行政院國家科學委員會專題研究計畫　成果報告

利用前置式區域搜尋技術增強機率模型構建式基因演算法之執行效能研究

計畫類別：個別型計畫
計畫編號：NSC94-2213-E-009-120-
執行期間：94年08月01日至95年07月31日
執行單位：國立交通大學資訊工程學系(所)

計畫主持人：陳穎平

報告類型：精簡報告
報告附件：出席國際會議研究心得報告及發表論文
處理方式：本計畫可公開查詢

中　華　民　國　95年10月24日

利用前置式區域搜尋技術增強機率模型構建式基因演算法之執行效能研究

Performance Speed-up for Probabilistic Model Building Genetic Algorithms by Using Prefix Local Searchers

中文摘要

機率模型構建式基因演算法 (probabilistic model building genetic algorithms) 為當前所可使用的最好的基因與演化演算法之中的一種。然而和傳統或是一般的基因演算法比起來，通常會需要比較大的人口數目 (population size) 方能達到預計的效能。此外，一般人口初始化的過程使得基因演算法和隨機搜尋演算法 (random search)幾近相同。本研究採用前置式區域搜尋技術(prefix local search techniques)來初始人口，用以增進機率模型構建式基因演算法之效能。奠基於機率模型構建過程的特性，前置式區域搜尋技術應當能夠製造出比完全隨機產生的人口，更適合於進行構建機率模型的初始人口。在本研究中，我們採用由 Harik 於 1999 年提出的延伸型緊緻基因演算法 (extended compact genetic algorithm)做為研究的標的，提出之新式人口初始化方法，除了可以使用於延伸型緊緻基因演算法，亦可以使用於其他的機率模型構建式基因演算法以獲得效能之增進。

關鍵字: 機率模型構建式基因演算法, 前置式區域搜尋技術, 效能增強技術

ABSTRACT

The probabilistic model building genetic algorithms are among the best genetic and evolutionary algorithms currently available. However, algorithms belonging to this class usually require larger populations compared to the traditional genetic algorithm. Otherwise, the common initialization procedure currently employed makes the algorithm close to the random search. This research project employs a prefix local searcher for initializing the population in order to improve the performance of the probabilistic model building genetic algorithms. Based on the properties of the probabilistic model building process, the prefix local searcher should be able to provide a population that is more appropriate for building probabilistic models. In particular, we use the extended compact genetic algorithm, which was proposed by Harik in 1999, as our study subject in this project. The developed initialization scheme should not be only applicable to the extended compact genetic algorithm but also applicable to other algorithms or techniques belonging to the class of probabilistic model building genetic algorithms.

Keywords: Probabilistic model building genetic algorithms, prefix local search, efficiency enhancement techniques

I. INTRODUCTION

The probabilistic model building genetic algorithms are among the best genetic and evolutionary algorithms currently available. However, as studies in the literature indicated, algorithms belonging to this class usually require larger populations compared to the traditional genetic algorithms. Moreover, according to the properties of genetic algorithms, in the first few generations, the individuals contained in the population are all virtually random. In this case, the algorithm is almost identical to the random search. In order to alleviate this situation and therefore to enhance the performance, we would like to develop an initialization scheme in this work.

In fact, modifying the initialization procedure or seeding the initial population is not new at all and has been broadly investigated in the early stage of the development of genetic and evolutionary algorithms. As reported by the existing publications, using exiting solutions in the initial population can offer the following interesting benefits: (1) avoiding "reinventing the wheel" and preventing waste of computational efforts; (2) directing the search into certain search space regions of interests; (3) better performance by mixing heuristic initialization and evolutionary search (Eiben & Smith, 2003). A number of different ways for this purpose have been proposed by researchers in this field, include *seeding*, which uses known good solutions in the initial population, *selective initialization*, which holds N k-way tournaments to form the initial population, conducting a local search for each member of the initial population, and using a starting individual with mass mutation to fill up the initial population.

This research project employs a prefix local searcher for initializing the population in order to improve the performance of the probabilistic model building genetic algorithms. Based on the properties of the probabilistic model building process, the prefix local searcher should be able to provide a population that is more appropriate for building probabilistic models. With a properly initialized population, the efficiency of the probabilistic model building genetic algorithms should be improved not only in the early stage but also during the whole evolutionary process. In particular, we use the extended compact genetic algorithm(Harik, 1999) as our study subject in this project. The developed initialization scheme should not be only applicable to the extended compact genetic algorithm but also applicable to other algorithms or techniques belonging to the class of probabilistic model building genetic algorithms.

The major steps to conduct this research project include (1) select the local searcher to incorporate, (2) determine the way to conduct local search, (3) study the ratio of the starting points and generated points, and (4) understand the effect of the scheme on the probabilistic model building process. The scheme proposed and developed in this project can be considered as an efficiency enhancement technique, which speeds up the underlying genetic and evolutionary algorithm, as well as a tool for us to better understand the probabilistic model building process in both theory and practice.

II. BACKGROUND

Genetic algorithms (GAs) are powerful stochastic, population-based search and optimization techniques loosely modeled after the paradigms of evolution (Goldberg, 1989; Holland, 1973). They guide the search through the solution space by using natural selection and genetic operators, such as crossover, mutation, and the like. Genetic algorithms are now widely applied to solve problems in a number of different fields and domains (Goldberg, 2002). However, most genetic algorithms employed in practice nowadays are simple, traditional genetic algorithms with fixed genetic operators and chromosome representations, which implicitly require their users to have the problem specific knowledge a priori such that appropriate chromosome representations can be designed for the genetic algorithm to work with. Unable to overcome the difficulty caused by the chromosome representation or variable arrangement, these common genetic algorithms suffer from the *linkage problem* (Goldberg, Korb, & Deb, 1989; Thierens, 1995; Thierens & Goldberg, 1993), which refers to the need of appropriately arranging or adaptively ordering the genes on chromosomes or variables in encoding strings during the evolutionary optimization process.

Because it is supposedly hard to guarantee that the user-designed chromosome representation provides correctly, tightly linked building blocks when the problem domain knowledge is unavailable, a variety of linkage learning techniques have been proposed and developed to handle the linkage problem during the run of a genetic algorithm (Kargupta & Bandyopadhyay, 2000; Larranaga & Lozano, 2001; Pelikan & Goldberg, 2002). Existing linkage learning techniques, according to the way they detect, process, and present linkage, can be broadly classified into the following three categories (Munetomo & Goldberg, 1999):

- Perturbation-based methods
  Perturbation-based methods detect the linkage between genes by perturbing the individual and observing the fitness difference caused by the perturbation, such as the *messy genetic algorithm* (mGA) (Goldberg, Deb, & Korb, 1990; Goldberg et al., 1989), the *fast messy genetic algorithm* (fmGA) (Goldberg, Deb, Kargupta, & Harik, 1993; Kargupta, 1995), the *gene expression messy genetic algorithm* (gemGA) (Kargupta, 1996), the *linkage identification by nonlinearity check* (LINC) (Munetomo & Goldberg, 1998), the *linkage identification by non-monotonicity detection* (LIMD) (Munetomo & Goldberg, 1999), and the *linkage identification based on epistasis measures* (LIEM) (Munetomo, 2002).
- Linkage adaptation techniques
  Linkage adaptation techniques employ specifically designed representations, operators, and mechanisms for adapting linkage along with the evolutionary process. The examples of linkage adaptation techniques include the *punctuation marks* (Schaffer & Morishima, 1987), the *metabits* (Levenick, 1995) for implementing the *differential crossover probability*, the *linkage evolving genetic operator* (LEGO) (Smith & Fogarty, 1996) for adapting the recombination strategy via evolution of linkage, and the *linkage learning genetic algorithm* (LLGA) (Chen, 2004; Harik, 1997) using a special representation and an expression mechanism for learning linkage naturally.
- Probabilistic model builders
  Probabilistic model builders learn linkage via building probabilistic models based on the current population and generating new individuals according to the obtained probabilistic models. Useful and popular probabilistic model building genetic algorithms include the *mutual-information-maximizing input clustering* algorithm (MIMIC) (De Bonet, Isbell, & Viola, 1996), the

combining optimizers with mutual information trees method (COMIT) (Baluja & Davies, 1997), the *bivariate marginal distribution algorithm* (BMDA) (Pelikan & Muhlenbein, 1999), the *IDEA framework* (Bosman & Thierens, 2001), the *Bayesian optimization algorithm* (BOA) (Pelikan, Goldberg, & Cantu-Paz, 2000), the *factorized distribution algorithm* (FDA) (Muhlenbein & Mahnig, 1999), and the *extended compact genetic algorithm* (Harik, 1999).

III. EXTENDED COMPACT GENETIC ALGORITHM

The extended compact genetic algorithm partitions the genes or variables into linkage groups, which can be considered as building blocks, and tries to find good probability distributions with the help of these linkage groups. The judging criterion for discriminating superior probability distributions from inferior ones is developed based on the principle of the minimum description length (MDL) (Rissanen, 1978, 1989). The probabilistic models used in the extended compact genetic algorithm is a class of probabilistic models known as marginal product models (MPMs), which are formed as a product of marginal distributions on a partition of the genes for variables, i.e., a linkage group. For a typical generation, the extended compact genetic algorithm selects the better individuals with the specified selection scheme, constructs the marginal product models with the individuals of the current population as well as the guidance given by the principle of the minimum description length, and then gets samples from the constructed probabilistic model as individuals of the population in next generation.

As pointed out by the studies in the literature, in order to acquire good enough probabilistic models for probabilistic model building genetic algorithms, such as the extended compact genetic algorithm, the population size grows at least proportionally in terms of the problem size (Pelikan, 2002). According to the properties of genetic algorithms, the individuals in these large populations in the first few generations are very close to totally random ones. As a consequence, in the early stage of the evolutionary computational process, the probabilistic model building genetic algorithms are not quite different from the random search. In order to avoid waste of time on the random search in the early stage and therefore speed up the whole run duration, the initial populations which may be more appropriate for building probabilistic models should be utilized in the probabilistic model building genetic algorithms, such as the extended compact genetic algorithm in this project.

IV. EXPERIMENTS

i. Prefix Local Search

Although many different initialization techniques have been introduced to the genetic and evolutionary algorithms, the proposed initialization scheme in this study is distinguishably different from those previously described in the literature in that a local searcher is incorporated in the initialization phase and more importantly, the path in the search space walked through by the local searching points are kept to help the probabilistic model building process.

To realize the novel initialization scheme is to select an appropriate local search method by which the search points generated along the local search process can actually help the later probabilistic model building process. Candidate local search methods for this purpose include the deterministic hill-climbing, the stochastic hill-climbing, the steepest gradient descent, and the like.

We use a prefix local searcher for initializing the population. First, part of individuals are randomly generated as seeds and evaluated. Second, the local search is conducted on each individual for specified steps to “fill” up the population. We use different mutation rates to produce offspring for local search. Compare the parent with its offspring, the better is held to next local search. Therefore the population size is the number of initial individuals multiplied by the number of local search steps. The parameters to initialize populations are presented in Table 1.

Table 1. The initialization parameters in our experiments.

Population size	Randomly initial individuals	Steps for local search
160	40	4
	20	8
	10	16
256	64	4
	32	8
	16	16
384	96	4
	48	8
	24	16
576	144	4
	72	8
	36	16
960	240	4
	120	8
	60	16
1904	476	4
	238	8
	119	16
Combined with mutation rates: 0.05, 0.1, and 0.2		

ii. Fitness function

For each individual, it is a binary string which is composed of ten 4-bit trap. The objective function *h(x)* for *i*th 4-bit trap is described as:

$$u(x) = \sum_{j=1..4} x_j$$

$$\text{if} \begin{cases} u(x) = 0, & h_i(x) = 5 \\ \text{else}, & h_i(x) = u(x) \end{cases}$$

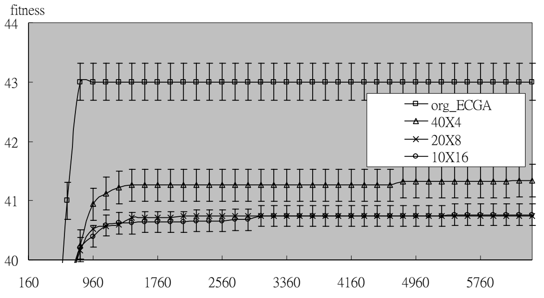
where x_j is a bit in a 4-bit trap x . Therefore the fitness function is:

$$f(X) = \sum_{i=1..10} h_i(x)$$

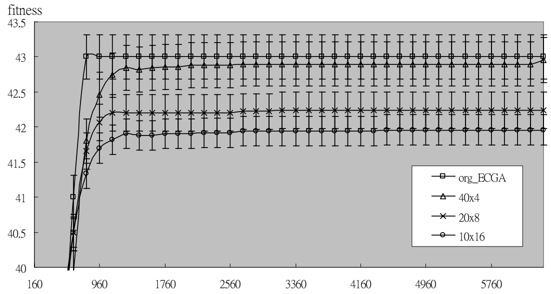
where x is an individual.

V. RESULTS

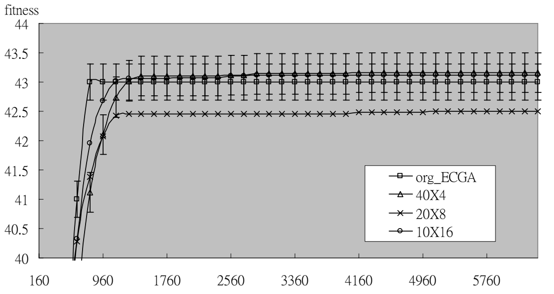
For each parameters setting (see Table 1), we performed 50 runs and each run has 250,000 evaluations. We get best fitness averaging 50 runs and compare with original ECGA. Figure (a)-(r) show the result of each generation in the different population size with different mutation rate. We can see that sometimes our approach is better than original ECGA, but usually isn't. When population size is large enough, the results between our approach and original ECGA are similar. For all results of our approach, the result in 4 steps for local search is always better than others. Besides, we get the final fitness in Figure (a)-(r) to show the variation of the fitness in different population size.(see Figure 1-3)



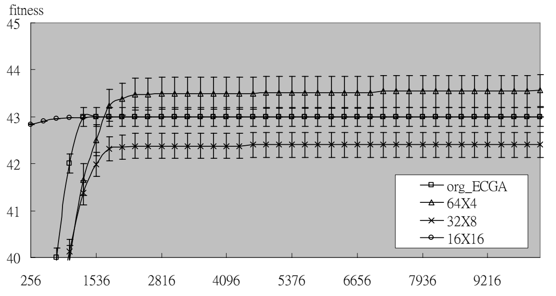
(a) 160_0.05



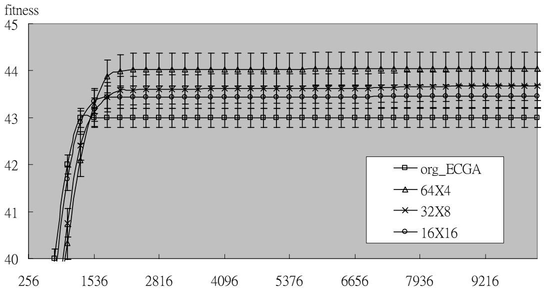
(b) 160_0.1



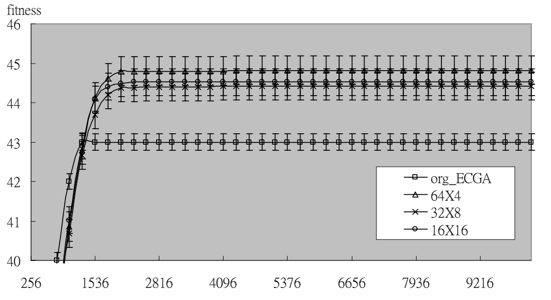
(c) 160_0.2



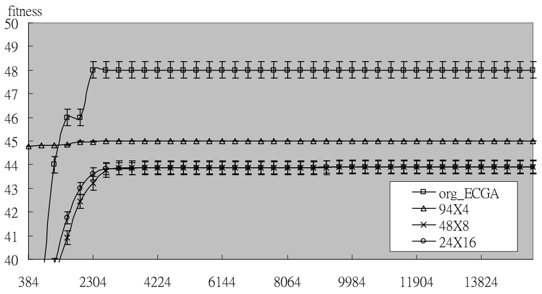
(d) 256_0.05



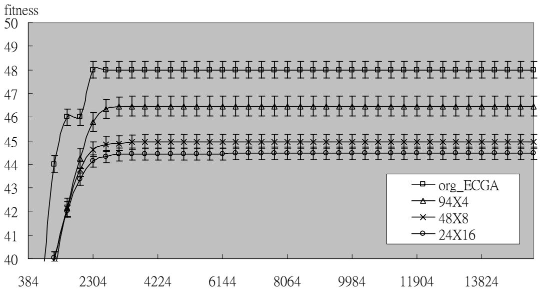
(e) 256_0.1



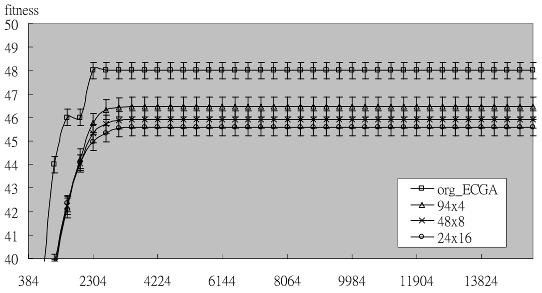
(f) 256_0.2



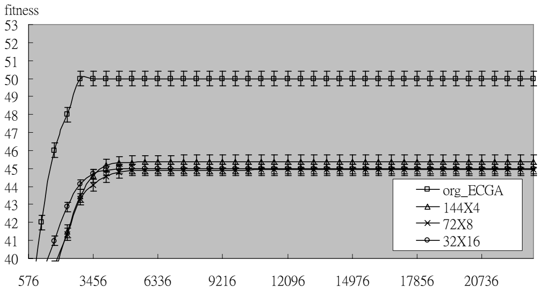
(g) 384_0.05



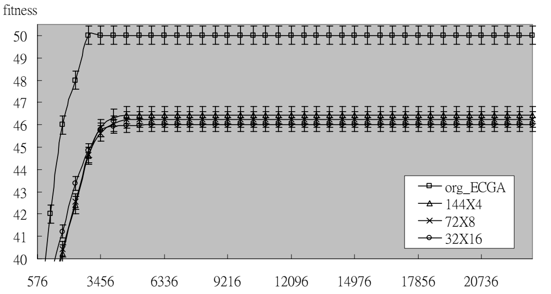
(h) 384_0.1



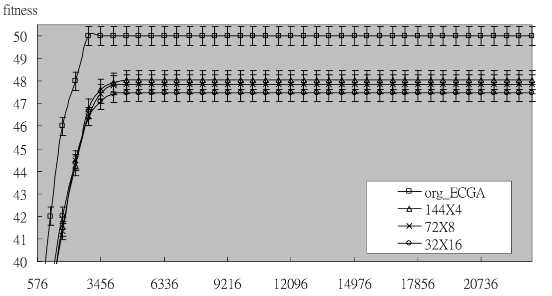
(i) 384_0.2



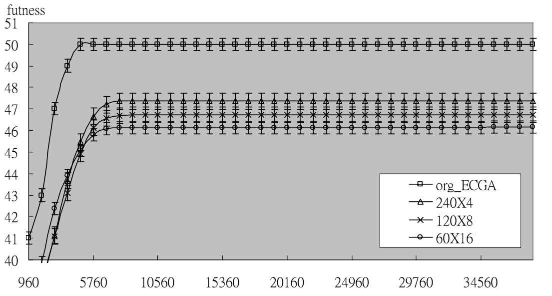
(j) 576_0.05



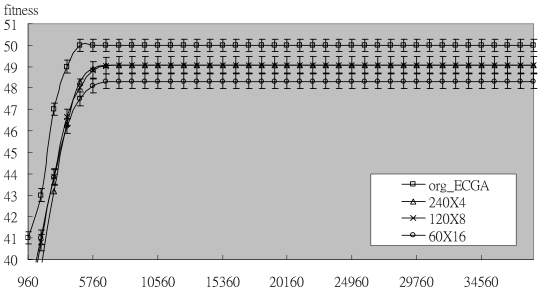
(k) 576_0.1



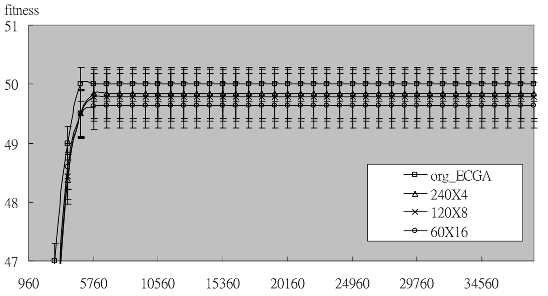
(l) 576_0.2



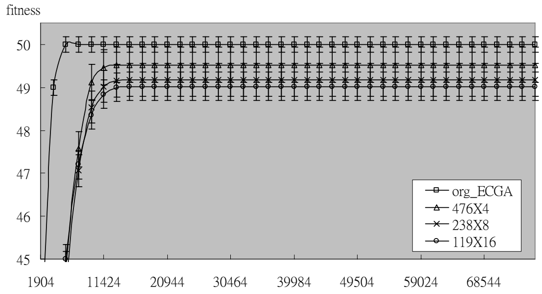
(m) 960_0.05



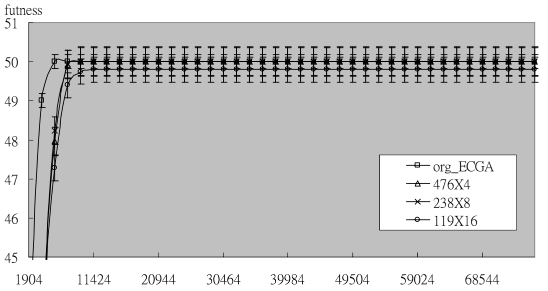
(n) 960_0.1



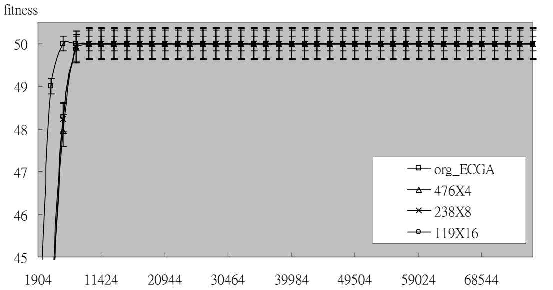
(o) 960_0.2



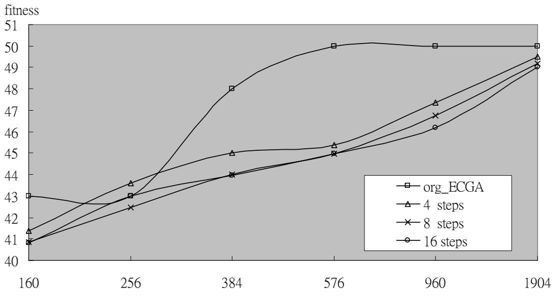
(p) 1904_0.05



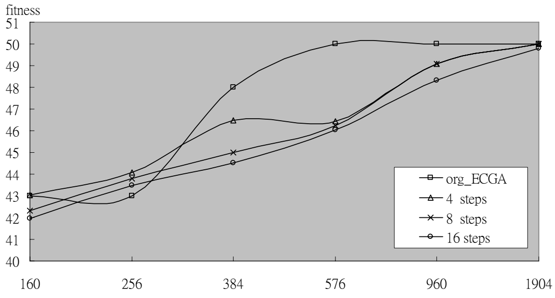
(q) 1904_0.1



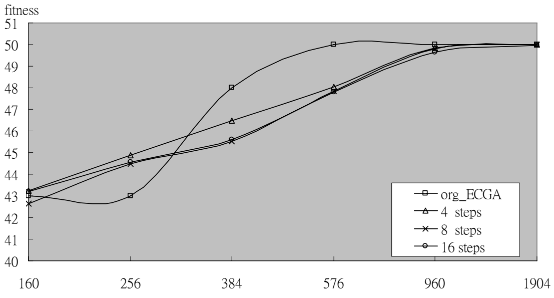
(r) 1904_0.2



(1) mutation rate = 0.05



(2) mutation rate = 0.1



(3) mutation rate = 0.2

VI. CONCLUSIONS

We attempted to design an approach to enhance the efficiency and performance of the extended compact genetic algorithm by using a prefix local searcher to assist to generate the initial population and therefore to help the probabilistic model building process utilized by the extended compact genetic algorithm. However, the experimental results indicate that the approach in the present study does not help ECGA to provide superior performance in either convergence speed or the best fitness value. There are several possibilities for this situation. One may be that the model building process intrinsically cannot be enhanced by the proposed approach. Another may be that the chosen local searcher is not compatible with ECGA and hence cannot assist the model building process. In order to clarify the cause of the failure and to gain further understanding of the nature of model building, more studies and experiments have to be done along this line of research.

VII. REFERENCE

Baluja, S., & Davies, S. (1997). Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. *Proceedings of the Fourteenth International Conference on Machine Learning*, 30-38.

Bosman, P. A. N., & Thierens, D. (2001). Advancing Continuous IDEAs with Mixture Distributions and Factorization Selection Metrics. *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference (GECCO-2001 OBUPM)*, 208-212.

Chen, Y.-p. (2004). *Extending the Scalability of Linkage Learning Genetic Algorithms: Theory and Practice.* Unpublished Doctoral Dissertation. University of Illinois at Urbana-Champaign, Urbana, IL, USA.

De Bonet, J. S., Isbell, C., & Viola, P. (1996). MIMIC: Finding Optima by Estimating Probability Densities. *Advances in Neural Information Processing Systems, 9*, 424-430.

Eiben, A. E., & Smith, J. E. (2003). *Introduction to Evolutionary Computing*: Springer.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*: Addison-Wesley Publishing Co.

Goldberg, D. E. (2002). *The Design of Innovation: Lessons from and for Competent Genetic Algorithms* (Vol. 7): Kluwer Academic Publishers.

Goldberg, D. E., Deb, K., Kargupta, H., & Harik, G. R. (1993). Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93)*, 56-64.

Goldberg, D. E., Deb, K., & Korb, B. (1990). Messy genetic algorithms revisited: Studies in mixed size and scale. *Complex Systems, 4*(4), 415-444.

Goldberg, D. E., Korb, B., & Deb, K. (1989). Messy Genetic Algorithms: Motivation, Analysis, and First Results. *Complex Systems, 3*(5), 493-530.

Harik, G. R. (1997). *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms*. Unpublished Doctoral Dissertation. University of Michigan at Ann Arbor, Ann Arbor, MI, USA.

Harik, G. R. (1999). *Linkage learning via probabilistic modeling in the ECGA*. University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. *SIAM Journal on Computing, 2*(2).

Kargupta, H. (1995). *SEARCH, polynomial complexity, and the fast messy genetic algorithm*. Unpublished Doctoral Dissertation. University of Illinois at Urbana-Champaign, Urbana, IL, USA.

Kargupta, H. (1996). The gene expression messy genetic algorithm. *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, 814-819.

Kargupta, H., & Bandyopadhyay, S. (2000). A perspective on the foundation and evolution of the linkage learning genetic algorithms. *Computer Methods in Applied Mechanics and Engineering, 186*(2-4), 269-294.

Larranaga, P., & Lozano, J. A. (2001). *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation (Vol. 2)*: Kluwer Academic Publishers.

Levenick, J. R. (1995). Metabits: Generic endogenous crossover control. *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA-95)*, 88-95.

Muhlenbein, H., & Mahnig, T. (1999). Convergence theory and applications of the factorized distribution algorithm. *Journal of Computing and Information Technology, 7*, 19-32.

Munetomo, M. (2002). Linkage Identification Based on Epistasis Measures to Realize Efficient Genetic Algorithms. *Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002)*, 1332-1337.

Munetomo, M., & Goldberg, D. E. (1998). *Identifying Linkage by Nonlinearity Check (No. IlliGAL Tech Report. 98012)*: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Munetomo, M., & Goldberg, D. E. (1999). Linkage identification by non-monotonicity detectio for overlapping functions. *Evolutionary Computation, 7*(4), 377-398.

Pelikan, M. (2002). *Bayesian Optimization Algorithm: From Single Level to Hierarchy*. Unpublished Doctoral Dissertation. University of Illinois at Urbana-Champaign, Urbana, IL, USA.

Pelikan, M., & Goldberg, D. E. (2002). A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications, 21*(1), 5-20.

Pelikan, M., Goldberg, D. E., & Cantu-Paz, E. (2000). Linkage problem, distribution estimation, and Bayesian networks. *Evolutionary Computation, 8*(3), 311-341.

Pelikan, M., & Muhlenbein, H. (1999). The bivariate marginal distribution algorithm. *Advances in Soft Computing-Engineering Design and Manufacturing*, 521-535.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica, 14*, 465--471.

Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry (Vol. 15)*: World Scientific Publishing Co.

Schaffer, J. D., & Morishima, A. (1987). An adaptive crossover distribution mechanism for genetic algorithms. *Proceedings of the Second International Conference on Genetic Algorithms (ICGA-87)*, 36-40.

Smith, J., & Fogarty, T. C. (1996). Recombination strategy adaptation via evolution of gene linkage. *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, 826-831.

Thierens, D. (1995). *Analysis and Design of Genetic Algorithms*. Unpublished Doctoral Dissertation. Katholieke Universiteit Leuven.

Thierens, D., & Goldberg, D. E. (1993). Mixing in genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93)*, 38-45.