Journal of Information Sciences and Computing Technologies(JISCT)
ISSN: 2394-9066

SCITECH
RESEARCH ORGANISATION|

Volume 5, Issue 3
June 30, 2016

Journal of Information Sciences and Computing Technologies
www.scitecresearch.com/journals

Review of Traveling Salesman Problem for the genetic algorithms

Blerina Zanaj[1], Elma Zanaj[2]

[1]*Department of Mathematics and Informatics, Agricultural University of Tirana, Albania bzanaj@ubt.edu.al*
[2]*Departaments of Electronics and Telecommunications, Polytechnic University of Tirana, Albania ezanaj@fti.edu.al*

Abstract

Genetic Algorithms (GAs) are an evolutionary technique that uses the operators like mutation, crossover and the selection of the most fitted element as solution for problems optimization. The Traveling Salesman Problem (TSP) finds a path with minimal length, closed within a weighted graph in all its nodes and it visits each of them once. This problem is found in many real world applications and where a good solution might help. There are applied many methods for finding a solution for the TSP, but during this study GAs are used as an approximate method of TSP.

Keywords: Genetic algorithms; TSP; crossover; mutation; selection.

1. Introduction

Genetic Algorithm is an optimization technique based on natural evolution that includes the idea of surviving of the most fitted element in a searching algorithm. Genetic algorithm is based in the natural process of evolution. In nature the most appropriate individuals have more possibilities of surviving and it will bring more adapted successors, those will be healthier than their parents. The same idea is applied by creating a new generation of solutions that are nearer to the demanded solution. A genetic algorithm consists of these steps: 1) encoding, 2) evaluation, 3) selection, 4) crossover, 5) mutation, 6) decoding. A suitable encoding for the solution is found by premeditating that every possible solution to the problem has a unique code like an array. Then a starting population is created with a preset number of solutions, in most of the case randomly chosen and for every individual are calculated the level of suitability. Crossover is the process where two individuals are recombined to create new individuals that become part of a new generation. Then the mutation happens. Similar to crossover after the mutation we have the creation of a new generation. This least process continuous till one stopping condition is fulfilled. At this point of the procedure the individuals that is the nearest to the optimal solution is decoded and the process is over. The whole implementation of the algorithm for the solution of TSP with the GA will consist on many partial implementations of the operators and the helping links that all of them are part of GA mentioned above. It is observed the performance evaluation of the GA in solving of real problems of TSP and its optimization.
The paper is organized as follows: Section II presents the standard Genetic Algorithms. The TSP description is shown in section III; in section IV are brought the simulations done and the Conclusions section brings a summary of the results of the work and also the future work.

2. The standard Genetic Algorithms

The standard GA will have the following steps [1, 2]:
1. Generate a set of possible solutions, with *n* candidates with a length of *l* bites (genes).
2. Calculate the *f(x)* function of suitability of each *x* (candidate) chromosomes of the population.
3. Repeat the following steps till there are created *n* successors:

a. Chose a couple of chromosomes from the actual population, the selection probability is to be an increasing trend of suitability. Each chromosome may be chosen more than once to become a selected parent of the required couple.
b. With probability *pc* (crossover probability) the couple is crossover at randomly chosen points (uniformly) in both parents to bring two successors. If there is no crossover then the successors are identical copies of their respective parents.
c. Each successor mutates in the found gene with *pm* probability (mutation probability), and set the created successors in the new population. If *n* is odd then one of the successors will be set randomly.

4. The old population is substituted with the new one.

5. Turn back to the second step.

The components of Genetic Algorithms (GA) are: encoding, evaluation, selection, crossover, mutation, decoding.

2.1 Encoding

There are many choices for the solutions that might be encoded to create a population, but they can be divided into four groups:
a) Binary encoding is usually used with problems with small complexity, or when the solution is expected to be in the form of a number;
b) Encoding with real numbers, is used with problems that look for optimization of the functions, and it is classified as more appropriate than any other type.
c) *Letter encoding with natural numbers* is a direct one. It is the best choice for listing or for a combinatorial problem.
d) Encoding with a general structure of data is used generally for real and complex problems of everyday.

2.2 Evaluation function

It performs an important role in genetic algorithm. The evaluation function is used to define "how well" a chromosome is.

2.3 Selection

Main principal where the GA is based on is essentially the natural Darwinian selection. The selection assures a guiding force to GAs [4,5,6,7]. With a strong force the GA will go to the end very early, with a weak force the evolution progress will become slower than normally. Usually a low pressure for the selection is suggested at the beginning of the genetic search in favor to the wide area of exploring space. Instead a high pressure in the selection is recommended at the end of the genetic search to tighten the search area and to reach to a convergence for the solution. Selection orients the research to the promising zone in the research area. There are different selections methods that are proposed, examined and compared between them and those are: a) Roulette wheel selection, b) (μ+λ)-selection, c) reproduction Stedy-state, d) Tournament selection, e) Ranking and scaling, f) Sharing.

a) Roulette wheel selection, is proposed from Holland, is the best prototype of selection that is recognized till now. The idea of this method is the defining of the selection probability or the surviving probability for each chromosome in proportion to the suitability value. So a model with roulette wheel can be used for showing those probabilities. The selection process is done by rotating the wheel equal in number to the size of population, where each time is selected a single chromosome for the new population. The wheel represents the method as a stochastic sampling procedure. The wheel is projected as a roulette wheel with the same number of spaces as that or population. The basic strategy over which is based the approximation is the keeping of expected number of chromosome couples in the new generation.

b) (μ+λ) selection. In contrast to the proportional selection, (μ+λ)- selection where the proposed (μ, λ) from Back are determining procedures that select the best chromosomes from parents and their successors. Both methods stop the selection of duplicated chromosomes from the population, some researches prefer to use this method, to work on with it for the problems of combinatorial optimization. Elite' selection is usually used as supplement in proportional selection to save the best chromosomes in the new generation even that those are not selected during the proportional selection process. Generate substitution substitutes a group of parents with their successors and it can be seen as another version deterministic approximation.

c) Reproduction of Steady-state of Whitely and Syswerda, belong to this class, in which the worst *n* parents are substituted with their successors (*n* is the number of successors).

d) Tournament selection. Another selective procedure contains the random and determining characteristics at the same time. An example of this group is the tournament selection of Goldberg, Korb, and Deb. This method randomly chooses chromosomes set and brings out the best from them for the crossover. The number of chromosomes in the set makes out the tournament size. Stochastic selection of tournament was suggested from Wetzel. In this method the selection probability is normally calculated continuously for the couples of chromosomes that were taken through the selective procedure of the roulette wheel. After getting a couple of chromosomes with the highest evaluation of suitability then those are brought into the new population. This process keeps going on till the new population is filled. Stochastic sampler with memory proposed from Bridle is a modified determining sampler. With this method each chromosome sample is allocated as the integer part of every expected number, so the chromosomes compete for the remaining places into the population, based on the fractional part of the expected number. In the proportional selective procedure the selection probability of each individual is proportional with its suitability. This simplified scheme shows an undesired characteristic. So, for example: in the earliest generates there is a tendency of some super chromosomes to dominate in the selection process. Instead in the last generate when the population widely converges, the fight between the chromosomes is weaker, so it can be used the random search approach.

e) Scaling and ranking mechanisms are proposed to release the problem mention in (d). Scaling do a mapping of objective function values without being elaborated in positive real values, and the surviving probability for every chromosome that is defined on these values. Suitability scaling have more attention: (1) for keeping a reasonable difference between the relative suitability values of chromosomes, and (2) to except the fast creation of a super chromosome that will put a limitation to the competition when it is created early and later on it can simulate it. Based on the type of used functions to transform the expectated values without being elaborated in the scaled suitability, scaling methods can be classified as linear scaling, logarithmic scaling, etc. In the transforming relation between the suitability scaling value with the unelaborated and constant, and so it is called as static scaling method. If the transformation will be variable depending on many factors it will be called a dynamic scaling method. Windowing techniques will represent a main line of movement toward the selection that is proportional to suitability to hold a constant pressure on selection. The normalized technique is one of dynamic scaling method that was proposed by Cheng and Gen.

f) Sharing technique that was introduced by Golberg and Richardson for a multi model optimizing function, those are used to conserve the variety of population. A spread function is used to define the devolution of suitability of an individual from another individual when it stands near to the other one. With the devolution the reproduction probability of individuals in the same group decreases while the other individuals are more encouraged to bring into life their successors.

2.4 Crossover

Crossover can be a straight forward procedure, but not every chromosome is used for the crossover. The evaluation function settles to each chromosome “points” that are used to define the probability that each chromosome will be selected for the crossover. By following the selection the chromosomes will be chosen for the crossover randomly but a greater probability is given to those with the highest “points”. It is used the spreading and gathering that is created in the evaluation process for the chromosomes selection by following different selection methods. Random numbers between 0 and 1 were generated and it is selected which chromosome corresponds in our spreading. It is repeated again to find the second and later on is done the crossover, both the new individuals become part of the new generation. The process keeps going like this till the new population is full. But the new one may or may not be better than the older generation. Every new chromosome can be checked to ensure that it actually does not exist in the new generation by doing like this is it is ensured the variety of each generation. There are possibilities that the chromosome with the best suitability passes directly to the new generation without performing any action. This means that the best choice can not become worse even for the case when the average of generate can be unsatisfactory, it still remains the best solution.

2.5 Mutation

Mutation is the technique which allows that the process get not stuck in a local optima. As a consequence of random process there are cases where the chromosomes are near to the local optima but not to the global one. For this reason are chosen the chromosomes near local optima for the crossover because those will have the greatest suitability but then the chances will be reduced for reaching the global optima.

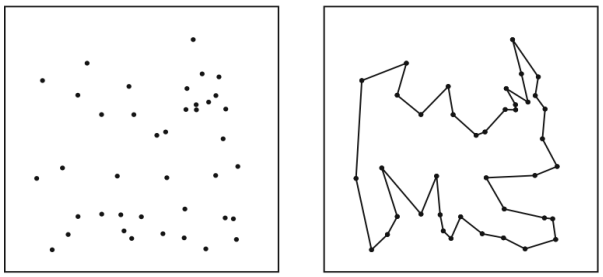
2.6 Decoding

Decoding, is the last step that follows in a genetic algorithm, it is the opposite process of encoding. It is applied when the data that comes as a response of the algorithm for the problem the output of it is in the chosen form of encoding. Those are answered in a reasonable response for the user or process.

3. The Traveling Salesman Problem

TSP treats the problem of finding a path to visit a predefined number of towns with the condition that each town will be visited only once and the journey will finish when the tour brings him to the town where it started and the length of the tour to be the shortest possible. The first example of TSP was brought by Euler in 1759, and the problem consisted on placing the horse of the chess in every position only once. The standard problem known as the problem of symmetric traveling salesman is expressed as:

There is a weighted graph G=(V, E), where c_{ij} is the weight of path between nodes *i* and *j*. It is a non negative value and it needs to find the connection of each node with a minimal total cost. The only way that guarantees an optimal solution for this problem independently of the problem size is by calculating each connection possibilities and the finding between them the one with the smallest cost [5,6]. Every connection possibilities for a predefined number of *n*-connected towns require *n!*-calculation of connection possibilities. When the number of towns is increased this method becomes inefficient and quite impossible to calculate the cost of each connection with a polynomial complexity in time. So, many optimizing methods are used during the efforts on finding a solution to the problem, Fig.1a.b, shows a case of TSP.



a) b)

Fig.1.a) The towns configuration; b) TSP.

The problem of many salesmen is the same as the standard TSP but here there is more than one salesman. It is needed to define where to send each salesman by doing so every town will be visited only once and will return back to starting town. The bottleneck with the TSP is called when it is wanted to minimize the long connections in the tour instead of the total cost. It means that it is required to minimize the maximal distance that a salesman needs to do to pass from one town to another. The problem of traveling salesman depending on time is like the standard TSP with the exception that in this case there is included the time periods. The cost now include the traveling cost from node *i* to node *n* in the time dimension *t*.

3.1 Algorithms for solving TSP

The straight forward way of solving TSP is not efficient in time and possibilities it offers [5,6,7]. And this is so because the straight forward solution has an exponential growth in cost and iterations, and it also depends on the number of nodes in the problem. It can be reached a polynomial growth only by sacrificing a bit of accuracy but gaining in velocity for finding the solution [10]. This can be achieved by heuristic algorithms for TSP, where their characteristics are high velocity and are nearer to optimality for the choice. The algorithm is divided in two main groups:

I) Constructive;
II) Optimizing.

I) Constructive has: a) nearest Neighbor $O(n^2)$,b) Greedy $O(n^2\log_2(n))$,c) insertion heuristics, d) Christofides.

a) Nearest Neighbor $O(n^2)$. It appears as the simplest and straight forward algorithm helping for the TSP. The key of this algorithm is to visit always the nearest town to the one where you actually are. Its steps are as follows:

1. Select randomly a town.
2. Find the nearest unvisited town and go there.
3. Are still other unvisited towns? If yes return to step 2.
4. Turn back to the starting first town.

The problem of this algorithm stands in long distances that it needs to pass through where there have remained only a few last towns unvisited.

b) Greedy $O(n^2\log_2(n))$. The helper Greedy gradually constructs a route by selecting repeatedly the shortest path and set them in the tour till those do not create a circuit with less than *N* paths, or it increase the grade of each node bigger than two. It is not possible to add twice a path.

The Greedy solution follows three steps:

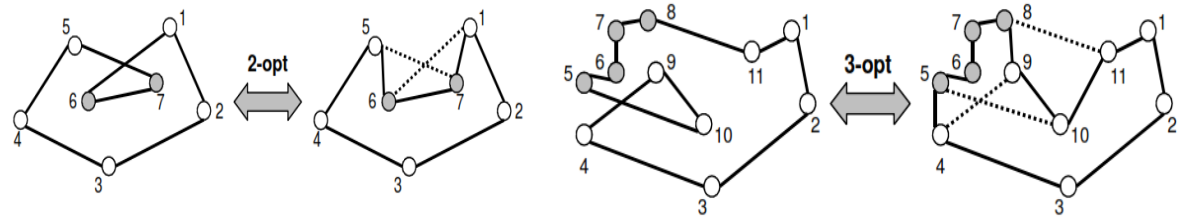
1. List every path (connection).
2. Chose the shortest path and add it to the tour if there it is not broken any of the above rules.
3. Are there N paths in the tour? If not return to step 2.

c) Insertion heuristics. The algorithm with heuristic insertion has some variants from which to choose and the most widely used is the nearest insertion. The ground of the algorithm of heuristic insertion is about the starting of a tour of a subgroup of towns, and then adding them continuously one after the other in an oriented way. The starting sub-tour mainly is a triangle compound of three towns. It may start as well as a single connection as sub-tour.

d) Christofides. Most of heuristic algorithms guarantee a report of worst case 2 (which means a tour with length 2 times longer than the optimal).Prof.Nikos Christofides went on with one of the algorithms and it reaches the conclusion that the report of the worst case is 3/2 and the algorithm is recognized as Chrisotfides heuristic.

II) ***Optimizer*** or tour improvers has: a)2-op and 3-opt, b)2-opt and 3-opt accelerated, c) k-opt, d)Lin- Kernighan, e)Tabu – search, f) simulated annealing.

a) 2-opt and 3-opt, 2-opt algorithm is based on removing of two connections in turn and creating of two others. This action is referred as 2-opt move. It exists only a way to be reconnected and it is done only on the case when the new connections created make the tour shorter. It continues till, there is no more improvements with 2-opt. The tour is called as 2-opt shown in the Fig.2.a. The 3-opt algorithm works similarly with 2-opt, but instead of removing two connections, it removes three. This means that there are two other ways of connecting with 3 passes for a valid tour. The research goes to the end when there have remained no other of the 3-opt move that can improve the tour, Fig.3.b. If a tour is 3-optimal it is as well a 2- optimal.



a) b)

Fig.2. Examples of: a) 2-opt move; b) 3-opt move

b) 2-opt and 3-opt accelerated. The complexity of the algorithm k-opt it brings into mind that a single move have a complexity O(n) to finish. A naive implementation of 2-opt is executed for a complexity of $O(n^2)$ where is included the selection of a connection (c1,c2) and the search of another connection (c3,c4). The movement is completed only if the dist(c1,c2)+dist(c3,c4)>dist(c2,c3)+dist(c1,c4). Another study done by Steiglitz and Weiner tells that the search may become shorter if dist(c1,c2)>dist(c2,c3). This means that we can reduce in good part our search by keeping as helping information the towns that are the nearest to each town. This information asks for extra time of computing ($O(n^2\log_2(n))$, and a considerable part of memory. By reducing the number of neighbors in our list it will allow to use practically this logic. By keeping only the *m* nearest neighbors to every town the complexity will be improved till O(mn). Anyway it is required to find the nearest neighbors town. But this information is static for each part of the problem, so it is required to be calculated only once and then can be used during each iterations of the algorithm that is executed. This acceleration will decline the guarantee of the 2-opt tour, but the loss in the quality of the tour will be controlled with a careful selection of *m*. So for instance by choosing *m=20* the maximum will reduce the quality very little or not at all. By choosing *m=5* it will bring a significant grow in the execution velocity with a small cost in quality. 2 and 3-opt accelerated is a very good improving algorithm used, it allows the programmer to make a trade-off between the quality and velocity depending on the problem treated.

c) K-opt. There is not only 3-opt but we keep going on with the 4-opt, 5-opt and so on. This type of algorithms are called as k-opt. Although their usage further than 3-opt different studies have reached to the conclusion that the increase in improvement between 2-opt and 3-opt is too small if we take into account the large time of execution that those require.

d) Lin-Kernighan. Lin and Kernighan constructed an algorithm based on the other optimization those are denoted by LK letters starting the name of the algorithm. The value of *k* is an integer variable during the application on the case. The algorithm selects the value of most suitable *k*, during each iterations passed. This made the algorithm more complex, and little was possible to improve in this aspect. The complexity of this algorithm is $O(n^{2.2})$, which is slower than 2-opt standard.

e) Tabu- Search. Neighbor algorithm searches between its neighbors the candidate to find the best. When it is applied NS in TSP, the moving of neighbors are normal 2-opt. The problem with the search algorithm is that it may

be very easily "stopped" in the best local solution not for the global one. This can be avoided easily by the Tabu-Search algorithm. Tabu search allows movements with negative gain if there is no positive found. There many implementations in the tabu list. One includes the adding of two removed connections from 2-opt movement in a list. The movement than is considered tabu if it tries to add the same movement again. Another way is by adding the shortest connection removed from the 2-opt and then every other movement that includes this movement remain tabu. A third way is by keeping every ending point of each movement, by doing that the movement between the ending points to be tabu. The biggest problem that characterized this algorithm is the long execution time for TSP, $O(n^3)$ making it a bit slower than the 2-opt.

f) Simulated annealing is an adapted algorithm to give a solution approximated for TSP. SA at the origin is a random and locally searching algorithm that allows movements with negative gain. This algorithm uses 2-opt movements to find the neighbors. It is not by chance that the resultant tour is comparable from the one that came from the normal 2-opt. Better results can be obtained by increasing the execution time of SA algorithm, by giving us comparable results with those obtained from other algorithm LK.

3.2 Genetic Algorithms as a method for solving the TSP

3.2.1 Encoding

a) Adaptability. One of the methods for TSP optimization is GA. In this work is studied how an algorithm helps based on the natural evolution for solving a problem as wide as TSP.
b) Suited Encoding. It can be used for vector encoding with numbers like 12345, a vector of letters ABCDE, as well in every other form a chain of symbols and those might have sense with the problem. It will bring below the main ways of representation of TSP so that it can be elaborated from the GA.
c) Matrix. Although a problem like TSP is only a graph. It can be built a connectivity matrix that is made of 1 and 0, where with 1 is defined a connection between *i* and *j* nodes with 0 separation. Then this matrix can be treated in unchanged form or to work with it by laying it linearly in a row. This leads toward a first method for encoding of the problem of the traveling salesman.
d) Array. The first form of representing the TSP can be done by cyclic symbols in the array type.

3.2.2Adapted evaluation

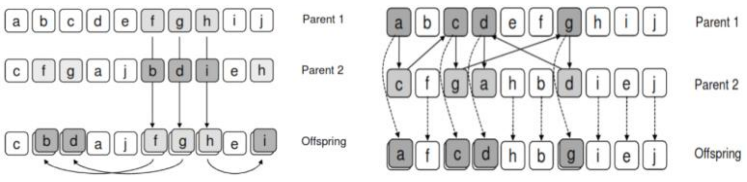
The process of evaluating the individuals created, that in this case are the connections one after the other in the shown tour. It will be the sum of each distance, the cost or weight from node to node. As the main task of our problem is based on achieving and finding of the shortest tour, the aim of finding a solution to this problem will be the minimizing of this evaluation. Memorizing the distances or weights between the nodes can be calculated as Euclidian distance, or in searching of efficiency of the algorithm which can be stored in a matrix by eliminating the need to continuously recalculate it over and over again.

3.2.3Adapted selection

Selection is one of the most important operators in GA. As the aim is to minimize the tour, it is asked that the individuals that in our case are tours with the greatest probability to select. Those are required to have the greatest adaptability so with an overall distance of the tour the smallest that is defined from the evaluation operator. All the selective methods can be used, but our request of selecting the best is fulfilled by the roulette wheel selection method. It is based on the individuals evaluation by defining their part in the wheel which than is swung.

3.2.4Adapted crossover

There are different methods developed for the crossover in the traveling salesman problem. Partially matched crossover (PMX), is when it is supposed that we are using the type of encoding with number arrays, and it reused the crossover with two points, Fig.3.a.



a) b)

Fig. 3. Representation of a partially crossover: a) PMX, b) cycling

The crossover will be meaningful if it will be represented in a cycle, because in this case it will conserve more from parent structure. If as in our case is used the first type of representation with numbers, the queue of visited towns will have many differences from parents to the child, and only a few of the connections that were before will be conserved. In the cyclic representation, Fig.3.b, many of the connections will be transferred without change. Although it will be used this kind of routine of crossover with the cyclic representation but it can achieve even non legal tours. It is required to derive a routine to repair and create legal tours from the results that the crossover brings by changing the less possible to maintain a similar structure.

3.2.5Adapted mutation

For the mutations done to ensure that successors are more suitable, instead of leaving this process to casualty for the TSP there are also other helping algorithms who check for the selection of the "gene" that undergoes to changes. First it will start with 2-opt operator. There are chosen randomly two connections (a,b) and (b,c) from our tour and it is checked if there can be found another way for the connectivity of those four nodes in order to achieve the lowest possible cost. To do this it is checked if:

$$Cab + Ccd > Cac + Cdb \tag{1}$$

If the inequality above is fulfilled then there are made substitutions of the connection of (a,b) and (c,d) with other connections (a,c) and (d,b). It is emphasized that it was supposed that a,b,c,d appear in order in the tour even that b with c are not connected. There is as well the 3-opt operator which checks 3 connections chosen randomly instead of two. If there are the connections (a,b), (c,d) and (e,f) it is checked if:

$$Cab + Ccd + Cef > Cac + Cbe + Cdf \tag{2}$$

If the condition is fulfilled then there are done substitutions of the connections for 6 nodes. The third operator called Or-opt is similar with the 3-opt. Randomly there is chosen a set of nodes connected and it is checked if the vector can be set between 2 other nodes that are connected and this is done to reduce the cost. This can be calculated by finding the total cost of the connections that were introduced and the total cost of the nodes that were removed. If the cost of connections that were removed is bigger than the cost of connections that were set then the change is done. There are also three other mutation operators but those are additions where randomly is selected a town and it is set at a random position. The displacement is done there where is selected a part of the tour and add it somewhere else. As well there are also exchanges of two towns' positions with each other.

4. Implementation and simulation with Matlab for the TSP

The whole implementation of the algorithm for the solution of TSP with the GA will consist on many partial implementations of the operators and the helping links that all of them are part of GA mentioned above [12,13,14]. It is observed the performance evaluation of the GA in solving of real problems of TSP and its optimization. The evaluation will be represented as a relative difference from the best choice known that it is defined like this:

$$\text{Diff\_Relative} = \left(\frac{\text{Result Obtained}}{\text{Optimal Result}} - 1\right) \times 100\% \tag{3}$$

GA are closely related with random operator, it is awaited that problem repetition even in the same conditions, to bring different results. So, due to this reason more than one simulation needs to be done for each problem that will be chosen, by defining like this an error margin [8,9,11]. Problem statement on same conditions is too important to get comparing results. So for each simulation will be used standard parameters of GA shown as in the Table 1.

Table 1. Parameters of standard Genetic algorithms

Generation (iteration)	100'000
Population size	120
Elitism scale	1
Selective operator	Roulette
Mutation probability	0.1 , 0.2 , 0.8

To obtain comprehensive conclusions from the simulation are chosen 5 main published problems of TSP, [14]: Bay29, Eli51, Berlin52, kroA100, kroA200. This selection is done following 2 criteria like this: a) number of towns

(nodes) in problems is different; b) Towns' dispersion, nodes to be different for problems with the near numbers of nodes. Below are treated every environment like a TSP apart, by emphasizing through simulations the approximation property of GAs.

Bay29. The first problem that will be simulated Bay29 is a problem constructed for the situation of 29 towns in Bavaria. With the best solution known 9074 published in [14]. Towns positioning for the 2 dimensions coordinates projected in Matlab, Fig.4.a.

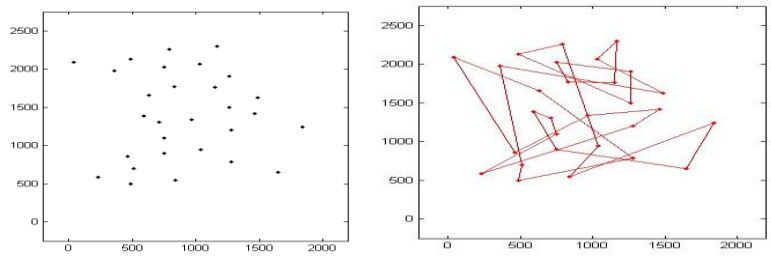
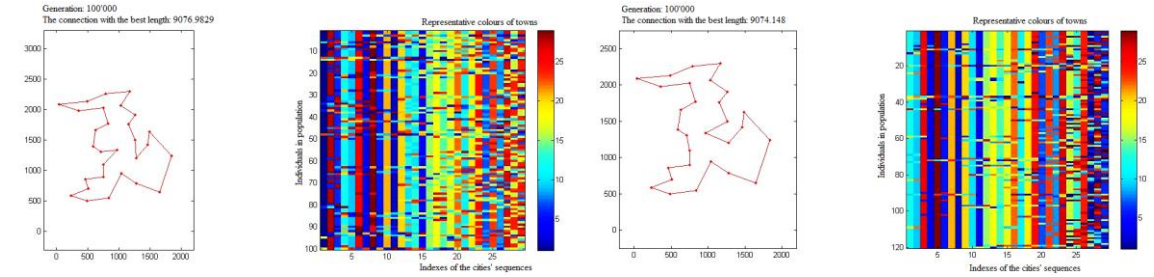


Fig.4. a) Towns dispersion for *Bay 29*; b) The best tour got from the random generated first generation

The created tour from random generation of starting population for the problem of traveling salesman bay29: In this phase of simulation, actually only after 5 generations none of the genetic algorithms operator is not yet used. So as expected the actual connection the best is far from the optimal.



a) b)

Fig. 5. Result of simulation *Bay 29*: a)first simulation b) second simulation

Table 2.Results for *Bay 29*

Bay29	Result	Optimal	Diff Relative
Sim.1	9076	9074	0.02 %
Sim.2	9074	9074	0.00 %
Sim.3	9074	9074	0.00 %

The best result: 9074 (0.00%).
Average result: 9075(0.01%).

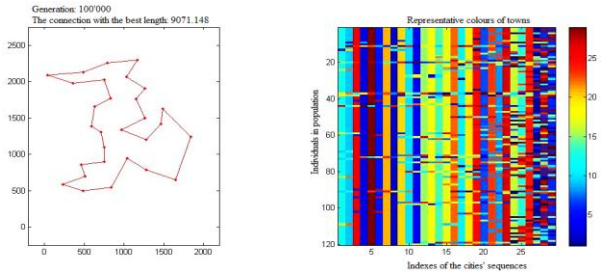


Fig. 6. Simulation result *Bay 29*, third simulation

Eli51. Eli51 is another example of positioning treated as TSP compound of 51 towns (nodes). The best known solution is 426. Towns positioning by 2 dimensions coordinate for Eli51, Fig.7.a, and the created tour from random generation of first population for the TSP, Eli51, Fig.7.b.

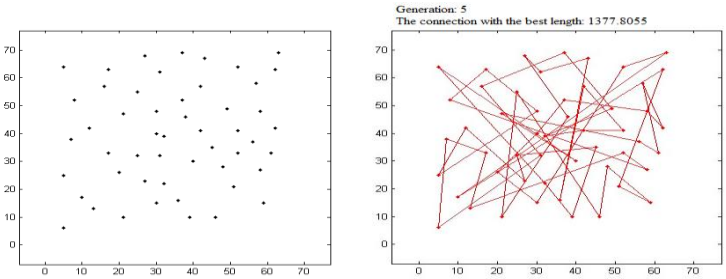
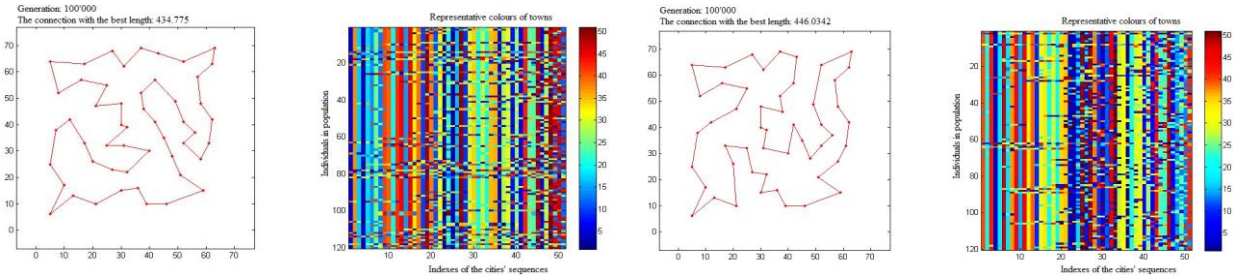


Fig. 7. a) Towns spreading for *Eli51* b) The best tour from the starting population randomly created



a) b)

Fig. 8. Result of *Eli51* simulation a) first simulation, b) second simulation

Table 3. Results for *Eli51*

Eli51	Result	Optimal	Diff Relative
Sim.1	434	426	1.87 %
Sim.2	446	426	4.69 %
Sim.3	444	426	4.22 %

The best result: 434 (1.87%).
Average result: 441 (3.52%).

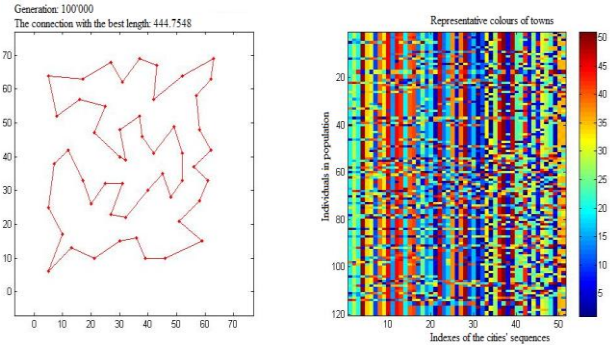
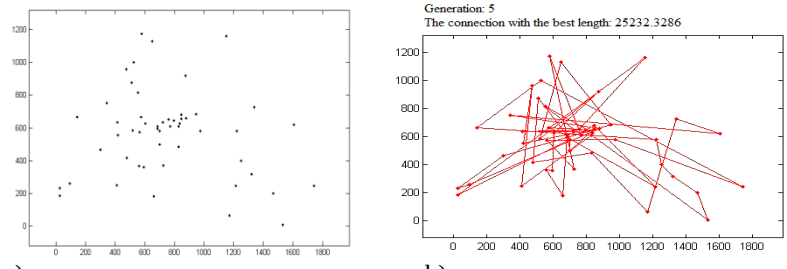


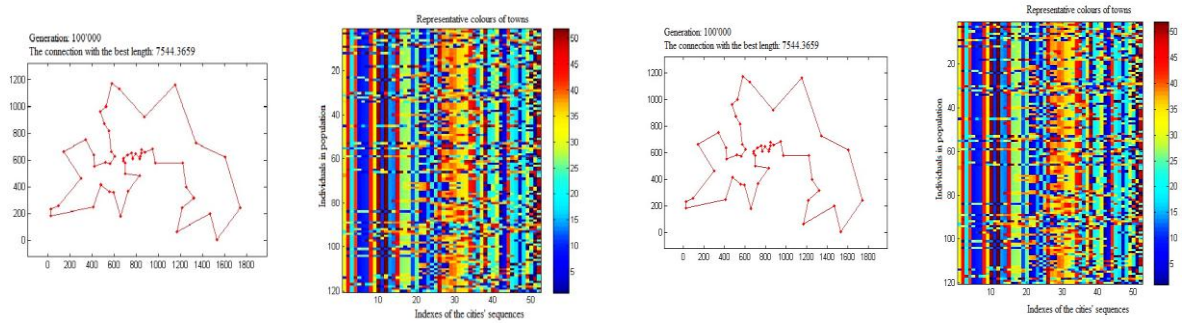
Fig. 9. Result of *Eli51* simulation, third simulation

Berlin52. Berlin52 is one of the maps that is used to study for different problem of optimization mainly treated like TSP. It is composed from 52 main positioning in Berlin. The best solution reached is 7542. Nodes position expressed as a two dimensions is like in Fig.10.a. The tour created from random generation of the first population, for the TSP, Berlin52, Fig.10.b.



a) b)

Fig. 10. a) Towns spreading for *Berlin52* .b) The best tour get from the starting population created randomly



a) b)

Fig.11. Result of *Berlin52* simulation: a) first simulation, b) second simulation

Table 4. Results for *Berlin52*

Berlin52	Result	Optimal	Diff Relative
Sim.1	7544	7542	0.02 %
Sim.2	7544	7542	0.02 %
Sim.3	8235	7542	9.10 %

The best result: 754(0.02%).
Average result: 7774 (3.07%).

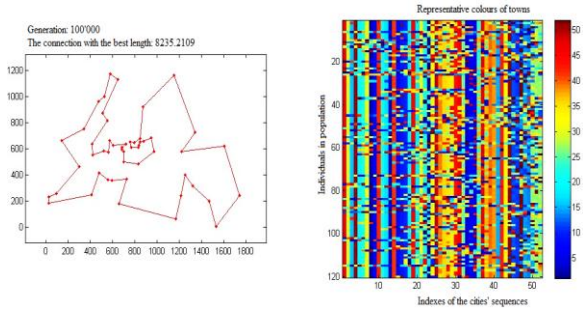
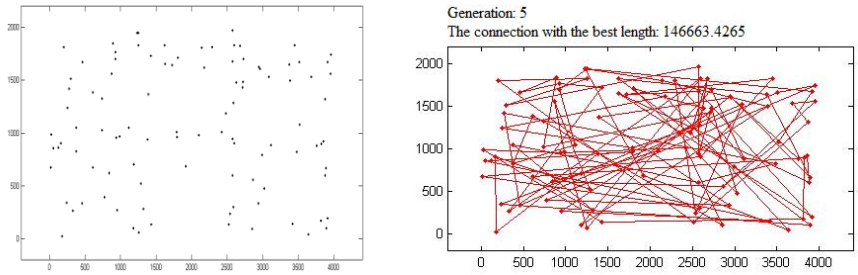


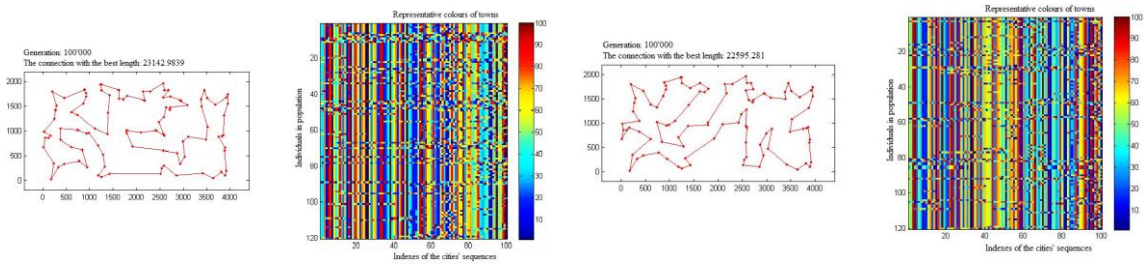
Fig. 12. Result of *Berlin52* simulation, third simulation

KroA100. KroA100 is an environment compound of 100 nodes; it is treated as TSP with a good solution known of 21282. Towns positioning following the 2-dimensions coordinates for KroA100, Fig.13.a. The tour created for starting population generation, for the problem of traveling salesman KroA100, Fig.13.b.



a) b)

Fig.13. a) Towns spreading for *KroA100;* b) The best tour get from the starting population created randomly



a) b)

Fig. 14. Result of simulation *KroA100:* a)first simulation b) second simulation

Table 5. Results for *KroA100*

KroA100	Result	Optimal	Diff Relative
Sim.1	23142	21282	8.73 %
Sim.2	22595	21282	6.16 %
Sim.3	21600	21282	1.49 %

The best result: 21600(1.49%).
Average result: 22445(5.46%).

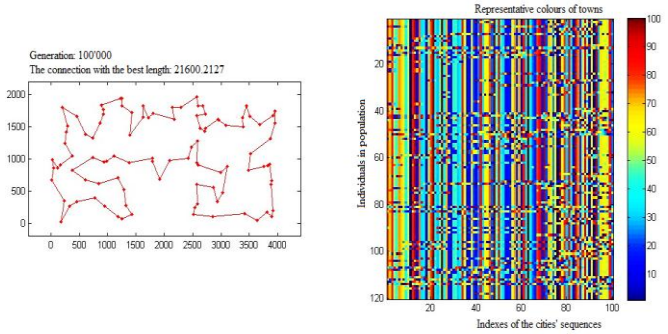
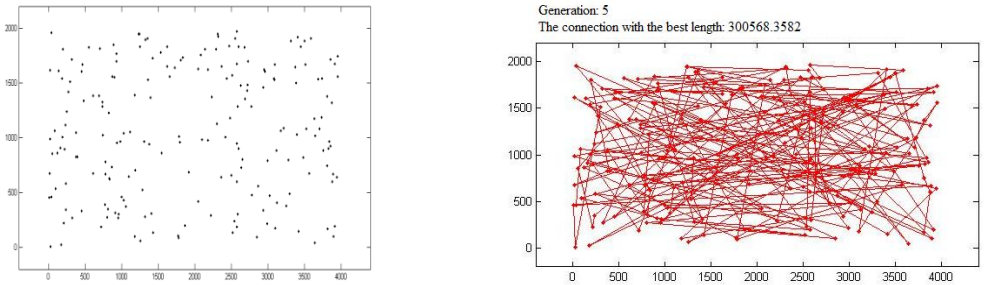


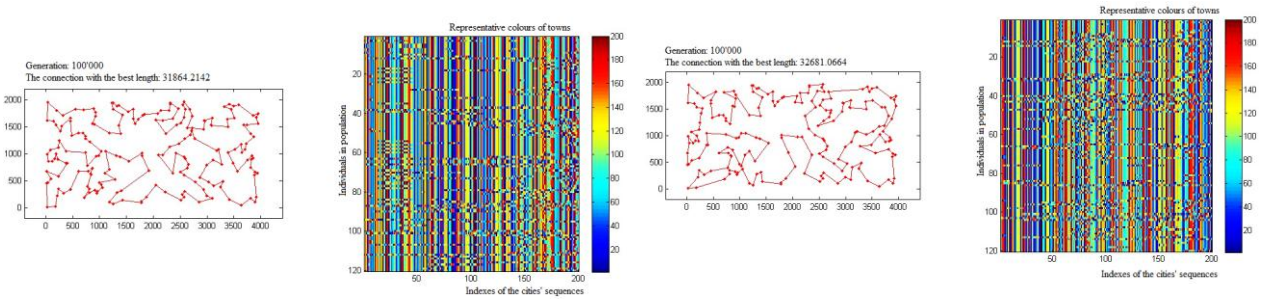
Fig.15. Result of simulation *KroA100*, third simulation

KroA200. KroA200 is an environment made of 200 nodes, treated as TSP with a good solution known of 29368. Towns positioning by 2-dimensions coordinates for KroA200, Fig.16.a. The tour created from the random generation of the first population for the traveling salesman with KroA200. Fig.16.b.



a) b)

Fig. 16. a) Towns spreading for *KroA200*; b) The best tour get from the first population generated randomly



a) **b)**

Fig. 17. Result of simulation *KroA200*: a)first simulation b) second simulation

Table 6. Results for *KroA200*

KroA200	Result	Optimal	Diff Relative
Sim.1	32681	29368	11.28 %
Sim.2	31864	29368	8.49 %
Sim.3	32652	29368	11.18 %

Best result: 31864(8.49%).
Average result: 32399(10.32%).

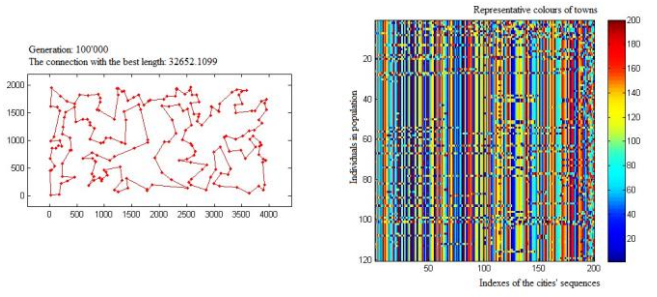


Fig. 18. Result of simulation *KroA200*, third simulation

Conclusions

GAs with their developing are having a very important role in solving different problems based on approximation and forecasting. Those have brought a further step in evolutionary calculation, by using the crossover, mutation and selection operators those have created a flexible and suitable algorithm for every kind of application. The TSP is a conceptual base of a problem that comes from many real applications. It is on the focus of many researchers due to its importance, it is widen so much in the aspects of different treatments also in the techniques and methods necessary for its solution. In this work is used GA for finding an approximation for the TSP based on simulations. The GAs are relatively good approximate tools even in the TSP. It reacts ideally for a number of nodes less than 30, and it also gives a satisfactory and fast solution also for problems with many nodes. As it can be observed from the Table 7, the dependency of relatively approximation calculated does not exist only from the number of nodes, but as well of their positions and weight. By observing the results of Eli51 and Berlin52 it can be concluded that the GA shows more efficiency in long distance of tours. In the problem where accuracy is primary, greater number of generation is required to take the best from it.

Table 7. Results of all simulations

	No. of nodes (towns)	Optimal result	The best approximation achieved from GA		Average approximation achieved from GA	
Bay29	29	9074	9074	0.00 %	9075	0.01 %
Eli51	51	426	434	1.87 %	441	3.52 %
Berlin52	52	7542	7544	0.02 %	7774	3.07 %
KroA100	100	21282	21600	1.49 %	22445	5.46 %
KroA200	200	29368	31864	8.49 %	32399	10.32 %

Like in many other applications the GA may serve with efficiency for achieving of a satisfactory approximation for the TSP. In the future it will bring higher level of efficiency as one of the most studied method of the present.

References

[1] Falkenauer E. (1998). Genetic Algorithms and Grouping Problems. *John Wiley and Sons*.

[2] David E. Goldberg (1989). Genetic Algorithms in Search, Optimization and Machine Learning. *Addison-Wesley*.

[3] Homaifar A., Guan Sh., and E. Liepins G. (1992).Schema analysis of the traveling salesman problem using genetic algorithms. *Complex Systems*.

[4] Jog P., Y. Suh J., and Van Gucht D. (1991) Parallel genetic algorithms applied to the traveling salesman problem. *SIAM Journal of Optimization*.

[5] Junger M., Thienel S., and Reinelt G.(1994). Provably good solutions for the traveling salesman problem. *Mathematical Methods of Operations Research*.

[6] Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G., and Shmoys D.B. (1986) The Traveling Salesman. *John Wiley and Sons*.

[7] Michalewicz Z..(1994) Genetic Algorithms + Data Structures = Evolution Programs. *Springer-Verlag*, 2nd edition.

[8] Reinelt G.(1994) The Traveling Salesman: Computational Solutions for TSP Applications. *Springer-Verlag.*

[9] Wroblewski J. (1996). Theoretical foundations of order-based genetic algorithms. *Fundament Informatica.*

[10] Gen M., Cheng R. (2000) Genetic.Algorithms and Engineering.Optimization.*Wiley*

[11] Mitchell M. (1991) An Introduction to Genetic Algorithms. *5ed.*

[12] Affenzeller M., S Wagner, S Winkler, A Beham (2009) Genetic Algorithms and Genetic Programming. *Crc Press.*

[13] Haupt R.L. and Haupt S.E. (2004) Practical Genetic Algorithms. 2ed. *Wiley*

[14] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/