應用演化式演算法發展回饋類神經網路之自動代理機器人

An Evolutionary Algorithm to Evolve Recurrent Neural Networks for Autonomous Agents

計畫編號：NSC 89-2213-E-002-069

執行期限：88 年 8 月 1 日至 89 年 7 月 31 日

主持人：高成炎　　國立台灣大學資訊工程學系

一、中文摘要

自動化代理機器人是提昇工業界競爭力的最重要關鍵技術，自動化代理機器人必須具備隨環境自我調適，同時具有容忍雜訊的能力，類神經網路具有彈性、健全與容忍雜訊的能力，因此類神經網路被廣泛應用在自動化代理機器人。在本計畫中我們發展了新的演化式演算法以解類神經網路的問題，此方法論可提昇類神經網路解的品質。我們應用此演算法於布林函數與兩個標竿問題上，以說明本演算的有效性，接著此方法也成功應用於人工螞蟻及簡單踢足球兩個自動化代理機器的問題上，我們也規劃應用此演算法到實際的自動化代理機器人上。

關鍵詞：自動化代理機器人，回饋式類神經網路、演化式計算、遺傳演算法、人工螞蟻

英文摘要

Autonomous agents are one of the most critical factors of promoting the competitive ability for industry. An autonomous agent must possess self-adaptive ability and tolerate noise to adapt its behavior to any changes of its environment. Neural networks have been considered to well achieve the demands and to avoid the bias of the designer in shaping the system development, because neural networks are flexible, robust, and tolerant of noise. In the project, we will develop a new evolutionary algorithm to evolve neural networks. The algorithm is ablt to avoid the disadvantages of back propagation to improve the solution quality of recurrent networks. To demonstrate that our approach is an efficient approach, our algorithm was applied to two benchmark problems, including Boolean function learning and regular language recognition. Furthermore, we studied two simple autonomous agent problems that are artificial ant problems and the simulation of playing football. We will continue to apply this algorithm to practical neural-based autonomous agents.

Keywords: Autonomous Agent, Recurrent Neural Networks, Evolutionary Computation, Genetic Algorithm, Artificial Ants.

二、計畫緣由與目的

An autonomous agent is a system situated within an environment that senses the environment and acts on it to fulfill a set of goals. Autonomous means that an agent operates without any external agent control and adaptation means that an agent is able to improve its performance over time. Behavior-based control systems have been successfully offered to design autonomous agents. An autonomous agent must possess self-adaptive ability and tolerate noise to adapt its behavior to any changes of its environment. Neural networks have been considered to well achieve the demands and to avoid the bias of the designer in shaping the system development, because neural networks are flexible, robust, and tolerant of noise.

Many learning approaches have been proposed to train ANNs. One of the most widely

used approaches is back propagation which is a gradient decent search algorithm. Back propagation is susceptible to being trapped into local optima and is inefficient in terms of searching for a global minimum of functions which are vast, multimodal, and nondifferentiable. In addition, back propagation produces worse recurrent networks than non-gradient methods when an application requires memory retention according to a theoretical perspective.

An evolutionary algorithm is a non-gradient method and a very promising direction for global search to avoid the disadvantages of back propagation. At the same time, evolutionary algorithms do not need the gradient and differentiable information when they are applied to train ANNs. Recently, they have been successfully applied to train or evolve various ANNs structures for many application domains. Several pertinent researches have demonstrated that the search speed of evolutionary algorithms is competitive with back propagation if genetic operators are well designed.

In this project a new method called family competition evolutionary algorithm (FCEA) is proposed for training neural networks. FCEA is a multi-operator approach which combines three mutation operators: decreasing-based Gaussian mutation, self-adaptive Gaussian mutation, and self-adaptive Cauchy mutation. The performance of these three mutations heavily depends on the same factor called *step size*. FCEA incorporates the family competition and adaptive rules for controlling step sizes to construct the relationship among these three mutation operators. The family competition is derived from *(1−λ)-ES* and acts as a local search procedure.

三、結果與討論

The basic structure of the FCEA is as follows (Fig. 1): N individuals (ANNs) are generated as the initial population. Then FCEA enters the main evolutionary loop, consisting of three stages in every iteration: decreasing-based Gaussian mutation, self-adaptive Cauchy mutation, and self-adaptive Gaussian mutation. Each stage is realized by generating a new quasi-population (with N ANNs) as the parent of the next stage. As shown in Fig. 1, these stages differ only in the mutations used and in some parameters. Hence we use a general procedure ``FC_adaptive" to represent the work done by these stages.

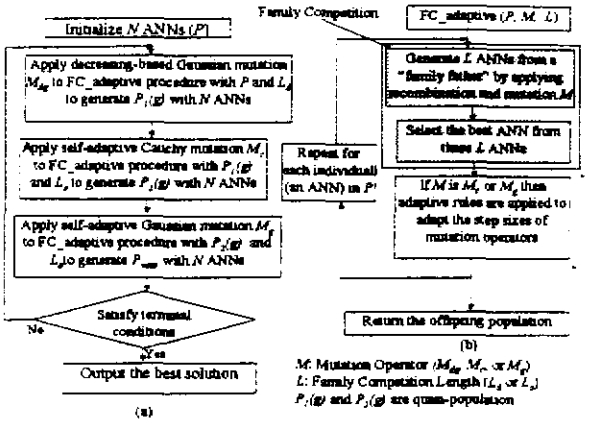

Fig. 1: Overview of FCEA

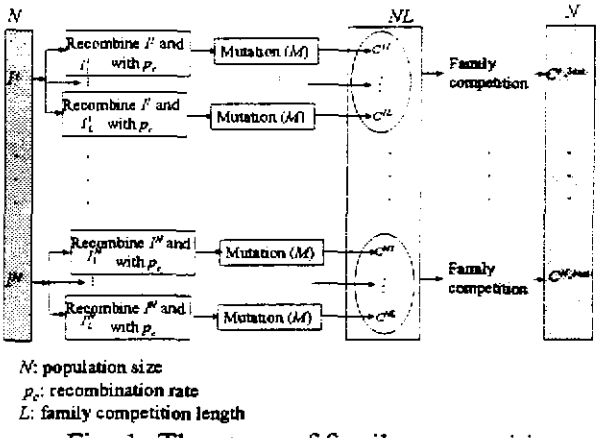

Fig. 1: The steps of family competition

The FC_adaptive procedure employs three parameters, namely, the parent population (P, with N solutions), mutation operator (M), and family competition length (L), to generate a new quasi-population. The main work of FC_adaptive is to produce offspring and then conduct the family competition (Fig. 2). Each individual in the population sequentially becomes the ``family father." Here we use $I^l$ as the ``family father" to describe the procedure of the family competition. With a probability p_c, this family father and another ANN (I^l_{rl}) randomly chosen from the rest of the parent population are used as parents to do a recombination operation. Then the new offspring or the family father (if the recombination is not conducted) is operated on by the mutation to generate a offspring (C^{l1}). For each family father, such a procedure is repeated L times. Finally L ANNs ($C^{l1}, \ldots, C^{lL}$) are produced but only the one (C^{l_best}) with the lowest objective value survives. Since we create

L ANNs from one ``family father'' and perform a selection, this is a family competition strategy. We think this is a way not only to avoid the premature but also to keep the spirit of local searches.

A. Artificial Ant Problems

An artificial ant problem, i.e., tracker task ``John Muir Trail'' is that a simulated ant is placed on a two-dimensional toroidal grid that contains a trail of food. The ant traverses the grid to collect any food encountered along the trail. This task requires to train a neural network, i.e., a simulated ant, that collects the maximum number of pieces of food during the given time steps.

Fig. 3 presents this trail. Each black box in the trail stands for a food unit. According to the environment of, the ant stands on one cell, facing one of the cardinal directions; it can sense only the cell ahead of it. After sensing the cell ahead of it, the ant must take one of four actions: move forward one step, turn right 90^o, turn left 90^o, and no-op (do nothing). In the optimal trail, there are 89 food cells, 38 no food cells, and 20 turns. Therefore, the number of minimum steps for eating all food is 147 time steps.

Fig. 4 depicts a typical search behavior and the traveled path of a simulated ant that is controlled by our evolved neural network. The number in the cell is the time step to eat the food. The symbol * denotes a cell traveled by an ant when the cell is empty. Fig. 4 indicates that the ant requires 195 time steps to seek all 89 food pieces in the environment.

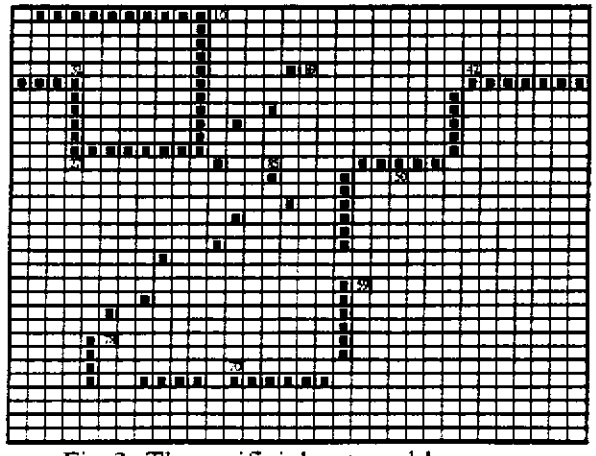

Fig 3. The artificial ant problem.

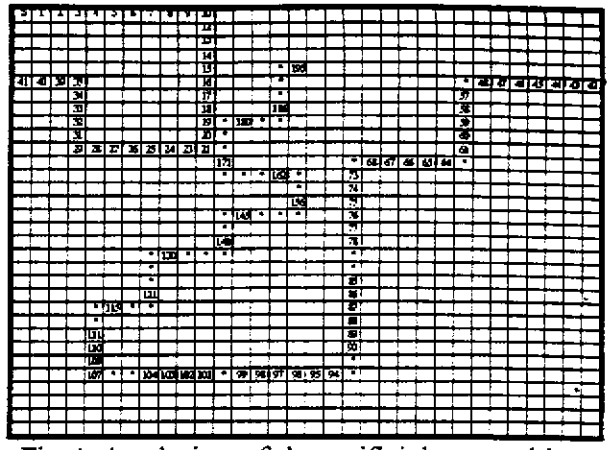

Fig 4. A solution of the artificial ant problem shown in Fig. 3.

B. The Simulation of Playing Football

FCEA aimed at developing a simulated robot capable of performing a sequence of behavior: an agent learns to play football. The simulation of Maniezzo's football environment is that both robot and ball are randomly set to any positions on a field. To shoot ball into opponent's goal, an agent must learn four sequential tasks: reaching the ball, getting the ball, dragging it and reaching opponent's goal, and last kicking it into the goal. In sequential behavior environment, the controller will be triggered when environment is changed, and then the controller will refer previous actions to decide next action. The intelligent agent needed seven sensory inputs: two inputs for the distance of ball, two inputs for the distance of opponent's goal, two inputs for own goal (an agent must intercept ball in two-player environment when opponent got the ball earlier) , and one input for ball status. The ball status (free or controlled by a player) can be considered as a trigger input in this problem.

Fig. 5 shows a typical evolutionary process and the results of one robot player environment. Fig. 5(a) shows an ideal path obtained from an experiment. Fig. 5(c) shows the paths of the best simulated robot player at 1^{th}, 20^{th}, 50^{th}, and 100^{th} generations.

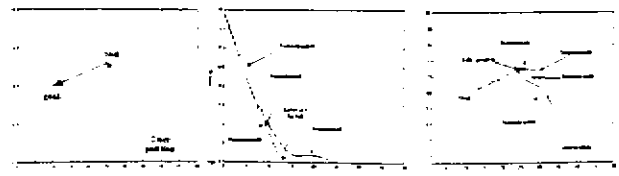

Fig. 5. A typical simulated results of one robot player environment.

四、計畫成果自評

This project presents that FCEA is a stable approach to train both feedforward and recurrent neural networks with the same parameter settings for the artificial ant problem and the simulation of a playing football. The experiments verify that the proposed approach is very competitive with other evolutionary algorithms, including genetic algorithms and evolutionary programming. We believe that the flexibility and robustness of our FCEA makes it a highly effective global optimization tool for other task domains. We have published two journal papers and two conference papers by using FCEA on training neural networks.

五、參考文獻

[1] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent Neural Networks," *IEEE Trans. on Neural Networks*, vol. 5, no. 1, 54-65, 1994.

[2] T. Bäck and H. P. Schwefel, "An overview of evolution algorithms for Parameter Optimization," *Evolutionary Computation*, vol. 1, no.1, 1-23, 1993.

[3] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robotics and Automation*. Vol. 2, no. 2, 14-23, 1986.

[4] R. A. Brooks, "Intelligent without representation. Artificial Intelligence," vol. 47, 139-159, 1991.

[5] M. Colombetti and M. Dorigo, "Train agents to perform sequential behavior. *Adaptive Behavior*." 2 (3), 247-275, 1994.

[6] M. Dorigo and U. Schnepf, "Genetic-based machine learning and behavior based robotics: a new synthesis," *IEEE Trans. on System, Man. and Cybernetics*, vol. 23,141-154. 1993.

[7] L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval-schemata," *In Foundations of Genetic Algorithms 2*. 187-202, 1993

[8] D. Floreano and F. Mondada, 1996. Evolution of homing navigation in a real Mobile robot," *IEEE Trans. on System. Man. and Cybernetic*. vol. 26, no. 6, 1996.

[9] D. B. Fogel, L. J. Fogel, and V. W. Porto. "Evolving neural networks," *Biological Cybernetics*, 487-493, 1990.

[10] Fogel D. B. and Atmar, J. W. 1993. Comparing genetic operators with Gaussian mutations in simulated evolutionary processes using linear systems. *Biological Cybernetic*. vol. 63. 111-114.

[11] D. E. Goldberg, *Genetic Algorithms in search, Optimization & Machine Learning*, Reading. MA: Addison-Welsley, 1989.

[12] C. Z. Janikow and Z. Michalewicz, "An experimental comparison of binary and floating point representations in genetic algorithms," *In Proceeding of the Fourth Int. Conference. on Genetic Algorithms*, 31-36, 1991.

[13] D. Jefferson, et al., "Evolution as a theme in artificial life: The genesys/tracker system," In *Artificial Life II: Proceedings of the Workshop on Artificial Life*. 549-577, 1991.

[14] J. Koza, "Genetic evolution and co-evolution of computer program," In *Artificial Life II: Proceedings of the Workshop on Artificial Life*. 603-629, 1991.

[15] V. Maniezzo, "Genetic evolution of topology and weight distribution of neural networks," *IEEE trans. On Neural Networks*. vol. 5 no. 1, 39-53, 1994.

[16] L. A. Meeden, "An incremental approach to developing intelligent neural network controllers for robots," *IEEE Trans. on System. Man. and Cybernetics*. vol. 26, no. 6, 474-485. 1996.

[17] D. Parisi, F. Cecconi, and D. Nolfi, "Econets: Neural networks that learn in an environment," *Networks*, vol. 1,149-168, 1990.

[18] J. D. Schaffer, R. A. Caruana, and L. J. Eshelman, "Combinations of genetic algorithms and neural networks: A survey of the state of the art," In *Proceeding International Workshop on Combinations of Genetic Algorithms and Neural Networks*. 1-37, 1992.

[19] S. W. Wilson, "Classifier systems and the animat problem," *Machine Learning*, vol. 2, 199-228, 1987.

[20] J.-M. Yang and C.-Y. Kao, "A Family competition evolutionary algorithm for automated docking of flexible ligands to Proteins," *IEEE Transaction on Information Technology in Biomedicine*. vol. 4, no. 3, pp. 225-237, 2000.

[21] J.-M. Yang, J.-T. Horng, and C.-Y. Kao, "A genetic algorithm with adaptive mutations and family competition for training neural networks," vol. 10, no.5. *International Journal of Neural Systems*. pp. 333-352, 2000.

[22] J.-M. Yang and C.-Y. Kao, "A robust evolutionary algorithm for training neural networks." to appear in vol. 10, no. 3. *Neural Computing and Application, 2001*.

[23] B. Yamauchi and R. D. Beer, "Integrating reactive, sequential, and learning behavior using dynamically neural networks," *In From Animals to Animats III: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*. 382-391. 1994.