Archives of
Transport System
Telematics

Volume 3

Issue 1

February 2010

Tunning Parameters of Evolutionary Algorithm in Travelling Salesman Problem with Profits and Returns

J. KOSZELEW[a], A. PIWOŃSKA[a]

[a] Faculty of Computer Science, Bialystok University of Technology, Wiejska 45a, 15-351 Białystok, Poland

EMAIL: j.koszelew@pb.edu.pl, a.piwonska@ pb.edu.pl

ABSTRACT

A huge number of papers studies Travelling Salesman Problem (TSP) in classical version. In standard TSP all cities must be visited and graph is completed. While this is indeed the case in many practical problems, there are many other practical problems where these assumptions are not valid. This paper presents a new evolutionary algorithm (EA) which solves TSP with profits and returns (TSPwPR). This version of TSP is often applied in Intelligent Transport Systems, especially in Vehicle Routing Problem (VRP). TSPwPR consists in finding a cycle which maximizes collected profit but does not exceed a given cost constraint. A graph which is considered in this problem can be not completed, salesman doesn't have to visit all cities and he can repeat (with zero profit) cities in his tour. The method was implemented and tested on real network which consists of 160 cities in eastern and central voivodeships of Poland. The main parameter which has the highest influence on quality of obtaining results is the size of population and our experiments are directed to determine an optimal value of this parameter

KEYWORDS: routing in transport networks, travelling salesman problem with profits, evolutionary algorithm

1. Introduction

Transport logistics and fleet management problems often fall into one class of the optimization problems. Finding an optimal set of routes for group of vehicles in the transport network under defined constraints is known as the Vehicle Routing Problem (VRP) [10]. VRP represents the NP- hard problem and the optimal solution cannot be reached in polynomial time. In case of solving practical vehicle routing problems that include a number of special additional constraints, the approximate solving methods should be used. The heuristic and meta-heuristic algorithms provide approximate solutions acceptable in practical application. In the case when only one vehicle is serving the customers and there are no additional constraints, the vehicle routing problem is reduced to the Travelling Salesman Problem (TSP).

In classical version TSP is formulated as follows. Given the set of n cities and distances between each pair of them, find a closed tour (a cycle) through all cities that visits each city only once and is of minimum length. The problem is known to be NP-hard, therefore many heuristics have been proposed to find near-optimal solutions [3]. While in standard TSP a salesman needs to visit each city, there are some situations where not all customers need to be visited when the optimization model is run. Consider the situation where all customers need to be visited but not necessarily in the same tour or set of tours, for instance in the cases where a customer has to be visited within

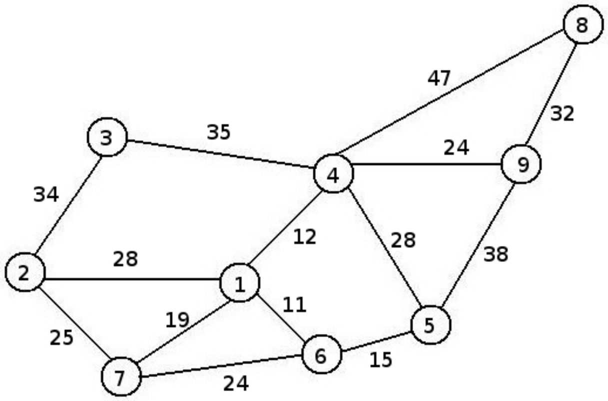

Fig. 1. A graph representation of a network of cities
Source: [own work]

a given time period, say three days. Then, when a tour or a set of tours has to be organized for a given day, there are customers that need to be visited but also customers that may be visited or whose visit may be postponed. In this case the lack of need to serve all customers in the same day comes from the dynamic nature of the problem. This feature gives rise to a number of problems which are called in the literature Travelling Salesman Problem with Profits (TSPwP) [1], [3] and [9]. In this group of problems, usually one of *n* cities has a special meaning - it is considered as a depot. In a one version of TSPwP described in the literature, the problem is to find an elementary cycle (i.e., a cycle such that each vertex is visited at most once) starting from a depot, that maximizes collected profit such that the tour length do not exceed a given constraint. This problem is called Orienteering Problem (OP) [1]. The name orienteering comes from an outdoor sport played on forest areas. Given a specified set of points, each competitor has to visit as many points as possible within a specified time limit. Like TSP, OP is NP-hard problem.

In this paper, additional assumptions to TSP were proposed. Firstly, we assume that a graph maybe not complete: some pairs of vertices are not connected. In the real world cities in some region often are not connected by through route. Despite of the fact that we can transform such a not complete graph in a complete one by introducing dummy edges, such an approach seems to be ineffective. It would result in a lot of unnecessary data introduced to the problem. The second assumption is that we allow repeated visiting of a given vertex: a cycle we are looking for must not be an elementary one. This assumption results from the fact that a graph is not complete. However, while a salesman can be in a given city more than once, a profit is realized only during first visiting. This assumption prevents from finding routes in which a city with a highest profit is continually visited while others are not. With these additional assumptions, the problem is more real-life. The variant of TSP with two above assumptions will be called Traveling Salesman Problem with Profits and Returns (TSPwPR). A new evolutionary algorithm (EA) which solves TSPwPR is presented in this paper. The method was implemented and tested on real-network which consists of 160 cities in voivodeships of Poland. Our tests are focused on verifying the size of population parameter influence on quality of obtaining tours.

The paper includes five sections. Next section describes formal definition of TSPwPR. Section 3 presents in detail a EA. Experimental results are presented in Section 4. The paper ends with some proposition of improving presented method.

2. Definition of TSPwPR

A network of cities in our model is represented by a weighted, undirected graph G = <V, E, d, p>, where V = {1, 2, …, n} is a set of n vertices, E is a set of edges, d is a function of weights and p is a vector of profits. Each node in G corresponds to a city in a network. Vertex 1 has a special meaning and is interpreted as the depot. An undirected edge $\{i, j\}\in E$ is an element of the set E and means that there is through two-way route from the city i to the city j. The weight d_{ij} for undirected edge {i, j} denotes a distance between cities i and j. Additionally, with each vertex a non-negative number meaning a profit is associated. Let $p=\{p_1, p_2, \ldots, p_n\}$ be a vector of profits for all vertices. Each value of a profit pi is a no-negative number. An important assumption is that a profit is realized only during first visiting of a given vertex. A graph representation of an exemplary network of cities is shown in Fig. 1. It is a simple example of the network which includes nine cities. The dij values are marked on the edges and the pi values are: {5, 4, 6, 2, 1, 2, 4, 3, 4}. One can see that the highest profit equals to 6 can be gained during visiting a depot. The TSPwPR can be formulated as follows. The goal is to find a cycle starting from the depot that maximizes collected profit such that the tour length do not exceed a given constraint c_{max}.

Assuming $c_{max} = 100$, for the graph presented in Fig. 1, one possible solution could be: 1 - 4 - 9 - 5 – 6 - 1. In this case the tour length equals to 100 and the collected profit equals to 14.

3. Description of EA

EAs are adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and heredity. First introduced by John Holland in the 60s, EAs has been widely studied, experimented and successfully applied in many fields [2], [5].

In a typical EA, a population of candidate solutions (called chromosomes) is evolved during artificial evolution. Traditionally, solutions are represented as binary strings, but other encodings are also possible. The EA starts from a population of randomly generated individuals. In each generation, the fitness of every individual is evaluated. Based on the fitness, individuals are stochastically selected from the current population to the next one. This step is called a selection. Individuals in the new population undergo genetic operators: crossover and mutation. The new population is then used in the next iteration of the algorithm. The algorithm usually terminates when a maximum number of generations has been reached [6].

The EA starts with a population of P solutions of TSPwPR. The initial population is generated in a special way. Starting at the depot, with equal probability we choose a city to which we can travel from the depot. We add the distance between the depot and the chosen city to the current tour length. If the current tour length is not greater than $c_{max}/2$, we continue, but instead of starting at the depot, we start at the chosen city. We again randomly select a city, but this time we exclude from the set of possible cities the city from which we have just arrived (the last city in a partial tour). This assumption prevents from continual visiting a given city but is relaxed if there is no possibility to choose another city. The most popular approach in EA community to handle constraints is to use penalty functions that penalize infeasible solutions by reducing their fitness values in proportion to their degrees of constraint violation. However, we don't use penalty function in our algorithm because we are sure that all individuals don't exceed cmax value. It is caused by condition that we must return in our generating tour when tour length is not greater than $c_{max}/2$.

The pseudocode of the EA described in this paper is presented below.

```
Pseudocode of EA for TSPwPR
Begin
 generate an initial population of size
 P;
 compute fitness function for each in-
 dividual;
 for i:=1 to ng do
  Begin
   select the population i from the po-
   pulation i-1 by means of tournament
   selection;
   with the group size equals to tsi-
   ze; divide population into disjont
   pairs; cross each pair if possible;
   mutate each individual if possible;
  End;
 choose the best individual from the
 final population as the result;
End;
```

If the current tour length is greater than $c_{max}/2$, we reject the last city and return to the depot the same way. In this case the tour length do not exceed c_{max} therefore the constraint imposed by the problem is preserved. One can see that such an idea of generating the initial population causes that individuals are symmetrical in respect of the middle city in the tour. However, experiments show that the EA quickly breaks these symmetries.

Let us construct an individual for the problem presented in Fig. 1 with the assumption that $c_{max} = 150$. We start at the node 1 and have to choose one node from the set {2, 4, 6, 7}. Let us assume that node 2 was selected. Since the distance between 1 and 2 equals to 28, the current tour length equals to 28 (and is not greater then $c_{max}/2$). The partial tour is 1 - 2. Starting at the node 2, we can select the node 3 or the node 7, with equal probability (we exclude node 1). Let us assume that the node 3 was selected. The current tour is now 1 - 2 - 3 with the length equal to 62. Starting from the node 3 we can only select the node 4 but this situation will cause crossing the threshold value $c_{max}/2$. We must reject the node 4 and return to the depot the same way. Our complete tour is 1 - 2 – 3 - 2 - 1 and has the length equal to 124.

The next step is to evaluate individuals in the initial population by means of the fitness function. The fitness of a given individual is equal to collected profit under the assumption that a profit is realized only during first visiting of a given vertex. For example, the fitness of the individual represented by the chromosome: 1 - 2 - 3 - 2 - 1 equals to 16.

Once we have the fitness function computed, the EA starts to improve initial population through repetitive application of selection, crossover and mutation. In our experiments we use tournament selection: we select t_{size} individuals from the current population and determine the best one from the group. The winner is copied to the next population and the whole tournament group is returned to the old population (randomizing with returns). This step is repeated P times. The parameter t_{size} should be carefully set because the higher t_{size}, the faster convergence of the EA.

We present a new heuristic crossover operator adjusted to our problem. In the first step, individuals are randomly coupled. Then, each couple is tested if crossover can take place. If two parents do not have at least one common gene (with the exception of the depot), crossover can not be done and parents remain unchanged. Crossover is implemented in the following way. First we randomly choose one common gene from the set of common genes in both

P1: 1 — 4 — **9** — 5 — 6 — 1 → O1: 1 — 4 — 9 — 4 — 1
P2: 1 — 4 — 8 — **9** — 4 — 1 → O2: 1 — 4 — 8 — 9 — 5 — 6 — 1

Fig. 2. The example of the crossover operator
Source: [own work]

parents (we exclude the depot from this set). This gene will be the crossing point. Then we exchange fragments of tours from the crossing point to the end of the chromosome in two parent individuals. If offspring individuals preserve the constraint c_{max}, they replace in the new population their parents. If one offspring individual does not preserve the constraint c_{max}, its position in the new population is occupied by better (more fitter) parent. If both children do not preserve the constraint c_{max}, they are replaced by their parents in the new population. The example of the crossover is presented in Fig. 2. This example concerns Fig. 1 with the assumption that $c_{max} = 200$.

The length of the tours represented by offspring are equal to 72 and 155, respectively. Since both off-spring individuals preserve the constraint c_{max}, they replace in the new population their parents.

The last genetic operator is a mutation. Each individual in the current population undergo mutation. It is performed in the following way. First we randomly select a position in a chromosome where a mutation will be performed. Then we try to insert a city (from the set of possible cities) at this position. If inserting a city do not violate the constraint c_{max}, we keep this new city in a tour otherwise it is rejected. For example, let us look at the individual O2 in Fig. 3. Let us assume that this individual is to be mutated and randomly selected position in the chromosome is marked with an arrow. The only city we can insert between the cities 6 and 1 is the city 7. Inserting this city will result in the tour length equal to 198. Since $c_{max} = 200$, we keep the city 7 on its position. The new mutated individual O2' replaces O2 in the population.

4. Experiments

We conducted many experiments on network which consists of 160 cities in Poland. The tested data of network are accessible on the website: http://piwonska.pl/research in two text files: cities.txt and distances.txt. Line number i in both files represents information about city number i. The number of lines in each file is equal to n. Format of line number i in cities.txt file is: i name-of-the-city pi. Format of line number i in distances.txt file is: i j_1 d_{ij1} ... j_k d_{ijk}., where j_1... j_k are number of destination cities and d_{ij1} ... d_{ijk} are values of distances to destination cities.

The data in file distances.txt for network in Fig. 1 are presented below:

1 2 28 4 12 6 11 7 19
2 1 28 3 34 7 25
3 2 34 3 35
4 1 12 3 35 5 28 9 24
5 4 28 6 15 9 38
6 1 11 5 15 7 24
7 1 19 2 25 6 24
8 4 47 9 32
9 4 24 5 38 8 32

The network which is written in cities.txt and distances.txt files was created from a real map, by including to a graph main segments of roads. Profits associated with a given city were determined according to a number of inhabitants in a given city. The more inhabitants, the higher profit associated with a given city. These rules are presented in a Tab. 1.

First experiment showed that EA generates the best tours (with the highest collected profit) in first hundred iterations and therefore 100 is enough value hundred is the optimal value for ng parameter.

Fig. 4 presents the best of ten EA runs for tested values of c_{max} (from 200 to 1000 km). On each plot we can see the profit of the best individual in a given generation. For all plots presented in this figure, the EA quickly finds the optimal (or suboptimal) solutions. Further generations do not bring an improvement. We present in Fig. 4 results for 60 generations only, because over this number profit of the best individual never grown up.

Second test examined the influence the value of population size parameter (P) on quality of resultant tours.

O2: 1 — 4 — 8 — 9 — 5 — 6 — 1
{7}

O2': 1 — 4 — 8 — 9 — 5 — 6 — 7 — 1

Fig. 3. The example of the crossover operator
Source: [own work]

Table 1. Profits associated with a given city

number of inhabitants	profit
under 10000	1
(10000, 20000]	2
(20000, 30000]	3
(30000, 40000]	4
over 40000	5

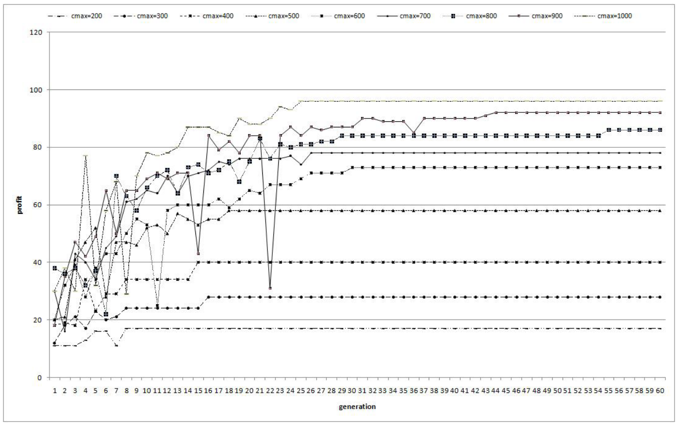

Fig. 4. The EA run
Source: [own work]

Table 2. Profits associated with a given city

P	length of tour	number of cities	collected profit
100	680	22	76
150	697	21	76
200	681	22	77
250	687	22	80
300	676	21	78
350	22	22	78
400	687	22	80

Table 3. Best results for cmax = 1400 and a given value of P

P	length of tour	number of cities	collected profit
100	1400	43	132
150	1379	42	138
200	1382	40	125
250	1395	41	142
300	1399	44	148
350	1398	45	147
400	1398	45	147

Table 4. Best results for cmax = 2100 and a given value of P

P	length of tour	number of cities	collected profit
100	2080	64	181
150	2081	56	182
200	2099	59	185
250	2100	62	195
300	2095	61	194
350	2097	61	193

Determination of optimal value for this parameter is very important because it influence on: quality of EA results (maximum value of collected profit) and time of realization of the algorithm. Too big value of P causes too high of average time of EA realization. Too small value of P takes effect of low quality of resultant tours (smaller value of collected profit).

Tests were performed for three c_{max} values: 700, 1400 and 2100 and for six values of P: 100, 150, 200, 250, 300, 350 and 400. Results of this experiment are presented in Tab. 2, Tab. 3 and Tab. 4. In each row of this table we present the values of the best collected profit and length of the best tour, obtaining as results of ten EA runs, for given values of c_{max} and P. Grey row marks the best collected profit. In Tab. 5 we see the resultant best tour (with maximum collected profit) for each considered value of c_{max} parameter.

In both experiments t_{size} parameter was equal to 3.

Results of conducted experiments show that EA can quickly improve the initial population of randomly generated solution. The tour lengths of the best individuals for all tested c_{max} values are close to given constraints. Another observation concerns the number of returns in generated tours. Only in 7% of tests resultant tours includes returns and 80% of all these cases were happened for limit distance not great then 700 km. However, to test a real quality of proposed algorithm we need to know the global optimum of tested network. To conduct such kind of experiments we must generate a graph in a special way, with known global optimum. It will be a subject of our future research.

We can also observe, on the base of Tab. 2, Tab. 3 and Tab. 4, that EA returns the best tour for value of P not greater than 300. For c_{max}=2100 the best result was obtained for P=250. Only for c_{max}=1400 the best tour was generated for population which size is greater than 250. Additionally, maximum collected profit for c_{max}=1400 and P=250 was only 12,5% worse than for P=300 and the same value of cmax. Therefore, we can conclude that optimal value of P is equal to 250 for tested network.

Table 5. Best tours for a given cmax

c_{max}	best tour
700	1, 10, 5, 2, 5, 12, 70, 69, 79, 61, 80, 62, 66, 68, 64, 63, 67, 73, 72, 71, 11, 1
1400	1, 13, 14, 9, 32, 96, 94, 98, 93, 95, 100, 81, 92, 91, 90, 89, 88, 133, 132, 131, 109, 107, 102, 106, 105, 104, 101, 114, 112, 103, 119, 68, 66, 64, 62, 80, 61, 79, 69, 70, 71, 72, 11, 1
2100	1, 13, 16, 18, 19, 20, 91, 97, 94, 98, 14, 9, 32, 96, 94, 93, 95, 99, 82, 100, 81, 92, 90, 89, 88, 133, 132, 131, 140, 109, 107, 102, 106, 102, 120, 101, 115, 103, 116, 117, 118, 67, 63, 64, 68, 66, 65, 62, 80, 61, 79, 69, 70, 71, 72, 11, 6, 12, 2, 5, 10, 1

5. Conclusions

In this paper we presented a version of TSP called Travelling Salesman Problem with Profits and Returns. Additional assumptions to this problem were proposed in the paper which make the problem more real-life. The aim of the work was designing a EA to deal with this problem. The EA proposed in the paper was tested on some voivodeships in Poland. The main parameter which has the highest influence on quality of obtaining results is the size of population and our experiments were directed to determine an optimal value of this parameter.

We can generally conclude that EA gives satisfactory results for tested network. We also determined, on the base of experiments, optimal value of two important parameters of EA: size of population and number of generation. These values are: ng=100 and P=250.

However, these results should be compared with results obtained by the other heuristic algorithms and other (bigger) networks.

Another issue which must be carefully studied is the mutation operator. Results of experiments have shown that this operator has significant role in the quality of received solutions. This operator should take into account (besides a distance) a profit associated with an inserted city. Other version of mutation can also be considered, for example inserting a fragment of tour or exchanging cities in a tour.

The another proposition of improving the algorithm will be using of penalty function. In actual version of algorithm we don't use penalty function because we are sure that all individuals don't exceed c_{max} value. It is caused by condition that we must return in our generating tour when tour length is not greater than c_{max}=2. However, this condition often causes necessity of returns in tours and consistently decrease a searching area in a graph.

It is also proper to consider some heuristic in process of generation of initial population. In actual version of the algorithm we completely randomly generate the initial population. This process can be improved by preferring choosing a city with better neighbor in a tour (a neighbor with the best profit or a neighbor with the least distance).

The results presented in this paper are results of preliminary experiments. The future work will be focused on testing the improved version of the EA on bigger and more dense networks, for example for cities from the whole Poland.

Acknowledgements
This work was supported by grants number: S/WI/2/2008 and S/WI/3/208 realized in Bialystok University of Technology

Bibliography

[1] FEILETT D., DEJAX P., GENDREAU M. : Traveling salesman problems with profits. In Transportation Science. Vol. 39, No. 2, pp. 188 205, 2005
[2] GOLDBERG D. E: Genetic algorithms and their applications. The publishing company, Warsaw, 2nd edition, 2005
[3] JOZEFOWICZ N. F. G., LAGUNA M.: Traveling salesman problems with profits. In Journal of Mathematical Modeling and Algorithms. Vol. 7, No. 2, pp. 177-195, 2008
[4] LAWLER E., LENSTRA J., SHYMOYS A. H. G.: The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. John Wiley&Sons, Chichester New York Brisbane Toronto Singapore, 1st edition, 1985
[5] MICHALEWICZ Z.: Genetic Algorithms+Data Structures=Evolution Programs. WNT, Warsaw, 2nd edition, 1996
[6] MITCHELL M.: An Introduction to Genetic Algorithms. MIT Press, Massachussets, 1st edition, 1998
[7] QIN H., LIM A., XU, D.: The selective traveling salesman problem with regular working time windows. In Studies in Computational Intelligence. Vol. 214, pp. 291-296, 2009
[8] SEVKLI Z., SEVILGEN F. E.: Variable neighbor hood search for the orienteering problem. In Lecture Notes on Computer Science. Vol. 4263, pp. 134-143, 2006
[9] TASGETIREN M. F. :A genetic algorithm with an adaptive penalty function. In Journal of Economic and Social Research. Vol. 4 (2), pp. 1-26. , 2002
[10] TONCI C., HRVOJE G., GALIC A., CAVAR I.: Vehicle Routing Problem Solving System as a CRO-GRID Application, Proceedings 4th European Congress on Intelligent Transport Systems, 24-26 May 2004, Budapest.