Mutation Can Improve the Search Capability of Estimation of Distribution Algorithms

Hisashi Handa

Okayama University, Tsushima-Naka 3-1-1,
Okayama 700-8530, JAPAN,
`handa@sdc.it.okayama-u.ac.jp`,
`http://www.sdc.it.okayama-u.ac.jp/~handa/index-e.html`

Abstract. The Estimation of Distribution Algorithms are a class of evolutionary algorithms which adopt probabilistic models to reproduce the genetic information of the next generation, instead of conventional crossover and mutation operations. In this paper, we propose new EDAs which incorporate mutation operator to conventional EDAs in order to keep the diversities in EDA populations. Experiments results shown in this paper confirm us the effectiveness of the proposed methods.

1 Introduction

Recently, Estimation of Distribution Algorithms (EDAs) have been attracted much attention in genetic and evolutionary computation community due to their search abilities [1]. Genetic operators such like crossover and mutation are not adopted in the EDAs. In the EDAs, a new population is generated from the probabilistic model constituted by a database containing the genetic information of the selected individuals in the current generation. Such reproduction procedure by using the probabilistic model makes EDAs more powerful search algorithms. However, it significantly decreases the diversity of the genetic information in the generated population when the population size is not large enough. Hence, EDAs with small population size are often trapped into premature convergences. In this paper, we discuss on the effectiveness of mutation in the case of EDAs. We propose new EDAs which incorporate mutation operator to conventional EDAs in order to keep the diversities in EDA populations.

2 Estimation of Distribution Algorithms with Mutation

In this paper, we incorporate a mutation operator into EDAs in a simple way: For all M sampled individuals, the mutation operation is performed. We adopt the bitwise mutation operator, which is the same as mutation operation in SGA, as the mutation operator of the proposed method. That is, at all loci for all M sampled individuals, whether bit-flip will be held or not is probabilistically decided by referring to the mutation probability. The reason why we adopt such simple way is that we can apply the notion of the proposed method into any kinds of

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 396–397, 2004.

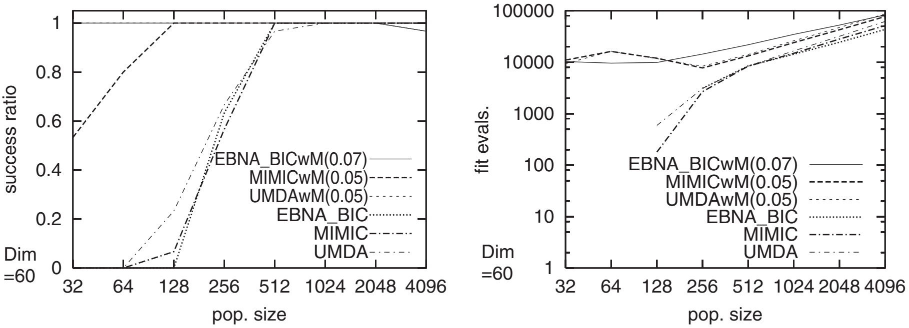

Fig. 1. Experimental results of Fc_4 function: SR (LEFT), NOE (RIGHT); dim = 60

EDAs. In this paper, the proposed methods for UMDA, MIMIC, and EBNA$_{BIC}$ are referred as UMDAwM, MIMICwM, and EBNA$_{BIC}$wM, respectively.

3 Experimental Results

We examine Fc_4 function for comparing the proposed methods with corresponding conventional methods [2]. The number of individuals M is set to be one of 32, 64, 128, 256, 512, 1024, 2048, and 4096, for each problem instance. The number of selected individuals is set to be half of the number of individuals M. We use the truncation selection method to constitute the selected individuals. In this paper, we adopt two indices to evaluate the effectiveness of algorithms: success ratio (SR) and the number of fitness evaluations until finding optimal solutions (NOE). The NOE in this paper is averaged value over "success" runs.

Fig. 1 shows the experimental results of the Fc_4 function. The conventional methods whose population size is less than 512 could not solve the Fc_4 function well. On the other hand, our proposed methods, i.e., UMDAwM and EBNA$_{BIC}$wM, could solved that function for all population sizes. The NOE of our proposed methods is grater than the corresponding conventional methods if the population size of both methods is the same. However, the lowest NOE in the all experiments whose SR = 1 is shown by the MIMICwM with population size = 256.

References

1. P. Larrañaga and J. A. Lozano Editors: *Estimation of Distribution Algorithms*, Kluwer Academic Publishers (2002)
2. J. S. De Bonet *et al.*: MIMIC: Finding optima by estimating probability densities, *Advances in Neural Information Processing Systems* **9** (1996)