods. In most practical problems, however, various parameters are so intertwined that it is another difficult problem to select more important parameters [5].

Finally, one phenotype is represented by several genotypes. Such encoding schemes are called *redundant encodings*. There are quite a few problems in which it is difficult to represent one phenotype by one genotype using traditional encoding schemes due to the characteristics of the problems. Redundant encodings are still being used in most of those problems.

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2723, pp. 862–873, 2003.

There are two representative groups of problems belonging to the third class. One is the group of *grouping problems* [2]. These are commonly concerned with partitioning a given item set into mutually disjoint subsets. Examples belonging to this group include k-way graph partitioning [6] [7] [8], graph coloring, bin packing, and workshop layouting [2]. The other is the group of the structural optimization problems. These are concerned with finding the optimal structures satisfying given constraints. In the problems, there are often several other solutions structurally equivalent to a solution. Example problems include sorting network optimization [9], neural network optimization [10], and RNA structure prediction [11].

Recently, there were discussions about the benefits of redundancy in encoding in evolutionary algorithms [12] [13] [14]. In particular, Shipman, Shackleton, and Ebner [12] [13] showed that some redundant encodings are useful to help mutation-based hill-climbing search by constructing many different neutral paths in the genotype space that enable moves at the same fitness level. However, the tackled problems were of small sizes and only mutation operators were considered. As mentioned in the next section, in the cases that the problems are of large sizes and highly epistatic, redundant encodings may lead to severe losses in the search powers of traditional crossovers (such as 1-point, multi-point, and uniform) and accordingly of genetic algorithms with them as well.

In order to alleviate the problems caused by redundant encodings, a number of approaches were proposed. In some problems, adaptive crossovers were developed that recombine parents in terms of phenotypes [6] [15] [16]. Van Hoyweghen *et al.* [17] pointed out problems in a certain class of redundant encodings and presented some solutions to them. *Normalization* transforms the genotype of a parent to another genotype to be consistent with the other parent. There were a few successful studies that used normalization [7] [8] [9].

Among those approaches, we focus on normalization. We review the normalization methods used in a few problems and state their merits associated with search space reduction. Finally, we support its validity with experimental results.

The rest of this paper is organized as follows. In Section 2, we argue that redundant encodings lead to severe losses of search power in genetic algorithms, in particular, with respect to traditional crossovers. In Section 3, we describe normalization as a method of alleviating the problems by redundant encodings and show that the normalization plays a role of reducing the search space to another space of less size. In Section 4, we review the redundant encodings and normalization methods used in some problems, and examine the effects of normalization on the search spaces by experiments. Finally, we make our conclusions in Section 5.

2 Drawbacks of Redundant Encoding

In general, *respectfulness* and *combination* are considered to be two major features that crossovers have to possess. Respectfulness means the property that the alleles common to the parents are transmitted to offsprings. The concept

of respectfulness stands on the basis of the commonality hypothesis: "schemata common to above-average solutions are above average" [18]. Many papers referred to the importance of respectfulness in the crossover design [1] [18] [19] [20]. Combination means the property of producing good schemata of higher orders by recombining good schemata of low orders in parents. The concept of combination is based on the building block hypothesis [21] and has received heavy and repeated emphasis in the literature [1] [19] [22] [23].

A redundant encoding represents an optimal phenotype by multiple genotypes and thus the genotype space becomes multimodal [21]. The difficulty of multimodal problems with respect to genetic algorithms has been pointed out in the literature [21] [24] [25]. Similarly to the optimal phenotypes, high-quality schemata are redundantly represented by the encoding: a phenotype schema in an above-average phenotype is represented by multiple genotype schemata that are equivalent in terms of phenotype.[1] On the other hand, traditional crossovers such as 1-point crossover and uniform crossover operate on the genotype space regardless of phenotypes. The redundant encoding is harmful to those crossovers in both aspects of respectfulness and combination, as described below.

- **Respectfulness:** Most traditional crossovers are basically respectful in the genotype space (e.g., 1-point, multi-point, and uniform crossovers). However, they are not respectful in terms of phenotypes. Those crossovers do not preserve "equivalent" genotype schemata of high-quality in parents if they have different representations. The probability that a common phenotype schema in parents is transmitted to the offspring decreases as the schema has more different representations in the genotype space.
- **Combination:** In highly epistatic problems,[2] a low-order schema in a genotype may have a positive or negative effect on the quality of the genotype, depending on the alleles on the other loci of the genotype, i.e., the genotype context [26]. Traditional crossovers recombine schemata in parents regardless of the genotype context. The change in the context may diminish the longevity of high-quality schemata with respect to crossover. For this reason, it is difficult to build desirable schemata of higher orders by juxtaposition. As a phenotype schema has more different genotypes, the probability that the schema emerges decreases.

Thus, in the problems of large sizes and high epistases, it is hard for genetic algorithms with traditional crossovers to cope with the genotype spaces expanded by redundant encodings. Traditional crossovers cannot handle phenotype schemata effectively. Normalization methods transform a given parent to another genotype so that the genotype contexts of the parents are as similar as possible in crossover. As mentioned in the next section, these methods help

[1] A phenotype schema in a phenotype is implicitly defined to be a partial property contained in the phenotype.

[2] Most of the problems mentioned in Section 1, in which redundant encodings are used, belong to this class.

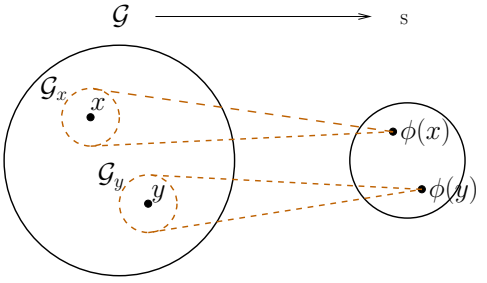


Fig. 1. Genotype-phenotype space relationship

traditional crossovers overcome the problems related to respectfulness and combination. They play a role of reducing the search space by considering multiple genotypes corresponding to a phenotype as a conceptual group.

3 Normalization and Space Reduction

Given the phenotype space $\mathcal{S}$ for a problem, a redundant encoding determines the genotype space $\mathcal{G}$ and a function $\phi : \mathcal{G} \longrightarrow \mathcal{S}$. For a genotype $x \in \mathcal{G}$, consider the subset $\mathcal{G}_x$ of $\mathcal{G}$ as follows:

$$\mathcal{G}_x = \{u \in \mathcal{G} | \phi(u) = \phi(x)\}. \tag{1}$$

$\mathcal{G}_x$ corresponds to the set of genotypes that share the phenotype of x. We call $\mathcal{G}_x$ the coset (neutral set) of x. The above definition of cosets gives an equivalence relation among the genotypes and $\mathcal{G}$ is partitioned by these cosets. We denote the set of cosets by $\bar{\mathcal{G}}$, and then $\bar{\mathcal{G}}$ and $\mathcal{S}$ are in one-to-one correspondence (Figure 1).

As mentioned before, it is necessary to recombine parents in a phenotype level in order to alleviate the problems of redundant encodings. Then, given parents $x, y \in \mathcal{G}$, how do we recombine the corresponding phenotypes, $\phi(x)$ and $\phi(y)$? The essence of normalization is to view the genotypes in a coset as a phenotype. Such a view allows that x is substituted by an element in $\mathcal{G}_x$ that is the most consistent to y. That is, given $\phi(x)$ and $\phi(y)$, normalization operators play a role of selecting the most appropriate $x' \in \mathcal{G}_x$ and returning it to the crossover operator so that high-quality schemata are well preserved and combined.

More formally, suppose a distance function $d : \mathcal{G} \times \mathcal{G} \longrightarrow \mathbb{R}$ and a consistency measure $s : \mathcal{G} \times \mathcal{G} \longrightarrow \mathbb{R}$ defined on genotype pairs ($\mathbb{R}$ denotes the set of real numbers). Given parents $x, y \in \mathcal{G}$, the normalization operator transforms x to $x' \in \mathcal{G}_x$ so that $s(\cdot, y)$ is maximized. Then the crossover recombine the two genotypes x' and y. In other words, the distances between genotypes are redefined by normalization and consequently the landscape of search space alters. Since $\bar{\mathcal{G}}$ and $\mathcal{S}$ are in one-to-one correspondence as mentioned, the genotype space is reduced to the phenotype space by normalization from the viewpoint of crossover (Figure 2).

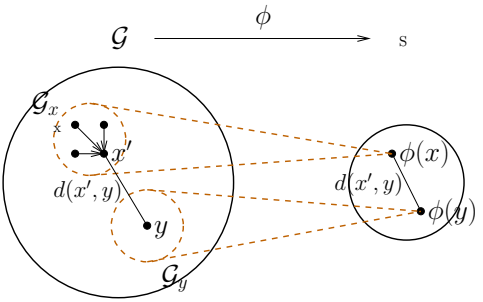


Fig. 2. Space reduction

Space reduction by normalization modifies the structure of the landscape of search space. Thus, it changes the difficulty of problem with respect to genetic algorithms and so affects the performances of the genetic algorithms. The change in the difficulty has much to do with the consistency measure s, and selecting a suitable measure s is crucial in the normalization operator design. It suggests that normalization is more than simply decreasing the genotype distance (e.g., Hamming distance) between parents.

To get insight into the effect of normalization on search space, we consider fitness distance correlation (FDC) [27], which is a popular measure for predicting problem difficulty [28] [29], in the next section. As seen in the section, normalization leads to considerable increases of FDC values.

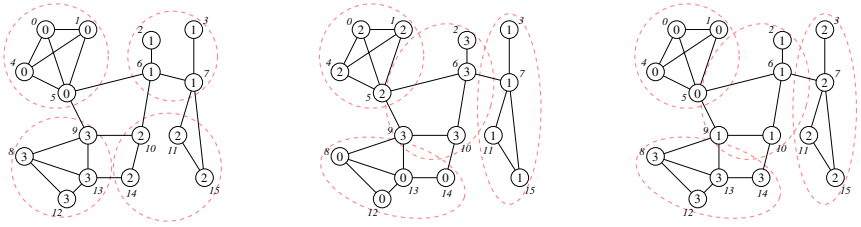
Normalization gives rise to the moves at the same level of fitness in the genotype space, so called, the neutral walks [30] by replacing parents with the genotypes in their cosets. It is related to the previous studies in the aspect of using the neutral paths created by redundancy [12] [13]. However, normalization attempts to alleviate the negative effects of neutral paths.

4 Normalization Examples

In this section, we review the normalization methods used in two problems. In addition, we consider the effects of normalization on space reduction and analyze the landscapes of the search spaces before and after normalization. Experimental results with respect to normalization are also provided. The problems that we consider in this section are the graph partitioning problem and the sorting network problem, which belong to the groups of grouping problems and structural optimization problems, respectively.

4.1 Graph Partitioning Problem

Given a graph $G = (V, E)$, where V represents the set of vertices and E represents the set of edges, k-way partitioning is grouping the vertex set V into k



(a) Parent 1 (b) Parent 2 (c) Normalized parent 2

Fig. 3. A normalization example in 4-way graph partitioning

disjoint subsets. In particular, two-way partition is called bisection or bipartition. In a k-way partition, the total number of edges whose endpoints belong to different subsets is called cut size. A k-way partition is said to be balanced if the difference of cardinalities between the largest and the smallest subsets is at most one. The k-way partitioning problem in this section is to find a balanced k-way partition with minimal cut size.

In the k-way partitioning problem, the k-ary encoding scheme, in which k subsets are represented by the integers from 0 to $k-1$, has been generally used [6] [7] [8] [15] [31]. In this case, a phenotype (k-way partition) is represented by $k!$ different genotypes according to the numbering of the subsets.

In the k-way partitioning problem, a normalization method was used in [8], which is based on the adaptive crossovers proposed in [6] and [15]. The method can be described as follows. First, it selects a partition from each parents so that they contain the most vertices in common, and then it assigns the partitions a common number. It repeats this process with the other partitions and remaining partition numbers. Figure 3 shows an example of normalization. The parent 2 was transformed to be as consistent as possible to the parent 1. Figure 4 shows an example crossover operator with and without normalization. The same parents as in Figure 3 were used. We can see that the offspring was too much perturbed without normalization; good characteristics of the parents were inherited little to the offspring.

Tables 1 and 2 show the experimental results of the cases without normalization (Ordinary) and with normalization (Normalization) for the bipartitioning problem and the 32-way partitioning problem, respectively. The experiments were performed for the instances from Johnson's benchmark [7]. In the tables, Ed and ρ indicate the average hamming distance and the fitness distance correlation coefficient (FDC) value [27], respectively, of 10,000 local optima. To get the local optima, the Kernighan-Lin heuristic [7] and the heuristic in [31] were used for bipartitioning and 32-way partitioning, respectively. Best, Avg., and CPU represent the best cut size, average cut size, and average running time over 100 runs of a hybrid genetic algorithm.

In both problems, the average distances among local optima considerably decreased by normalization. In the 32-way partitioning problem, the distances before normalization were close to the chromosome lengths, which shows that

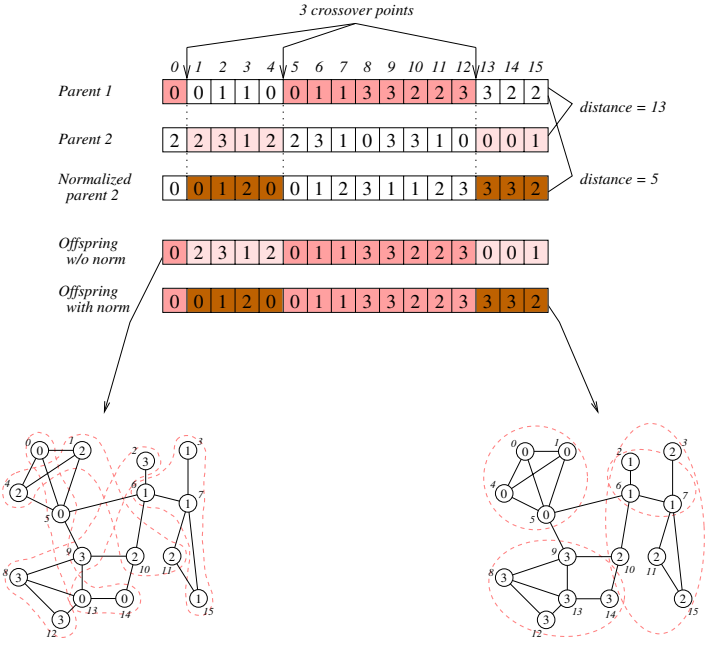


Fig. 4. A normalization example in 4-way graph partitioning

common schemata are rare in genotypes. The distances were dramatically reduced after normalization. In another experiment, we observed that the average distances among local optima were almost the same as those among random solutions in the case without normalization. However, in the case with normalization, we observed that the average distances among local optima were much less than those among random solutions.[3] This implies that the local optima got closer to one another in the space after normalization.

The considerable increase of FDC values by normalization are observed from the tables. This indicates that the landscapes of the search spaces altered by normalization, so that the difficulty of search with respect to genetic algorithms diminished. As expected, normalization led to considerable improvement in performance. Figure 5 shows the change of landscape of an instance G500.005 in the 32-way partitioning problem. The x-axis represents the distance from the best out of $10,000$ local optima. Before normalization, the solution quality had little to do with the distance from the best local optimum. After normalization, they showed a strong correlation.

[3] Experiments showed that, for bipartitioning problem, the average distances among $100,000$ random solutions were 250.00 and 241.02 without and with normalization for 500-node instances and 500.00 and 487.32 for 1000-node instances. For 32-way partitioning problem, they were 484.37 and 432.58 for 500-node instances and 968.75 and 899.54 for 1000-node instances.

Table 1. Comparison of experiments in bipartitioning

Graph	Ordinary					Normalization				
	Ed	ρ	Best	Avg.	CPU	Ed	ρ	Best	Avg.	CPU
G500.005	249.99	0.003	51	52.58	0.82	213.22	0.343	50	52.03	0.89
G500.04	250.05	0.012	1744	1745.65	4.51	215.52	0.292	1744	1745.77	4.35
G1000.0025	499.96	-0.018	97	100.57	2.39	445.01	0.398	95	98.18	2.78
G1000.02	500.00	0.005	3382	3385.65	23.54	447.83	0.487	3382	3385.18	22.60
U500.05	249.94	0.009	2	4.80	1.09	215.82	0.312	2	3.48	1.24
U500.40	250.23	-0.004	412	412.00	1.68	33.38	0.867	412	412.00	1.24
U1000.05	499.90	-0.005	1	5.23	2.77	447.85	0.290	1	2.49	3.24
U1000.40	498.24	0.030	737	737.00	9.88	249.13	0.754	737	737.00	7.85

Table 2. Comparison of experiments in 32-way partitioning

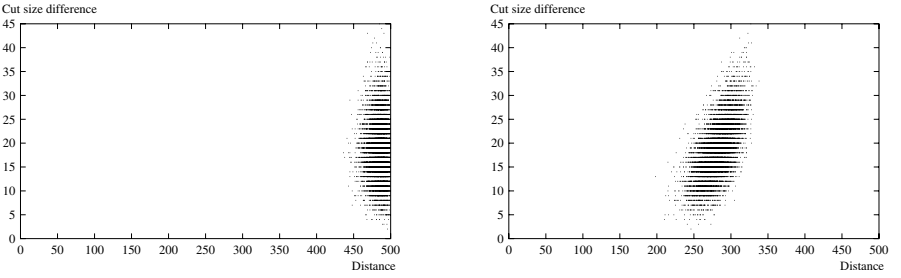
Graph	Ordinary					Normalization				
	Ed	ρ	Best	Avg.	CPU	Ed	ρ	Best	Avg.	CPU
G500.005	483.59	0.019	181	184.92	163.05	300.61	0.621	179	181.63	166.97
G500.04	484.36	0.013	4039	4049.56	1410.56	358.82	0.412	4037	4045.03	1197.93
G1000.0025	968.15	0.009	316	328.55	666.85	723.75	0.472	312	321.32	630.88
G1000.02	968.78	0.030	7824	7839.46	4500.29	813.99	0.303	7818	7832.49	4350.69
U500.05	484.00	0.005	113	120.86	278.33	170.76	0.602	113	116.59	290.75
U500.40	484.04	0.016	5363	5391.38	1197.35	173.56	0.338	5359	5380.15	1161.56
U1000.05	968.47	0.021	127	143.74	702.39	525.48	0.610	117	126.04	1199.85
U1000.40	969.04	0.006	7406	7425.16	3405.49	225.09	0.672	7398	7416.85	3218.18

4.2 Sorting Network Problem

A sorting network is a hardware sorting logic in which the comparisons and exchanges of data are carried out in a prescribed order. A sorting network is composed of buses and a number of homogeneous comparators, where each comparator c(a,b) performs the elementary operation that compares the a^{th} and b^{th} buses; if the values are in order, ignore them, otherwise exchange them. We call a sorting network for n inputs an n-bus sorting network. The sorting network problem is to find an n-bus sorting network with minimal comparators for a given input size n.

In the sorting network problem, a sorting network is generally represented by a sequence of comparators [9] [32] [33] [34]. Choi and Moon [9] defined sorting network isomorphism in terms of construction scheme and showed that the sequence enumeration is a redundant encoding. Contrary to the case of the graph partitioning problem, the number of genotypes for each phenotype may be different from phenotype to phenotype.

Figure 6 shows an example of two isomorphic networks. Between the two networks, there is a permutation $\begin{pmatrix} 0\,1\,2\,3\,4\,5 \\ 5\,4\,3\,2\,1\,0 \end{pmatrix}$ of bus indices. Although they are quite different in appearance, they are the same in function.



(a) Ordinary (b) Normalization

Fig. 5. Normalization effect in 32-way partitioning

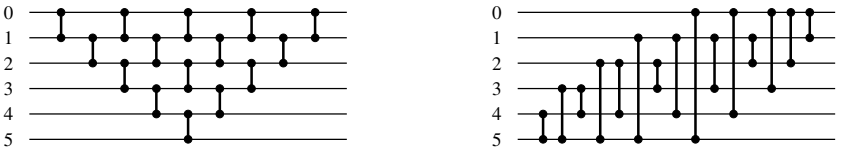
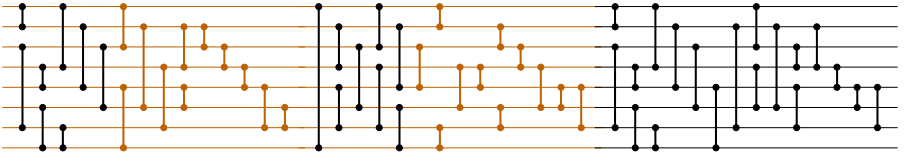


Fig. 6. Two isomorphic networks

In sorting networks, it is known that the comparators in the front part (left side) strongly affect the subsequent comparators [34] [35]. This indicates that the comparators in the front part have stronger effects on the qualities of networks than others. In this context, a normalization method was proposed in [9] that transforms a parent so that the front part is as consistent as possible with the other parent.

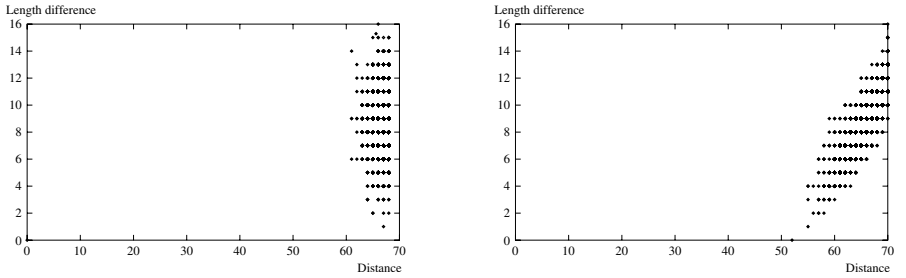
For example, consider the parents 1 and 2 in Figure 7. Although the subnetworks with the first eight comparators in the two networks are totally different, they are isomorphic. Figure 7(c) shows the network after the parent 2 is normalized with respect to the parent 1. The actual crossover is conducted between the parent 1 and the normalized parent 2.

For 16-bus sorting network problem, we performed similar experiments to those in the graph partitioning problem. In this case, the local heuristic and hybrid genetic algorithm in [9] were used. Figure 8 shows the change of the landscapes obtained from 10,000 local optima generated by the local heuristic.



(a) Parent 1 (b) Parent 2 (c) Normalized parent 2

Fig. 7. A normalization example in the 8-bus sorting network problem



(a) Ordinary (b) Normalization

Fig. 8. Normalization effect in 16-bus sorting network problem

Like in the graph partitioning problem, the FDC value considerably increased by normalization (from 0.079 to 0.847) and significant improvement in performance was observed. (Average running time to find 60-comparator networks known as the best decreased from 1,316 to 717 minutes.)

5 Conclusion

We argued that redundant encodings, in which a phenotype is represented by multiple genotypes, lead to severe loss of search power in genetic algorithms, in particular, with respect to traditional crossovers. As a method of alleviating the problem of redundant encodings, normalization considers the genotypes corresponding to a phenotype as a group. We also showed that the normalization alters the search space.

We reviewed the redundant encodings and normalization methods used in the graph partitioning problem and sorting network problem. Through some experiments, we examined the change of search space, the change in problem difficulty, and the performance improvement with respect to genetic algorithms. In both problems, the normalization methods decreased the problem difficulties and led to remarkable improvement in performance.

We are currently working on the effective design of normalization methods for other problems such as neural network optimization and magic tortoise optimization.

Acknowledgements. The authors thank Jong-Pil Kim and Yong-Hyuk Kim for insightful discussions about graph partitioning problem. This work was partly supported by KOSEF through the Statistical Research Center for Complex Systems at Seoul National University (SNU) and Brain Korea 21 Project. The RIACT at Seoul National University provided research facilities for this study.

References

1. N. J. Radcliffe. Forma analysis and random respectful recombination. In *Fourth International Conference on Genetic Algorithms*, pages 222–230, 1991.
2. E. Falkenauer. *Genetic Algorithms and Grouping Problems*. Wiley, 1998.
3. J. Yang, J. Horng, and C. Kao. A continuous genetic algorithm for global optimization. In *Seventh International Conference on Genetic Algorithms*, pages 230–237. Morgan Kaufmann, 1997.
4. Y. K. Kwon and B. R. Moon. A genetic hybrid for CHF function approximation. In *Genetic and Evolutionary Computation Conference*, pages 1119–1125, 2002.
5. M. J. Martin-Bautista and M. A. Vila. A survey of genetic feature selection in mining issues. In *Congress on Evolutionary Computation*, pages 1314–1321, 1999.
6. G. Laszewski. Intelligent structural operators for the k-way graph partitioning problem. In *Fourth International Conference on Genetic Algorithms*, pages 45–52, 1991.
7. T. N. Bui and B. R. Moon. Genetic algorithm and graph partitioning. *IEEE Trans. on Computers*, 45(7):841–855, 1996.
8. S. J. Kang and B. R. Moon. A hybrid genetic algorithm for multiway graph paritioning. In *Genetic and Evolutionary Computation Conference*, pages 159–166, 2000.
9. S. S. Choi and B. R. Moon. Isomorphism, normalization, and a genetic algorithm for sorting network optimization. In *Genetic and Evolutionary Computation Conference*, pages 327–334, 2002.
10. C. Igel and P. Stagge. Effects of phenotypic redundancy in structure optimization. *IEEE Trans. on Evolutionary Computation*, 6(1):74–85, 2002.
11. P. Schuster. Molecular insights into evolution of phenotypes. In J. P. Crutchfield and P. Schuster, editors, *Evolutionary Dynamics — Exploring the Interplay of Accident, Selection, Neutrality and Function*. Oxford Univ. Press, 2002.
12. R. Shipman. Genetic redundancy: Desirable or problematic for evolutionary adaptation. In *Fourth International Conference on Artificial Neural Networks and Genetic Algorithms*, pages 337–344. Springer-Verlag, 1999.
13. M. A. Shackleton, R. Shipman, and M. Ebner. An investigation of redundant genotype-phenotype mappings and their role in evolutionary search. In *Congress on Evolutionary Computation*, pages 493–500, 2000.
14. K. Weicker and N. Weicker. Burden and benefits of redundancy. In *Foundations of Genetic Algorithms*, volume 6, pages 313–333. Morgan Kaufmann, 2001.
15. H. Mühlenbein. Parallel genetic algorithms in combinatorial optimization. In *Computer Science and Operations Research: New Developments in Their Interfaces*, pages 441–453, 1992.
16. R. Dorne and J. K. Hao. A new genetic local search algorithm for graph coloring. In *Parallel Problem Solving from Nature*, pages 745–754. Springer-Verlag, 1998.
17. C. Van Hoyweghen, B. Naudts, and D. E. Goldberg. Spin-flip symmetry and synchronization. *Evolutionary Computation*, 10(4):317–344, 2002.
18. S. Chen. *Is the Common Good ? A New Perspective Developed in Genetic Algorithms*. PhD thesis, Robotics Institute, Carnegie Mellon University, 1999.
19. G. Syswerda. Uniform crossover in genetic algorithms. In *Third International Conference on Genetic Algorithms*, pages 2–9, 1989.
20. S. Chen and S. Smith. Commonality and genetic algorithms. Technical Report CMU-RI-TR-96-27, Robotics Institute, Carnegie Mellon University, 1996.

21. D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, 1989.
22. L. Davis. *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, 1991.
23. R. A. Watson and J. B. Pollack. Recombination without respect: Schema combination and disruption in genetic algorithm crossover. In *Genetic and Evolutionary Computation Conference*, 2000.
24. S. Forrest and M. Mitchell. Relative building-block fitness and the building-block hypothesis. In *Foundations of Genetic Algorithms*, volume 2, pages 109–126. Morgan Kaufmann, 1993.
25. J. Horn and D. E. Goldberg. Genetic algorithm difficulty and the modality of fitness landscapes. In *Foundations of Genetic Algorithms*, volume 3, pages 243–270. Morgan Kaufmann, 1995.
26. S. A. Kauffman. Adaptation on rugged fitness landscapes. In D. Stein, editor, *Lectures in the Sciences of Complexity*, pages 527–618. Addison Wesley, 1989.
27. T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In *Sixth International Conference on Genetic Algorithms*, pages 184–192. Morgan Kaufmann, 1995.
28. P. Merz and B. Freisleben. Fitness landscapes, memetic algorithms and greedy operators for graph bi-partitioning. *Evolutionary Computation*, 8(1):61–91, 2000.
29. P. Merz and B. Freisleben. Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE Trans. on Evolutionary Computation*, 4(4):337–352, 2000.
30. M. Huynen. Exploring phenotype space through neutral evolution. *Journal of Molecular Evolution*, 43:165–169, 1996.
31. J. P. Kim and B. R. Moon. A hybrid genetic search for multi-way graph partitioning based on direct partitioning. In *Genetic and Evolutionary Computation Conference*, pages 408–415, 2001.
32. W. D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. In C. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II.* Addison Wesley, 1992.
33. G. L. Drescher. Evolution of 16-number sorting networks revisited. Unpublished manuscript, 1994.
34. S. S. Choi and B. R. Moon. A hybrid genetic search for the sorting network problem with evolving parallel layers. In *Genetic and Evolutionary Computation Conference*, pages 258–265, 2001.
35. S. S. Choi and B. R. Moon. A graph-based approach to the sorting network problem. In *Congress on Evolutionary Computation*, pages 457–464, 2001.