Ant colony for the TSP

Yinda Feng

Master Thesis
Computer Engineering
Nr: E3839D

2010

Högskolan Dalarna
www.du.se

DEGREE PROJECT
Computer Engineering

Programme Computer Engineering	Reg number E3839D	Extent 15 ECTS
Name of student Yinda Feng	Year-Month-Day 2010-6-7	
Supervisor Mr. Pascal Rebreyend	Examiner Professor Mark Dougherty	
Company/Department	Supervisor at the Company/Department	
Title Ant colony for TSP		
Keywords TSP, ant colony, pheromone, combinatorial optimization		

Abstract

The aim of this work is to investigate Ant Colony Algorithm for the traveling salesman problem (TSP). Ants of the artificial colony are able to generate successively shorter feasible tours by using information accumulated in the form of a pheromone trail deposited on the edges of the TSP graph. This paper is based on the ideas of ant colony algorithm and analysis the main parameters of the ant colony algorithm. Experimental results for solving TSP problems with ant colony algorithm show great effectiveness.

Acknowledgement

I sincerely thank my supervisor Mr. Pascal Rebreyend for his advices, support and efforts of guidance in my thesis work.

I also acknowledge my family and my friend who give me suggestion and totally support during the thesis period.

Content

List of Figures

List of Tables

1. Introduction

Combinatorial optimization problem is an important embranchment of operational research. The mathematical methods can be used to search optimization arrange, grouping, sequence or ridding of the discrete events. These Problems belong to the Non-polynomial-complete (NPC) questions, which can't be solved in polynomial time. With the enlargement of the scale of question, the question space makes the characteristic of exploding up, is unlikely solved with general methods. As a NPC Problem, Traveling Salesman Problem (TSP) is a classical combinatorial optimization question, which now can only be solved by meta-heuristics to get the approximate solution.

Since creating bionics in middle period of 1950's, people are being inspired from the mechanism of the biological evolution constantly. Many new methods had been applied to solve the complicated optimization problems are proposed. Such as neural network, genetic algorithm, simulated annealing, and evolution computation. These new methods had been successfully applied to solve the practice problems.

1.1. Ant Colony Algorithm

Ant Colony Algorithm [1] is a random search algorithm, it based on the research of the nature ants behaviors, simulate real ant colony collaborative process. Which proposed by the Italian scholar M．Dorigo V．Manierio，A．Collomi. Taking into account the similarity of ants searching of food and the traveling salesman problem, using the Ant Colony Algorithm to solving traveling salesman problem.

Real ants are capable of finding the shortest path from a food source to the nest [2] without using visual cues [3]. Also, they are capable of adapting to changes in the environment, for example finding a new shortest path once the old one is no longer feasible due to a new obstacle [2].

1.2. Traveling Salesman Problem

Combinatorial optimization problem [4] is to calculate the variable combination to minimize (or maximize) the objective function, under the given constraints. Traveling Salesman Problem (TSP) [5] is a classical combinatorial optimization question. Therefore it is not sufficient to find an arbitrary solution. Instead, one is interested in the best (or at least a very good) solution.

The travelling salesman problem is quite simple: a travelling salesman has to visit customers in several towns, exactly one customer in each town. Since he is interested in not being too long on the road, he wants to take the shortest tour. He knows the distance between each two towns he wants to visit. So far, nobody was able to come up with an algorithm for solving the traveling salesman problem that does not show an exponential growth of run time with a growing number of cities. There is a strong belief that there is no algorithm that will not show this behavior, but no one was able to prove this (yet). But one was able to prove that the traveling salesman problem is a kind of prototypical problem for a big class of problems (the famous class NP) [6] that show this exponential behaviour.

2. The model of ant colony algorithm

2.1. The biological model of ant colony algorithm

Real ants follow a path between nest and food source. An obstacle appears on the path: Ants choose whether to turn left or right with equal probability. Pheromone is deposited more quickly on the shorter path. All ants have chosen the shorter path. As figure 1:

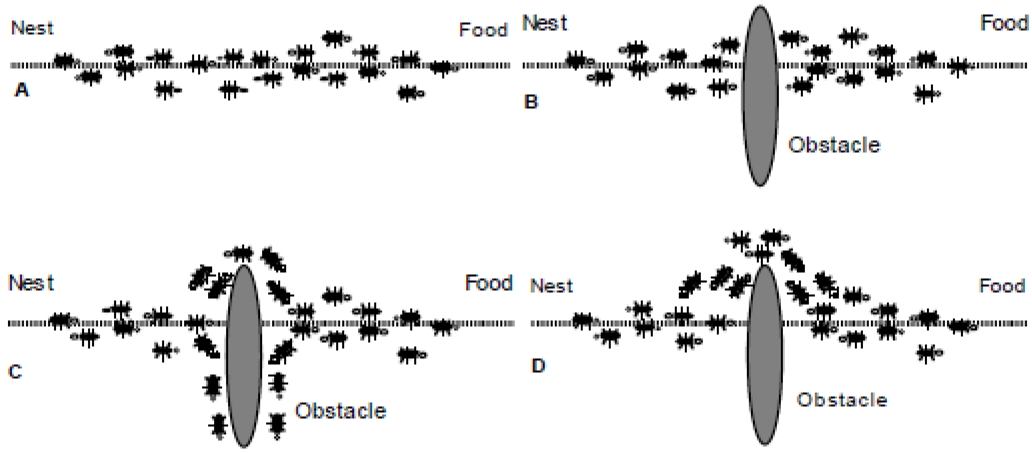


Figure.1. Ants foraging process

2.2. The basic theory of ant colony algorithm

The study found that the ant leaves volatile hormone when it travels, and the ants use the hormone exchange information. Ants tend to follow the path which accumulated more hormone. Ants which find the shortest path, always be the first one return to the nest, thus leaving more hormone on the path. As the shortest path has accumulated more hormones, more and more ants choose this path, in the end all the ants will tend to choose this shortest path.

Now we use figure 2 to descrip the theory of ants finding the shortest path:

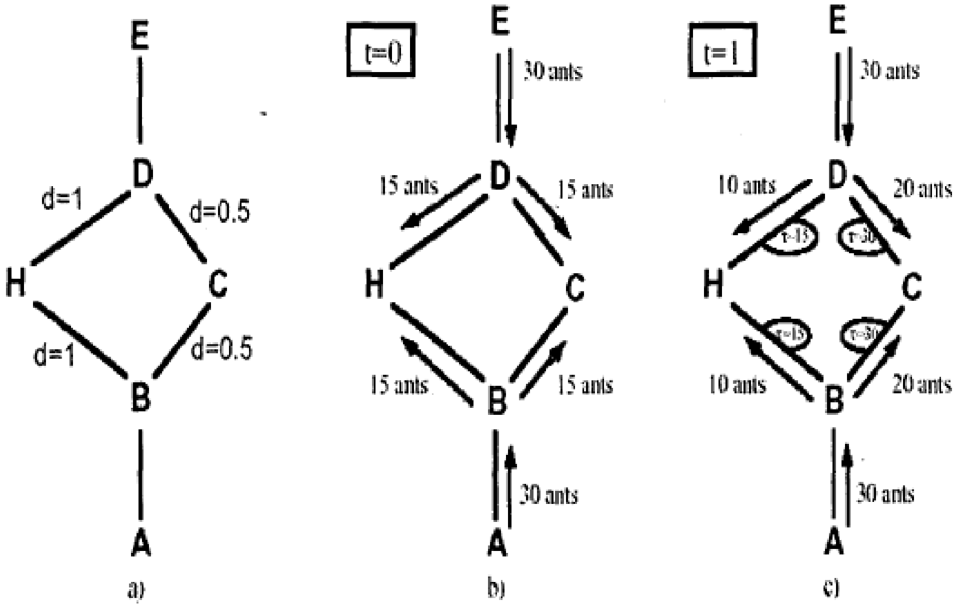


Figure.2. The theory of ants finding the shortest path

Shown as Fig.2.(a), A is the ants nest, E is the food source, HC is an obstacle, because of the obstacle, the ants can only reach E through H or C from A. Assume that the distance between D and H, H and B, B and D (through C) are 1. At a unit time, there are 30 ants reached B from A, and 30 ants reached D from E. The ant left 1 pheromone after it pass, the ant moves at the speed of 1 per unit time. At the moment t=0, because of there is no pheromone exists on the path: BD, BC, DH, DC, the ants at B and D can randomly select their path to walk. From the statistical point of view, they have the same probability to choose the path BH, BC, DH, DC. Therefore, in average, form the direction B and D, there are 15 ants travel to H, and 15 ants travel to C, as Fig.2.(b). At the moment t=1, there are 30 new ants arrived at B from A. They found that the pheromone concentration on the road leading to H is15, which is the sum of the pheromone left by the 15 ants choose the direction H from B to D and the 15 ants choose the direction H form D to B. And the pheromone concentration on the road leading to C is 30, thus, the probability of selecting path is changed, as Fig.2.(c).Thus, the ants to the C direction will be the twice of ants to the H direction, respectively 20 and 10.The ants starting to reach D from E, will do the same. This process continues, in the end all the ants will choose the shortest path.

The ant colony algorithm simulated ant colony foraging behavior introduces a new artificial intelligence model, the algorithm is based on the several assumptions as follows:

- Ants make communication by using pheromone.
- At the individual level, each ant to make independent choices based on the environment. But ant colony can have orderly group behavior through self-organization process.

Ant colony algorithm includes two basic stages: adaptation phase and collaboration phase. In the adaptation phase, the more ants walking through the path, the greater amount of pheromone the path will have, then the probability be chose will be greater. The longer, the less amount of pheromone will be.

2.3. The mathematical model of ant colony algorithm

In the artificial ant colony systems, ants have several characteristics as:

- Ants select the path with probability. The probability depends on the function of distance between two cities and the residual pheromone in the path.
- Ants have memory function. In each cycle, each ant can only choose the path that it never travelled when it transfer path.
- Ants leave some pheromone in its path. The pheromone left in the path will decay gradually over time.

An artificial ant k in city i chooses the city j to move to among those which do not belong to its working memory M_k by applying the following probabilistic formula:

$$j = \begin{cases} \underset{j \notin M_k}{\arg\max} \left\{ \left[\tau_{ij}(t) \right]^{\alpha} \left[\eta_{ij} \right]^{\beta} \right\} & \text{if } q < q_0 \\ J & \text{otherwise} \end{cases} \tag{1}$$

Where τ(i, j) is the amount of pheromone trail on edge (i, j), η(i, j) is a heuristic value which was chosen to be the inverse of the distance between cities i and j, β is a parameter which weighs the relative importance of pheromone trail and of closeness, q is a value chosen randomly with uniform probability in [0,1], q0 (0≤q0≤1) is a parameter, and J is a random variable selected according to the following probability distribution, which favors edges which are shorter and have a higher level of pheromone trail:

$$P^{k_{ij}(t)} = \begin{cases} \dfrac{\left[\tau_{ij}(t) \right]^{\alpha} \left[\eta_{ij} \right]^{\beta}}{\sum_{j \notin M_k} \left[\tau_{ij}(t) \right]^{\alpha} \left[\eta_{ij} \right]^{\beta}} & \text{if } j \notin M_k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Where $P^{k_{ij}(t)}$ is the probability with which ant k chooses to move from city i to city j. M is the amount of ants, $d_{ij}(i, j = 1, 2, ..., n)$ is the distance between city i and city j, $\tau_{ij}(t)$ is the amount of pheromone on edge (i, j) at time (t). Each edge has the same pheromone at the initial time $\tau_{ij}(0) = C$.

Over time, the pheromone on the edge will decay gradually. ρ is the pheromone residue level, then 1- ρ is the pheromone volatile level. After artificial ants have completed their tours, the pheromone on each edge adjusted according to the following formula:

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{3}$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{\ k}(t) \tag{4}$$

Where $\Delta\tau_{ij}^{\ k}(t)$ is the incremental pheromone ant k leave on the edge between city i and city j at time t. For the ant k, the incremental pheromone can be calculated by applying the following formula:

$$\Delta \tau_{ij}{}^{k}(t) = \begin{cases} \frac{Q}{L_k} \\ 0 \end{cases} \tag{5}$$

Where Q is a given constant, L_k is the length of path the ant traveled so far.

Ant colony algorithm implementation steps are as follows:

1) Initialization:

Set the initial amount of pheromone on path (i, j), $\tau_{ij}(0) = C$, $\Delta \tau_{ij}{}^{k}(0) = 0$ .Put M ants in N cities randomly, bulid taboo list for each ants at the same time. Set initial valu for the parameters: alpha, beta, rho, Q, Set the number of iterations.

2) Iterative process:

```
while not end conditions do
for i=1 to n-1Do(Traverse all cities)
      for k =1 to M Do(loop for M ant)
         for j=1 to n Do(loop for N cities)
                 According to the formula (1) and (2), ant k choose the next city
                 j,movie the ant k to city j,put the city j into the taboo list；
         end
      end
  end
```

caculate the loop obtained from the ants. According to the formula (3), (4) and (5), update the pheromone on the path (i, j);

nc=nc+1;

end while

3) Given output, end of the algorithm

3. Experiment and analysis

The main feature of Ant Colony Algorithm is that sacrife the accuracy of the results for the efficiency of the results. Therefore it is not possible to get the optimal solution in each implementation of the algorithms. Often it just continuous approach the optimal solution. This feature determines the algorithm has a great room for adjustmen in the process of solving problems. The adjustment of the algorithm embodied in the choice of parameters.

The parameters has an important impact on the performance of the ant colony algorithm. In the ant colony system there are five main parameters influence the efficient of the algorithm:

- The amount of ants: m
- Pheromone factor: α
- Heuristic factor: β
- The residues coefficient of pheromone: ρ
- The amount of pheromone: q

Now analysis the parameters on the performance of the ant colony algorithm according to the experiment result. There use some data to test the affect of the parameters.

3.1. Experiments

1. At first, test the affect of the parameter: α . Keep the other parameter still, adjust

the value of α to get results.

m=50

α=3

β=3

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.3 and Figure.4:

Elapsed time is 100.391684 seconds. shortest_length = 7847.8095

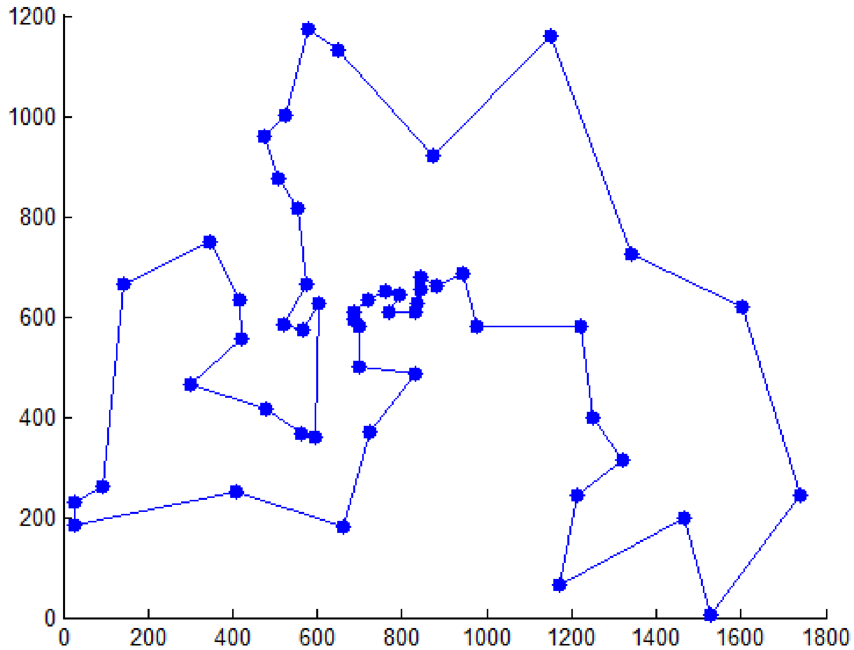


Figure.3 Shortest path

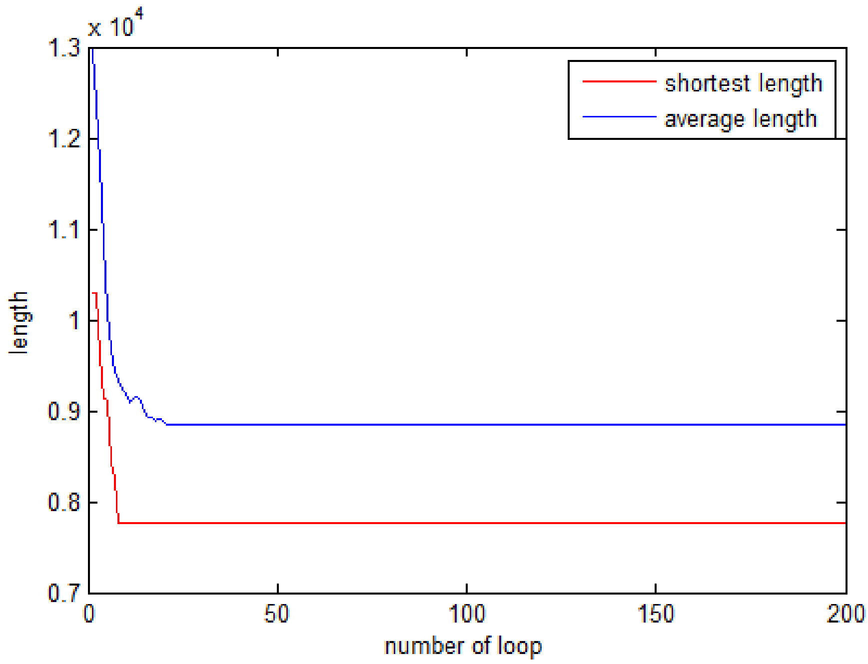


Figure.4 Evolution of the best length

Program was run for 200 iterations, the result of eil51 as Figure.5 and Figure.6:

Elapsed time is 95.715819 seconds. shortest_length =482.5811

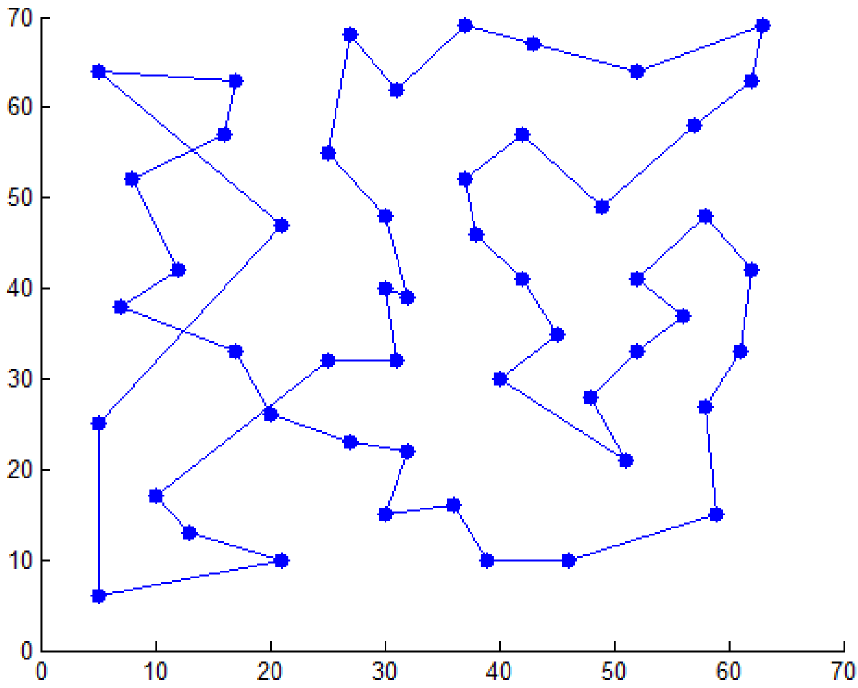


Figure.5 Shortest path

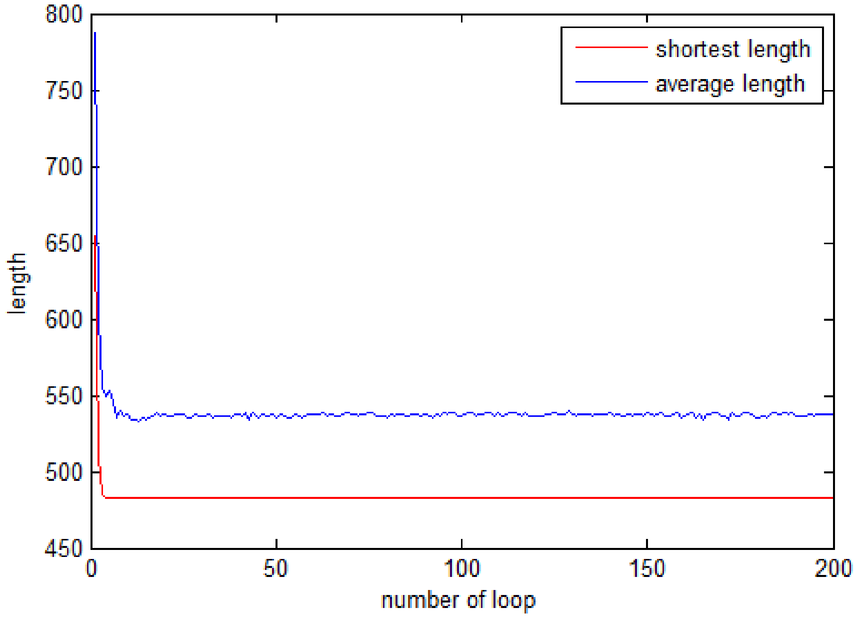


Figure.6 Evolution of the best length

m=50

α=1

β=3

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.7 and Figure.8:

Elapsed time is 96.780620 seconds. shortest_length =7663.5851

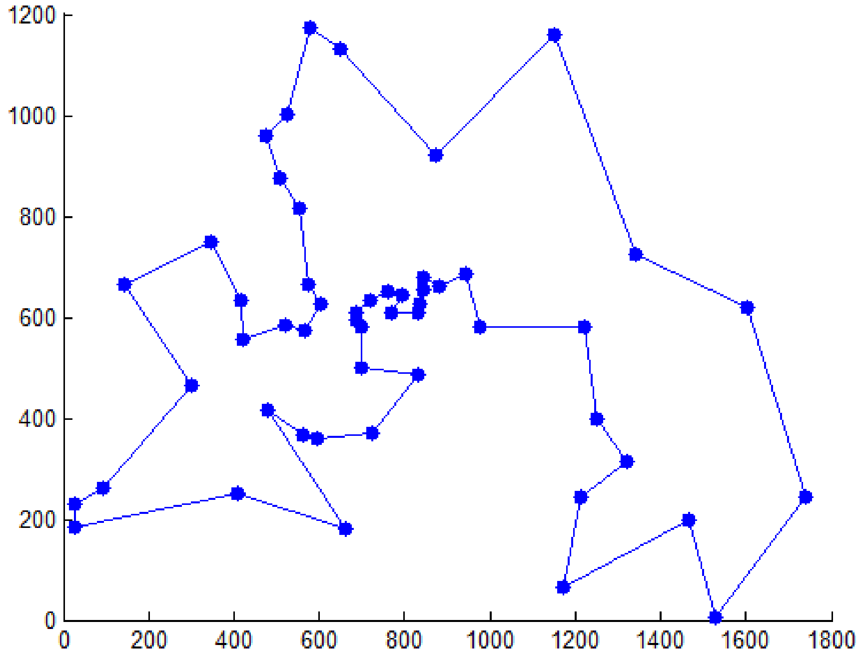


Figure.7 Shortest path

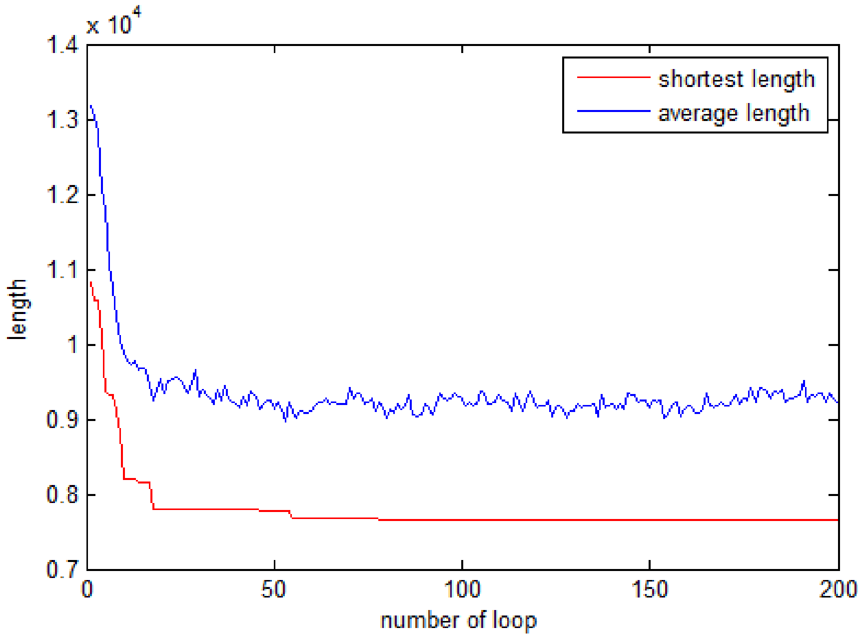


Figure.8 Evolution of the best length

Program was run for 200 iterations, the result of eil51 as Figure.9 and Figure.10:

Elapsed time is 92.781528 seconds. shortest_length =446.8725

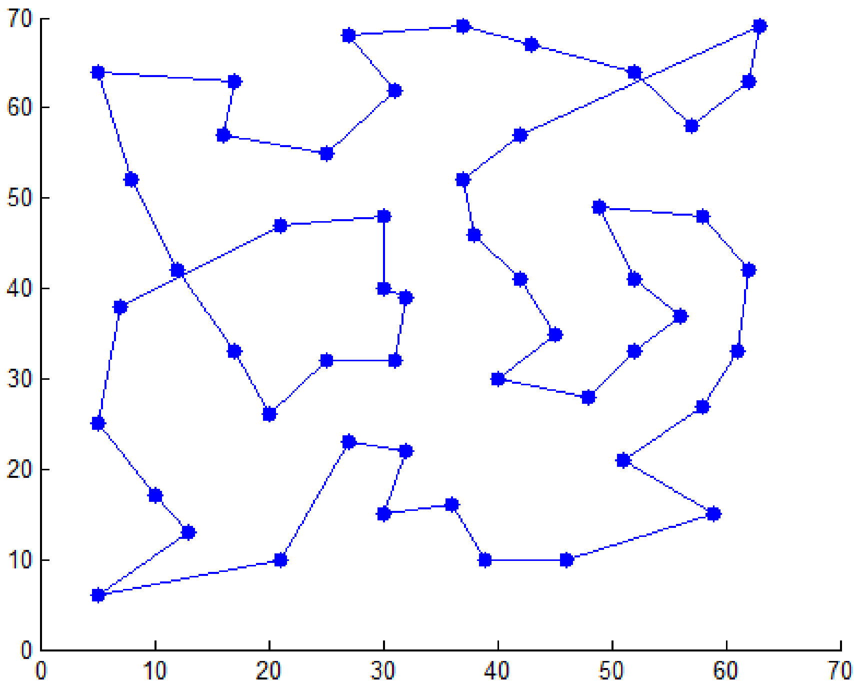


Figure.9 Shortest path

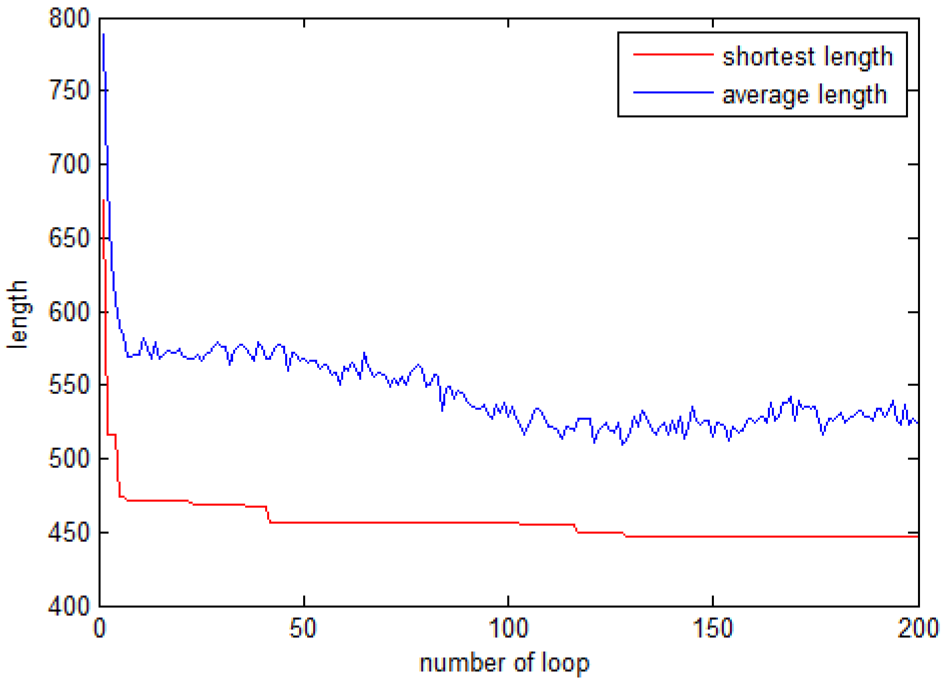


Figure.10 Evolution of the best length

The two group of parameters have similar the best result, but the evolution of average length are quite different. The higher value of the parameter α has very smooth evolution of average length, and the evolution of average length become tortuous with the lower value of parameter α.

2. Then, test the affect of the parameter: β. Keep the other parameter still, adjust the value of β to get results.

m=50

α=1

β=1

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.11 and Figure.12:

Elapsed time is 97.226648 seconds. shortest_length =8347.4485

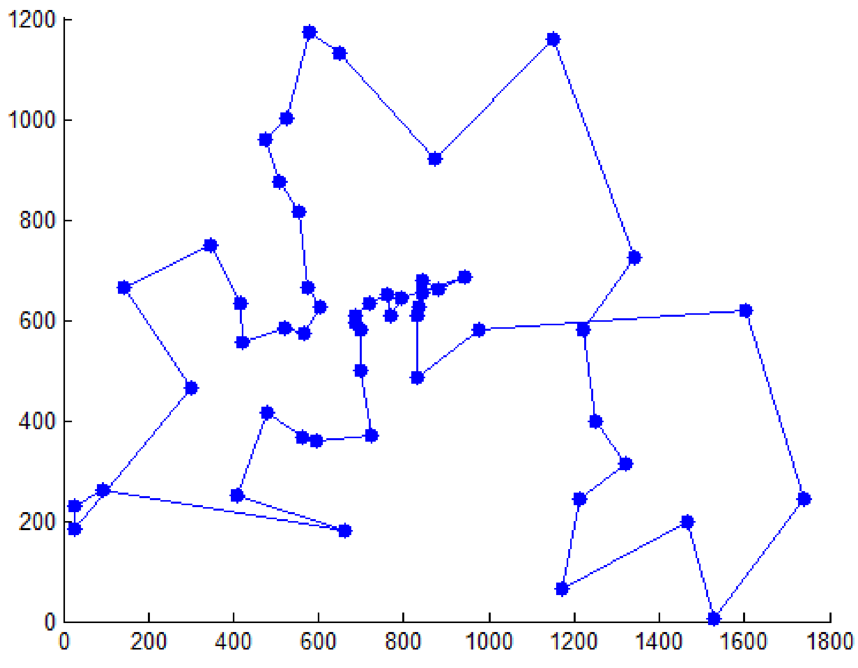


Figure.11 Shortest path

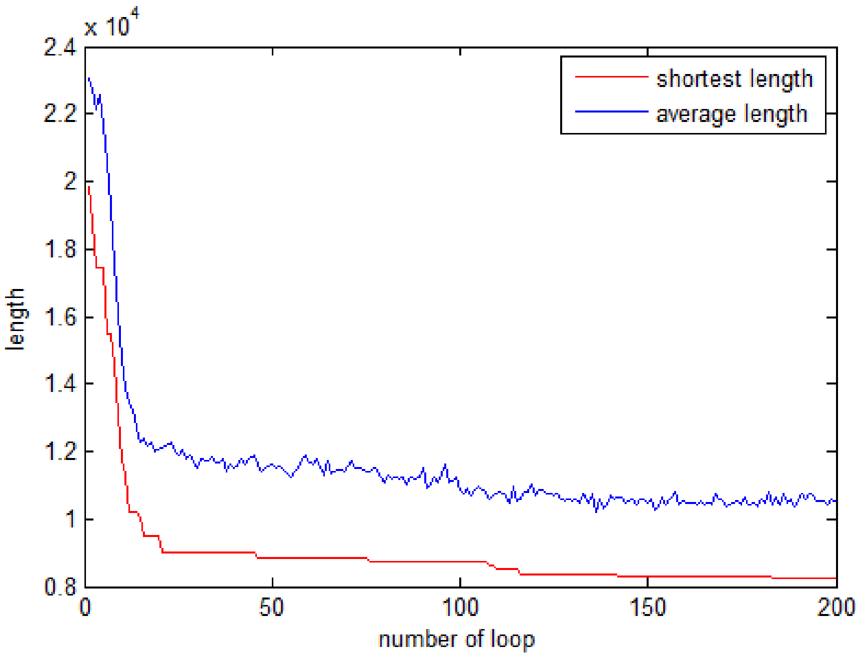


Figure.12 Evolution of the best length

Program was run for 200 iterations, the result of eil51 as Figure.13 and Figure.14: Elapsed time is 93.932650 seconds. shortest_length =460.7313

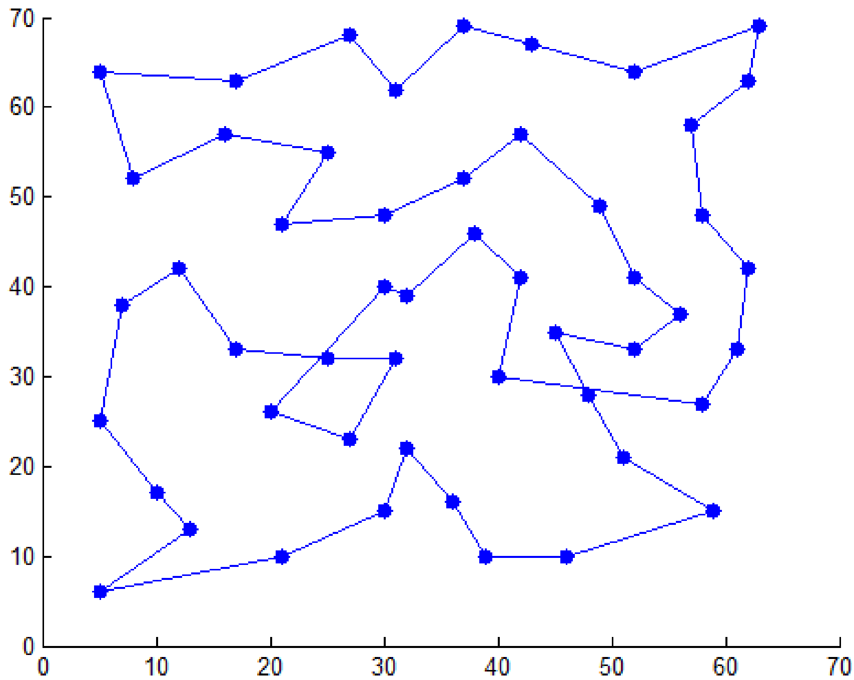


Figure.13 Shortest path

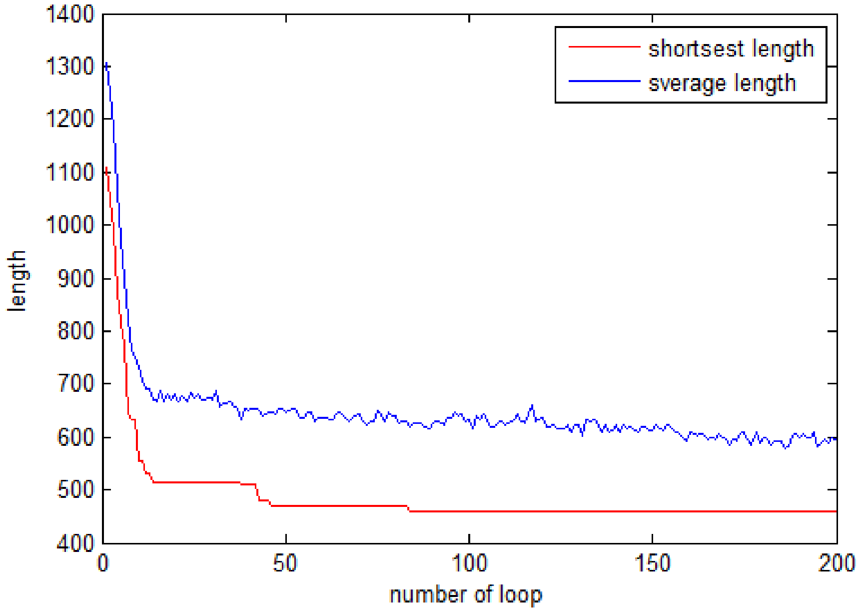


Figure.14 Evolution of the best length

m=50

α=1

β=5

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.15 and Figure.16:

Elapsed time is 102.839641 seconds. shortest_length =7654.2141

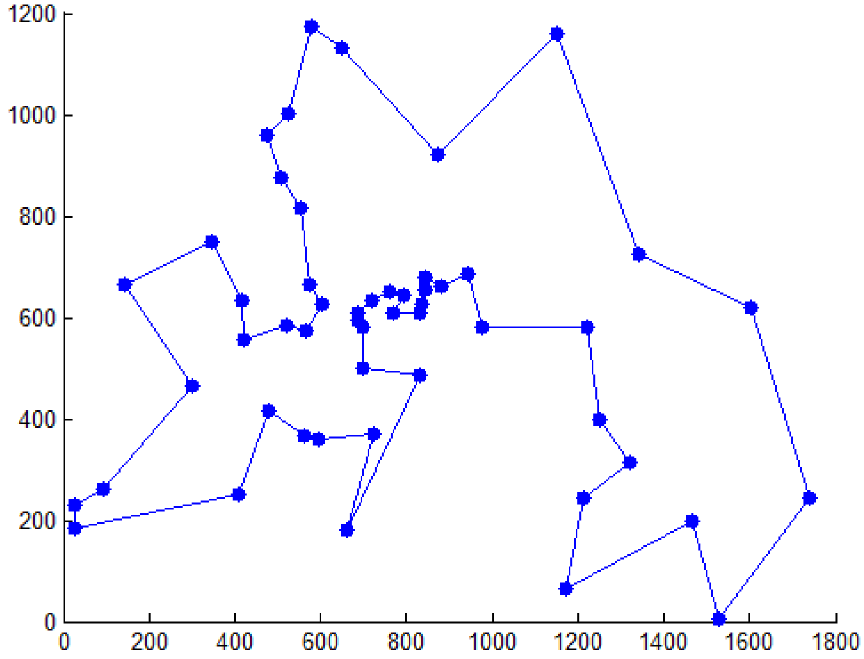


Figure.15 Shortest path

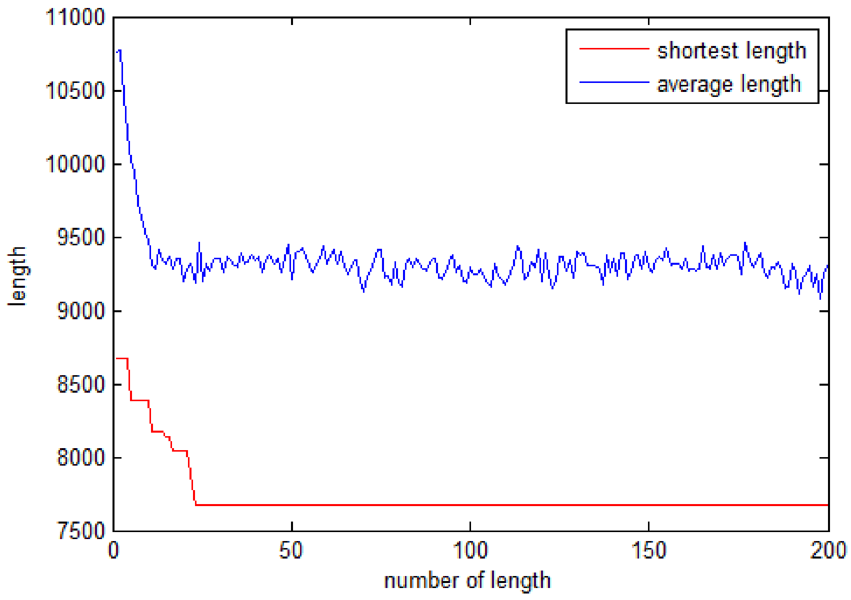


Figure.16 Evolution of the best length

Program was run for 200 iterations, the result of eil51 as Figure.17 and Figure.18:

Elapsed time is 98.920805 seconds. shortest_length =456.8866

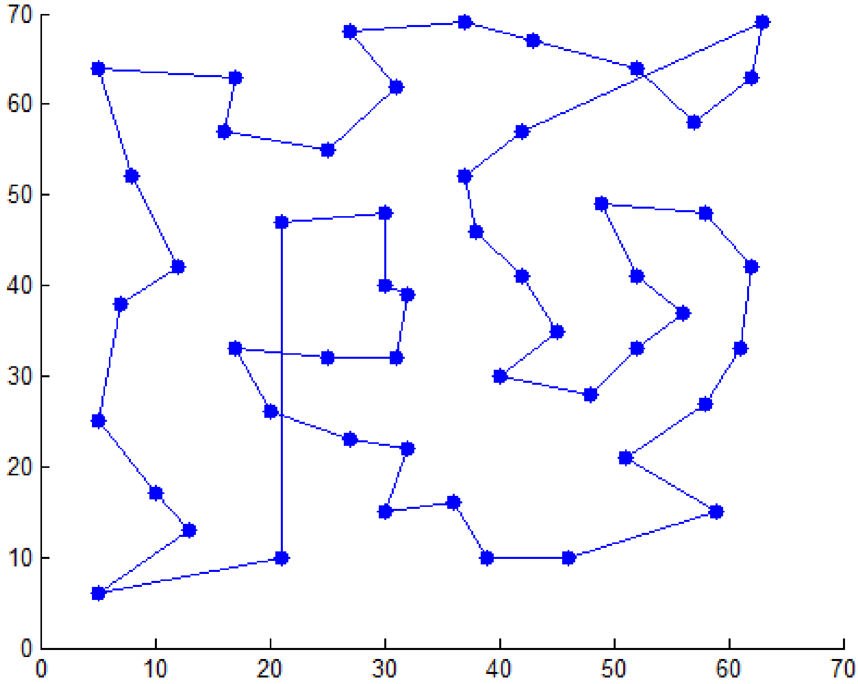


Figure.17 Shortest path

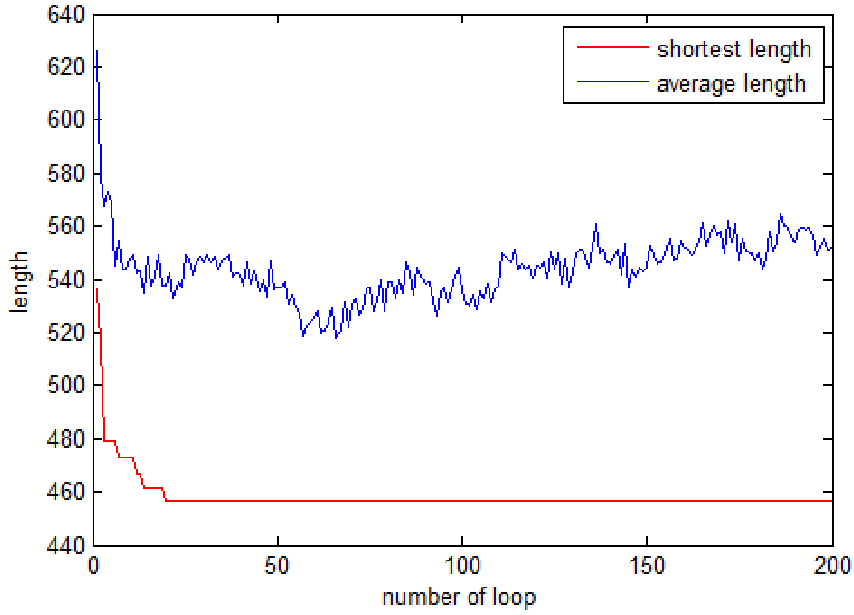


Figure.18 Evolution of the best length

The higher value of the parameter β get the better ruslt, and has fast convergence rate. The evolution of average length of bigger parameter β is more tortuous than the smaller parameter β.

3. Then, test the affect of the parameter: ρ. Keep the other parameter still, adjust the value of ρ to get results.

m=50

α=1

β=5

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.15 and Figure.16 mentioned before.

Elapsed time is 102.839641 seconds. shortest_length =7654.2141

Program was run for 200 iterations, the result of eil51 as Figure.17 and Figure.18 mentioned before.

Elapsed time is 98.920805 seconds. shortest_length =456.8866

m=50

α=1

β=5

ρ=0.1

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.19 and Figure.20:

Elapsed time is 103.023667 seconds. shortest_length =7681.4537

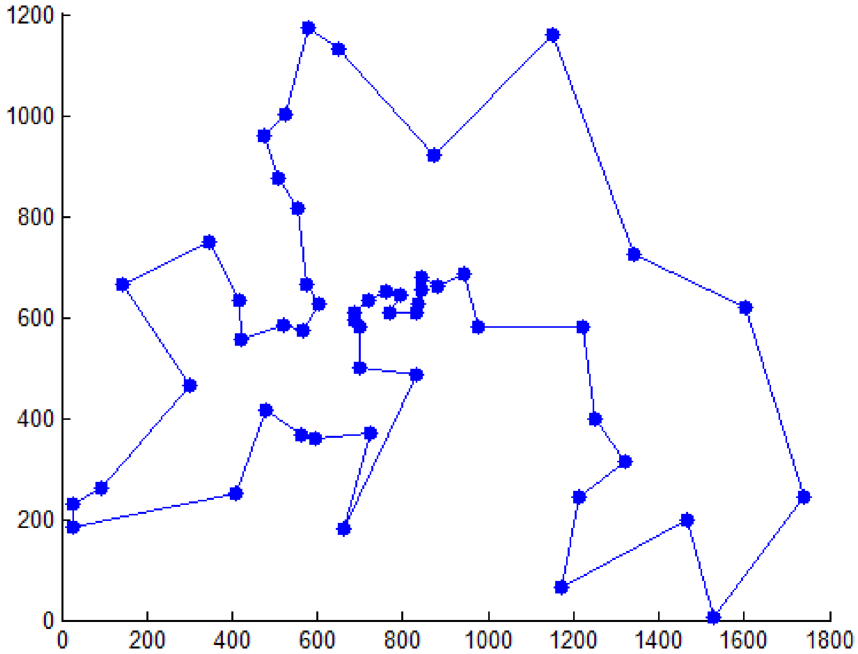


Figure.19 Shortest path

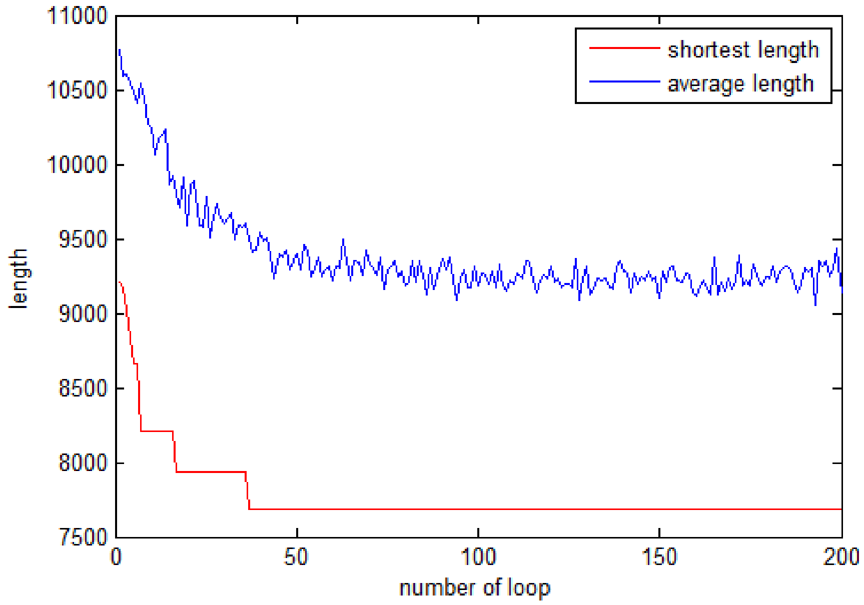


Figure.20 Evolution of the best length

Program was run for 200 iterations, the result of eil51 as Figure.21 and Figure.22: Elapsed time is 93.826971 seconds. shortest_length =429.2145

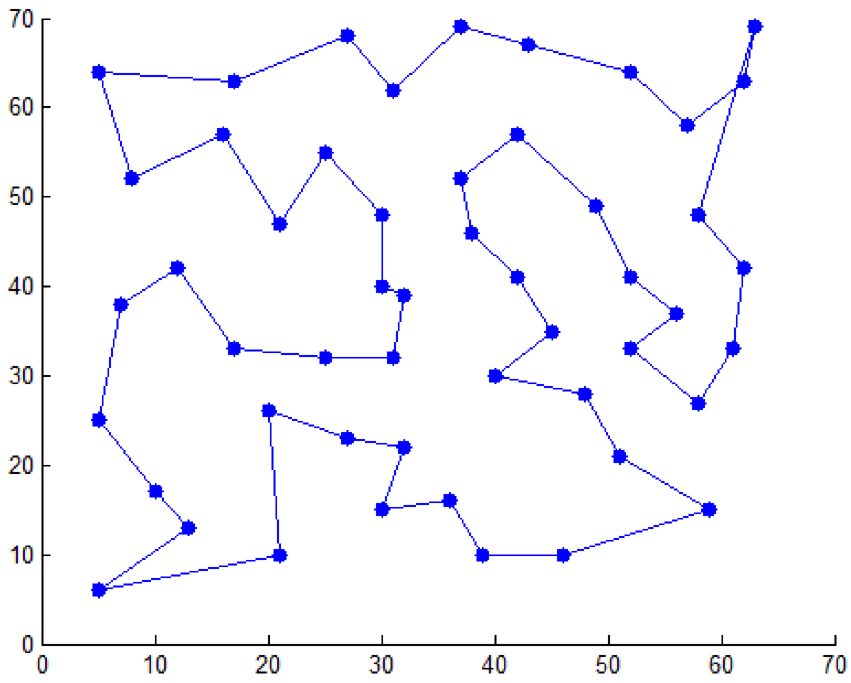


Figure.21 Shortest path

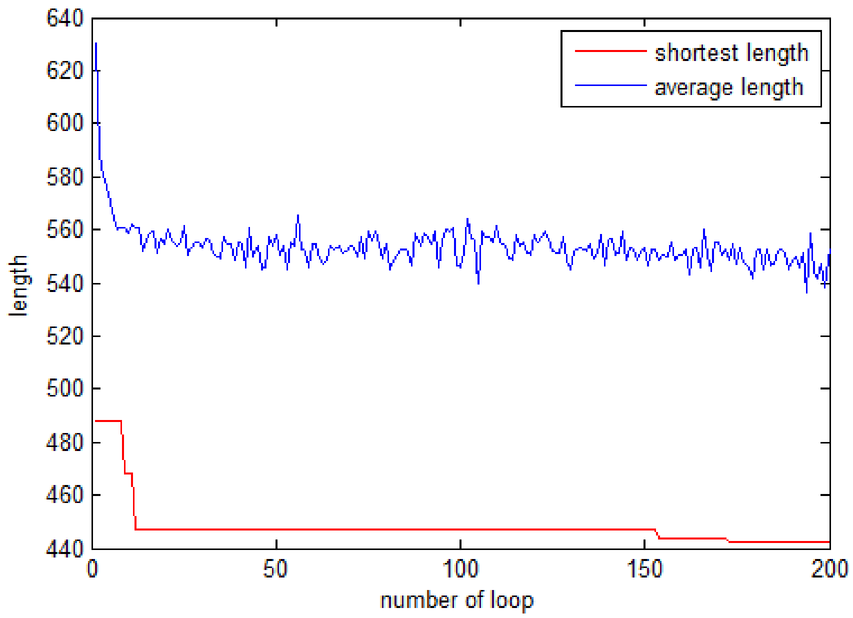


Figure.22 Evolution of the best length

The two group of parameters get nearly the same results. The smaller parameter ρ get the more tortuous evolution of average length, the bigger parameter ρ has the faster convergence rate.

4. Then, test the affect of the parameter: q. Keep the other parameter still, adjust the value of q to get results.

m=50

α=1

β=5

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.15 and Figure.16 mentioned before.

Elapsed time is 102.839641 seconds. shortest_length =7654.2141

Program was run for 200 iterations, the result of eil51 as Figure.17 and Figure.18 mentioned before.

Elapsed time is 98.920805 seconds. shortest_length =456.8866

m=50

α=1

β=5

ρ=0.5

q=50

Program was run for 200 iterations, the result of berlin52 as Figure.23 and Figure.24:

Elapsed time is 99.275866 seconds. shortest_length =7663.5851

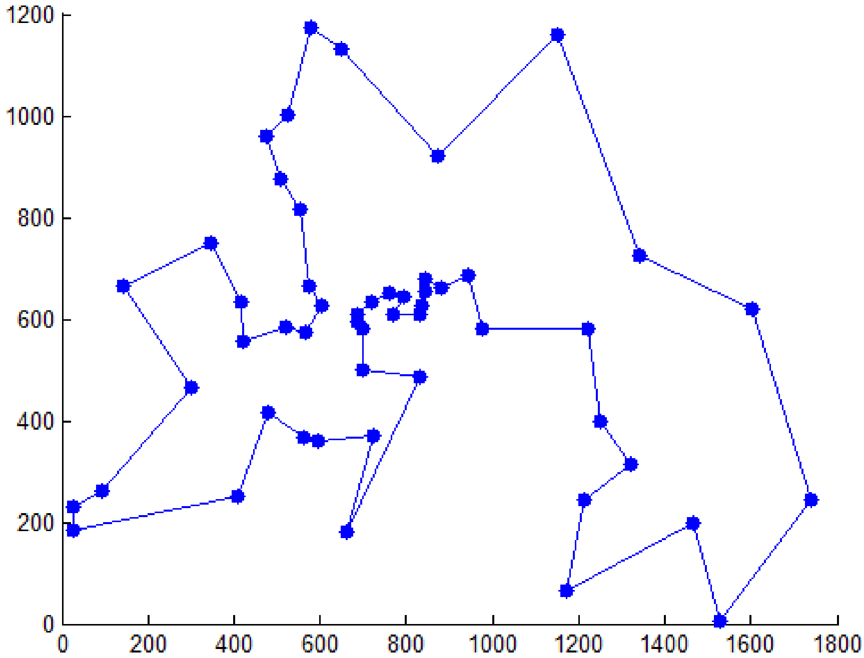


Figure.23 Shortest path

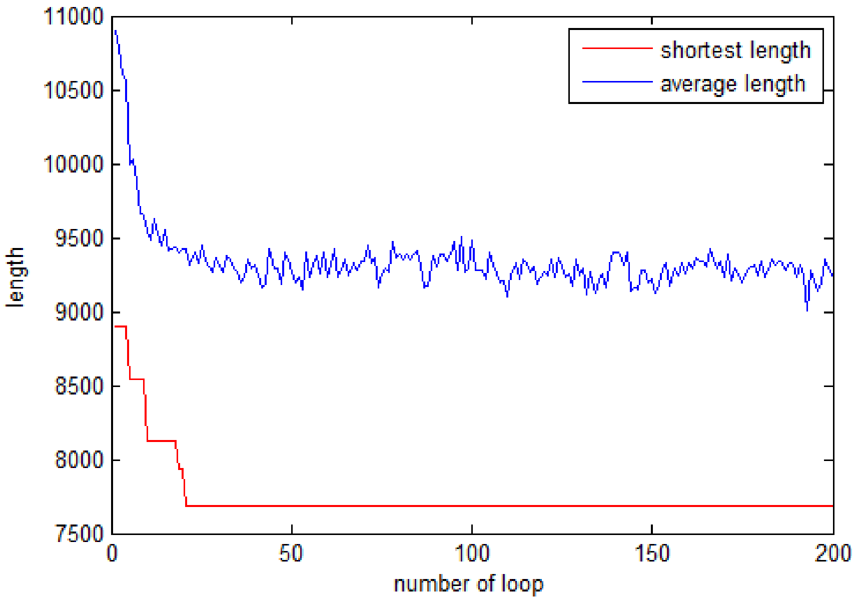


Figure.24 Evolution of the best length

Program was run for 200 iterations, the result of eil51 as Figure.25 and Figure.26: Elapsed time is 93.568189 seconds. shortest_length =450.3589

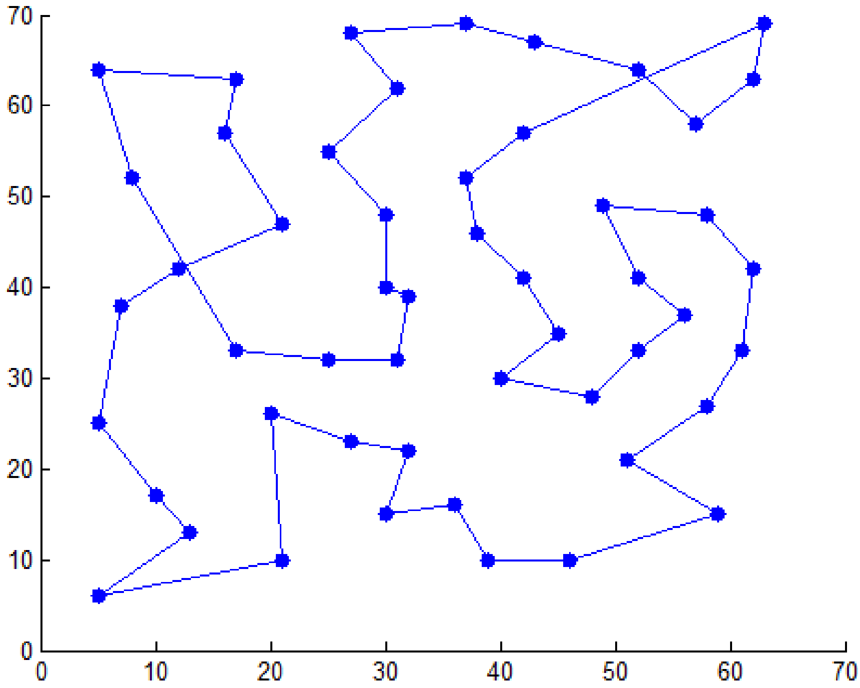


Figure.25 Shortest path

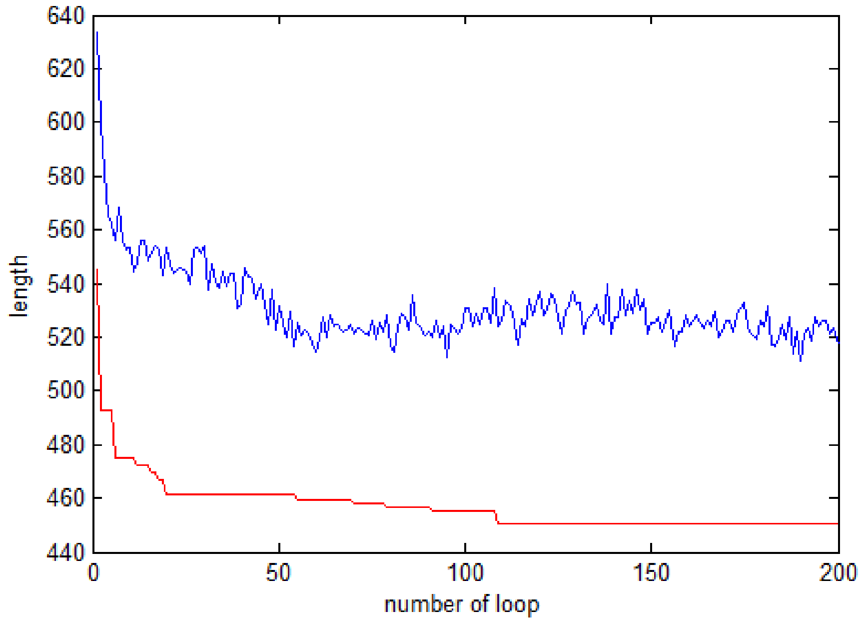


Figure.26 Evolution of the best length

The results of two group of parameter are nearly the same, the parameter has Little effect on the algorithm.

5. Then, test the affect of the parameter: m. Keep the other parameter still, adjust

the value of m to get results.

m=50

α=1

β=5

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.15 and Figure.16 mentioned before.

Elapsed time is 102.839641 seconds. shortest_length =7654.2141

Program was run for 200 iterations, the result of eil51 as Figure.17 and Figure.18 mentioned before.

Elapsed time is 98.920805 seconds. shortest_length =456.8866

m=100

α=1

β=5

ρ=0.5

q=100

Program was run for 200 iterations, the result of berlin52 as Figure.27 and Figure.28:

Elapsed time is 202.793259 seconds. shortest_length =7687.8621

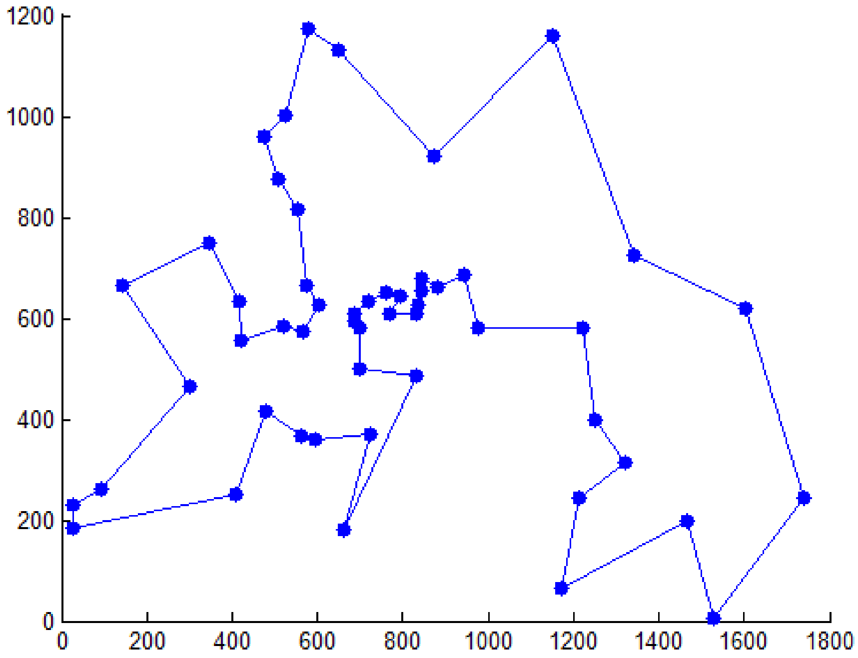


Figure.27 Shortest path

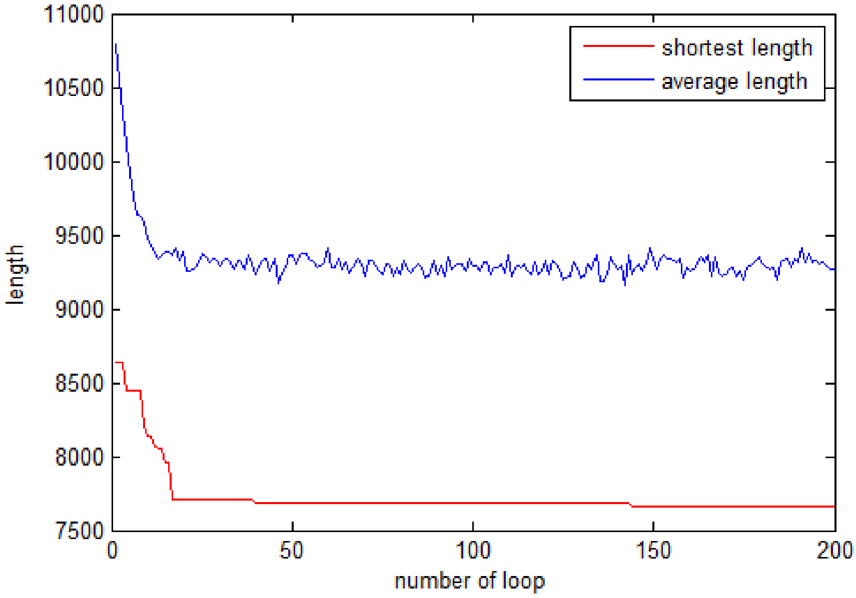


Figure.28 Evolution of the best length

Program was run for 200 iterations, the result of berlin52 as Figure.29 and Figure.30:

Elapsed time is 198.128342 seconds. shortest_length =429.6081

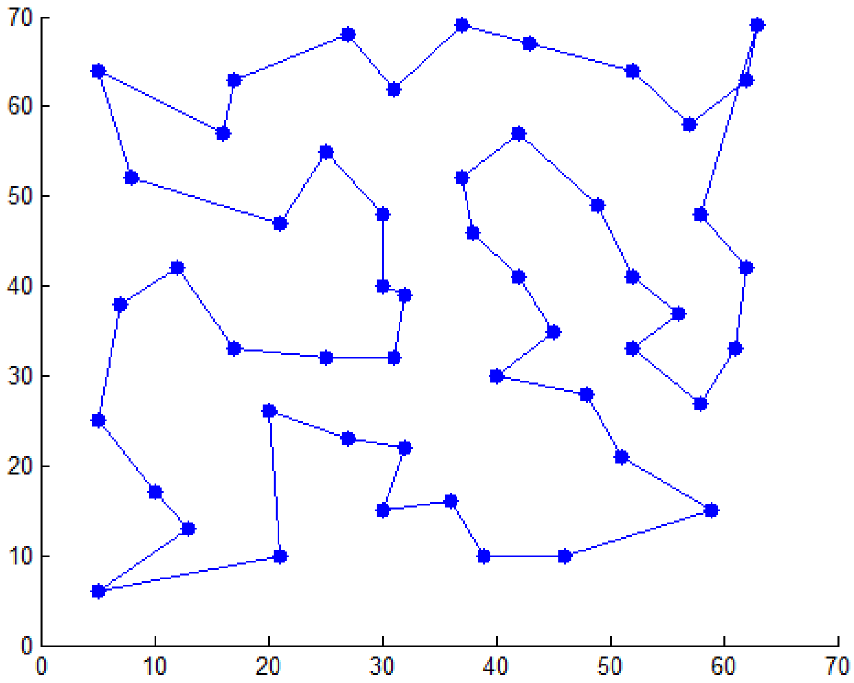


Figure.29 Shortest path

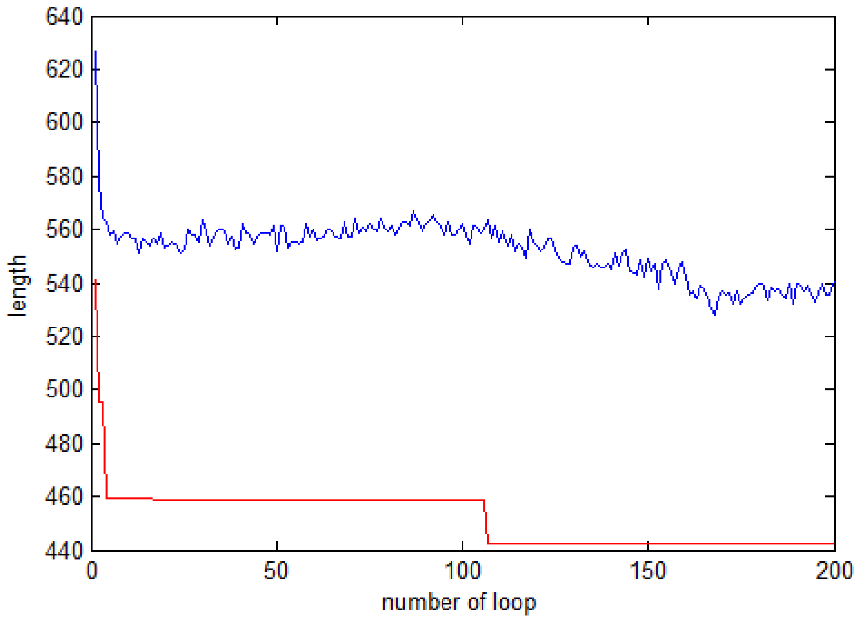


Figure.30 Evolution of the best length

The two groups of parameters get nearly the same results. The smaller parameter m get the more tortuous evolution of average length, the bigger parameter m has the faster convergence rate. But the bigger amount of ants cost much more time than the smaller amount.

The tables below are the simulation results, for facilitating analysis:

Table.1 Simulation results

TSP Instance name	Number of nodes	m	α	β	ρ	q	Elapsed time	Best solution for AC	Best Known solution
Berlin52	52	10	1	3	0.3	100	20	7727	7542
		25	1	3	0.3	100	49	7677	
		50	1	3	0.3	100	96	7548	
		75	1	3	0.3	100	139	7548	
		100	1	3	0.3	100	190	7548	
		150	1	3	0.3	100	286	7548	

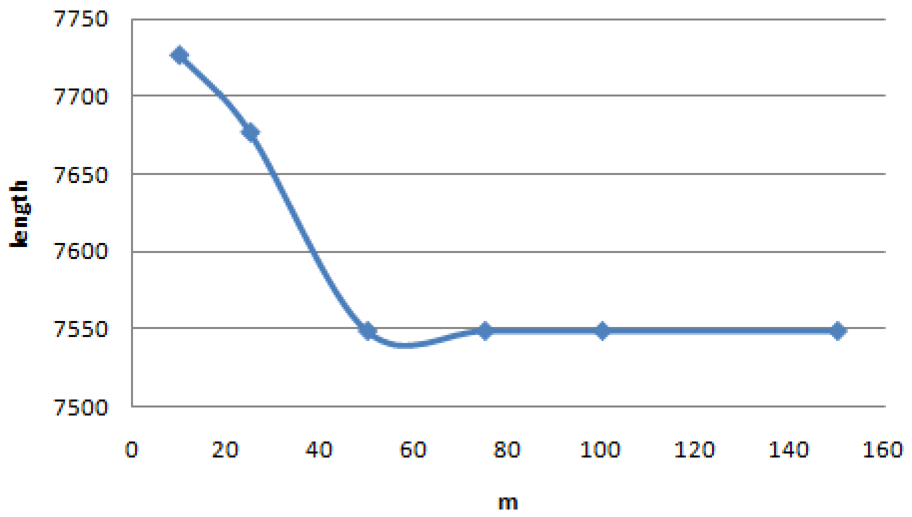


Table.2 Simulation results

TSP Instance name	Number of nodes	m	α	β	ρ	q	Elapsed time	Best solution for AC	Best Known solution
Berlin52	52	50	1	3	0.3	100	96	7548	7542
		50	3	3	0.3	100	103	7855	
		50	5	3	0.3	100	101	7999	
		50	7	3	0.3	100	100	8080	
		50	9	3	0.3	100	105	8229	

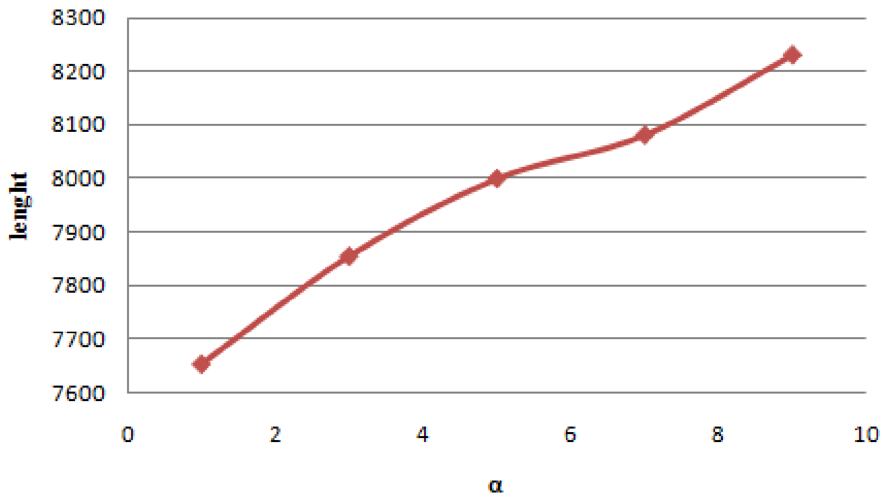


Table.3 Simulation results

TSP Instance name	Number of nodes	m	α	β	ρ	q	Elapsed time	Best solution for AC	Best Known solution
Berlin52	52	50	1	1	0.3	100	102	8312	7542
		50	1	3	0.3	100	105	7548	
		50	1	5	0.5	100	101	7681	
		50	1	7	0.5	100	95	7742	
		50	1	9	0.5	100	100	7681	

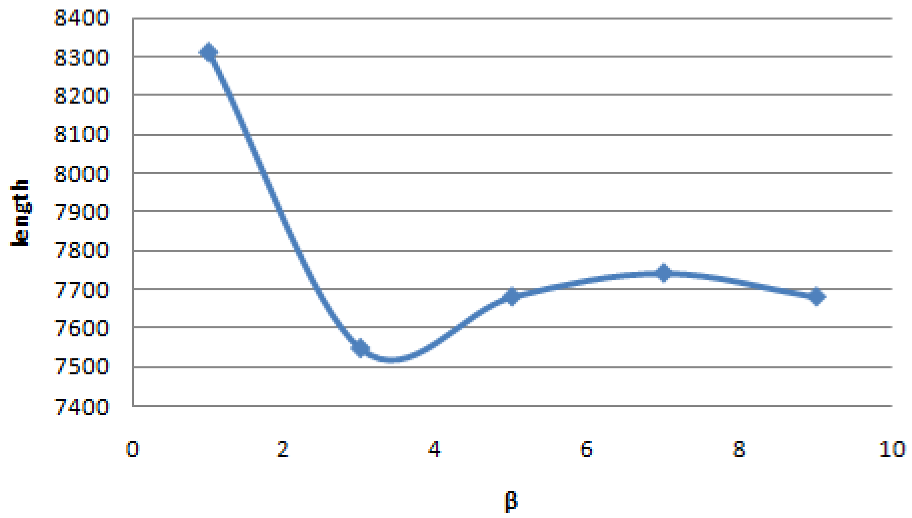


Table.4 Simulation results

<table>
<tr><th>TSP Instance name</th><th>Number of nodes</th><th>m</th><th>α</th><th>β</th><th>ρ</th><th>q</th><th>Elapsed time</th><th>Best solution for AC</th><th>Best Known solution</th></tr>
<tr><td rowspan="5">Berlin52</td><td rowspan="5">52</td><td>50</td><td>1</td><td>5</td><td>0.1</td><td>100</td><td>103</td><td>7548</td><td rowspan="5">7542</td></tr>
<tr><td>50</td><td>1</td><td>5</td><td>0.3</td><td>100</td><td>103</td><td>7548</td></tr>
<tr><td>50</td><td>1</td><td>5</td><td>0.5</td><td>100</td><td>108</td><td>7681</td></tr>
<tr><td>50</td><td>1</td><td>5</td><td>0.7</td><td>100</td><td>106</td><td>7681</td></tr>
<tr><td>50</td><td>1</td><td>5</td><td>0.9</td><td>100</td><td>105</td><td>7681</td></tr>
</table>

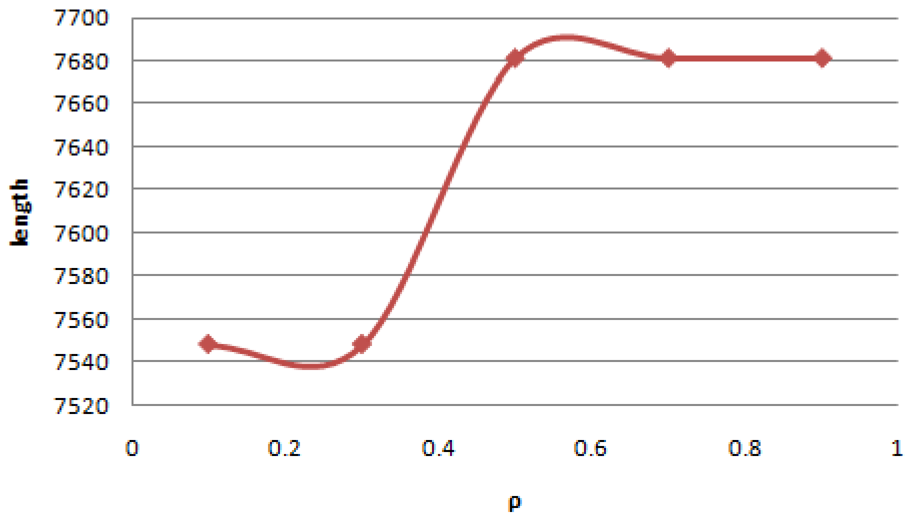
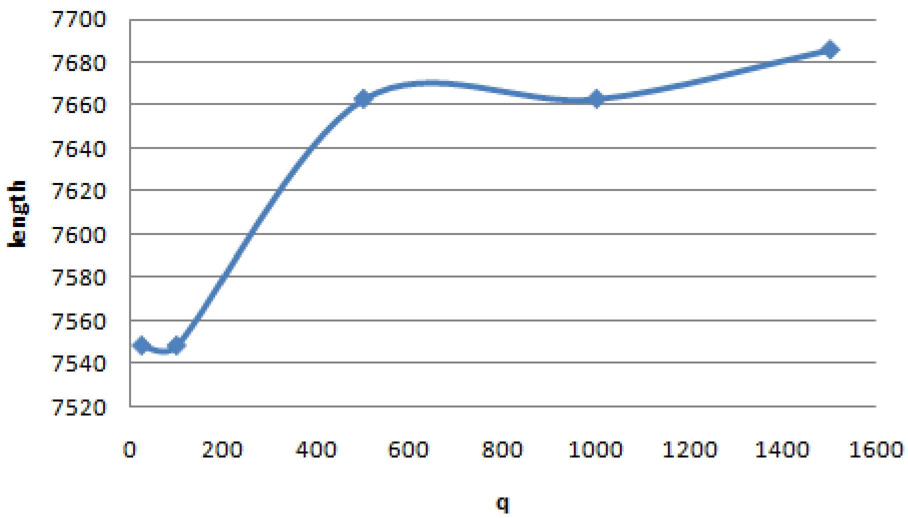


Table.5 Simulation results

TSP Instance name	Number of nodes	m	α	β	ρ	q	Elapsed time	Best solution for AC	Best Known solution
Berlin52	52	50	1	5	0.1	25	97	7548	7542
		50	1	5	0.1	100	104	7548	
		50	1	5	0.1	500	101	7663	
		50	1	5	0.1	1000	103	7663	
		50	1	5	0.1	1500	101	7686	



As the results showed in the tables, the best ant colony model for the problem berlin52 has the parameters as m=50 α=1 β=3 ρ=0.3 q=100.And the result of this group of parameters is 7548 as follow, which equals the best result we have got so far, very close to the optimal solution.

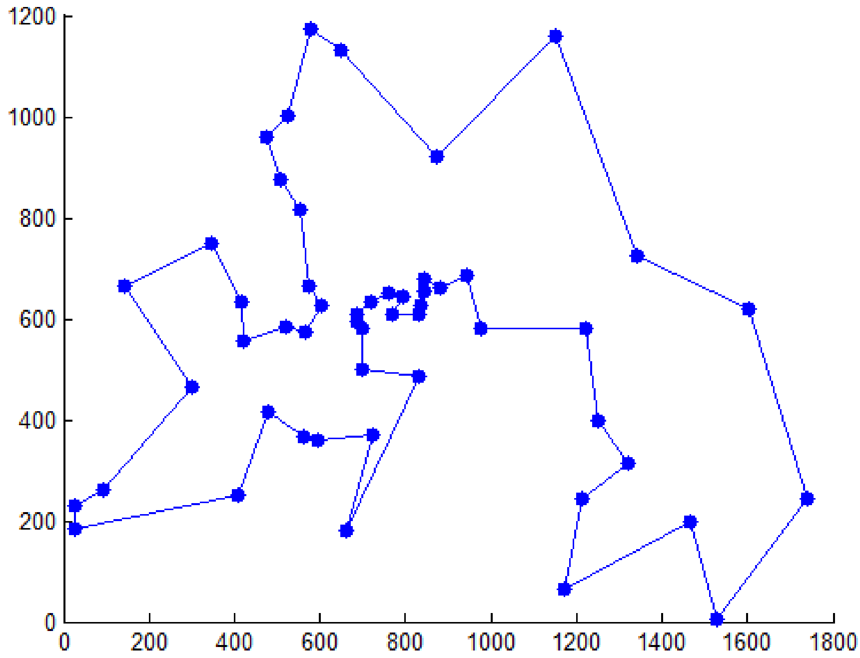


Figure.31 Shortest path

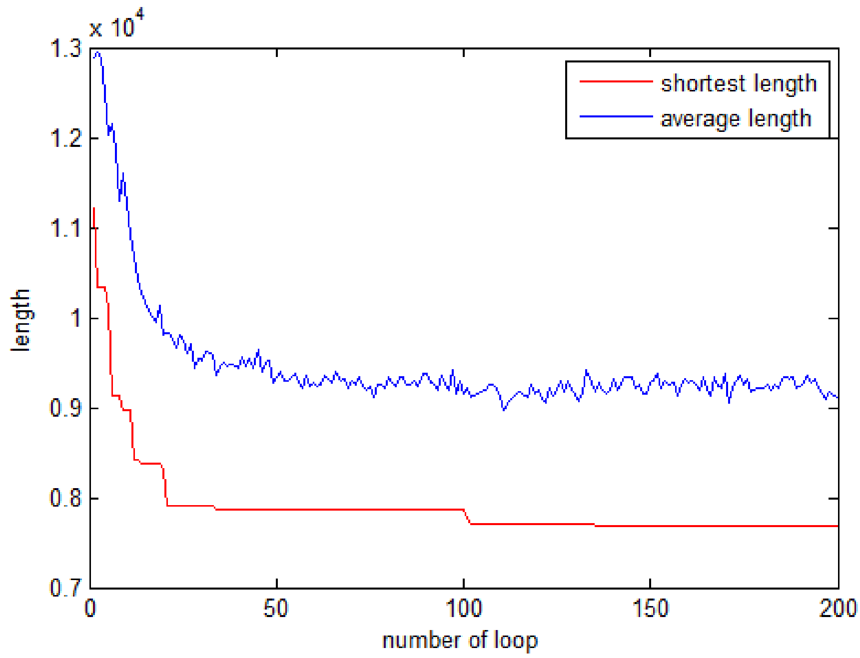


Figure.32 Evolution of the best length

3.2. Analysis of pheromone factor α and heuristic factor β

The pheromone factor α and heuristic factor β represent the importance of pheromone τ_{ij} and heuristic value η_{ij} during the algorithm implementation. In the guiding ants searching process, parameter α reflects the importance of accumulated amount of pheromone in the ant's travel; and the parameter β reflects the importance of heuristic value in the ant's travel. The value of α reflects the intensity of random factor. We can see form the Table2 and Talbe3, the solution become bad when the α increase, and get the best result when β is 3. As the increase of α, the greater possibility of ants choose the path belong to the previous tour will be, in the new tour. Parameter α affects the randomness of ants search. Parameter β reflects the importance of the intensity of heuristic value. As the increase of β, the possibility of ants choosing the local shortest path will increased, the speed of convergence gets faster, but the randomness of search optimal solution will decrease.

To get optimal solution, algorithm must get balance between global optimization and fast convergence. Here, for the ant colony algorithm is the choice of parameter α and β.

3.3. Analysis of the residues coefficient of pheromone ρ

In the ant colony system, the pheromone left before will gradually residue as the time passed. The residues coefficient ρ directly related to the global search capability and convergence rate of ant colony algorithm. The residues coefficient ρ is a value between 0 and 1, stand for the pheromone residue level. As the result of Table4 when ρ is 0.2 ACS get good result, the solution is better when ρ is 0.3, but when ρ become bigger, the quality reduced. The parameter ρ show the intensity of affect between the individual ants. Because of the residues coefficient ρ, when the problem size become lager, the pheromone on the path which never be searched before will reduce to near 0, it reduces the global search capability of algorithm. The greater the value is, the pheromone will exit longer, the effect of the following ants will increase, the greater possibility of ants choose the path belong to the previous tour will be, it affect the randomness and global searching. At the same time, the different pheromone concentration between each path became smaller, the evolution toward the optimal solution will slow down. So, smaller ρ value can improve the random performance and global search capability of algorithm, but it reduces the solution quality, that pheromone volatilize to fast may cause the individual ants ignore the possible interaction of each other, make the algorithm far from the optimal solution.

3.4. Analysis of the amount of ants m

Ant colony algorithm is a random search algorithm, through the evolution of the group of the candidate solutions to get the optimal solution. The reason why the ants have complex and orderly behaviors, is the cooperation and exchange between individual ants. Each path through which the individual ants complete a tour represents one solution, m ants' tour in each loop is the subset of the set of solution. The larger subset (ie, amount of ants) can improve the global search ability and stability of the ant colony algorithm. As the results in Table1 the shortest length decrease when m become bigger, and it stopped at m=50. But the elapsed time increased violent. However, after increasing the number of ants, the comparison changes of pheromone on the path has been searched become average. The feedback of information is not obvious.

Although the increase of the ants amount is benefit for the searching. However, when the number of ants is much larger than the problem size, the convergence of the algorithm become slow. Excessive number of ant algorithm will only improve the performance of algorithm a little, just increase the time complexity of the algorithm.

3.5. Analysis of the amount of pheromone q

Q is the total amount of pheromone released in the path by the ants in one loop (ie, through all of the city once). It is a constant in the algorithm. As the results in Table5 when the problem size is small, smaller q can get better result. The solution quality decreased when the q become bigger. The greater the total amount of pheromone q, the faster accumulation of pheromone on the path, Strengthen the feedback effect in the search. In the ant colony algorithm, the role of each parameter is closely related to each other, the major parameter influence the algorithm performance is the pheromone factor α, heuristic factor β and the residues coefficient of pheromone ρ. The effect of amount q to the performance depends on the three parameters (pheromone factor α, heuristic factor β and the residues coefficient of pheromone ρ).

3.6. Test and compare

Now use TSP instances from TSPLIB to test the performance of the ant colony algorithm. These test problems were chosen either because there was available data to compare our results with those obtained by other methods or with the optimal solutions, or to show the ability of ant colony algorithm in solving difficult instances of the TSP.

The performance of ACS was compared with the performance of other naturally inspired global optimization methods: simulated annealing (SA) genetic algorithm (GA), results as follows:

Table.6 Comparison results

TSP Instance name	Number of nodes	Genetic Algorithm	Simulate Annealing	Ant Colony	Best known solution
Berlin52	52	7559	7554	7548	7542
Eil51	51	438	443	429	426
Eil76	76	546	545	555	529
Kroa100	100	21761	22523	21416	21282
Lin105	105	14557	14742	14446	14379

Now test the ant colony algorithm on some bigger problems to study its behavior for increasing problem dimensions. The quality of the produced solutions is given in terms of the relative deviation from the optimum, that is $\omega = \frac{100*(ac - opt)}{opt}$, where ac denotes the cost of the optimum found by ant colony algorithm, and opt is the cost of the optimal solution. The following tables are the results of the ant colony algorithm:

Table.7 Test results(1/2)

TSP Instance name	Number of nodes	Ant Colony	Best known solution	Relative deviation (%)
Kroa100	100	21416	21282	0.630
Lin105	105	14446	14379	0.466
Pr124	124	59584	59030	0.939
Bier127	127	120279	118282	1.688
Pr144	144	58704	58537	0.285
Ch150	150	6565	6528	0.567
D198	198	16000	15780	1.394
Kroa200	200	29826	29368	1.560
Ts225	225	128745	126643	1.660
A280	280	2644	2579	2.520

Table.7 Test results(2/2)

TSP Instance name	Number of nodes	Ant Colony	Best known solution	Relative deviation (%)
Lin318	318	43756	42029	4.109
Fl417	417	11985	11861	1.045
PCB442	442	52443	50778	3.279
Pat575	575	6954	6773	2.672
D675	675	50791	48912	3.842
U724	724	43411	41910	3.842
Pcb1173	1173	60490	56892	6.324
D2103	2103	86941	80450	8.063
Rl5915	5915	631301	565530	11.62
Rl11849	11849	1046571	923228	13.36

The results show that ant colony is able to achieve good performance for the benchmark problems. While the results reported in table 3 show that ant colony algorithm has good performance compared to other approaches. Table 4 shows that there is still room for improvement in ant colony model when size of the problem instances increase. There are a few reasons to this. One of them is the parameter settings which need to be fine-tuned to cater for different scenarios.

4. Conclusion

Ant colony algorithm is a good feedback mechanism for the universal problem. This paper is about a bionic optimization algorithm - Ant Colony Algorithm. In this paper, introduced the ant colony algorithm and did a detailed analysis to the main parameters of the algorithm. Main tasks as the following: Modeling the biological model, the mathematical model and algorithm for the ant colony algorithm. Analyze the main parameter of the ant colony algorithm.

A ant colony model based on the ants' behaviours has been introduced to solve TSP. The model has been tested on a set of benchmark problems. Although the ant colony algorithm has many advantages, but there also are some flaws. Algorithm requires a long search time. Because at the beginning, the pheromone on each path has little different, after a long period of time, the better path has more pheromone than the other path. Ant colony algorithm sometimes has stagnation phenomenon. Ants always tend to the path with the strongest pheromone, sometimes all the ants are concentrated in this local optimum path, Stagnation phenomenon happens.

We hope to improve the model further to achieve optimal values for the list of benchmark problems. There are many ways in which ant colony system can be improved when it face larger problem instances. Parameter setting is quite important to the performance of ant colony algorithm Constructive heuristics can be integrated in preparing initial solutions. This helps the ant colony algorithm to have a better starting point in its search for optimal solutions First, local optimization heuristic like 2-opt, 3-opt can be used in the ant colony algorithm. Second, the algorithm is amenable to efficient parallelization, which could greatly improve the performance for finding good solutions. For example: distributing ants on different processors, the same TSP is then solved on each processor by a smaller number of ants, exchange the best tour among processors.

References

[1]. M．Dorigo V．Manierio，A．Collomi. Ant system: Optimization by a colony of cooperating agents. IEEE Trans on SMC, 1996, 26(1):28-41

[2]. Beckers, R., Deneubourg, J.L. and Goss, S., 1992, Trails and U-turns in the selection of the shortest path by the ant Lasius niger. Journal of Theoretical Biology, 159, 397–415.

[3]. Hölldobler, B. and Wilson, E.O., 1990, The ants (Springer-Verlag, Berlin).

[4]. E.L.Lawler, J.K.Lenstra, A. H.G Rinnoy-Kan, Eds. The Traveling Salesman Problem. New York: Wiley, 1985

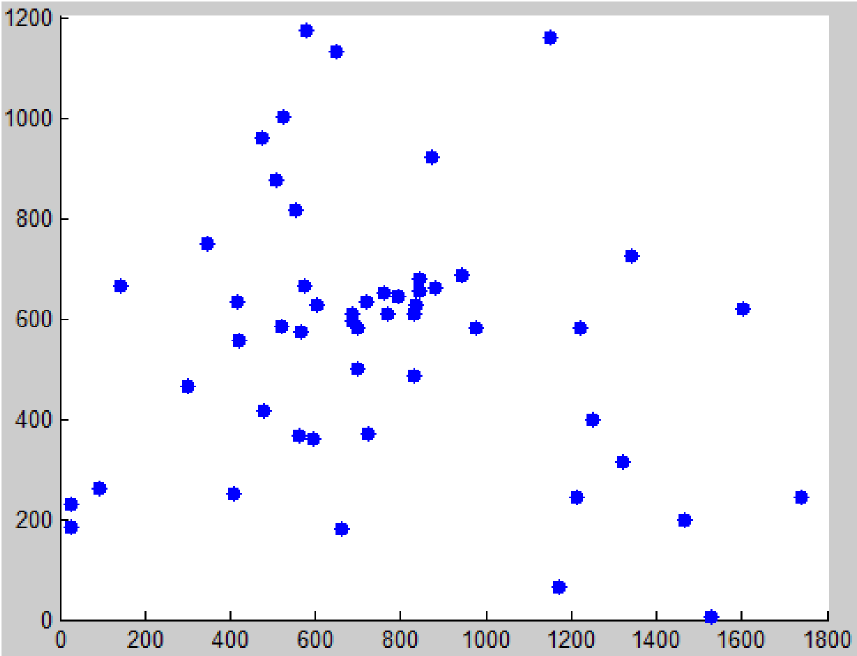
[5]. Luca Maria Gambardella and Marco Dorigo. Solving symmetric and asymmetric TSP's by ant colonies. Proc. IEEE Int. Conf. Evolutionary Computation,IEEE-EC 96,1996,pp.622-627

[6]. Christos H. Papadimitriou and Kenneth Steiglitz. Combinatorial Optimization: Algorithms and Complexity. New Jersey: Prentice-ha11,1982

[7]. http://elib.zib.de/pub/PacKages/mp-testdata/tsp/tsplib/tsplib.html

Appendix A: Test data files

The test data berlin52.txt, has 52 nodes:



Node number X coordinate Y coordinate

1 565.0 575.0

2 25.0 185.0

3 345.0 750.0

4 945.0 685.0

5 845.0 655.0

6 880.0 660.0

7 25.0 230.0

8 525.0 1000.0

9 580.0 1175.0

10 650.0 1130.0

11 1605.0 620.0

12 1220.0 580.0

13 1465.0 200.0

14 1530.0 5.0

15 845.0 680.0

16 725.0 370.0

17 145.0 665.0

18 415.0 635.0

19 510.0 875.0

20 560.0 365.0

21 300.0 465.0

22 520.0 585.0

23 480.0 415.0

24 835.0 625.0

25 975.0 580.0

26 1215.0 245.0

27 1320.0 315.0

28 1250.0 400.0

29 660.0 180.0

30 410.0 250.0

31 420.0 555.0

32 575.0 665.0

33 1150.0 1160.0

34 700.0 580.0

35 685.0 595.0

36 685.0 610.0

37 770.0 610.0

38 795.0 645.0

39 720.0 635.0

40 760.0 650.0

41 475.0 960.0

42 95.0 260.0

43 875.0 920.0

44 700.0 500.0

45 555.0 815.0

46 830.0 485.0

47 1170.0 65.0

48 830.0 610.0

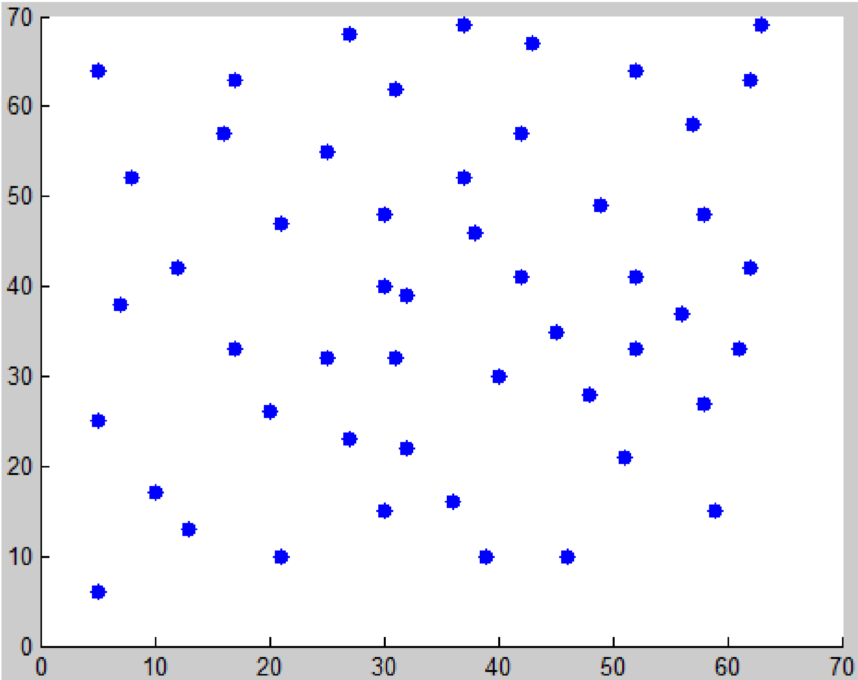
49 605.0 625.0

50 595.0 360.0

51 1340.0 725.0

52 1740.0 245.0

The test data eil51.txt, has 51 nodes:



Node number X coordinate Y coordinate

1 37 52

2 49 49

3 52 64

4 20 26

5 40 30

6 21 47

7 17 63

8 31 62

9 52 33

10 51 21

11 42 41

12 31 32

13 5 25

14 12 42

15 36 16

16 52 41

17 27 23

18 17 33

19 13 13

20 57 58

21 62 42

22 42 57

23 16 57

24 8 52

25 7 38

26 27 68

27 30 48

28 43 67

29 58 48

30 58 27

31 37 69

32 38 46

33 46 10

34 61 33

35 62 63

36 63 69

37 32 22

38 45 35

39 59 15

40 5 6

41 10 17

42 21 10

43 5 64

44 30 15

45 39 10

46 32 39

47 25 32

48 25 55

49 48 28

50 56 37

51 30 40