COMPARISON BETWEEN GENETIC ALGORITHMS AND ANT ALGORITHMS

Urszula Boryczka
Institute of Computer Science, University of Silesia
Żeromskiego 3, 41–200 Sosnowiec, Poland
phone/fax: (+48 32) 291 82 83
e–mail: uboryczk@us.edu.pl

ABSTRACT: Genetic algorithms and ant systems are the examples of algorithms applying the stochastic searching, parallel investigation, autocatalitic process or stigmergy to solve the optimization problems. In this paper, we concentrate on the representation of the one from the optimization problems - TSP for the genetic algorithm and the ant algorithm. We present sets of parameters and operators influenced the data structure in these algorithms. We discuss two different structures of the algorithms emphasizing the selection problem. We also describe the experiments we performed (e.g. 50 cities TSP). The essence of the comparison is the idea of the evolution - exploited in GA, but can improve the performance of the AA.

1 Introduction

The purpose of this work is to apply genetic algorithms (GA) and ant algorithms (AA) to optimization problems, in particular, to the Travelling Salesman Problem (TSP).

In our work we concentrate on the comparison of two different approaches to solve the TSP.

Our intentions are not to show one more method, that may be better than those mentioned above, but we are anxious to point out as well common features of those algorithms as the differences between them. One of these differences causes some of the algorithms to exploit the idea of competition, and others — the idea of co–operation. In the following sections, we will try to show that this is not the only difference between those algorithms.

The first problem we will discuss is the representation of the TSP for the GA and AA–based issues. Then, different operators acting on the problem representing structure will be discussed.

The parameters play an important part in both algorithms, so they will be analyzed in the further part of our work. We will emphasis there the differences between the parameters of both algorithms.

In the GA, the selection problem has the great importance. This problem will be also discussed for the AA. For the GA, the structure of populations of individuals will be presented, and for the AA, we will discuss the structure of populations of the agents–ants.

In the paper, we will also describe the experiment that we performed. Here, we would like to emphasize the similarity in the performance of both algorithms, from the viewpoint of finding the global minimum and searching the space of solutions. The essence of this comparison is the idea of the evolution (exploited in the GA), that can improve the performance of the AA.

Nowadays, exploiting of algorithms applying the stochastic searching to solve the optimization problems is the more and more popular. Such methods are the GA and the AA. These algorithms may be included to the group of methods that mimic the nature. In the case of the GA we mean the genetics (Holland (1992)), and in the case of the AA — the mimicry of collective insects, very primitive individuals having the possibility of communicate with other ones (Goss e.a. (1990)).

Both of these methods look at the analyzed problem different from the traditional optimizing methods. They see the problem as a set of potential solutions in a special coded shape. These methods use probabilistic, non–deterministic rules of choice. The GA hold the searching process, starting from a population of individuals. Similarly, in the AA, solutions are searching by many individuals — agents, called ants. The basic difference between GA and AA is that in the GA the starting point is fixed in the stochastic way. In the AA, there is an initial population of individuals performing this step during the first cycle of the algorithm (using a special form of memory and methods of communication).

At this point, we must emphasis, that when we say the AA, we mean the ant–cycle algorithm, because there are three ant algorithms with different rules of updating the pheromone trails (Colorni e.a. (1991)). The ant–cycle algorithm is the best one. In this algorithm, the pheromone trail is laid on the paths connecting cities after each cycle, and it helps another ants to find the better and better solutions.

2 Representation of the TSP in GA and AA

In this work, we do not concentrate on the genetic terminology and on searching the analogy in biology (especially in genetics). Thus, we look at the GA in a very special way — from the point of view of a whole chromosome. The chromosome is in opposition to a gene, which is a basic element of the representation of the problem, and which undergoes changes according to the rules of genetics.

In the case of the AA, an element of the TABU–list is the smallest unit of the representation of the problem. Its value may be compared with the value of the gene in the particular genetic representation of the problem (e.g. path representation).

The definition of the TSP in terms of GA is not very difficult. The TSP has the natural evaluation function — we have to calculate the total length of the tour, where the distances between cities are given. When we have the population of tours, we can easily compare any two of them. But the clear definition of the tour representation is very difficult.

The most natural representation of the tour is a list of cities visited during this tour. However, this representation is not very suitable for genetic operators, especially for crossover operator.

The binary representation of the tour is not very good for the TSP, too.

There are also three vector representations for the TSP (Goldberg (1989), Michalewicz (1996)):

- adjacency representation
- ordinal representation
- path representation

In all these representations, a tour is described as a list of cities, but position of each city in this list and its code is different.

The path representation is very useful and it is similar to the representation used by the AA, where the sequence of cities representing a tour is stored directly in the TABU–list. This is because the TABU–list includes cities visited by an ant during one cycle of the ant algorithm.

Recapitulating, we may say, that the representation of the TSP is very similar in both GA and AA. The main difference lies in the moment of creation of the first permutation of cities. For the GA, the initial permutation of cities, as the initial population, is created randomly. For the AA, this is the solution that we can observe after the first iteration of the algorithm, when each agent–ant remembers the visited cities in the concrete order. In addition, this process goes on for the agents in the parallel way.

3 Operators and Factors altering current solutions

Our considerations we start at the moment of the algorithms, where for the GA we have a population of the potential solutions represented by the sequence of numbers (in the path representation), and for the AA — we have empty agents' memories, which must be filled in an application form. In the case of the AA, we make a special swaps or sweeps in accordance with the rules described by the genetic operators (Goldberg (1989), Michalewicz(1996)).

Apart from the classical genetic operators: mutation, inversion and crossover, the following three new crossover operators for the path representation come into existence:

- partially–mapped crossover (PMX)
- order crossover (OX)
- cycle crossover (CX)

The PMX was proposed by Goldberg and Lingle (Michalewicz (1996)). Using this operator, a new individual is created choosing a subsequence of a tour from one parent and preserving the order and position of as many

cities as possible from the other parent. A subsequence of a tour is selected by choosing two random cut points, which make boundaries for swapping operations.

The OX (Goldberg (1989)) builds the offspring by selecting a subsequence of a tour from one parent and preserving the relative order of cities from the other parent. The OX crossover exploits a property of the path representation — the order of cities.

The CX, proposed by Oliver (Michalewicz (1996)), makes an individual in a such way, that each city (and its position) comes from one of the parents. The CX operator guards the absolute position of the elements in the path sequence.

The operators: PMX, OX and CX have a serious disadvantage. They do not take into account the length of arcs connecting cities. This fault is confirmed by the results of different experiments. The OX operator is the best, because it holds the shortest edges in its representation, and then gives the best solution (the results of the OX are about 11% better then for the PMX and about 15% better then for CX (Michalewicz (1996))).

Grefenstette (Michalewicz (1996)) proposed a class of heuristic operators that emphasizes the length of edges. His algorithm defines a probability distribution over selected edges based on their cost. The selection of the edges is based on the distribution (depending on the length of the edge).

All these operators work with the path representation in the TSP.

The heuristic operator, proposed by Grefenstette, works very similar to a state transition rule in the AA. The state transition rule used by the AA, called a random–proportional rule (given by eq.(1), which describes the probability with which ant k in city r chooses the city s to move to.

$$p_k(r,s) = \begin{cases} \frac{[\tau(r,s)]\cdot[\eta(r,s)]^\beta}{\sum_{u \in J_k(r)} [\tau(r,u)]\cdot[\eta(r,k)]^\beta} & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where:

- τ corresponds the pheromone
- $\eta = \frac{1}{d}$ is the reciprocal of the distance $d(r,s)$
- $J_k(r)$ is the set of cities that remains to be visited by ant k positioned on city r
- β is a parameter which determines the relative importance of pheromone versus distance ($\beta > 0$).

In the case of the AA, the role of the genetic operators plays the transition rule which causes alterations during the performance of the algorithm, and makes it to fill in the TABU–list.

In conclusion, Table I. shows the elements which influence the current solutions (in the GA) or build the new solutions.

GA operator	AA rule
mutation operator	the state transition rule
inversion operator	
crossover operator: a) simple b) permutational c) heuristic	the global and local updating rule (influencing the state transition rule)

Table I: Elements influencing solutions.

4 The sets of parameters of the algorithms

The parameters that occur in the mentioned algorithms may be divided into two groups:

a) connected with the size of the population, the number of cycles/generations

b) connected with the consequences of operations performed by the algorithms

Within the first group we find:

- for the GA — the number of individuals in a population
- for the AA — the number of agents–ants solving the problem
- for both algorithms — the number of iterations/populations

The differences between quantitative and qualitative parameters we show in Table II.

The kind of the parameter	GA	AA
quantitative	– a number of individuals l – a number of populations (cycles)	– a number of ants k – a number of cycles
qualitative	– a probability of mutation p_m – a probability of crossover p_c – a probability of inversion p_i	– a pheromone decay parameter α (similar to the learning coefficient) – a parameter which determines the relative importance of pheromone versus distance ($\beta > 0$) – an initial pheromone level τ_0

Table II: Quantitative and qualitative parameters.

The qualitative parameters may be adapted to the current stage of the searching process and to the current topology of the space being searched. The quantitative parameters play rule of control parameters. All these values are adaptive. Now, in the GA a new adaptive procedure was developed. It adapts the frequencies of the operators on the basis of their efficiency.

The values of the parameters within both groups depend on the size of the solving problem.

The parameters from the second group we see in the global pheromone–updating rule. Once all ants have completed their tours, the pheromone trail is updated on all edges according to the formula:

$$\tau(r,s) = (1-\alpha)\cdot\tau(r,s) + \sum_{k=1}^{m}\Delta\tau_k(r,s)$$

where

$$\Delta\tau_k(r,s) = \begin{cases} \frac{1}{L_k} & \text{if}(r,s) \in \text{tour performed by ant } k \\ 0 & \text{otherwise} \end{cases}$$

$0 < \alpha < 1$ and L_k is the length of the tour performed by ant k, and m is the number of ants.

The parameters from the qualitative group have a conclusive (decisive) influence on the speed and quality of the algorithms.

5 Steps of the GA and the AA (with emphasis of the selection problem)

The GA are the best–known techniques in the class of evolutionary computation. They maintains a population of individuals $P(t)$ for iteration t. Each individual represents a potential solution, and each solution is examined to obtain a value of its fitness. Then, a new population (in iteration $n+1$) is formed by the selected, fitter individuals. Some individuals of the new population undergo transformations (alterations) under influence of the genetic operators.

The structure of a genetic algorithm is as follows:

```
procedure EvolutionProgram;
begin
  t := 0;             { the discrete time      }
  initialize P( t );  { the initial population }
  evaluate P( t );
  while ( not TerminationCondition ) do
    begin
      t := t + 1;
      select P( t ) from P( t - 1 );
      alter P( t );
      evaluate P( t )
    end
end;
```

The structure of an ant algorithm may be presented in the notation of an evolutionary computing in the following way:

1. Initialization step — ants are positioned in starting towns (they do not perform any move).
2. Alter the population:
   (a) move every ant through all towns, applying a state transition rule,
   (b) put the pheromone trail on edges L_k (using the pheromone updating rule) — the real formation of the new population,
   (c) calculate the fitness function, i.e. the length of the path for each ant,
   (d) select the best–adapted individuals (ants) within one generation by calculating the shortest paths (one or more ants with the best solution).
3. If the path found in step 2d) is not the global minimum then repeat step 2.
4. Stop the algorithm.

As one can see, in this algorithm we do not accentuate evolution of the particular individuals. They neither intercross each other nor evolve in the meaning of GA. For the present, we can only show the analogy between these two methods based on the fact that many individuals try to reach the fixed goal.

To achieve a better performance of the AA to take advantage of the adapting itself to achieve the best solution.

In both these algorithms: GA and AA, the following phases are being repeated permanently: initialization, formulation a new population and selection. Selection is executed on the two different levels. In the AA with the global– best or local–best rules (Colorni e.a. (1991)), it ends the cycle and determines a recapitulation of the whole algorithm (in the GA it works in the similar way). When we think about a selection process as a choice mechanism — it occurs during a creation of the following element of the new solution (the next element in the TABU–list) and we have used a roulette wheel-like in the GA (a proportional selection).

For the selection of a new population (based on a fitness function) and for the selection of a new city to move to, a roulette wheel is used. At the first, we can use some scaling mechanism, where we calculate a special probabilities (in these algorithms) and we select a single chromosome or the city number according to a random number generated (from the appropriate range).

The most important role in both algorithms plays a fitness function (evaluation function), but we do not discuss this problem more detail, because in TSP this problem is very simple.

6 Experiments and results

The GA and the AA arose as the result of researches of the new optimizing methods that can solve problems with high computing complexity. They were created as a new approach to the optimizing methods and they may be included to the group of methods connected with the Evolutionary Computing (Michalewicz (1996)).

The mentioned algorithms were compared looking at their structure, the way they act and their effectiveness. The analysis may be performed according to several aspects, described below:

- probability in algorithms
- GA and AA as autocatalytic algorithms
- GA and AA as distributed and parallel algorithms

These two algorithms find their solutions in the similar way — both of them exploit probability in their researches.

In the AA, a single ant placed in the town i chooses the town of destination (j), with the probability depending on the visibility of the town j and on the intensity of the pheromone trail connecting the towns i and j.

The acting of the GA is also based on probabilistic choice. The main stage in the GA is the selection of individuals for reproduction and creation of the new population. In the classic GA, the individual i is selected with the probability is depending on the value of the fitness function.

Both the GA and the AA are based on the positive feedback called autocatalysis.

One of the fundamental features of the AA is the exploitation of the positive feedback. It means that the final result of one cycle is the basis for the next ones. Owing to this positive feedback, better results are gaining in the following cycles. In the AA, the pheromone trail laid by the ants on the edge (belonging to the minimal path) that is connecting the towns is the mechanism of transferring information between cycles.

Looking at the GA, it is easy to find the similar schema of proceeding. Rising of the succeeding population in the following cycle is based on the individuals of the previous population. In every population there is information derived from the previous cycles. As an effect, in the following populations the average value of the fitness function increases — these populations are better and better. Figure 1. illustrates the described problem.

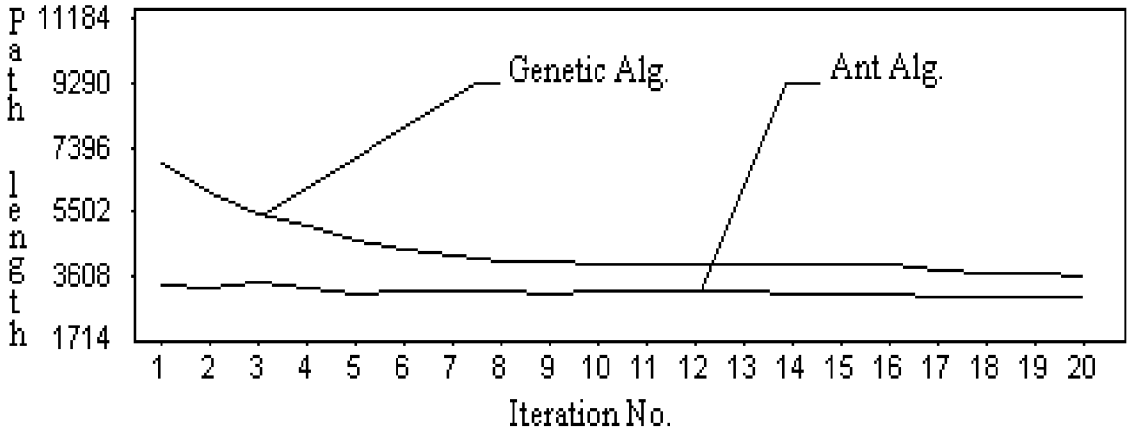

Figure 1: Comparison of algorithms.

One of the most important features of the AA is that it is the multiagent system, where the global task is worked out by sub–units called agents (ants in our approach). In the AA, the agents are the autonomous individuals, provided with elementary memory and acting in the discrete time. They can communicate and exchange with their experiences using the pheromone (Colorni e.a. (1991)).

Independently of other ants, every ant chooses the next town to go to. Therefore, the ants check many different paths simultaneously. Such construction of the algorithm allows implementing such algorithm as a parallel one, which may run in the multitasking environment (Colorni e.a. (1991)). Every ant may be treated as a single task and the set of pheromone trails is the common supply. The system clock is the additional task, which should generate succeeding time quanta, synchronizing tasks connected with the ants.

The GA act in the slightly different way. It is hard to distinguish the autonomous units. The most important functions are realized here by the one structure — the whole population. Referring to the whole population (existence of the single individuals is mentioned within the population only) and difficulty in dividing this algorithm into independent tasks, that can work simultaneously, do not allow to call it the fully parallel and multiagent algorithm. However, analyzing the way the algorithm searches for the succeeding solutions, it is

easy to notice that, in fact, it is really a parallel algorithm analyzing many possible solutions at the same time. Holland called this property of the GA the implicit parallelism (Goldberg (1989)).

In addition to the analysis of the theoretical problems, the examination of practical performance of the algorithms was observed. The computer implementations of the AA with the Ant Cycle version and the GA using the path representation were produced. In the implementation of the GA two operators: heuristic crossover and PMX (partially mapped) were applied. The task was to find solution of the TSP problem for 50 towns.

Behavior of the algorithms strongly depends on several circles. One can observe the brave influence of the choice of the parameters' values (the parameters typical for the algorithm), as well as the solving problem itself. This problems were analyzed in our previous works, e.g. Boryczka & Boryczka (1997), Boryczka (1997b).

As the examined properties of the algorithms, the average, the best and the worst solutions for the given iteration of the algorithms were selected. The three phases of the behavior of every algorithm were considered:

- the initial phase — first cycles, where the solution is not to good and an algorithm starts learning,
- the active phase — where expected good solutions appear,
- the final phase — final cycles of the experiment.

Figure 2. and Figure 3. illustrate the mentioned above properties.

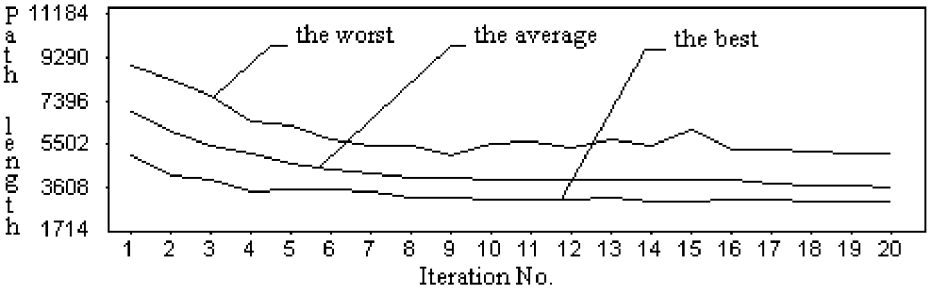

Figure 2: The Genetic Algorithm.

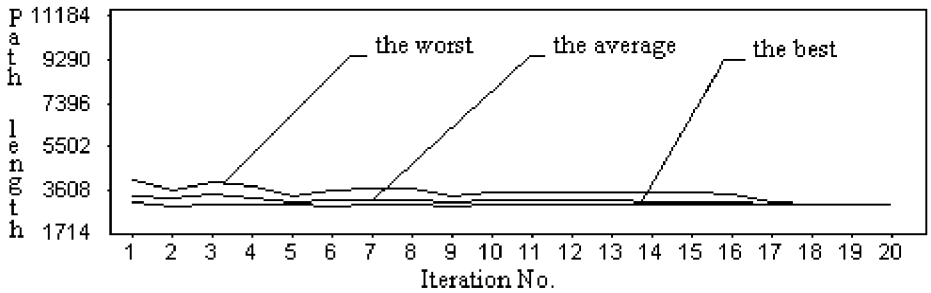

Figure 3: The Ant Algorithm.

For both algorithms, the average and the best solutions are near each other, especially close to the end of the cycle (iteration). Here, the algorithms, almost always, afford very good solutions, which come close together. One can notice the greater disproportion in the final phase for the Genetic Algorithm — the result of exploiting of the crossover operator.

The analysis of the average path lengths in following iterations shows that the Ant Algorithm improves its solutions in first 10–20 iterations, while in the Genetic Algorithm such inclination goes on in first 30 (sometimes 50–60) iterations. In the final phase, in both algorithms, the results are weakly differentiated. This situation is called the stagnancy in the solutions in demand.

References

Boryczka, M., Boryczka, U., 1997, "Generative policies in ant system", Proceedings of the conference EUFIT'97, Aachen, pp. 857–861.

Boryczka, U., 1997, "Influence of the problem representation over the selection of the genetic operators in the genetic algorithms", Proceedings of the conference "Expert Systems", Wroclaw (in polish).

Boryczka, U.; Boryczka, M., 1998, "Generative policies in Ant System for scheduling", Proceedings of the conference EUFIT'98, Aachen, pp. 382-386.

Carlier, J.; Pinson, E., 1989, "An algorithm for solving the Job-Shop problem", Management Science, 35, 2(1989), pp. 164–176.

Colorni, A.; Dorigo, M.; Maniezzo, U., 1991, "Positive Feedback as a search Strategy", Technical Report no.91–016, Politecnico di Milano.

Colorni, A.; Dorigo, M.; Maniezzo, U., 1992, "An Investigation of same Properties of an Ant Algorithm", Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92), Brussels, Belgium, Elsevier Publishing.

Colorni, A.; Dorigo, M.; Maniezzo, U.; Trubian, M., 1994, "Ant System for Job–Shop Scheduling", JORBEL — Belgian Journal of Operations Research, Statistics and Computer Science, 34, 1(1994), pp. 39–53.

Gambarella, L.M.; Dorigo, M., 1995, "AntQ: A Reinforcement Learning approach to the traveling salesman problem", Proceedings of ML–95, Twelfth International Conference On Machine Learning, Morgan Kaufmann Publishers, pp. 252–260.

Goldberg, D.E., 1989, "Genetic Algorithms in Search, Optimization and Machine Learning", Addison–Wesley.

Goss, S.; Beckers, R.; Deneubourg, J.L.; Aron, S.; Pasteels, J.M., 1990, "How Trail Laying and Trail Following Can Solve Foraging Problems for Ant Colonies", R. N. Hughes (ed.), Behavioural Mechanisms of Food Selection, NATO ASI Series, Vol. G 20, Berlin: Springer–Verlag.

Graham, R.L.; Lawler, E.L.; Lenstra, J.K.; Rinnooy, Kan A.H.G., 1979, "Optimization and approximation in deterministic sequencing and scheduling: a survey", Annals of Discrete Mathematics, 5, pp. 287–326.

Grassé, P.P., 1959, "La reconstruction du nid et les coordinations interindividuelles chez Bellicositermes Natalensis et Cubitermes sp. La théorie de la stigmergie: essai d'interprétation du comportement des termites constructeurs", Insectes Sociaux, 6, pp. 41–48.

Holland, J., 1992, "Adaptation in Natural and Artificial Systems", MIT Press.

Michalewicz, Z., 1996, "Genetic Algorithms + Data Structures = Evolution Programs", Springer Verlag, Berlin, Second extended edition.