行政院國家科學委員會專題研究計畫　成果報告

鏈結學習型基因演算法之設計改進研究

計畫類別：個別型計畫
計畫編號：NSC93-2213-E-009-156-
執行期間：93 年 10 月 01 日至 94 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

計畫參與人員：陳穎平

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 94 年 10 月 29 日

鏈結學習型基因演算法之設計改進研究

Design Enhancement of Linkage Learning Genetic Algorithms

中文摘要

本研究計畫測試與驗證先前於文獻中為鏈結學習型基因演算法 (linkage learning genetic algorithm) 所提出之子染色體表現方式 (subchromosome representation) 的功能與效用。子染色體表現方式的設計目的，乃在於程式執行時期，將建構基石被處理的個數有效地減低，以避開在鏈結學習型基因演算法的收斂時間模型 (convergence time model) 中所指出的執行效能極限。於先前的文獻中，已提出子染色體表現方式的設計與初步實作嘗試。在本研究中，我們希望經由再次的測試與驗證，來呈現出於鏈結學習型基因演算法使用子染色體表現方式，確實有效顯著的功能與效用。我們的實驗結果顯示出，在鏈結學習型基因演算法使用子染色體表現方式，除了可以明顯地改進其在處理含有貢獻一致之建構基石的問題時之效能，同時在當預先得知的基因鏈結資訊存在時，子染色體表現方式也能做為充分利用此種資訊的良好機制。

關鍵字：鏈結學習、基因演算法、表現方式、建構基石

ABSTRACT

This research work examines and verifies the use of subchromosome representations previously introduced to the linkage learning genetic algorithm (LLGA). The subchromosome representation is utilized for effectively lowering the number of building blocks in order to escape from the performance limit implied by the convergence time model for the linkage learning genetic algorithm. A preliminary implementation was developed to realize subchromosome representations in the literature. In this study, we once more examine and verify the use of subchromosomes in the linkage learning genetic algorithm. The experimental results indicate that the representation can improve the performance of the linkage learning genetic algorithm on uniformly scaled problems, and the implementation provides a potential way for the linkage learning genetic algorithm to incorporate prior linkage information when such knowledge exists.

Keywords: Linkage learning, genetic algorithms, representations, building blocks

I. INTRODUCTION

Linkage learning, making genetic algorithms (GAs) capable of detecting associations among genes or variables and properly arranging these closely related genes to form building blocks, is one of the key challenges of the genetic algorithm design. In order to ensure a genetic algorithm works well, the building blocks represented on the chromosome have to be tightly linked (D.E. Goldberg *et al.*, 1993; Thierens & Goldberg, 1993). One way to alleviate the burden of choosing an appropriate chromosome representation for genetic algorithm users is to employ the genetic linkage learning technique. Among the existing linkage learning methods, such as *perturbation-based techniques* (Kargupta, 1996; Munetomo & Goldberg, 1999) , *model builders* (Baluja, 1994; Muhlenbein & Mahnig, 1999; Pelikan *et al.*, 1999) , and *linkage learners* (Levenick, 1995; Smith & Fogarty, 1996), is the linkage learning genetic algorithm (LLGA), using an evolvable genotype capable of learning genetic linkage during the evolutionary process.

While the linkage learning genetic algorithm achieved successful linkage learning on problems with badly scaled building blocks, it was less successful on problems consisting of uniformly scaled building blocks. The convergence time model for the linkage learning genetic algorithm (Chen & Goldberg, 2004) explains the difficulty faced by the linkage learning genetic algorithm and indicates the performance limit of the linkage learning genetic algorithm on uniformly scaled problems. This research project seeks to enhance the design of the linkage learning genetic algorithm based on the time models to improve the performance of the linkage learning genetic algorithm on uniformly scaled problems.

In particular, we would like to examine and improve a previously proposed design, called *subchromosome* representations of the linkage learning genetic algorithm. The subchromosome representation was developed to avoid the performance limit implied by the convergence time model for the linkage learning genetic algorithm.

In the remainder of the report, we will first briefly review the linkage learning genetic algorithm in the next section, followed by the discussion on the limit to competent of linkage learning genetic algorithm. The subchromosome representation will then be described, followed by the experiments for verficiation purpose. Finally, this report will be concluded in section VI.

II. THE ESSENSE OF THE LINKAGE LEARNING GENETIC ALGORITHM

The key elements of the linkage learning genetic algorithm (G. R. Harik, 1997) include the chromosome representation, the exchange crossover, and the probabilistic expression. The linkage learning genetic algorithm is capable of learning genetic linkage in the evolutionary

process without the help of extra measurements and techniques. A modified version of the algorithm working with promoters (Chen & Goldberg, 2002) will be used in this project.

The LLGA's chromosome representation is mainly composed of moveable genes, non-coding segments, probabilistic expression, and promoters. Moveable genes are encoded as (*gene number*, *allele*) pairs on the LLGA chromosome, and an LLGA chromosome is considered as a circle. These genes are allowed to move around and reside anywhere on the chromosome. Non-coding segments are inserted into the chromosome to create an evolvable genotype capable of learning linkage. Non-coding segments act as non-functional genes residing between functional genes to form gaps for precisely expressing genetic linkage.

Probability expression (PE) was proposed to preserve building-block level diversity. For each gene, all possible alleles coexist in a PE chromosome at the same time. For the purpose of evaluation, a chromosome is interpreted with a point of interpretation. The allele for each gene is determined by the order according to which the chromosome is traversed clock-wisely from the point of interpretation. A complete string is then expressed.

Consequently, each PE chromosome represents not just a single solution but a probability distribution over the range of possible solutions. If different points of interpretation are selected, a PE chromosome might be interpreted as different solutions. Furthermore, the probability of a PE chromosome to be expressed as a particular solution depends on the length of the non-coding segment between genes critical to that solution. It is the essential technique of the linkage learning genetic algorithm to capture the knowledge about linkage and to prompt the evolution of linkage.

The use of promoters was proposed (Chen & Goldberg, 2002) to handle separation inadequacy and to improve nucleation potential. Promoters are special non-functional elements on the chromosome. While in the linkage learning genetic algorithm without promoters, all genes and non-coding segments can be the points of interpretation of the child created by crossover, only promoters can be the points of interpretation in the linkage learning genetic algorithm with promoters.

The exchange crossover operator is another key mechanism to make the linkage learning genetic algorithm capable of learning genetic linkage. In the linkage learning genetic algorithm with promoters, although the grafting point can still be any genes or non-coding segments, the point of interpretation of the offspring is no longer the grafting point. Instead, the new point of interpretation is the nearest promoter before the grafting point on the chromosome. After the grafting point is randomly chosen, the first promoter in front of the grafting point is the point of interpretation of the offspring. The genetic material is then transferred in the following order: (1) the segment between the promoter and the grafting point, (2) the segment chosen from the donor, and (3) the rest of the recipient. Exchange crossover in the linkage learning genetic algorithm with promoters selects only one cutting point at random. The other cutting point is always the element (either functional or non-functional) just before the point of interpretation of the donor.

The linkage learning genetic algorithm has been studied on problems containing multiple building blocks in two forms---the uniformly scaled problem and the exponentially scaled problem---not only because of their prevalence in the literature but also because they are abstract versions of many decomposable problems (D.E. Goldberg, 2002). Uniformly scaled problems resemble those with subproblems of equal importance, and exponentially scaled problems represent those with subproblems of distinguishable importance. As reported previously (G. R. Harik, 1997), when the building blocks of a problem are exponentially scaled, the linkage learning genetic algorithm can solve the problem in a linear time function of the number of building blocks. However, when the building blocks are uniformly scaled, the linkage learning genetic algorithm either needs a population size that grows exponentially with the problem size or takes exponential time to converge. In order to explain LLGA's seemingly inconsistent behavior, the tightness time models (Chen & Goldberg, 2003) and the convergence time model (Chen & Goldberg, 2004) were previously proposed to understand how the linkage learning genetic algorithm works.

III. LIMIT TO COMPETENCE OF THE LINKAGE LEARNING GENETIC ALGORITHM

First of all, before providing a solution to the performance issue of the linkage learning genetic algorithm, we need to understand the implication of the convergence time model and then enhance the design of the linkage learning genetic algorithm accordingly. In addition to describing the way the linkage learning genetic algorithm works on uniformly scaled problems as well as explaining the seemingly inconsistent behavior of the linkage learning genetic algorithm on problems with building blocks of different scalings, the convergence time model for the linkage learning genetic algorithm also reveals a critical limit to competence of the linkage learning genetic algorithm that the computational time for the linkage learning genetic algorithm to solve uniformly scaled problems grows exponentially with the number of building blocks in the problem.

By examining the proposed convergence time model more carefully, we can find that the parameters involved in the model are either the properties of the problem to solve, such as the order of building blocks, k, and the number of building blocks, m, or the uncontrollable constants and variables, such as the linkage-skew coefficient, c_s, and the level of linkage, ε. Unlike many facetwise models that contain algorithmic parameters and can shed light on how to appropriately set these parameters to enable the genetic algorithm, little guidance can be obtained from the time model for setting the existing algorithmic parameters of the linkage learning genetic algorithm. For example, the schema theorem (De Jong, 1975; David E. Goldberg, 1989; Holland, 1975) describes the market share growth of building blocks in terms of selection pressure and crossover probability. It shows us the correct way to choose these two parameters. However, the convergence time model cannot

provide us such practical indications because there is no algorithmic parameters involved in the model. Therefore, instead of trying to adjust those existing algorithmic parameters, another way to improve the performance of the linkage learning genetic algorithm on uniformly scaled problems has to be taken, such as the one examined in this project.

IV. SUBCHROMOSOME REPRESENTATIONS

The subchromosome representation in the linkage learning genetic algorithm separates a LLGA chromosome into several parts, called *subchromosomes*. The structure of a subchromosome is identical to that of a regular LLGA chromosome. Like a regular LLGA chromosome, a subchromosome contains moveable genes, non-coding segments, as well as promoters and is interpreted with probabilistic expression. All subchromosomes belonging to one individual form a complete LLGA chromosome. In subchromosome representations, there is no separate fitness measurement for each subchromosome. The fitness obtained from interpreting the complete chromosome is used by all subchromosomes. Therefore, the unimetric characteristic of the linkage learning genetic algorithm is still maintained because there is no extra measurement regarding the structure of the chromosome.

The goal of subchromosome representations is to create a flexible encoding that makes LLGA chromosomes capable of grouping closely related building blocks to form higher-level ones in addition to moving genes together on the chromosome to form the first-level building blocks. Similar to genetic linkage learning, the process of forming higher-level building blocks should be integrated with the evolutionary and problem-solving process. The subchromosomes of an LLGA chromosome can be considered as building blocks of the second level. The subchromosome representation can be designed to hierarchically express building blocks of even higher levels, such as the third level, the fourth level, and so on.

V. EXPREIMENTS

The experiments to observe the effect of using the subchromosome representation in the linkage learning genetic algorithm are presented in this section. First, the parameter settings of the experiments are described in detail. Then, the experimental results are shown in the remainder of this section.

V. 1 Experimental Settings

Trap functions (Ackley, 1987; Deb & Goldberg, 1993; Deb *et al.*, 1993) are used for examining the effect of adopting subchromosome representations in the linkage learning genetic algorithm because trap functions provide decent linkage structures, and good linkage is required in order to solve problems consisting of traps. The experiments here were done for order-4 traps, and all traps contribute equally to the fitness.

To simulate the infinite-length chromosome, we let one order-4 building block embedded in 250 genes, including functional genes and non-coding elements. For example, for five order-4 building blocks, the 20 genes are embedded in a 1,250-gene chromosome with 1,230 non-coding elements. The total number of building blocks in one experiment is the number of building blocks per subchromosome times the number of subchromosomes. From 2 to 8 building blocks per subchromosome, all conditions for the total number of building blocks less than or equal to 80 are included in the experiments.

The gambler's ruin model (G. Harik *et al.*, 1997)is utilized for population sizing. Other parameters are set as follows. The crossover rate is 1.0 such that the crossover event always happens. The maximum number of generations is 100,000. The number of promoters on each subchromosome is set to $2m$, where m is the number of building blocks on the subchromosome. Finally, each experiment was repeated with 50 independent runs.

V. 2 Experimental Results

For each experiment, the success rate is calculated according to the results obtained in the 50 independent runs. Here, a success is determined by the final solution quality. The solution quality is the ratio between the number of correctly solved building blocks in the end of the run and that of the total building blocks in the trial. For example, if in a particular run for solving 20 building blocks, 12 building blocks are correctly solved, the solution quality of this run is 0.6. If the final solution quality of a run is equal to or greater than 0.9, the run is recorded as a success. The success rate is therefore the ratio between the number of success trials and that of runs.

Figures 1, 2, and 3 give the success rates of all the experiments with the total number of building blocks less than or equal to 80. The number of building blocks distributed on each subchromosome varies from 2 to 8. The results for each number of building blocks on subchromosomes are shown with different line-point styles in the figures. As shown in these figures, utilizing subchromosome representations in the linkage learning genetic algorithm can significantly improve the performance of the linkage learning genetic algorithm on the uniformly scaled problems. Compared to the previously reported results, the linkage learning genetic algorithm with the subchromosome representation can solve uniformly scaled problems about five times larger in terms of the total number of building blocks than that can be solved by the linkage learning genetic algorithm without subchromosomes.

In addition to the performance improvement, the figures also show that the limit for the linkage learning genetic algorithm with the subchromosome representation in our first attempt of implementation to solve uniform scaled problems seems to be around 50 in terms of the total number of building blocks. Even with different numbers of building blocks distributed on one subchromosome, no successful trial was found among all the experiments with totally 60 or more building blocks. However, according to the importance of building-block identification and exchange, we had to use a crossover probability as high as

1.0 in the linkage learning genetic algorithm without subchromosomes in order to effectively promote the linkage learning process, which proceeds with only the *differential selection of linkage*, which is generated indirectly from the schema theorem.

Such a high crossover probability not only promotes the linkage learning process but also raises the probability to disrupt building blocks. Therefore, by using the same setting of the crossover probability in these experiments, the procedure to apply exchange crossover on subchromosomes may cause serious building-block disruption, as it does in the linkage learning genetic algorithm without subchromosomes. Based on the discussion, the exchange crossover operator is slightly modified as follows. Instead of applying exchange crossover to all pairs of subchromosomes, each pair is now chosen with a probability of 0.5 for applying exchange crossover in order to reduce building-block disruption as much as possible without influencing the mixing rate too much. The previous experiments were repeated to check the effect of adjusting the crossover probability. The results are shown in Figure 4 and indicate that the subchromosome works well in the range of these experiments.

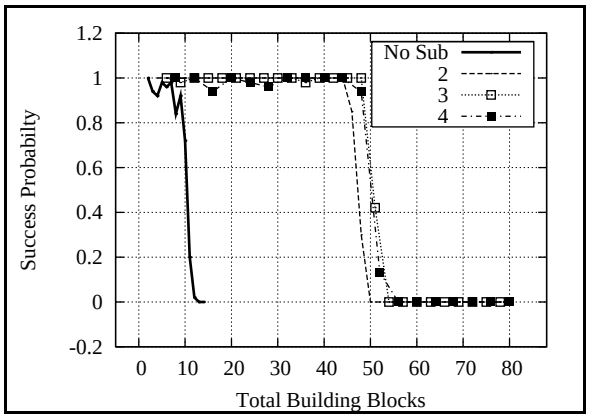

Figure 1: 2, 3, & 4 Subchromosomes.

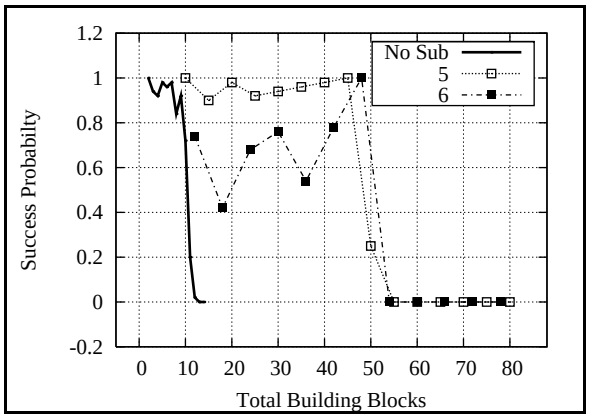

Figure 2: 5 & 6 Subchromosomes.

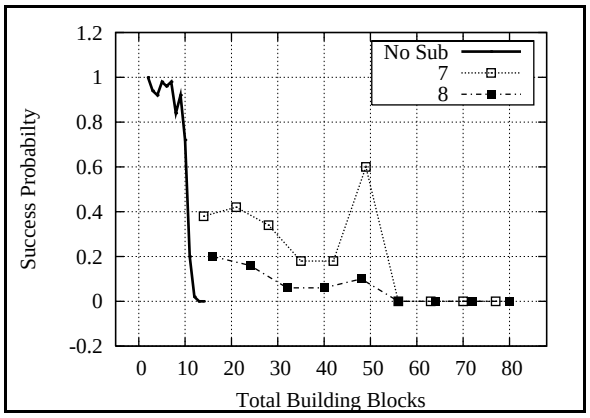

Figure 3: 7 & 8 Subchromosomes.

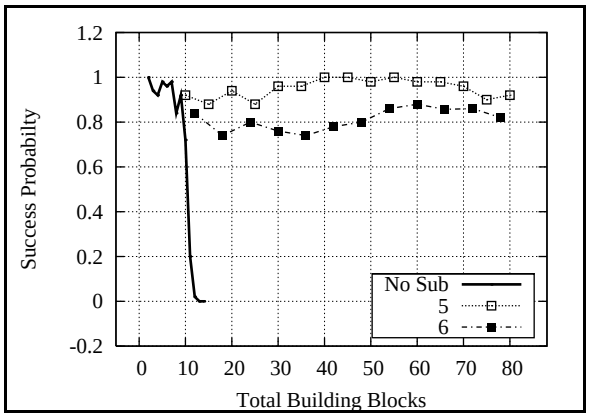

Figure 4: 5 & 6 Subchromosomes w/ pc=0.5.

VI. CONCLUSIONS

We started with a brief review of the linkage learning genetic algorithm, including the chromosome representation, exchange crossover, linkage learning mechanisms, and time models. The subchromosome representation was developed and employed in the linkage learning genetic algorithm for effectively lowering the number of building blocks to escape from the limit implied by the convergence time model. An initial step to realize subchromosome representations in the linkage learning genetic algorithm was taken. The preliminary experimental results of using subchromosomes in the linkage learning genetic algorithm indicated that the proposed scheme can improve the performance of the linkage learning genetic algorithm on uniformly scaled problems.

In addition to showing that the subchromosome representation helps the linkage learning genetic algorithm to solve larger uniformly scaled problems, the initial step for implementation the proposed representation in the current work also leads a possible way in making the linkage learning genetic algorithm capable of incorporating the prior linkage information. With the use of subchromosomes, the distribution of genes, non-coding segments, and building blocks can be determined according to the available linkage information of the problem. In the linkage learning genetic algorithm without subchromosomes, utilizing prior linkage information is extremely difficult. Overall, the results reveal a promising path for achieving scalable genetic linkage learning techniques.

Finally, although the ultimate goal of the research project was not successfully accomplished, several minor objectives were achieved, including extending the scale of problems that can be solved by the linkage learning genetic algorithm, providing a controllable mechanism for the linkage learning genetic algorithm to adjust the gene distributions, and developing a way for the user to incorporate the prior linkage information into the linkage learning genetic algorithm. The research results of this project have been compiled in the form of academic papers and have been submitted to important international conferences in the field of evolutionary computation. Hopefully, there will be positive outcomes and the research work along this line can be carried on.

VII. REFERENCES

Ackley, D. H. (1987). *A connectionist machine for genetic hill climbing*. Boston: Kluwer Academic.

Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning* (No. CMUCS-94-163). Pittsburgh, PA: Carnegie Mellon University.

Chen, Y.-p., & Goldberg, D. E. (2002). Introducing start expression genes to the linkage learning genetic algorithm. *Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature (PPSN VII)*, 351--360.

Chen, Y.-p., & Goldberg, D. E. (2003). Tightness time for the linkage learning genetic algorithm. *Proceedings of Genetic and Evolutionary Computation Conference 2003 (GECCO-2003)*, 837--849.

Chen, Y.-p., & Goldberg, D. E. (2004). Convergence time for the linkage learning genetic algorithm. *Proceedings of the 2004 Congress on Evolutionary Computation (CEC2004)*, 39--46.

De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems.* Unpublished PhD Dissertation, University of Michigan, Ann Arbor, MI.

Deb, K., & Goldberg, D. E. (1993). Analyzing deception in trap functions. *Foundations of Genetic Algorithms, 2*, 93--108.

Deb, K., Horn, J., & Goldberg, D. E. (1993). Multimodal deceptive functions. *Complex Systems, 7*(2), 131--153.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: 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., & Thierens, D. (1993). Toward a better understanding of mixing in genetic algorithms. *Journal of the Society of Instrument and Control Engineers, 32*, 10--16.

Harik, G., Cantu-Paz, E., Goldberg, D. E., & Miller, B. L. (1997). The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation*, 7--12.

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

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press.

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

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). Fda - a scalable evolutionary algorithm for the optimization for the optimization of additively decomposed functions. *Evolutionary Computation, 7*(4), 353--376.

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

Pelikan, M., Goldberg, D. E., & Cantu-Paz, E. (1999). Boa: The bayesian optimization algorithm. *Proceedings of Genetic and Evolutionary Computation Conference 1999 (GECCO-99)*, 525--532.

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