Е. С. Семенкин, М. Е. Семенкина

ПРОЕКТИРОВАНИЕ АНСАМБЛЕЙ ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ САМОКОНФИГУРИРУЕМЫМ АЛГОРИТМОМ ГЕНЕТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Самоконфигурируемый алгоритм генетического программирования с модифицированным оператором равномерного скрещивания, использующим селективное давление на этапе рекомбинации, применяется для автоматического формирования ансамблей интеллектуальных информационных технологий. Символьные выражения, нейронные сети или их комбинации являются членами ансамблей, которые генерируются также автоматически при помощи самоконфигурируемого алгоритма генетического программирования. Сравнительный анализ эффективности подхода был проведен на тестовых и реальных практических задачах.

Ключевые слова: генетическое программирование, само-конфигурация, нейронные сети, символьная регрессия, ансамбли, автоматическое проектирование, задачи классификации.

UDK 519.68

O. E. Semenkina

EFFECTIVENESS COMPARISON OF ANT COLONY AND GENETIC ALGORITHMS FOR SOLVING COMBINATORIAL OPTIMIZATION PROBLEMS*

This paper considers ant colony optimization, genetic algorithm and parallel versions of these methods for solving traveling salesman problem.

Keywords: Genetic algorithm, Ant colony optimization, Traveling Salesman Problem, Parallelization.

One of the most interesting and topical methods for optimization problems solving are stochastic algorithms working with many current solutions at the same time. This paper considers ant colony optimization (ACO) [1] and genetic algorithm (GA) [2] and also parallel versions of these methods. These algorithms can be very useful for finding approximate solutions of complex optimization problems. Both algorithms are nature-inspired optimization metaheuristics, where ACO is based on the behavior and organization of ant colonies in their search for food whereas GA is based on some principals of evolution.

Investigation of effectiveness of ACO and GA was conducted by solving well-known problem of combinatorial optimization namely Traveling Salesman Problem (TSP). Software implementation of these algorithms was programmed in C++ and the library MPICH2 was used for parallelization.

In the genetic algorithm for the TSP a chromosome is represented as permutation of the n numbers (numbers of cities). That is why some standard operations have a few changes, but many adjustable parameters in GA remains such as mutation probability, the type of selection – tournament selection (parameter is the size of the tournament), proportional and rank selection (linear or exponential ranking), population size and number of generations. Solutions in ACO are also represented as permutation of n cities and ants chose next city using taboo-list and pheromone matrix at every stage. The same to GA, ACO have rather adjustable parameters: ant colony size (m), number of iteration, evaporation rate (ρ), relatively importance of previous search experience (α) and relatively importance of the distance between cities (β).

At first efficiency of each algorithm on a test task was investigated (a grid 5 on 5 cities) and regularities in dependence of algorithms efficiency on its settings were revealed. ACO algorithm shows the best results at the following settings: $\alpha = 1$, $\beta = 10$, $\rho \in [0.5, 0.7]$ and quantity of ants approximately equals to number of the cities. The best settings for GA are tournament selection with small tournament size and low or very low mutation rate. This test task solution ACO showed better results than the GA because ACO rely on distance between the cities at initialization and begins to work with rather a quite good route while GA begins with an absolutely random population. Therefore, ACO requires considerably less computational resources on the problem with a lot of decisions. Therefore reliability of algorithms was investigated on more complex test task with a number of the cities equals to 30 and called Oliver30. After the completion of algorithms work local search was applied to the best solutions. For both algorithms identical number of calculations, namely 30 individuals (ants) and 10000 generations (iterations) was given.

*The study was supported by The Ministry of education and science of Russian Federation, project № 16.740.11.0742, 14.740.12.1341 and 11.519.11.4002.

Reliability was calculated by averaging on 30 runs of algorithms. GA found the shortest route at least once for all variants of the settings whereas ACO didn't find the best route in all cases. The average length of a route for genetic algorithm varied in an interval from 429.172 to 459.028 whereas for ant algorithm it is in an interval from 423.8171 to 490.5966. It means that ACO has a wide scatter of reliability and more strongly depends on a choice of settings. In comparison with genetic algorithm ACO can find both considerably the best route and essentially the worst at an unsuccessful choice of parameters. Thus it is possible to say that for this task the GA works better, than ACO because it is more constant and predictable for its various settings.

GA and ACO show good reliability at enough amount of resources, but this amount and, therefore, the operating time of algorithms grows quickly with the dimension increase of the task. One of this problem solution is parallelization.

Many different variants of parallelization do exist, but this paper considers the most interesting type – Coarse-Grained evolutionary algorithms [3]. In this case there are some independent populations on every core of processor. They periodically exchange a few individuals, and this process is called migration. That is why two additional parameters are necessary – interval of exchange and number of migrating individuals. Parallel evolutionary algorithms of this type are very popular, but hard enough to understand because the effects of migration are not fully investigated. In this research clique topology of individuals exchange was chosen.

Reliability of algorithms was investigated at various settings of parameters and at various quantities of cores, and respectively populations. The interval between exchanges of individuals was established on 10 generations and in each population one best individual migrated. At increase in quantity of cores, i. e. separate populations, reliability of algorithms didn't change essentially, though some changes of reliability for various settings were observed besides operating time of algorithms was essentially reduced. Using one core for GA takes about 30 seconds for one run, using two cores it takes about 16 seconds, and using four ones – about 8 seconds. For ACO these values are 0.55, 0.27 and 0.16 seconds respectively. That is parallelized algorithm can significantly reduce the running time without reliability loss. Thus, it is possible to say that if the migration rate is sufficient, the parallelization of algorithms at least doesn't reduce efficiency of algorithms. However it is necessary to find more precisely migration rate at which algorithms work best of all.

For a grid 5 on 5 cities with fixed quantity of resources efficiency of both algorithms on 4 cores for their various settings and the various intervals sizes of exchange of one best individual was considered. At increase in an interval between migrations, reliability increased to a certain limit, and then started to decrease. It can be explained by several facts. At low migration rate population on the cores become almost isolated from each other and work independently. At the same time if the migration rate is high, there is no essential difference with one big population. Besides there is a known principle in genetics: favorable signs extend quicker when communities are small, than when they are big. Thus it is possible to draw a conclusion that parallelization essentially reduces operating time of algorithms without reducing reliability and even increasing it if settings is right.

As a part of future plans, may be proposed search for more effective methods of parallelization to accelerate the work of the algorithm due to the effective interaction of diverse populations, developing on different cores and different topologies sharing individuals, and various topologies sharing individuals.

References

1. Dorigo M., Gambardella L. M. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem // IEEE Transactions on Evolutionary Computation. 1997. P. 53–66.

2. Holland J. H. Adaptation in Natural and Artificial Systems // The University of Michigan Press. 1975.

3. Grosso P.B. Computer Simulations of Genetic Adaptation: Parallel Subcomponent Interaction in a Multilocus Model // Unpublished doctoral dissertation ; The University of Michigan. 1985.

О. Е. Семенкина

СРАВНЕНИЕ ЭФФЕКТИВНОСТИ МУРАВЬИНОГО И ГЕНЕТИЧЕСКОГО АЛГОРИТМОВ ПРИ РЕШЕНИИ ЗАДАЧ КОМБИНАТОРНОЙ ОПТИМИЗАЦИИ

Рассматриваются алгоритм муравьиных колоний, генетический алгоритм и параллельные версии этих методов для решения задачи коммивояжера.

Ключевые слова: генетический алгоритм, алгоритм муравьиных колоний, задача коммивояжера, параллелизация.