ITM Web of Conferences **34**, 02006 (2020) https://doi.org/10.1051/itmconf/20203402006
*Third ICAMNM 2020*

# Genetic algorithm for solving transportation problems on networks with one source and multiple sinks

*Tatiana* Paşa[1]*

[1]Moldova State University, Chişinău, Republic of Moldova

**Abstract.** In this paper we propose a genetic algorithm for solving the nonlinear transportation problem on a network with multiple sinks and concave piecewise cost functions. We prove that the complexity of one iteration of the algorithm is $O(n^2)$ and the algorithm converges to a local optimum solution. We show that the algorithm can be used to solve large-scale problems and present the implementation and several testing examples of the algorithm using Wolfram Language.

## 1 Introduction

There are often situations where a manufacturer of a type of product has sale contracts with traders from various cities and/or countries. In turn, traders have many warehouses in various cities, from which they deliver the product to other entrepreneurs. Finally, after the product is transported through several intermediary points, it is placed on store shelves and bought by the customer. This structure can be described by a transportation network with one source, several destinations and a set of intermediary points.

Despite being able to solve the problem, the algorithms presented in [1], [2], [3] and [4] have an unreasonably big execution time for transportation networks of big dimensions. There are no known polynomial algorithms that would provide the solution to these large-scale problems. This is due to the NP-difficult problem with concave cost functions, which can have several local minima, to which, in most cases, the algorithm often converges.

Even if no genetic algorithms [5] are known that are able to deliver the exact solution for all optimization problems, their use is recommended, because they do not need the gradient or Hessian information. These algorithms are resistant to blockages in an optimal local even if the structure of the restrictions that describe the range of admissible solutions is quite complex. Genetic algorithms are successful for solving large-scale nonlinear optimization problems.

A first step in using a genetic algorithm is the correct coding of the problem, i.e. the description of the chromosomes that represent the coding of an admissible solution of the problem. When we talk about the solution obtained following the execution of a genetic algorithm [5], [6] a balance must be kept between the exploration of as many of the admissible solutions as possible and the exploitation of the solution as close to the optimal solution as possible. That means that we have to carefully choose the size of the population generated at the initial step, which depends on the size of the network describing the problem. Another

*e-mail: pasa.tatiana@yahoo.com

important factor is the number of iterations after which the algorithm is stopped and the solution associated with the chromosome with the best characteristics is accepted as the optimal solution.

The elitism [7] implemented in the proposed algorithm is a technique that allow us to keep the chromosomes with the best characteristics from one population to another. Otherwise, individuals may be lost due to non selection or modification through mutation. Incorporating techniques that improve the solution in the genetic algorithm produces a hybrid that can obtain better results [8].

## 2 Problem formulation

In the following, the non-linear transportation problem on a network with a source and several destinations is solved. The network is described by the acyclic connected graph $G = (V, E)$, $V$ the set of vertices, $|V| = n$ and $E$ the set of arcs, $|E| = m$. On the set of vertices, the consumption and production function is defined $q : V \longrightarrow R$, and on the set of arcs, the piesewise non-decreasing concave cost functions $\varphi_e(x_e)$ are defined. Thus, the following problem must be solved:

$$F(x^*) = \min_{x \in X} F(x) \tag{1}$$

$$\sum_{e \in E^+(v)} x(e) - \sum_{e \in E^-(v)} x(e) = q(v) \tag{2}$$

$$x(e) > 0, \forall (e) \tag{3}$$

where the set of constraints defines the set of admissible solutions $X$ of the formulated problem. The consumption and production function has the form:

$$q(v) = 0, v \in (V \backslash V_t) \backslash \{v_0\}\,; q(v_0) = \sum_{i=1}^{k} q(v_i), v_i \in V_t \tag{4}$$

where $v_0$ source of the network, $v_i \in V_t$, $i = \overline{1,k}$ destination $i$ of the network, $V_t \subset V$ the set of all destinations and $|V_t| = k$. It is assumed that all data are real values and it is desired to obtain an optimal solution that describes the minimum cost of transport.

The nonlinear transportation problem on a network (1) - (3), (4) with piecewise concave cost functions can be solved using finite algorithms that are based on researching all spanning trees. In case the problems are described by dense and/or large graphs, for the solution of which the verification of all solutions, associated with the spanning trees, is impossible, it is recommended to use genetic algorithms. Such algorithms allow the keeping of promising candidates and the elimination of candidates with high transport costs. Thus, significantly reducing the number of admissible solutions that need to be researched in order to obtain a good result.

### 2.1 Description of the genetic algorithm AG

The size of the population generated in Step 1 and the method to code the problem are important elements that determine the amount of memory needed to store the data. In the following algorithm each chromosome is described only by the arcs that are part of the spanning tree that describes an admissible solution to the problem. Each chromosome, which describes an admissible solution, contains $n - 1$ genes given by pairs of numbers $(i, j)$, which indicate that the arc starts from vertex $i$ and enters the vertex $j$. At the same time, the size of the population

must allow the processing of a sufficient number of admissible solutions so that the right one can be found in a reasonable time.

Selection applies a local search to the population, so that chromosomes with good characteristics of the current population are selected and transferred to the new population to participate in the creation of new chromosomes.

Crossover involves the transmission of genes from two parent chromosomes and the mutation involves the modification of a randomly selected gene. The application of crossover and mutation operators in the AG algorithm described below, can lead to situations where the restrictions of the existence of the network flow are not respected, which means that an admissible solution cannot be associated to the chromosome. In such situations, methods from graph theory are used to adapt the chromosome, which brings them to a shape that allows after decoding to obtain admissible solutions.

**The genetic algorithm AG** proposed for solving the transportation problem with concave non-decreasing piecewise functions consists of the following steps:

**Step 1** *Initialization.* A random initial population of $4n$ chromosomes is generated: each chromosome is described by a list of the form $T = \left\{(i, j) \mid i, j = \overline{1, n}\right\}$, where $T$ is a spanning tree of the graph $G = (V, E)$, $|T| = n - 1$ and each $(i, j)$ is an arc that starts from the vertex $i$ and enters the vertex $j$. Duplicates are deleted and the dimension of the population is adapted (see Remark 2.1).

**Step 2** *Decoding and Evaluation of the chromosomes.* In the decoding process, to each chromosome an admissible solution is associated. For this purpose:

- a zero flow is associated with the arcs of the initial graph that are not part of the spanning tree that describes the chromosome;
- flow equal to the amount required by each destination is associated with the arcs that enter the respective destination;
- by traversing the tree in depth, the remaining arcs $(i, j)$ are associated with a flow equal to the total flow going passing through the outgoing arcs of the vertex $j$.

Evaluation involves determining the value of the objective function for each of the solutions obtained.

**Step 3** *Chromosome selection.* Chromosomes are sorted in increasing order of the value of the objective function associated with the chromosome. Chromosomes in the first half of the population are transferred to the new population.

**Step 4** *Chromosome crossover.* It takes place between the chromosomes transferred to the population $P(i)$ from the population $P(i-1)$ through selection. The same random cut is applied to both parent chromosomes, in which the set of destinations is divided in two. The set of destinations $V_t = 1, 2, ..., k$ is divided in two subsets $V_{t_1} = \{v_1, v_2, ..., v_{k_1}\}$ and $V_{t_2} = \{v_{k_1+1}, v_{k_1+2}, ..., v_k\}$, such that $V_t = V_{t_1} \bigcup V_{t_2}$ with $|V_t| = \left|V_{t_1}\right| + \left|V_{t_2}\right|$. Chromosome crossover implies:

- the first offspring obtains the arcs that form the paths from the source to the destinations from the set $V_{t_1}$ of the mother-chromosome and the arcs that form the paths from the source to the destinations from the set $V_{t_2}$ of the father-chromosome;
- the second offspring obtains the arcs that form the paths from the source to the destinations from the set $V_{t_1}$ of the father-chromosome and the arcs that form the paths from the source to the destinations from the set $V_{t_2}$ of the mother-chromosome.

To each offspring the missing vertices are added together with the paths from the source to each vertex in the following way:

1. the first offspring gets the paths from the mother-chromosome;

2. the second offspring gets the paths from the father-chromosome.

In order to be able to associate each offspring chromosome with an admissible solution, a spanning tree is generated on the obtained graph. In this way, every pair of parent chromosome produces two offspring and the size of the population remains constant.

**Step 5** *Mutation.* One gene of an offspring chromosome is mutated with the probability $\epsilon \in [0.1, 0.5]$. Through mutation, one arc $(i, j)$ is deleted and a new arc $(i^*, j)$ is chosen from the initial graph.

**Step 6** *Checking the stopping condition.* This implies stopping the algorithm after $k$ iterations. As the solution to the problem serves the solution with the minimal objective function associated with a chromosome from the last population.

In the following we will exemplify how the coding of an admissible solution takes place and also how a chromosome is decoded.

**Example (coding)** Consider the transportation network, in Figure 1, described by the graph with the set of vertices $1, 2, 3, 4, 5$, where 1 is the source and $4, 5$ are destinations, and the set of arcs $\{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$. It is required to transport to destination 4 product flow of 45 u. c. and to destination 5 product flow of 55 u. c. from the source which contains 100 u. c. such that the transportation cost is minimal. A chromosome, in this case, is a set of 4 arcs which describe a spanning tree of this graph.

**Figure 1.** Network with a sourse and two destinations

For example, a spanning tree is constructed, which contains the set of arcs $\{e_1, e_4, e_5, e_7\}$. In this case, a chromosome of the population is: $T = \{(1, 2), (2, 4), (2, 3), (3, 5)\}$, which means that only on the arcs of this tree flow is transported, and not on the arcs $e_2, e_3, e_6$, so a zero flow is associated.

**Example (decoding)** For the generated chromosome $T = \{(1, 2), (2, 4), (2, 3), (3, 5)\}$ the admissible solution $x = (x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ is constructed in the following way:

1. initially, a zero flow is associated with the arcs that are not part of the spanning tree: $x_2 = 0$ u.c., $x_3 = 0$ u.c., $x_6 = 0$ u.c.;

2. flow is associated with the arcs that enter the destinations and do not have already zero flow: $x_5 = 45$ u.c., $x_7 = 55$ u.c.;

3. the remaining arcs are associated with flow: $x_4 = 0 + 55 = 55$ u.c., $x_1 = 45 + 55 = 100$ u.c..

The obtained admissible solution is $x = (100, 0, 0, 55, 45, 0, 55)$. Knowing the admissible solution, the cost of transport on each arc and the value of the objective function are calculated, i.e. the total cost of transporting the flow from the source to all the destinations of the network.

In the following we will exemplify how the application of the crossover and mutation operator takes place on a chromosome of the population.

**Example (Crossover)** In Figure 2 a) and b) two spanning trees are presented that describe the two parent-chromosomes that are taking part in crossover:

Mother-chromosome: $\{(1,2),(2,3),(3,4),(4,7),(7,9),(1,8),(1,10),(2,6),(4,5)\}$

Father-chromosome: $\{(1,2),(2,6),(6,7),(7,8),(7,9),(2,10),(2,3),(3,4),(2,5)\}$.

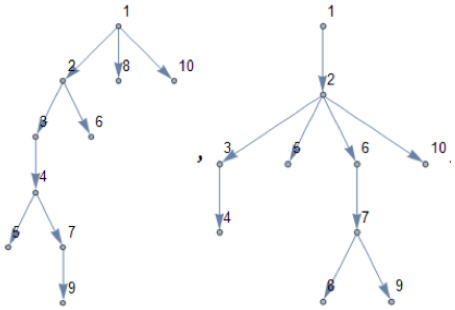


**Figure 2.** Parent chromosomes a) and b)

Crossover implies randomly splitting the set of destinations in two sets, for example $V_{t_1} = \{8\}$ and $V_{t_2} = \{9, 10\}$. The path to destination 8 from the parent chromosome from Figure 2 a) is $\{(1,8)\}$, to destination 9 $\{(1,2),(2,3),(3,4),(4,7),(7,9)\}$, and to destination 10 $\{(1,10)\}$. The path to destination 8 from the parent chromosome from Figure 2. b) is $\{(1,2),(2,6),(6,7),(7,8)\}$, to destination 9 $\{(1,2),(2,6),(6,7),(7,9)\}$, and to destination 10 $\{(1,2),(2,10)\}$.

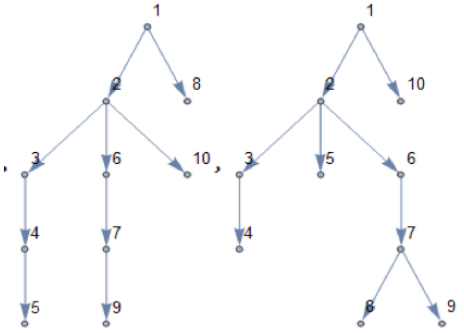


**Figure 3.** Offspring chromosomes a) and b)

To obtain the offspring from Figure 3. a), the path from source to destination 8 from Figure 2. a) is joined with the paths from source to destinations 9 and 10 from Figure 2. b). Missing vertices are added together with the paths that connect them to the source in Figure 2. a), with duplicates of arcs beeng deleted. To obtain the offspring from Figure 3. b), the path from source to destination 8 from Figure 2. b) is joined with the paths from source to destinations 9 and 10 from Figure 2. a). Missing vertices are added together with the paths that connect them to the source in Figure 2. b), also deleting duplicates of arcs. Finally, spanning trees are constructed over the obtained graphs. After these operations, the two new chromosomes can be added to the population.

Thus, the offspring chromosomes are:

Offspring chromosome 1: $\{(1,8),(1,2),(2,6),(6,7),(7,9),(2,10),(2,3),(3,4),(4,5)\}$

Offspring chromosome 2: $\{(1,2),(2,6),(6,7),(7,8),(7,9),(1,10),(2,3),(3,4),(2,5)\}$.

**Example (Mutation)** Mutation plays an important role in the AG algorithm. It allows the generation of new solutions and guarantees the avoidance of situations where the algorithm is blocked in a local solution. It is recommended that the mutation is applied with the probability $\epsilon = 0.5$, especially for big graphs. In Figure 4. a) and b) are presented the offspring chromosomes from Figure 3 after mutation. Mutation is applied only to the chromosome from Figure 4. a), the one in Figure 4. b) remains unmodified and is added to the population in its original form. The arc $(4,5)$ is deleted from the offspring from Figure 4. a) and the arc $(2,5)$ is added.

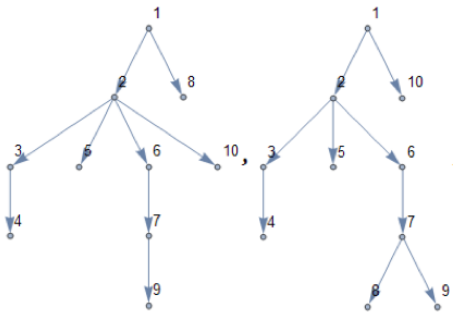


**Figure 4.** Mutation a) and b)

Thus, when mutation is applied to the chromosome:

Offspring: $\{(1,8),(1,2),(2,6),(6,7),(7,9),(2,10),(2,3),(3,4),(4,5)\}$

the following chromosome is obtained:

Mutated offspring: $\{(1,8),(1,2),(2,6),(6,7),(7,9),(2,10),(2,3),(3,4),(2,5)\}$.

The number of iterations required to stop the algorithm depends on the size of the studied transport network. Multiple iterations are required to obtain a chromosome with good characteristics.

## 2.2 Theoretical results

**Remark 2.1** *Because of the random generation of spanning trees in the initialization step, duplicates can appear. These are deleted from the population, so that the algorithm is not be forced to tend to a local minimum, instead of the global one. After eliminating the duplicates, the size of the population is reduced to the biggest number of the form 4p that is smaller than the number of available spanning trees. If necessary, other randomly chosen trees may have to be deleted in this step. This new size of the population is used in the steps 2-6 to determine the solution of the problem.*

**Remark 2.2** *Algorithm AG can be applied when the network satisfies the following conditions:*

1. *The graph of the transportation network is convex and acyclic;*

2. *Each vertex of the graph, except the destinations, has at least one outgoing arc;*

3. *The destination vertices do not have outgoing arcs and are more than one;*

4. *There is at least one path from the source to each destination.*

The following theorems are formulated and proved:

**Theorem 2.1** *The algorithm AG requires memory of the order* $O(n^2)$.

*Proof.* The graph $G = (V, E)$ is described by an adjacency list of size $m < n^2$. Every chromosome consists of a list $T$ of size $n - 1$. Thus, a population of $4n$ chromosomes is of size $4n(n-1)$. The population is renewed at each step in-place, without any additional memory. Therefore, the algorithm AG requires $O(n^2)$ memory. □

**Theorem 2.2** *The complexity of a single iteration of the algorithm AG is* $O(n^2)$.

*Proof.* To fill the adjacency list of the graph $G = (V, E)$, $O(m)$ operations are needed. The generation of a single chromosome has the complexity of $O(n)$. The generation of whole population requires $O(n^2)$ operations. The evaluation of a single chromosome requires $O(n)$ operations. Because a population has $4n$ chromosomes, evaluating all chromosomes of a population requires $O(n^2)$ operations. Crossover and mutation both have the complexity $O(n)$ for a single chromosome and $O(n^2)$ for a whole population. Therefore, a single iteration of the algorithm AG has the complexity $O(n^2)$. □

**Remark 2.3** *From Theorem 2.2. it results that the execution time of the AG algorithm is* $O(Un^2)$*, where U is the number of iterations necessary to obtain a solution associated with a chromosome with good characteristics.*

**Theorem 2.3** *Algorithm AG converges to a local optimum.*

*Proof.* From the $P(i-1)$ population, chromosomes, for which the value of the objective function is the smallest, are transferred to the $P(i)$ population. The chromosomes constructed at each iteration represent spanning trees and each chromosome has an associated admissible solution. After a finite number of iterations, a chromosome will be found which has the solution associated, which describes a local optimum. Therefore, the AG algorithm converges to a local optimum. □

## 3 Implementation and testing of the AG algorithm

Let the transportation network described by the graph in Figure 5 be given.

The set of vertices is associated with a production and consumption function of the form:

$$q(v) = \begin{cases} 100, & v = 1 \\ 0, & v = 2, ..., 9 \\ -14, & v = 8 \\ -65 & v = 9 \\ -21 & v = 10 \end{cases}.$$

Each arc $\{e_1, e_2, ..., e_{23}\}$ is associated with a cost function of the form $\varphi_1(x) = \begin{cases} x, & x \leq 1 \\ 1 & x > 1 \end{cases}$ or $\varphi_2(x) = \begin{cases} 2x, & x \leq 2 \\ 4 & x > 2 \end{cases}$. It is required to determine a flow that minimizes the cost of transporting 100 c.u. of flow from the source $\{1\}$ to the destinations $\{8, 9, 10\}$ with the restriction that the flow satisfies the consumption and production functions. The optimum solution of this problem will be of the form $x = \{x_1, x_2, ..., x_{23}\}$.

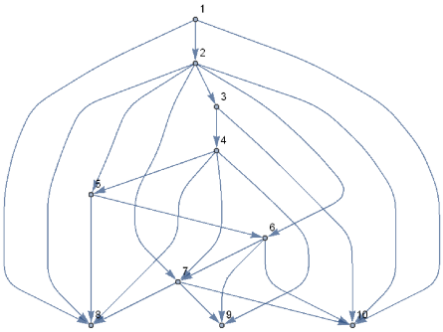


**Figure 5.** Transportation network

***Initialization*** involves randomly generating a population of chromosomes. Each of them describes a spanning tree and has associated an admissible solution. After the value of the objective function is calculated, the chromosomes are sorted in increasing order and the first half of the chromosomes will be transferred to the next population. They will participate in crossover and each pair of parent – chromosomes will have 2 child – chromosomes.

***Iteration 1.***

For the population obtained in the previous step $F_T(x) = 3224$ c.u.. The solution associated with the chromosome

$$\{1 \to 2, 1 \to 10, 2 \to 3, 2 \to 6, 2 \to 7, 2 \to 8, 3 \to 4, 4 \to 5, 6 \to 9\}$$

with the best characteristics from the current population is

$$x = \{79, 0, 21, 0, 0, 65, 0, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 65, 0, 0, 0, 0\}$$

and the objective function for this solution is $F(x) = 42$ c.u..

***Iteration 2.***

For the population obtained in the previous step $F_T(x) = 2913$ c.u.. The solution associated with the chromosome

$$\{1 \to 2, 2 \to 3, 3 \to 4, 2 \to 5, 5 \to 6, 4 \to 7, 1 \to 8, 6 \to 9, 2 \to 10\}$$

with the best characteristics from the current population is

$$x = \{86, 14, 0, 0, 65, 0, 0, 0, 21, 0, 0, 0, 0, 0, 0, 65, 0, 0, 65, 0, 0, 0, 0\}$$

and the objective function for this solution is $F(x) = 42$ c.u..

***Iteration 3.***

For the population obtained in the previous step $F_T(x) = 2586$ c.u.. The solution associated with the chromosome

$$\{1 \to 2, 1 \to 10, 2 \to 3, 2 \to 6, 2 \to 7, 2 \to 8, 3 \to 4, 4 \to 5, 6 \to 9\}$$

with the best characteristics from the current population is

$$x = \{79, 0, 21, 0, 0, 65, 0, 14, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 65, 0, 0, 0, 0\}$$

and the objective function for this solution is $F(x) = 41$ c.u..

***Iteration 4.***

For the population obtained in the previous step $F_T(x) = 2417$ c.u.. The solution associated with the chromosome

$$\{1 \to 2, 1 \to 8, 1 \to 10, 2 \to 3, 2 \to 5, 2 \to 6, 3 \to 4, 6 \to 9, 4 \to 7\}$$

with the best characteristics from the current population is

$$x = \{65, 14, 21, 0, 0, 65, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 65, 0, 0, 0, 0\}$$

and the objective function for this solution is $F(x) = 39$ c.u..

If we make only 4 iterations, then the optimum solution to the problem will be considered the last one.

**Remark 3.1** *Although at iteration 1 and 2 the chromosome with the best characteristics had the same value of the objecteve function of the associated solution, the algorithm exits this block and finds a better solution due to the operations of selection, crossover and mutation.*

The AG algorithm is implemented in the Wolfram language and tested on a set of examples transportation networks of various dimensions with different number of vertices, destinations and arcs.

In Table 1, the results of solving a set of transportation problems on networks with $n$ vertices, $m$ arcs and $d$ destinations are presented, for which the concave piecewise cost functions are known, which depend on the transported flow and the consumption and production function.

Note that the cost functions are piecewise functions and are the same as $\varphi_1(x)$ and $\varphi_2(x)$ from the example above.

**Table 1.** The change in the values $F(x)$ and $F_T(x)$ for the AG algorithm (u.c.)

| **Iteration** | **n/m/d 50/619/10** | **n/m/d 100/2422/20** | **n/m/d 150/4887/50** | **n/m/d 250/14881/70** | **n/m/d 500/57146/100** |
|---|---|---|---|---|---|
| **I** | 130/36497 | 317/156257 | 496/365616 | 614/774821 | 841/2066006 |
| **II** | 106/32340 | 246/143659 | 496/349826 | 610/733735 | 786/1951033 |
| **III** | 92/29141 | 231/133008 | 469/335819 | 571/698441 | 780/1853123 |
| **IV** | 86/26969 | 221/123762 | 425/322657 | 556/667725 | 744/1773558 |
| **V** | 72/23877 | 212/116216 | 425/311097 | 528/641933 | 710/1705995 |
| **VI** | 72/21886 | 212/109691 | 409/299570 | 528/618762 | 701/1646399 |
| **VII** | 72/20358 | 205/103967 | 401/288482 | 509/598623 | 679/1597838 |
| **VIII** | 72/18900 | 191/98440 | 382/277867 | 499/579515 | 657/1555438 |
| **IX** | 68/16663 | 180/90601 | 382/268272 | 497/562084 | 655/1516751 |
| **X** | 64/15759 | 175/87347 | 373/259505 | 454/545225 | 653/1482755 |

By analyzing the table, it can be observed, that $F_T(x)$ always decreases from one iteration to another, which means that the population after each iteration contains more chromosomes with better characteristics than the ones after the previous iteration. Although the $F(x)$ value is repeated in several consecutive iterations, the algorithm is able to exit from such a blockage through the operators of mutation, selection and crossover.

The execution time of the algorithm depends on the number of vertices of the graph, which is proved by the results presented in Table 2.

The tests were made on an Intel i5-2500 machine with 4 Cores and 8 GB of DDR3 memory in Wolfram Mathematica 12.

**Table 2.** The change to the runtime of AG (seconds)

| | **m / d** | **Time** | | **m / d** | **Time** |
|---|---|---|---|---|---|
| **n=10** | 20 / 3 | 0.2812 | **n=100** | 2264 / 15 | 22.3906 |
| | 22 / 4 | 0.3125 | | 2527 / 20 | 22.6094 |
| | 27 / 5 | 0.3593 | | 2569 / 30 | 22.6719 |
| **n=20** | 101 / 3 | 1.0156 | **n=150** | 4777 / 30 | 51.3750 |
| | 119 / 5 | 1.0312 | | 5178 / 40 | 52.1250 |
| | 119 / 7 | 1.2656 | | 5297 / 50 | 52.7813 |
| **n=30** | 203 / 4 | 2.1256 | **n=200** | 9271 / 50 | 95.8125 |
| | 217 / 7 | 2.1406 | | 9291 / 60 | 97.8906 |
| | 231 / 9 | 2.1562 | | 9329 / 70 | 96.9375 |
| **n=50** | 590 / 5 | 5.4843 | **n=300** | 20322 / 70 | 246.5940 |
| | 621 / 9 | 5.6093 | | 20663/100 | 249.2660 |
| | 657 / 12 | 5.6875 | | 20640/120 | 250.6720 |
| **n=60** | 860 / 9 | 7.9062 | **n=400** | 37944 / 80 | 512.9130 |
| | 917 / 12 | 7.8125 | | 37652/100 | 489.2340 |
| | 972 / 15 | 7.8906 | | 38073/120 | 507.7500 |
| **n=70** | 1191 / 9 | 11.0469 | **n=500** | 59771/100 | 918.5310 |
| | 1206 / 15 | 11.2344 | | 60756/120 | 917.5630 |
| | 1312 / 20 | 11.2500 | | 60012/150 | 918.4380 |

The algorithm was executed for 10 iterations for each test on graphs with $n$ vertices, $m$ arcs and $d$ destinations. Theorem 2 is proved experimentally, because a direct relation can be seen between execution time and the number of vertices. The number of arcs and destinations does not influence the execution time.

## 4 Conclusions

The paper contains a study of the nonlinear transportation problem on a network with one source and several destinations, for which a genetic algorithm AG is proposed, that can solve the problem in reasonable time even in the case of large-scale problems.

Based on the obtained results the following conclusions can be drawn:

1. The description of a correct coding of the problem allows the decoding to always obtain an admissible solution of the problem.

2. The operators of crossover and mutation are described so that regardless of how many times they are applied, the obtained chromosome can be decoded to an admissible solution.

3. It is proven experimentally that the algorithm can exit blockages in local solutions, so that after several iterations a new chromosome with better characteristics is found, for which the associated solution has a lower transportation cost.

4. The solution obtained by the AG algorithm depends on the population generated in Step 1.

5. Experimentally, the direct relation between the complexity of an iteration and the number of vertices of the graph is proven.

## References

[1] D. Lozovanu, T. Pasha, *An algorithm for solving the transport problem on network with concave cost functions of flow on arcs*, Computer Science Journal of Moldova **10** 3 (30), 341-347 (2002).

[2] T. Paşa, V. Ungureanu, *Solving the transportation problem with piecewise - linear concave cost function*, Review of AFA, The Scientific Informative Review **XV** 2 (34), 49-56 (2017).

[3] T. Paşa, V. Ungureanu, *Applying sequential and parallel programming to solve a nonlinear transport problem*, ICMCS, Chişinău, Republic of Moldova, October 19 - 21, 247-251 (2017).

[4] T. Paşa, V. Ungureanu, *Non-Linear Concave Transportation Problem Solving and Implementation using Wolfram Language*, Proceedings of ITSN, 30-39 (2017).

[5] J. H. Holland, *Adaptation in Natural and Artificial Systems* (University of Michigan, 1975) 232 pp.

[6] A. A. Hopgood, *Inteligent Systems for Engineers and Scientists, 3rd Edition* (CRC Press, Taylor and Francis Group, 2012) 451 pp.

[7] D. Jong. Kenneth, *An Analysis of the Behaivior of a Class of Genetic Adaptive Systems* (Doctoral thesis, Departament of Computer and Communication Sciences, University of Michigan, Ann Arbor, 1975).

[8] S. Sheng, Z. Dechen, X. Xiaofei, Genetic Algorithm for the Transportation Problem with Discontinuous Piecewise Linear Cost Function, IJCSNS **6** (7A), 182-190 (2006).

[9] I. Yu-Hua Gu, and E. Styvaktakis, Electric Power Systems Research **66**, 83-96 (2003).