ROMAI J., v.15, no.2(2019), 91–99

SOLVING NON-LINEAR MULTI-INDEX TRANSPORTATION PROBLEMS

Tatiana Paşa

Moldova State University, Chişinău, Republic of Moldova

pasa.tatiana@yahoo.com

Abstract In this paper we study the non-linear multi-index transportation problem with concave piecewise cost functions [2], [5], [7]. The aim of this paper is to propose a genetic algorithm to solve the problem. The proposed algorithm solves problems of large sizes in reasonable time, which was proven by the various tests shown in this paper. The algorithm was implemented Wolfram Mathematica and tested on problems on various dimensions.

Keywords: non-linear programming, concave function, transportation problem, index.
2010 MSC: 68W25, 90B06, 90C05, 90C26, 90C30, 90C35, 90C59.

1. INTRODUCTION

Product supply management is becoming more and more important in the economy for the efficient operation of companies. The transportation costs can be described by the cost of renting a vehicle, landing fee at the airport, parking taxes, fuel costs, driver's salary, etc. In order to bear minimum costs, it is necessary to implement an economical and fast transport system. Transportation models play an important role in supply logistics to reduce costs and improve services.

Problems involving the transport of products may vary depending on the types of products (construction materials, food, petroleum products) transported, the number and type of transport [1] (train, ship, truck, plane), number of sources and destinations, but also the transport route (air, water, roads, pipes, cables).

The 4-index transportation problem (4ITP) is a mathematical model that describes very well the activity of companies, which attracts attention of researchers, because the problem assumes that the capacities of the sources, the demands of the destinations to be supplied, the types and quantities of products and the types and capacities of the transports are known.

The problem is a part of the 4ITP group which was studied by R. Zitouni, who proposes an original algorithm [8] and compares it with other methods [7]. A modification of the classical algorithm was described by A. Djamel et. al. [2]. T-H. Pham and Ph. Dott propose an exact method of solving the problem [5], [6]. In [3] the nonlinear 4-index transportation problem (N4ITP) and its proprieties was formulated and an algorithm was proposed that simplifies the problem to a linear problem for which the simplex algorithm can be applied. Applying genetic algorithms to solve non-linear multi-index problems was discussed in [4].

This paper describes a genetic algorithm for solving the transport problem with concave cost functions with 4 index described by sources, destinations of several types of products and some types of transports circulating through the network. The algorithm was implemented in the Mathematica System and tested on several transport networks of different sizes, which means different numbers of sources, destinations, types of products and types of transports.

2. PROBLEM FORMULATION

In the following we are given:

- n sources A_1, A_2, ... , A_n which possess a quantity of the respective supply of products α_1, α_2, ... , α_n;
- m destinations B_1, B_2, ... , B_m with the respective demand of products β_1, β_2, ... , β_m;
- p different types of products P_1, P_2, ... , P_p of respective quantities γ_1, γ_2, ..., γ_p;
- q different types of transport T_1, T_2, ... , T_q with the respective transportation capacities of δ_1, δ_2, ... , δ_q;

N4ITP consists in determining a flow x^* that minimizes the function:

$$F(x) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q}\varphi_{ijkl}(x_{ijkl})$$

where $\varphi_{ijkl}(x_{ijkl})$ are piecewise non-decreasing concave cost functions.

We have to solve the non-linear global optimization problem:

$$F(x) \longrightarrow min \qquad (1)$$

$$\begin{cases} \sum_{j=1}^{m}\sum_{k=1}^{p}\sum_{l=1}^{q} x_{ijkl} = \alpha_i & i = 1, ..., n \\ \sum_{i=1}^{n}\sum_{k=1}^{p}\sum_{l=1}^{q} x_{ijkl} = \beta_j & j = 1, ..., m \\ \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{l=1}^{q} x_{ijkl} = \gamma_k & k = 1, ..., p \\ \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} x_{ijkl} = \delta_l & l = 1, ..., q \\ x_{ijkl} \geq 0 & \forall(i, j, k, l) \end{cases} \qquad (2)$$

To respect the constraints on positiv values, the following variables $\alpha_1, \alpha_2, ..., \alpha_n$, $\beta_1, \beta_2, ..., \beta_m$, $\gamma_1, \gamma_2, ..., \gamma_p$ and $\delta_1, \delta_2, ..., \delta_q$ must be positive.

Our problem have $n+m+p+q$ constraints. The general form of the solution for the N4ITP (1)-(2) is $x^* = (x^*_{1111}, x^*_{1112}, ..., x^*_{nmpq})$.

3. THEORETICAL RESULTS

In the following, we aim to solve the N4ITP problem, applying a genetic algorithm, which involves encoding the problem so that each chromosome in the population has a single admissible solution associated to it. Each newly created population keeps the chromosomes with the best characteristics from the previous population, so that the low-cost solutions are not lost.

Since each chromosome must contain information about each index that describes the problem, it is divided into 3 sections:

- *the first:* has the randomly ordered list of arcs;
- *the second:* contains a permutation of the numbers $i = 1, ..., n$ associated with the types of products to be transported;
- *the third:* contains a permutation of the numbers $j = 1, ..., m$ associated with the types of transport that can be used to supply the destinations.

The description of the genetic algorithm G: proposed for N4ITP consists of the following stepts:

Step 1 *Initialization.* The initial population of chromosomes is randomly generated, each chromosome is described by a list:

$$P = \{\{p(i,j)|i = 1, ..., n, j = 1, ..., m\} \,|\, \{p(1), p(2), ..., p(p)\} \,|\, \{p(1), p(2), ..., p(q)\}\}$$

Step 2 *Decoding and evaluating the chromosomes.* Decoding involves associating each chromosome with an admissible solution by pushing the flow of product transported from sources to destinations. The evaluation involves determining the cost for each arc and determining the value of the objective function for each of the solutions obtained.

Step 3 *Selecting the parent chromosomes* that participates in the creation of a new population. The chromosomes are sorted in order of increasing value of the objective function in the solution associated with the chromosome. The first half of the chromosomes are transferred to the new population.

Step 4 *Chromosome crossover* is performed between the pairs of chromosomes transferred into population $P(i)$ from population $P(i-1)$ by selection. The same cut is applied to both parent chromosomes on the elements of section one. The first descendant obtains the order of the arcs up to the cut and the order of distribution of the products of the mother-chromosome and the other arcs maintain the order of the arcs and the order of transport of the father-chromosome. The second descendant obtains the order of the arcs up to the cut and the order of distribution of the products of the father-chromosome and the other arcs maintain the order of the arcs and the order of transport usage of the mother-chromosome.

Step 5 *Mutation* of a chromosome gene occurs at a rate $\epsilon \in [0.1, 0.5]$ and involves:

- it is randomly chosen on which sections of the chromosome, *the second* - products or *the third* transports, the mutation is applied;
- from the chosen section two elements are chosen at random and their values are swapped.

Step 6 *Verifying the stopping condition* involves stopping the algorithm after generating k populations. The solution of the problem is the one corresponding to the chromosome in the last population for which the value of the objective function is minimal. In case the stopping condition is not satisfied, proceed to step 2.

The algorithm executes a fixed number of iterations, therefore it doesn't guarantee that the obtained solution is the global minimum.

Note 3.1. *Algorithm G is applied to networks that meet the following conditions:*

- *The graph that describes the transport problem is connected and acyclic;*
- *There are at least two sources, two destinations, two types of products and two types of transport;*
- *The sources do not have incoming arcs and the destinations do not have outgoing arcs.*

Let us consider a network with 2 sources, 2 destinations, 3 types of products and 3 types of transport have 4 arcs. Let the folowing two parent-chromosomes that participate in the crossover:

Mother {{1,4,2,3}, {1,2,3} , {1,3,2}}

Father {4,1,3,2}, {3,2,1}, {3,1,2}}.

The cut is applied to the middle of the set of arcs in section one. Then the descendants are:

Descendant1 {{1,4,3,2},{1,2,3},{3,1,2}}

Descendant2 {{4,1,2,3}, {3,2,1}, {1,2,3}}.

The mutation involves the modification of a random gene of the descendant. Which involves the random selection of the set describing the order of the products or the transports. In the example presented below, the transports were chosen for which two elements were swapped:

initial *Descendant* is: {{1,4,3,2}, {1,2,3}, {3,1,2}}

after mutatition *Descendant* is: {{1,4,3,2}, {1,2,3}, {2,1,3}}.

The following theorems can be formulated:

Theorem 3.1. *Algorithm G requires memory of the order O(nm(n+m+p+q)).*

Proof. The input data described by the constraint table is of size $nmpq$. In order to keep the solution associated with the chromosome with the best characteristics of the population, $nmpq$ memory is required. A single chromosome requires $nm + p + q$ and for the entire population $nm(nm + p + q)$ memory. Thus, algorithm G requires memory of the order $O(nm(nm + p + q))$. ∎

Theorem 3.2. *The complexity of a single iteration of the G algorithm is of the order $O(n^2m^2pq)$.*

Proof. The complexity of initializing the input data described by the restriction table is $O(nmpq)$. Generation of a single chromosome requires $O(nm+p+q)$ operations and the generation of the entire population requires $O(nm(nm+p+q))$ operations. Evaluating the solution associated with a chromosome requires $O(nmpq)$ operations. Because the population contains nm chromosomes, the complexity of evaluating all chromosomes is $O(n^2m^2pq)$. The crossover has a complexity of $O(nm+p+q)$ for one chromosome and $O(nm(nm+p+q))$ for the entire population. Thus, a single iteration of algorithm G has the complexity of the order $O(n^2m^2pq)$. ∎

Note 3.2. *From Theorem 3.2 it follows that the execution time of algo-rithm G1 is $O(Un^2m^2pq)$, where U - the number of iterations required to obtain a solution associated with the chromosome with good characteristics.*

Theorem 3.3. *The G algorithm converges to a local optimum.*

Proof. From the population $P(i-1)$ chromosomes are transferred to $P(i)$ for which the value of the objective function in the associated solution is minimal. The chromosomes of the populations constructed at each iteration are associated with an admissible solution, so we can say that after executing a finite number of iterations, the chromosome to which the solution describing the local optimum is associated will be detected. So, the G algorithm converges to a local optimum. ∎

4. PRACTICAL RESULTS

Let us be given a transportation problem described by a bipartite graph with the sources A_1 and A_2 containing the products of type P_1, P_2 and P_3 which is transported using the transport of type T_1, T_2 and T_3 to destinations B_1 and B_2.

Implementation of the algorithm

It is required to determine such a flow that the transport of 100 u. c. of product from sources to destinations should be of minimal cost with the restriction that the flow starting from sources fully satisfies the needs of the destinations. The optimal solution of the problem is of the form $x = \{x_{111}, x_{1112}, , x_{nmpq}\}$.

Initialization. Initialization involves the random generation of a population of $4nm$ chromosomes. Each cromosome is described by a random list of arcs describing the network, a permutation of product types and a permutation of transport types. Each admissible solution is assigned to each chromosome after decoding. After calculating the value of the objective function, the chromosomes are sorted in order of increasing value of the objective function and the first half of chromosomes are transferred to the new population, which then participates in the crossover so that each two parent-chromosomes generates two descendants. In the following we can see how the value of the total objective function, the function associated with the chromosome with the best characteristics of the current population and the solution associated with the chromosome for two iterations changes from one iteration to another.

Iteration 1. The solution associated to the chromosome $\{\{1,3,4,2\}, \{2,1,3\},\{2,3,1\}\}$ with the best characteristics of the current population is $x = (7,0,1,0,7,16,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,0,0,0,0,13,0,0,0, 0,0,49,0,0)$ and the objective function calculated in this solution $F(x) = 11$ u. c. Total objective function is $F_T(x) = 223$ u. c.

Iteration 2. The solution associated to the chromosome $\{\{1,3,4,2\}, \{2,1,3\},\{3,1,2\}\}$ with the best characteristics of the current population is $x = (7,1,0,0,6,17,0,0,0,0,0,0,0,0,0,0,0,0,7,0,0,0,0,0,0,0,0,13,0,0,0, 0,0,49,0,0)$ and the objective function calculated in this solution $F(x) = 10$ u. c. Total objective function is $F_T(x) = 204$ u. c.

If only 2 iterations are performed and therefore 2 populations are generated, we can say that from source 1 to destination 1 is transported a flow of 7 u. c. of product type 1 with transport type 1, from source 1 to destination 1 is transported a flow of 1 u. c. of product type 1 with transport type 2, from source 1 to destination 1 is transported a flow of 6 u. c. of product type 2 with transport type 2, from source 1 to destination 1 is transported a flow of 17 u. c. of product type 2 with transport type 3, from source 2 to destination 1 is transported a flow of 7 u. c. of product type 1 with transport type 1, from source 2 to destination 2 is transported a flow of 13 u. c. of product type 1 with transport type 1, from source 2 to destination 2 is transported a flow of 49 u. c. of product type 3 with transport type 1, with the total cost of 10 u. c.

Usage of the Wolfram Language

The G algorithm was implemented in Wolfram language and tested on a number of transportation problems of different sizes regarding the number of sources, the number of destinations, the number of products transported and the number of transports that can be used to supply destinations.

The results of solving a series of problems of different dimensions can be seen in Table 1 and 2. Based on the function of production and consumption, the total of the products available in sources is equal to the total of the products needed in destinations and is equal to 100 u. c. The size of the transportation problem is described by n - sources, m - destinations, p - types of products transported, q - types of transport used.

Table 1 Execution Time for G

n/m/p/q (seconds)	*3/3/3/3*	*3/3/4/4*	*4/4/4/4*	*4/4/5/5/*	*5/5/5/5*
	0.4843	0.8437	2.5781	3.8125	9.3593
	0.4531	0.8281	2.5312	3.9062	9.4218
	0.4843	0.7968	2.5468	3.8906	9.5937
	0.4843	0.8125	2.4843	3.9218	9.4834
	0.4843	0.8750	2.5781	3.8281	9.4218
Unknowns	*81*	*144*	*256*	*400*	*625*
n/m/p/q (seconds)	*6/6/7/7*	*7/7/7/7*	*8/8/8/8*	*9/9/9/9*	*10/10/10/10*
	38.8438	72.8125	159.375	334.359	597.641
	39.4063	74.4219	157.547	331.313	598.750
	39.4688	72.2031	159.969	316.266	597.703
	39.7813	71.3125	159.797	316.016	598.531
	39.5156	71.2500	162.203	317.766	604.328
Unknowns	*900*	*2401*	*4096*	*6561*	*10000*

Table 1 gives the execution time for the G algorithm in the case of problems of different dimensions regarding the number of sources, destinations, types of products and types of transports, for solving are used 81 unknowns for the problem with 3 sources, 3 destinations, 3 products and 3 types of transport, and up to 10000 unknown for the problem with 10 sources, 10 destinations, 10 types of products and 10 types of transports. As we can see, the algorithm allows to obtain local solutions in a reasonable time in case of large transport problems, the time being given after generating 10 populations.

Table 2 shows the results of solving a series of transport problems of different dimensions where for each of the 10 generated populations it is possible to observe how the value of the objective function calculated for the solution associated with the chromosome with the best characteristics of the respective population is modified.

Table 2 The reduction of $F(x)$ and $F_T(x)$ for G

Iter	*4/4/4/4*	*5/5/5/5*	*6/6/6/6*	*7/7/7/7*	*10/10/10/10*
I	19 / 1564	19 / 3253	27 / 5718	33 / 9772	52 / 27616
II	18 / 1507	18 / 2978	27 / 5433	30 / 9477	47 / 26788
III	16 / 1473	18 / 2852	27 / 5255	30 / 9322	47 / 26191
IV	16 / 1374	18 / 2741	24 / 5034	30 / 9099	47 / 25761
V	16 / 1365	18 / 2704	24 / 4982	30 / 8756	47 / 25455
VI	16 / 1291	17 / 2552	24 / 4861	30 / 8574	47 / 25144
VII	16 / 1255	16 / 2390	24 / 4612	30 / 8374	47 / 24877
VIII	16 / 1215	16 / 2364	21 / 4510	30 / 8304	44 / 24577
IX	14 / 1160	16 / 2229	21 / 4340	24 / 8127	44 / 24346
X	14 / 1118	14 / 2001	21 / 4209	24 / 7848	41 / 23941

From the Table 2 we can observe that total objective function $F_T(x)$ always decreases from one iteration to another, which means that each new population contains more chromosomes with better characteristics. Although we can see that in some cases at several consecutive iterations the same $F(x)$ value is repeated for the same solution, we observe that the algorithm exits such a blocks, resulting in a new chromosome with better characteristics after a few iterations.

5. CONCLUSIONS

The transportation problem is a complex one that requires in-depth study, especially for non-linear cost functions. The solution of the non-linear problem obtained from the algorithm described above depends largely on the initial population. Because the initial population is taken at random, it is not guaranteed to obtain global optimal solution, often resulting in a local minimum.

References

[1] O. Diaz-Para, A. Ruiz-Vanoye, B. B. Loranca, A. Fluentes-Penna, R. A. Barrera-Camara, *A survey of transportation problem*, HPC J. of AM, ID848129, http://dx.doi.org/10.1155/2014/848129, (visited July 2018), 17 pages.

[2] A. Djamel, N. Amel, L. T. Hoai, Z. Ahmed, *A modified clasical algorithm ALPT4C for solving a capacitated four-index transportation problem*, ACTA Mathematica Vietnamica, **37**, 3, 2012, 379-390.

[3] T. Paşa, *Multi-index transport problem with non-linear cost functions*, ROMAI Journal, https://rj.romai.ro, **14,** 2, 2018, 129-137.

[4] T. Paşa, V. Ungureanu, *Solving the non-linear 4-index transportation problem*, The Fifth Conference of Mathematical Society of the Republic of Moldova (p. 346). Chişinău: Vladimir Andrunachievici Institute of Mathematics and Computer Science, 2019.

[5] T.-H. Pham, P. Dott, *An exact Method for Solving Four Index transportation Problem and Industrial Application*, American Journal of Operational Research, **3**, 2, 2013, 28-44.

[6] T.-H. Pham, P. Dott, *Four indexes transportation problem with interval cost parameter for goods allocation planning*, LINDI 2012, 4th IEEE International Symposium on Logistics and Industrial Informatics, September 5-7. Smolenice, Slovakia, 2012.

[7] R. Zitouni, M. Achache, *A numerical comparison between two exact simplicial methods for solving a capacited 4-index transportation problem*, Journal of Numerical analysis and aproximation theory, **46**, 2, 2017, 181-192.

[8] R. Zitouni, A. Keraghel, D. Benterki, *Elaboration and Implementation of an Algorithm Solving a Capacitated Four-Index Transportation Problem*, Applied Mathematical Sciences, **53**, 1, 2007, 2643-2657.