

APPLICATION OF INTELLIGENT INFORMATION TECHNOLOGIES IN THE OPTIMIZATION OF TRANSPORTATION

National University "Yuri Kondratyuk Poltava Politechnics"

Introduction

The rapid development of the digital economy requires the appropriate development of information systems (IS) and information technologies (IT), changes the direction of the state strategy to overcome the current challenges of the need to build relationships with the global giants of the IT industry (Google, Apple, Facebook and Amazon), total capitalization which exceeded the GDP of many developed countries of the European Union. For Ukraine, the question arises about the choice of priorities, which requires an extremely rapid transformation of the traditional economy into its traditional paradigm to the modern information intellectual digital economy, which involves a synergistic effect from building the foundation on the concepts of "digital economy", "knowledge economy", "information society". In this context, the process of "consumption" of IT products becomes more significant. That is - not only to produce powerful IT solutions for other countries, but also to implement modern IS and IT in the national manufacturing sectors. In addition, the analysis of deep trends in energy and raw material geopolitics shows a sharp increase in energy, natural resources and food needs, which will exacerbate tensions due to access to these resources. Producers and at the same time consumers of these resources are complex territorially-distributed systems (CTDS) or - economic entities for which the tasks of management of large pools of resources (labor, financial, material, etc.) are the most relevant and significant.

In the unpredictable directions of development of the world production system, the use of intelligent information technologies is a necessary means for making strategic decisions in the following areas: development of strategies of state regulation of

market segments, integration processes of the economy of Ukraine and the EU, increase of investment attractiveness of national economic entities, including AH, vertically-integrated national companies (VINC's), cluster development of territorial entities.

Currently, logistics is one of the fastest growing areas of productive activity. This process is associated not only with the growing demand for logistics services, but also with the strengthening of mutual integration of business opportunities while logistics and infrastructure development of the Territories. The most significant development of methods and algorithms of interaction of the logistics processes is due to the rapid development of information technology, which resulted in a widespread network of organizational forms of business, on-line flow of documents, the transition to electronic payment systems, virtualization logistic processes, etc. On the basis of the information infrastructure of innovation Logistics takes you to a new level of intelligent control processes, the formation of new logistics concepts «PartyLogistics».

1. Methods for solving NP - complex problems

One of the modern bionic principles of solving a wide class of applications, which are intractable tasks by classical methods, especially in the area of NP-complete optimization problems, is the use of GA - adaptive search methods that implement evolutionary computation, based on the genetic processes of biological organisms. The efficiency and quality of transport complexes (TC) largely depends on the rational coordination of the various modes of transport, optimal redistribution between traffic volumes and the formation of the necessary management decisions. Solution these and other scientific and applied tasks requires the formation of appropriate organizational and functional structures at the regional and national levels, which should assume the functions associated with the various modes of transport as horizontally (different modes of transport), and vertical (industrial, regional and state levels).

2. Locations for the application of transport tasks

Effective organization of work in different types of transport requires the solution of a number of important scientific and applied problems, the main ones are:

- formation of state and regional TC as a single functional units in the structure of public administration transport system;
- development of mathematical models of effective functioning of the individual elements of the TC in the modern market conditions;
- development of methods and algorithms of optimization of TC;
- formation of informational and functional organizational structures and systems, organizational management, production processes TC;
- development of methods of forecasting the economic production and financial activity of the TC;
- formation of the market of transport-forwarding services.

At the moment only in the technological essence of the process of cargo transportation at different levels and stages of their life cycle operation is divided interests of individual actors are not coordinated and, in General, focused on economic confrontation, there is a system interaction customers and transporters. The existing system of production organization and management does not provide the optimal choice of the economic-organizational criteria and does not direct transport to the high efficiency of their performance. Integration processes that occur in TC at this stage of its operation, are not able to provide its innovative development and do not meet the requirements for structural and temporal adequacy of the global logistics processes.

The organization of the consolidated transportation process should reduce the cost of processes of transportation in comparison with the option to self-perform a full range of logistics operations by the consignee. It is necessary to ensure the development of transport infrastructure in points of direct interaction between different modes of transport and create multifunctional TC, as well as information and logistics centres . Thus, the problem of substantial improvement of organizational production management in TC countries and regions at the present

stage is urgent and requires an appropriate solution based on the implementation of modern technical, technological, organizational, and economic activities.

One way of saving resources in the transportation of goods is the application of the systems decision support (DSS) in the field of transport logistics. The development of software packages, comprehensively solve the problem of the subject area requires serious research with the aim of obtaining efficient algorithms suitable for use in everyday practice.

Intelligent information system (IIS) , underlying any DSS must have the application architecture and have the user-friendly interface for ease of use.

- IIS should have the following functionality:
 - ✓ Providing information management support transportation.
 - ✓ Solution the transportation problem by different methods.
 - ✓ Universal class for working with genetic algorithms.
 - ✓ Finding the optimal solution for scheduling transportation.
- The purpose of the transportation activity is considered to be reached when performing six conditions:
 - ✓ the desired product;
 - ✓ the required quality;
 - ✓ in sufficient quantities;
 - ✓ at a certain time;
 - ✓ at the appointed place;
 - ✓ at minimum cost.

By now we know a lot of algorithms for solving problems of vehicle routing problem (VRP). For the most part this is a heuristic methods used in the presence of a matrix of distances or information about the location of the vertex on the plane. VRP is an NP-hard problem, therefore, the most intensive search is conducted in the direction of approximate algorithms. Proposed exact methods for solving the VRP, as, for example, a method of branches and borders, but the computation time when their use is growing too quickly. VRP involves a significantly larger number of options for viewing than gears with the same number of vertices, while the

application of the method of branch and bound to solve the gear is already difficult for datasets with 30 vertices and more.

3. Classical methods for solving transport problems

Among the classical methods for solving VRP can enumerate the following[1]:

1. Construction algorithms - algorithm Clarke-Wright and its extensions, the sequential insertion of Mole -James, sequential algorithm for insertion Christofides - Mingozi - Toth,
2. Two-phase classical algorithms.
3. The covering algorithm, the algorithm petals
4. The algorithm of Fischer - Jaikumar, the algorithm Bramel - Simchy - Levy.
5. Classic improving the algorithms.

Formal mathematical models of decision making currently, all more fully reflect the complexity of real-world problems that, on the one hand, makes them more adequate to real systems, and on the other, leads to the need to tackle more complex optimization problem. Basic properties of real-world optimization problems - the presence of many criteria, significant restrictions , variables measured in different and algorithmic job functions make it impossible to use traditional methods. The way out of this situation is the use of adaptive stochastic algorithms, successfully overcoming these difficulties.

4. Basic Genetic Algorithm

One of the most frequently used in this setting approaches are evolutionary algorithms [2] that represent stochastic optimization procedures that mimic the processes of natural evolution, in particular genetic algorithms (GA) as in Fig. 1.

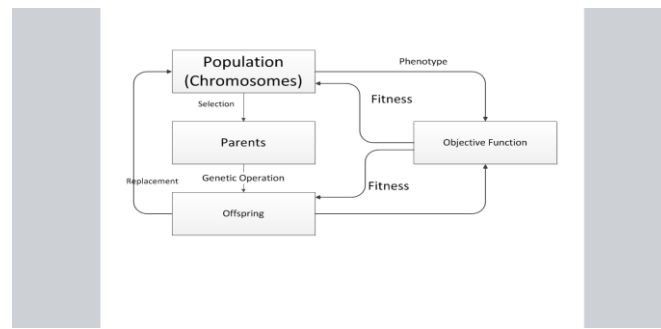


Fig.1. - Basic Genetic Algorithm

Outline of the Basic Genetic Algorithm:

1. *[Start]* Generate random population of n chromosomes (suitable solutions for the problem)
1. *[Fitness]* Evaluate the fitness $f(x)$ of each chromosome x in the population
2. *[New population]* Create a new population by repeating following steps until the new population is complete
3. *[Selection]* Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
4. *[Crossover]* With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
5. *[Mutation]* With a mutation probability mutate new offspring at each locus (position in chromosome).
6. *[Accepting]* Place new offspring in a new population
7. *[Replace]* Use new generated population for a further run of algorithm
8. *[Test]* If the end condition (for example number of populations or improvement of the best solution) is satisfied, stop, and return the best solution in current population
9. *[Loop]* Go to step 2.

Algorithmic job functions and variables measured in different do not pose additional difficulties for GA, which are binary representations of the solutions and do not require information about the properties of the objective functions. However, there are many criteria and limitations hinder the use of GA in practical tasks, since most of the approaches proposed in the field of evolutionary optimization, focused only on one problem, i.e., either there are many criteria, or the presence of restrictions. Approaches combining both directions, are rare, and their efficiency is not always satisfactory [3]. Thus, the improvement of existing and development of new efficient adaptive search algorithms conditional multi-objective optimization is an actual scientific problem.

5. Formalization of the transport problem

It is known that the classical algorithms do not have the possibility of parallelization and have exponential increase in execution time from the dimension of the problem. I.e., the number of mathematical operations (commands) grows exponentially, and the development of processor elements (increase the clock frequency, reducing the number of cycles instruction execution, the delay to fetch data from memory) does not compensate for the growing (with increasing problem dimension) needs to classical algorithms. Exact methods of solving transportation problems (TP), allow you to find a solution only for problems with a small number of clients (i.e., up to 50 clients). To solve large-scale problems exact methods are not effective in relation to their lot of time. However, right now, requires efficient algorithms for solving large-scale problems, because currently, there is a globalization of the economy. This leads to the necessity of planning transport operations with a large number of customers, i.e., TP of high dimensionality. Thus, the solution TP high dimensionality is an important task. One of the classes TP is TP with a time limit. This class of problems is difficult to solve, but it is necessary and widely used in practice. Model TP-limited time describes: banking and postal delivery, transportation of people, the collection of industrial and household waste,

food delivery, delivery of fuel and materials to enterprises. An important drawback of these algorithms is that they are not designed to solve problems with changing the number of requests from clients, which is important today in connection with the development of mobile communications, enabling them to track the route of the vehicle and to transmit the modified route. Therefore, the task of developing complex algorithms for solving TP with static and dynamic customer requests is an actual scientific problem.

TP with the time limit belongs to the class of NP-hard problems, exact solution methods for problems of this type is effective in a small number of clients (i.e., up to 50 clients). The main methods of solving this problem with a large number of clients are heuristic and metaheuristics methods. The best results in solving test problems give hybrid algorithms, guided by the global heuristic (metaheuristics), which, in turn, in the process of looking at the intermediate stages uses various methods to improve the route based on the method of local search. Effective are different postoptimization procedures that improve specific final decision and techniques to adapt the algorithm to the current task (when at different stages of the search are used in different parts of the algorithm).

Mathematically TP with limited time can be represented as a graph $G=(N,A)$, where: N is the set of vertices corresponding to the set of customers) (peaks 1, 2, ..., n) and the source depot, which begin and end their route all vehicles (vertices 0 and $n+1$); A - set of arcs connecting the vertices of the graph. We introduce the notation:

C - many clients, $|C| = n$;

i, j is the i -th and j -th clients ;

A_{ij} is an arc that connects the i -th and j -th vertex;

d_i is the demand of the i -th customer;

t_{ij} is the travel time of arc (i, j) , consisting of the service time of customer i and the travel time of the vehicle from customer i to customer j ;

c_{ij} - the cost of car travel from customer i to customer j ;

V is the number of identical vehicles of capacity q ;

k - k -th vehicle ;

$[a_i, b_i]$ is the time window - the period of time during which must be serviced by the i -th customer; S_i^k - arrival time of the k -th vehicle to the i -th customer; the departure time from the depot for all cars equal to 0 (i.e., $S_{0k} = 0$); X_{ij}^k is a variable taking values $\{0, 1\}$ and characterizing the direction of movement of the car:

$X_{ij}^k = 1$ - from customer i to customer j , $X_{ij}^k = 0$ - in the opposite direction.

6. The solution of the transport problem using genetic algorithms

. TC with the time limit can be described as follows [4]. There are a number of vehicles, one warehouse (depot) and a number of clients. For each vehicle you want to make a route over which a vehicle visits a number of clients (for example, to deliver any cargo). On the route of each vehicle there are a number of limitations. The main limitation is represented by the formulas (1) to(3).

$$\sum_{k \in V} \sum_{j \in N} X_{ij}^k = 1, \forall i \in C \quad (1)$$

$$\sum_{i \in C} d_i \sum_{j \in N} X_{ij}^k \leq q, \forall k \in V \quad (2)$$

$$a_i \leq S_i^k \leq b_i, \forall i \in N, \forall k \in V \quad (3)$$

Constraint (1) believes that every client is served by only one transport mean and only once. Variables X_{ij}^k take values $\{0, 1\}$, where 1 means that the car moves from vertex i to vertex j , 0 - otherwise. The upper index k is denoted by the corresponding car, where $k \in V$, V is the number of identical transport mean of capacity q . Constraint (2) determines that the transport mean cannot serve more customers than its capacity d_i is the demand of the corresponding client, $i \in C$, C - many clients. Constraint (3) is the time limit; the arrival of the vehicle to the customer must be within the time window, where S_i^k is the arrival time of the corresponding vehicle to a specific client, and $[a_i, b_i]$ - period of time, the so-called time window (*time window*) for which must be serviced by the customer. For this task are formulated the following goal (objective function): the primary goal is to minimize the total number of vehicles required to service all clients; secondary is to minimize the total service time of all customers and the total distance traveled by all transport means.

Exact methods based on the methods of directed enumeration, are not effectively solve the problem of high dimensionality. Heuristic and metaheuristics methods, which include genetic algorithms, give more acceptable results.

7. Designing a class library for genetic algorithms

Software implementation of the class library GA is implemented on the platform .NET Framework and shell development MS Visual Studio, programming language - C# [5].

Created a generic class library that implements the basic steps GA:

1. The creation of a population
2. Selection
3. Crossing
4. Mutation

Created a base class *BaseSpecies*. All types must be derived from the base class *BaseSpecies* *<TSpecies>* , where *TSpecies* - specific derived type:

abstract public class BaseSpecies<TSpecies>:

{

Comparable where

TSpecies: BaseSpecies<TSpecies>

{...}

static protected Cross (double x, double y)

{...}

static protected double Mutation (double val)

{...}

static protected double TestChromosomes()

{...}

}

As you can see from the code, *BaseSpecies* is an abstract class that contains static protected methods for crossing and mutation. Let us examine some of the

methods in more detail. Let's start with the methods that must be implemented in derived classes : *static protected double TestChromosomes()*. This method should set the internal protected Boolean variable *m_Dead* that shows fit chromosomes of the limitations that they impose (for example, in our example, are the values of the chromosomes in the interval [-5.0; 5.0]). If the value of the chromosomes suit us, then *m_Dead* is set to *false*, otherwise - *true*. With this variable (obtained through the property *Dead*) population rejects obviously unsuccessful. The next method that you must implement in derived class, it is abstract public *TSpecies Cross (TSpecies Species);*.

This method creates a new view, crossing himself and another type that is passed as an argument. Here you need to make some notes. Usually, if the view contains multiple chromosomes, and hybrid chromosomes "one species", i.e. in our case *a0* cross itself (i.e. *this*) and *a0* to another similar cross *a1* and *a1*, i.e. it makes no sense to breed *a0* yourself and *a1* of another class. The sense of crossing is that you take one part of the chromosome itself and the second part of another type and create a new chromosome, which will be held in a new form. Often chromosome in bitove. The essence bitwise crossing is that the first randomly selected break point (crossing) of the chromosome, then create a new chromosome, which consists of a left part of the first chromosome and the right second. For example, suppose we have two eight-bit chromosome : 10110111 and 01110010. Randomly selected break point (marked by the symbol '|'): 101 | 10111 and 011 | 10010 .

From here we can get the following two chromosomes - 101 10010 and 011 10111. You can also search for two or more points of intersection. Thus, we must have a constructor that creates the appearance of the chromosomes. For this class *BaseSpecies* included the implementation of known static methods for mating chromosomes of some types (*Double*, *Int64* and *Int32*). Let us consider the first two methods. In this case, there are two intersection points: first, in the middle of a word, the second the sign of the number, that is, first crossed chromosomes in bitove without regard to sign, and then also randomly selected for a sign from one of the chromosomes - *public void Mutation()*.

The next step is mutation. Mutation operates on a single chromosome. In theory, when mutations can occur any changes. But in this implementation the mutation changes one bit in the word. Mutations occur relatively rarely. For example, often in practice, the probability of mutation is set 5-10%, but sometimes we try to make it a little more (about 30%). The process of selecting chromosomes for mutation can be solved by using random selection (as the simplest option - random number generator). Quite often as a mutation for small numbers it is better to use not written above function, but simply a random number generator that generates a random number in a certain interval.

Implementation of population(class Population) -

public class Population<TSpecies> where TSpecies:BaseSpecies<TSpecies>.

As you can see, it has a generic parameter that is a type, which must be derived from *BaseSpecies <TSpecies>*.

Let's start with the properties that must be configured before you begin algorithm:

- *MaxSize (Int32)* - maximum number of species that can contain the population. This is the size to which "cut" the population after mating. The default is 500.
- *CrossPossibility (Double)* - the probability of crossing. It must be in the interval (0.0; 1.0). If this condition is not met, implement an *ArgumentOutOfRangeExceptionException*. By default this value is set equal to 0.95.
- *MutationPossibility (Double)* - the probability of mutation. It must be in the interval (0.0; 1.0). If this condition is not met, implement an *ArgumentOutOfRangeExceptionException* exception.

In the work value of the probability of selection is taken equal to 1.0. Otherwise there would be no need to remove the worst kinds of populations according to the values of a certain probability. But in any case, the species whose chromosomes do not fall within the desired interval, or do not meet any other specified conditions (the so-called dead species), should be removed anyway.

Now let's look at the public methods that will need to address:

- *public void Add (TSpecies species)* - add new type in the population. The necessary number of species should be added in-hand. Before the algorithm requires that the

population was at least two types. The number of species in a population may be smaller than the set value of *MaxSize*. If after breeding population size is less than *MaxSize*, just not removed the worst kinds (even the dead).

- *void NextGeneration ()* - get the next population. This method may take a lot of time, so it is best to call from a separate thread.

8. The overall plan for the use of the library

Consider the General plan of the steps when using the library:

- ✓ Create a class type that is derived from *BaseSpecies* *<TSpecies>*, where the argument of generalization in *BaseSpecies* is passed the name of the class type.
- ✓ Inside the class type to create a set of chromosomes necessary types.
- ✓ Abstract method overloading public *TSpecies Cross (TSpecies Species)*, which should create a new look. As a method parameter is an individual of the same species, which must be crossing.
- ✓ Overloading the property (only *get public double FinalFunc ()*), that returns the value of the objective function. If the target function complex, it is convenient to store the value of the objective function in a separate variable whose value and returns this property, and the calculation of the objective function to be implemented in the constructor of the form.
- ✓ Overloaded method *public void Mutation ()*.
- ✓ Overloaded method *public override void TestChromosomes ()*, which sets the value of the variable *m_Dead* to true if this type we do not knowingly arranges, for example, *e* is the value of its chromosomes falls within the prescribed limits, and to *false* otherwise.
- ✓ Create population *Population <TSpecies>*, which will keep individuals as required.
- ✓ Call the *Reset ()* method population for removal of all kinds, if they were

already in the population.

- ✓ Through *MaxSize* population to set the maximum size of the population.
- ✓ To establish the probability of mutation through the property *MutationPossibility*. The value of this property must lie in the range from 0 to 1. The default value is set to 0.1.
- ✓ To establish the probability of crossing through the property *CrossPossibility*. The value of this property must lie in the range from 0 to 1, with zero probability of crossing is not allowed. The default value is set to 0.95.
- ✓ Add to the population needed the number of individuals of the respective species through method *void Add (Tspecies)*.
- ✓ Call method *void NextGeneration ()* as long, while in the population need be, for example, until it reaches the desired value tolerance.
- ✓ Through the property *Tspecies BestSpecies* can get the best at a given iteration (in this generation) an individual of the species, in which the objective function is smaller than the other.

9. Software implementation

Thanks created the class library, you can implement any modification of the genetic algorithms . So, use it to implement the following algorithm:

Step 1. Construction of the initial population of a certain size (the initialization statement).

Step 2. Select two parent chromosomes from the population (operator selection).

Step 3. Copy the selected chromosomes and application of genetic operators to create new chromosomes (operators of crossover, mutation).

Step 4. The selection and subsequent removal of the chromosomes from the population to recover to its original size.

Step 5. Until not completed the specified number of steps, return to step 2; otherwise, end the algorithm.

The first step is finding the initial population, as in Fig.2, this would solve the transportation problem by the method of the North-West corner” and the method “minimum elements”.

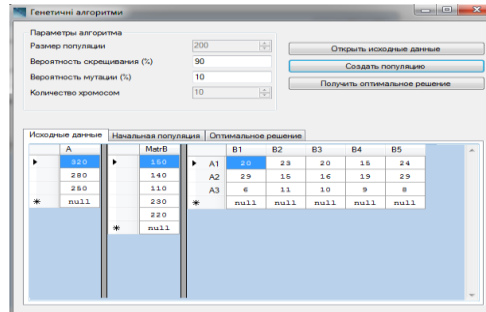


Fig.2. - Finding the Initial Population

The resulting table will be the source samples as in Fig.3.

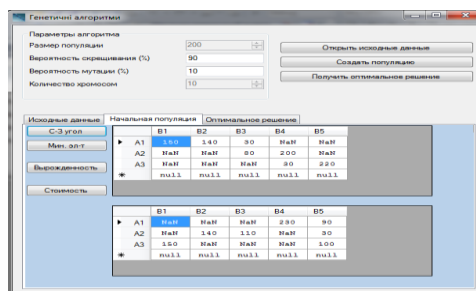


Fig.3. - Original Sample

After sampling software package performs crossover and mutation. Then there is a formation of the optimal solution as in Fig.4.

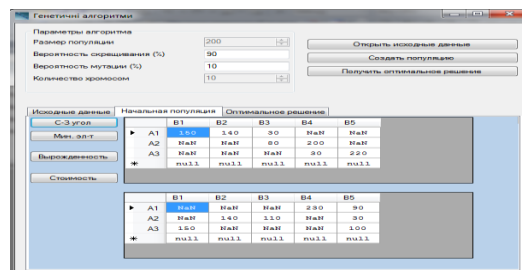


Fig.4. - Optimal Solution

CONCLUSIONS

The problem of decision making under uncertainty has an important place in the General problem of decision making. The successful solution of this problem is currently impossible without the use of new information technologies, as an integral part of the intellectual tools of information processing. The use of various modifications of HA allows to solve the problem of choosing the optimal parameters of methods, models and algorithms for decision support under uncertainty in an acceptable time. Based on the proposed algorithm software developed software environment that allows you to solve practical problems in management decisions. Studying the problem of optimization of logistics processes for managing objects, we can conclude that the use of genetic algorithms for solving such problems is a powerful mathematical tool and can be used for a wide range of applied problems, which can be enabled and those that are difficult or impossible to solve by other methods. The following functions are implemented as software:

- ✓ Create a generic class for working with genetic algorithms.
- ✓ The solution of the transport problem by different methods.
- ✓ Finding the optimal solution for scheduling transportation.

The main advantage of evolutionary, genetic methods can be called relatively simple (compared to traditional methods of decision) adaptation of the algorithm to different variants of the task. Using the same version of the algorithm can solve a number of similar tasks.

REFERENCES

1. Pozhidaev, M. C., Kostiuk, Y. L. (2009), "Balanced vehicle routing problem and the problem of practical application metaheuristics strategies", *Information technologies and mathematical modeling (ICCM 2009)*, Tomsk , TU, November 13- 14, 2009, pp. 233-235.
2. Charles C. Peck, Atam P. Dhawan, "Genetic algorithms as global random search methods: An alternative perspective", *Evolutionary Computation*, Volume 3 Issue 1, MIT Press, March 1995.

3. Gumennikova, A.V. (2004), “ An evolutionary approach to solve multicriteria problems constrained optimization ” , *VIII International scientific – practical conference “System Analysis in Planning and Management”*, Snt – Petersburg, October 25-27, 2004, pp. 72-76.
4. Emelyanova , T.S. (2008), “Research and development of algorithms solutions to transport problems using genetics methods”, There is abstract for Cand. Sc. (Engineering), 05.13.01, Technological Institute of Southern Federal University, Taganrog, Russia.
5. Herbert Schildt (2011), C# 4.0. The Complete Reference, Translated by Bershtane, I.V., Williams.

*ПРИМЕНЕНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ ПРИ ОПТИМИЗАЦИИ ТРАНСПОРТНЫХ ПЕРЕВОЗОК*

*ЗАСТОСУВАННЯ ІНТЕЛЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ
ТЕХНОЛОГІЙ ПРИ ОПТИМІЗАЦІЇ ТРАНСПОРТНИХ ПЕРЕВЕЗЕНЬ*

CONTENS

Introduction

1. Methods for solving NP - complex problems
2. Locations for the application of transport tasks
3. Classical methods for solving transport problems
4. Basic Genetic Algorithm
5. Formalization of the transport problem
6. The solution of the transport problem using genetic algorithms
7. Designing a class library for genetic algorithms
8. The overall plan for the use of the library
9. Software implementation

Conclusions