A Study on the Memory Schemes for Genetic Network Programming

YE, Fengming

情報生産システム工学専攻
進化型計算システム研究

2011年03月
In computer science, evolutionary computation whose essential concept comes from Darwin's Evolution Theory is a subfield of Artificial Intelligence (more particularly Computational Intelligence). It uses the iterative progress, such as growth or development in a population. In this population, a guided random search is executed using parallel processing to achieve the desired end. Such processes are often inspired by biological mechanisms of natural evolution.

As an effective way to solve optimization problems, evolutionary computation has been drawing attentions and endeavors for decades. A large number of studies on evolutionary computation techniques have been executed and many significant research achievements have been obtained, such as Genetic Algorithm (GA) by J. Holland, Genetic Programming (GP) by J. Koza, Evolutionary Programming (EP) by L. Fogel and Evolutionary Strategy (ES) by I. Rechenberg and H. Schwefel.

Traditionally, research on evolutionary algorithms (EAs), e.g., GA has focused on stationary optimization problems, whose environment situations remain fixed during the evolution process and they are precisely given in advance. For these stationary problems, the aim of EAs is to quickly and precisely locate the optimal solutions in the search space of a large scale. However, the environments of many real-world problems are more dynamical and complicated for which the aim of EAs is to find successive behaviors for agents making judgments and taking proper actions for the current environment. So, the optimal solutions of these problems are no longer the locus in the solution space, but a series of action regulations for agents. So, the traditional EAs such as GA usually fail to solve these dynamical problems. But, some researchers have introduced a kind of memory schemes, which stores historical informations on good solutions and reuse them later, to enhance the performance of EAs in dynamical problems. The adoption of the memory schemes has proved to be able to enhance EA's performances in many applications, especially in dynamical environments. The basic principle of the memory schemes is to store the information, e.g., good solutions, from the current generation and reuse it in later generations.

In the last decade, a new graph-based evolutionary algorithm named Genetic Network Programming (GNP) was developed. It is devised to deal with problems in dynamic environments effectively and efficiently and it is found that GNP has some advantages over traditional evolutionary computation techniques especially in dynamical environments. As the name suggests, GNP adopts directed graphs rather than trees as their phenotype, so, in some research communities, GNP is considered as a special variation of GP.
Since many research has shown significant improvements when EAs are equipped with memory schemes. So, it is expected that GNP will be enhanced if it also utilizes a well designed memory scheme. Originally, as GNP is devised to solve dynamical problems effectively, we think GNP will be also enhanced by employing the memory scheme in dynamical problems considering the profit of GA using the memory scheme. The main objective of the research is to develop a memory scheme to enhance the architecture of standard GNP and improve its performance when solving dynamical problems.

In GNP, the route of GNP which consists of a series of successive GNP transitions from node to node corresponds to the agent's actions. However, it is generally found that not all of the GNP nodes and connections, but only a part of them is included in the route and others are not used by agents at all. Thus, one of the points of the research is that we can extract and accumulate the information on the transitions that are carried out by agents and reuse the accumulated information later. Another point is that we can design a precise criterion to evaluate all the branches of nodes over the population and make use of them to guide the evolution process.

In Chapter 2, the memory scheme named GNP with rules was proposed. In this method, the node branches are focused on, which are defined as GNP rules. The judgment and processing functions embedded in the nodes are carried out by the agents in the problem environments. So, GNP rules can be considered as the set of rules to guide the agents during task execution. The memory stores the GNP rules and evaluates them using the fitness values of individuals in each generation. Before undergoing the genetic operators, the worst individuals are replaced by the new ones constructed by the selected rules from the memory. And the selection of the rules obeys a probabilistic policy that the rules with higher evaluation have higher probability to be selected.

In Chapter 3, another memory scheme named GNP with reconstructed individuals (GNP-RI) is studied, which stores the whole node transitions used by the agents, i.e., the GNP routes. Because only the part of the nodes and connections are used during the execution of the task in GNP, so the solution of GNP for the concrete problem is represented only by the used nodes and connections instead of the whole individuals. The aforementioned GNP rule is a connection from node to node indicating only one judgment or action of the agents. But, the complete node transition consists of successive judgments and actions of the agents in GNP-RI.
which can be considered as the series of regulations guiding the agents to solve concrete problems. In GNP-RI, the GNP routes of the best individuals are also stored in the memory in each generation. And they are used to modify the gene structures of the worst individuals before undergoing genetic operators. So, the worst individuals can learn more knowledge from the elite ones than those of GNP with rules.

In Chapter 4, a new explicit memory scheme for GNP named GNP with route nodes (GNP-RN) is proposed to overcome the following disadvantages of the memory reuse mechanism in GNP-RI. Firstly, the gene structures of the worst individuals become more similar to the elite ones' in GNP-RI which means a loss in population diversity. Secondly, the worst individuals learn experiences from the elite ones without considering their own situation but other ones' experiences may not suitable for them at all. In GNP-RN, although the memory also stores the best GNP routes, a new kind of nodes: route nodes are added into the individuals. When the agents transfer to the route nodes, they will directly follow the judgments and processings on the selected GNP route. If the agent meets a judgment node on the route, it will make the judgment of the node using the current condition in the environment. If the judgment result satisfies the condition indicated by the route, the agent will move to the next node on the route, otherwise, it will directly jump to the judgment node right after the next processing node on the route. So, individuals can learn knowledge from elite ones without modification of the gene structures and the learned knowledge is suitable for their own environment situation.

In Chapter 5, an architecture named adaptive mutation in SARSA learning of GNP (GNP-SLAM) is studied, which uses SARSA learning to evaluate the node branches and records its information during the evolution. When agents are carrying out the tasks, the SARSA learning is used to evaluate the Q values of the nodes and their branches. The Q value information is kept on updating and recorded in the memory, where it is used to adjust the mutation rates adaptively and guide the mutation directions. In the adaptive mutation, the node branches with lower Q values have higher probability to be changed, in addition, the nodes to which the mutated branches should change are guided according to the Q information.

In conclusion, simulation results show that the proposed memory schemes can enhance the architecture of GNP. In addition, GNP-RN can obtain better performance than GNP-RI due to its memory reuse mechanism and cooperating with adaptive mutation can further strengthen GNP-RI as well.