Agent-Based Evolutionary System for Traveling Salesman Problem

Rafał Dreżewski, Piotr Woźniak, Leszek Siwik

Department of Computer Science
AGH University of Science and Technology, Kraków, Poland
drezew@agh.edu.pl

Abstract. Evolutionary algorithms are heuristic techniques based on Darwinian model of evolutionary processes, which can be used to find approximate solutions of optimization and adaptation problems. Agent-based evolutionary algorithms are a result of merging two paradigms: evolutionary algorithms and multi-agent systems. In this paper agent-based evolutionary algorithm for solving well known Traveling Salesman Problem is presented. In the experimental part, results of experiments comparing agent-based evolutionary algorithm and classical genetic algorithm are presented.

1 Introduction

Evolutionary algorithms are heuristic techniques for finding (sub-)optimal solutions of continuous and discrete optimization problems [2]. They can be used in the case of highly multi-modal problems because, after using some special mechanisms for maintaining population diversity (niching techniques), the population of individuals can simultaneously detect multiple optima basins of attraction or avoid basins of attraction of local optima.

Evolutionary multi-agent systems (EMAS) are agent-based evolutionary algorithms resulting from research on merging multi-agent systems and evolutionary computations. Merging of the two paradigms leads to decentralization of evolutionary processes. In EMAS approach we usually have environment composed of computational nodes ("islands"), agents, and resources. Agents can reproduce, die (when they run out of resources) and interact with environment and other agents. There is also competition for limited resources among agents. In order to realize selection process "better" (what means that they simply better solve the given problem) agents are given more resources from the environment and "worse" agents are given less resources. This results in decentralized evolutionary processes in which individuals (agents) make independently all their decisions concerning reproduction, migration, etc., taking into consideration conditions of the environment, other agents present within the neighborhood, and resources possessed. In CoEMAS (co-evolutionary multi-agent system) approach we additionally have many species and sexes of agents which interact with each other [3]. Thus we can model co-evolutionary interactions and sexual selection mechanisms, which promote diversity of population.

Traveling Salesman Problem (TSP) is old and well known NP-hard problem. Many real-life problems can be modeled as TSP or some variants of it. The TSP problem has

quite simple concept. Traveling salesman has to visit all cities just one time, and in the end go back to the starting city. He knows cost of traverse between cities and has to plan the shortest route (with minimal cost). Because TSP is NP-hard problem we must rather try to find satisfying sub-optimal solution using some heuristic technique (simulated annealing, ant colony optimization, evolutionary algorithm) because search for global optima would be too expensive. Evolutionary algorithms are well suited to solve such problems because they usually find approximate solutions, or saying this in other words—they locate basins of attraction of optima. An introduction to solving TSP with the use of evolutionary algorithms may be found for example in [2, 4]. Good results are usually obtained when some heuristic technique is combined with a local search technique (hybrid approaches).

When we try to solve TSP with evolutionary algorithm we will realize that it has quite simple fitness function. Each of solutions is a permutation of cities. To compare the quality of each solution, it is sufficient to calculate length of the route for each permutation of cities and choose the shortest one. But the choice of route representation and genetic operators is usually not so obvious. It's rather not recommended to use binary representation in TSP. In the last few years, mainly three representation techniques were used: adjacent method, order method and path method. Each of them has its own appropriate genetic operators, but the route is connected sequence of cities in all techniques.

The main goal of this paper is to apply agent-based model of evolutionary computations to TSP and to compare the obtained results with those obtained by standard genetic algorithm. In the following sections we will present the architecture of the agent-based evolutionary system for solving TSP, and results of preliminary experiments.

2 Agent-Based Evolutionary System for Solving TSP

In this section the architecture and algorithms of the agent-based evolutionary system for solving TSP are presented. The system was implemented with the use of agent-based evolutionary platform JAgE [1]. This platform has all necessary mechanism for the realization agent-based models of (co-)evolutionary computations: computational environment composed of nodes (islands), agents, agent-agent and agent-environment communication mechanisms, and mechanisms for distributed computations.

The platform has component architecture, what means that every component is responsible for selected tasks of the system. There can be distinguished four basic components [1]:

- *Workplace*—main component which controls all computation processes. Every computation node has its own instance of *Workplace*. It includes one (main) agent which controls all descendant agents. *SimpleWorkplace* is a basic implementation of *Workplace*. It stores addresses which are needed for exchanging information between agents.
- *ConnectionManager* is responsible for connections, communication, and transmission of objects between computational nodes.
- *ControlManager*—its main task is monitoring and controlling computation processes.

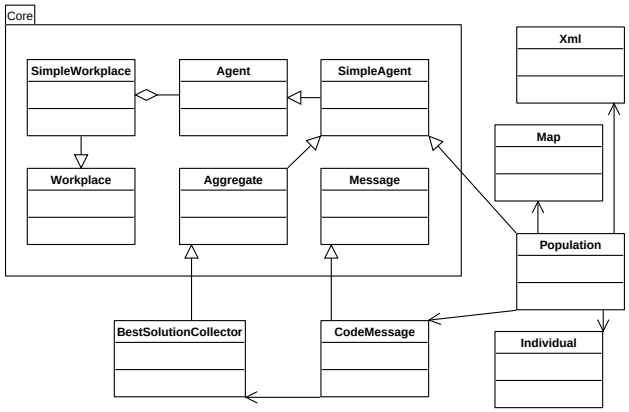

Fig. 1. Classes of the agent-based evolutionary system for solving TSP and their relations with jAgE core classes

– *ResourceManager* is a component which includes methods for system resource management.

Agent-based evolutionary system for solving TSP was implemented as a component of jAgE platform (see fig. 1). *XML* class is responsible for supplying external data to evolutionary algorithm solving TSP. *Individual* class includes definition of chromosome, methods for route length calculation and stores auxiliary information about individuals. The core of computations—where evolutionary process is realized—is the *Population* class. In this class evolutionary algorithm solving TSP was implemented.

Algorithm 1. Genetic algorithm

```
1  Generate random population;
2  Calculate length of route for each individual;
3  Calculate their absolute and relative fitness;
4  while stopping condition is not fulfilled do
5  |  Send the best individual to best solution collector class;
6  |  for i ← 0 to NumberOfIndividuals do
7  |  |  Add to the base population individual which was selected via roulette selection
8  |  end
9  |  for i ← 0 to NumberOfIndividuals/2 do
10 |  |  crossover two randomly selected individuals;
11 |  end
12 |  Perform mutation of the offspring;
13 |  Calculate length of route for each individual;
14 |  Calculate their absolute and relative fitness;
15 end
```

In the system described, two algorithms were implemented: agent-based evolutionary algorithm—EMAS (see alg. 2) and genetic algorithm—GA (see alg. 1). Genetic algorithm served as a reference point during experiments.

Algorithm 2. Agent-based evolutionary algorithm

```
1  Generate random population of agents;
2  Calculate length of route for each agent;
3  Calculate their absolute and relative fitness;
4  Distribute resources between agents;
5  while stopping condition is not fulfilled do
6      if agent enters the given population as a result of migration then
7          Add it to the population;
8      end
9      Find the best agent in population;
10     if it is better than previous best one then
11         Send him to the best solution collector class;
12     end
13     if if the amount of resources of the agent that wants to crossover is above the
       minimum then
14         Select partner and perform crossover;
15         Assign children to temporary population;
16     end
17     Mutate offspring;
18     Merge temporary population with base population;
19     Give back the rest of agents' resources to the environment;
20     if agent's age exceeds limit then
21         Remove it from the system;
22     end
23     Calculate length of route for each agent;
24     Calculate their absolute and relative fitness;
25     Distribute resources between agents;
26     if agent can not perform any action then /* agent has too few resources */
27         Remove it from the system;
28     end
29     if k steps of the algorithm passed from the previous modification then
30         Modify parameters of mutation and crossover;
31     end
32 end
```

3 Experimental Results

In this section results of experiments, which goal was to compare implemented agent-based evolutionary system (EMAS) and genetic algorithm (GA), are presented. The basic features of algorithms which were compared during experiments include the working time and the number of steps needed to find an (sub-)optimal solution. The results of experiments presented will also answer the question of how each parameter influence the course of evolutionary processes.

Input data should be identical for all algorithms in order to correctly estimate the quality of each technique. The problem consisted of 30 cities, identical for all techniques, was used during experiments. Certain number of cities from this problem was chosen to change the number of cities in consecutive experiments. In experiments 10, 15, 20, 25 and 30 cities samples were used. During experiments also Euclidean problem with 92 cities from TSP library [5] was used.

During preliminary experiments many different parameters' values were tested. It was observed that some configurations give better results, but others result in shorter working time of the algorithm. In the case of GA there were not too many possibilities:

Table 1. Values of parameters used during experiments

	EMAS	**GA**
Number of individuals	100 on each of 5 islands	100
Minimal number of individuals	10	-
Probability of crossover	0.8	0.7
Probability of mutation	0.01	0.1
Migration cost	0.3	-
Crossover cost	0.18	-
Mutation cost	0.15	-
Environment resources	equal to the initial number of individuals	-
Minimal amount of resources	0.04	-
Frequency of migration	after each 500 steps	-
Individual's life time limit	15 steps	-

only the number of individuals and parameters used during crossover and mutation. EMAS gives much more options. Optimal values of parameters used during main part of experiments for each technique are shown in table 1. During experiments the maximal number of steps (stopping condition) was set on the basis of improvements of average fitness of the population.

Number of individuals It is very important parameter of evolutionary algorithm. In the case of multi-modal optimization problems it affects the number of located basins of attraction of local and global optima. In the case of multi-objective optimization problems it affects the coverage of Pareto frontier. Relatively, greater number of individuals is desirable in the case of hard computational problems, with many local optima, but the size should be carefully chosen because too great value can considerably slow down the computational process.

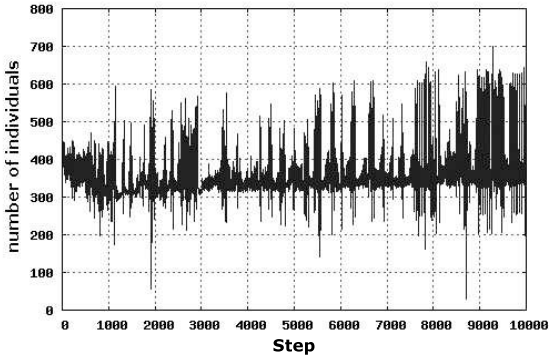

Fig. 2. Number of individuals in EMAS (92 cities, average from 5 experiments)

In the case of GA, the number of individuals is the same in all stages of algorithm. Each base and descendant population has the same size, so the number of individuals never changes. Results of experiments show that the number of individuals in EMAS grows during initial faze but then stabilizes at the given level, which depends on the total amount of resources present in the system. Number of individuals oscillates around this value during the consecutive steps of algorithm (see fig. 2).

There was also observed the fact that some parameters' values influence the number of individuals in the EMAS. One of them is minimal resource amount, which agent must possess in order to perform some activities in the system. Low minimum amount needed for crossover and mutation causes that average number of individuals grows and evolutionary process runs faster (and less precisely as well). It's connected with the fact that resources of the environment are limited and dependent on the initial number of individuals. When the population size is very large it results in limited activities of the agents because each of agents receives much less resources from the environment than in the case when there are less agents in the population. Other parameter which influences the population size is crossover cost. When it is too high it causes that less agents are able to initialize the process of reproduction and crossover, and as a result less new agents enter the population.

Population diversity In our experiments population diversity was measured as the number of different solutions present within the population in the given step.

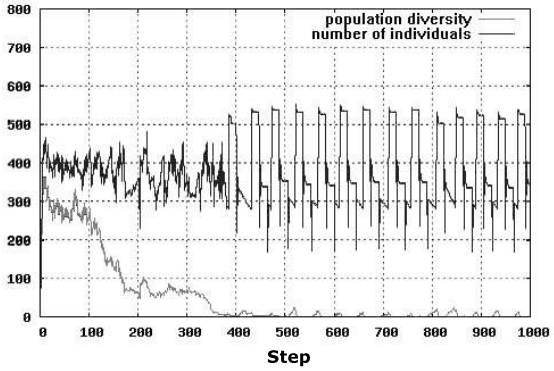

Fig. 3. Population diversity in EMAS (25 cities, average from 5 experiments)

In the case of EMAS it's easy to observe the tendency to slowly reduce the diversity during the experiment (see fig. 3). This phenomena is connected with the process of approaching the areas where optima are located—the search is then limited only to the very small areas. At the time of reducing diversity algorithm usually found good solution for 25 and 30 cities. Interdependence of diversity and the number of individuals present in the population can also be observed. When the diversity goes down then

the number of individuals oscillates with greater amplitude (see fig. 3). This is due to the fact that there are more very similar individuals (with the same or almost the same fitness) in the population, and as a consequence resources from the environment are more evenly distributed between agents. Dying off and reproduction of agents is realized in cyclic way.

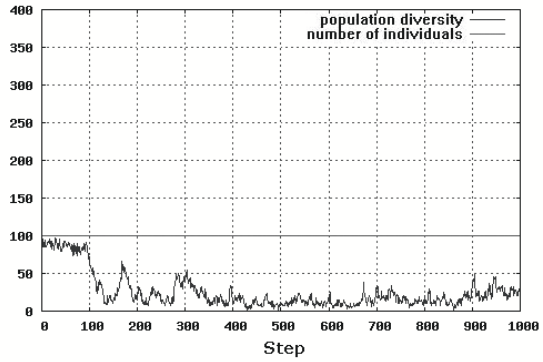

Fig. 4. Population diversity in GA (25 cities, average from 5 test). Genetic algorithm

GA with roulette selection shows tendency to significant loss of diversity during first 100 steps (see fig. 4). In the next steps, diversity keeps on safe stage. Contrary to the EMAS, increasing the number of cities does not result in differences in the diversity of population.

Step effectiveness Step effectiveness shows how fast (in how many steps) algorithm finds the best solution. Experiential test of the stability of both techniques was carried out. Algorithm was recognized as stable when in most runs it was giving similar solutions. GA was finding 15% worse solution then recognized as global minimum. Some specific schema of this method was noticed. Algorithm quickly finds better solution in first steps (usually after 100 or 200 steps) and then gets stuck in the local minimum or during the rest time of experiment only slightly improves founded solution. Tests with 25 and 30 cities showed advantage of EMAS over GA. First method still returned good solutions, 10% worse than recognized as global minimum. GA went considerably poorer, and found 30% worse solution for 25 cities and 35% worse for 30 cities.

Time effectiveness During experiments we also tried to compare GA and EMAS on the basis of time needed to find good solution. Unfortunately, it turned out to be the very hard task to prove superiority of some method over other looking only on the algorithms' working time. Presented techniques differ from each other on various levels and in so many details that comparison is not easy. First significant difference is population. GA has constant population size and it is equal to 100. In EMAS there are 5 islands with separated sub-populations. Initially, each island has 100 or 300 individuals. If we

Table 2. Working time of the algorithms for different problem sizes (average from 10 experiments)

	10	15	20	25	30	92
EMAS	10 sec	14 sec	21 sec	27 sec	37 sec	7 min
GA	8 sec	11 sec	14 sec	18 sec	24 sec	5 min

compare it with GA, the EMAS has five times more individuals in the initial step than GA. If we add the fact that in EMAS the number of individuals changes along the time, the difference in system load in the case of each algorithm is significant. These facts cause that EMAS should be, and in fact is, slower (see tab. 2). Other side effect of the fact that EMAS is more complex algorithm than GA is that it is much more configurable than GA, what gives more possibilities to tune the algorithm on the basis of the problem being solved.

4 Summary and Conclusions

In this paper agent-based evolutionary system for solving TSP was presented. There were two evolutionary algorithms compared during experiments: agent-based one and classical one. Experiments were carried out with the use of two problems with varying number of cities.

Results of preliminary experiments showed that EMAS obtained better results than GA, although the time needed to solve the problem was longer than in the case of GA. Longer run time mostly resulted from the fact that EMAS used larger initial population.

This was the first attempt to apply agent-based evolutionary algorithm to TSP—previously such approach was used mostly to solve multi-modal and multi-objective problems. Future research will certainly include further experimental verification of the proposed approach and the comparison to other "classical" and modern versions of evolutionary algorithms for solving TSP. Also, the application of EMAS to solving more complex problems like Vehicle Routing Problem (VRP) or VRP with Time Windows (VRPTW) is included in future research plans.

References

1. Agent-based evolution platform (jAgE). http://age.iisg.agh.edu.pl.
2. T. Bäck, D. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, 1997.
3. R. Dreżewski. A model of co-evolution in multi-agent system. In V. Mařík, J. Müller, and M. Pěchouček, editors, *Multi-Agent Systems and Applications III*, volume 2691 of *LNCS*, pages 314–323, Berlin, Heidelberg, 2003. Springer-Verlag.
4. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer -Verlag, 1996.
5. P. Moscato. TSPBIB home page. http://www.ing.unlp.edu.ar/cetad/mos/TSPBIB_home.html.