The decision of a knapsack problem on a methods of genetic programming

by Eugene Ledak

THE TABLE OF CONTENTS

	Introduction	3
1.	Statement of a problem	4
2.	Method of Monte Carlo against the Method of elite populations	5
3.	The description of algorithms	6
4.	Results of testing	12
	Conclusions	14
	References	15

INTRODUCTION

The modern level of scientific researches and development of market economy assumes fast and effective information processing. The set of applied problems concerns to the theory of discrete programming: for example, a traveling salesman problem and a knapsack problem. The exact decision of such problems is possible for finding for final number of iterations. Therefore, to such problems we apply a method of simple picking all possible values (later, we would call this group of methods an *extensive methods*). However, in a case when the set of allowable decisions will consist of very big number of points, the finding of the decision can be delayed by this method for years. Therefore, for the decisions of problems of discrete programming have been developed special, including, and an extensive methods that try to narrow amount of touched points.

There are two approaches to the decision of problems of discrete programming. The first has received the name combinatory and is based on purposeful, partial picking all the private decisions. Methods of this group are dividing on exact and approached. To exact the method of branches and borders concerns. The approached algorithms are based on some heuristic ideas which are taking into account specificity of the solved problem. Naturally, the approached methods do not give the exact decision, but search of the decision by such methods occupies in tens times less time.

The second approach is the methods of separation. The idea of these methods will be, that the set of allowable decisions forms some convex closed contour. Otherwise, such contour is under construction is artificial. The extreme problem is solved on this set by cutting off of parts of set. The part, which is not containing the extreme decision, is rejected on some algorithm. The rest is exposed to a section until there will be a contour not giving in to the further section and containing the optimum decision. The representative of the given group is Gomori algorithm for the decision of problems of integer linear programming.

1. THE DESCRIPTION OF THE KNAPSACK PROBLEM

The tourist gathers in a hike. He has a backpack into which it's possible to pack things in total weight in p units, and a set from n indivisible things that the tourist would like to take with him in a hike. Each thing possesses weight a_i units and costs c_i monetary units. It is required to find such a set of things which would be maximal at total cost but which total weight did not exceed capacity of a backpack.

Let

$$x_i = \begin{cases} 1, \mathit{if}\ \ i - st\ thing\ \ is\ \ packed\,; \\ 0, \mathit{if}\ \ i - st\ thing\ \ isn't\ \ packed\,. \end{cases}$$

Then in the size describing cost of some i-st thing, will be $c_i x_i$, and in the size describing weight of some i-st thing, will be $a_i x_i$. The set as a whole is characterized by total cost and total weight. The total cost is the sum of all values, that represent items' cost. In turn, the total weight is the sum of all values, representating weights of things.

Statement of a problem in a mathematical kind will look as follows:

$$\sum c_i x_i \ \rightarrow\ \max;$$

$$\begin{cases} \sum a_i x_i \ \le\ p; \\ x_i \in \{0, 1\}; \\ i = \overline{1, n}\,. \end{cases}$$

2. A METHOD OF MONTE CARLO AGAINST THE METHOD OF ELITE POPULATIONS

In the given work comparison of methods of Monte Carlo and a method of the genetic programming based on elite populations has been spent.

For the beginning, some words about the reasons of comparison of these algorithms are necessary to tell, in fact the most traditional for the decision of a knapsack problem is the method of branches and borders, instead of a method of Monte Carlo. In a basis of this choice that fact lays, that the method of branches and borders cannot be used with a plenty of selected things. At presence in 'store-room' more than 50 – 100 subjects the method of branches and borders does not surpass a method of a casual choice of subjects, i.e. a method of Monte Carlo, on accuracy. It is obvious, that programming of a method of branches and borders represents much more complexity, than programming of a method of Monte Carlo. This circumstance also has played the leading part in a choice of the competitor for genetic algorithm. Certainly, there is a set of versions of a method of Monte Carlo, but the author was not familiar with them for the period of creation of the program. Besides during programming a method of elite populations the author has encountered unforeseen complications that have considerably reduced time spent for search and a choice of algorithms of the decision of the given problem.

3. THE DESCRIPTION OF ALGORITHMS

3.1. A method of Monte Carlo.

1. Small trick: the method of casual selection of subjects will show the best results if to sort all set of the subjects kept in a 'store-room' by criterion of utility. Utility of u_i-st thing is calculated according to the formula:

$$u_i = \frac{c_i}{p_i}, \textit{ where}$$

c_i *– cost of i-st thing in a* 'store-room',

p_i *– weight of i-st thing in a* 'store-room'.

2. Since the most useful thing (u_j = max u_i, where $i=\overline{1,10}$) we touch the subjects accessible to a choice. About probability of 50% the thing gets in required set.

3. After picking a thing into the set, amount of things of the given type decreases for unit. The thing ceases to be accessible to a choice as soon as the amount of things of the given type becomes equal to zero.

4. Process lasts until in a 'store-room' will be no thing that could be placed in a knapsack.

3.2. Elite populations method.

3.2.1. General provisions.

In a year of 1975 John Holland has offered algorithm which task was modeling process of evolution. The given algorithm will transform two populations: B (t), named a base population, and B (t+1), named a population of descendants. These populations contain identical amount of individuals. On a step of initialization, the individuals generated by casual image fill the base population. At a stage of fitness evaluation for each of individuals value of fitness function is calculated. After initialization of a base population B (0) we pass to a basic cycle of the program in which is determined process of artificial evolution.

In the beginning there is a reproduction based on crossover of casually chosen individuals from a base population. Probabilities of a choice, however, are not identical: individuals with greater value of fitness function have greater chances of a reproduction. As a result of a reproduction it is possible to expect, that for crossover a lot of copies of the most adapted individuals will be chosen. The chosen individuals are located in a population of descendants B (1).

After filling a population of descendants, individuals from B (1) are exposed to genetic operations – to crossover and a mutation. During crossover an individual are grouped in pairs, and for each separate pair it is made a decision on realization of crossover. If the decision is positive, there is a crossover, and the received individuals - descendants, replace the parents. Otherwise, the pair remains without changes. The probability of carrying out of crossing $\boldsymbol{p_c}$ is a parameter of algorithm. Further, the mutation being change of its genotype is made for each individual from B (1). The individuals received thus are estimated by function evaluation of fitness and form a population of descendants B (1). This population becomes a new base population for the following turn of a cycle. The counter of iterations t is increasing.

Process passes cyclically until a condition of a stop is satisfied. It can be, for example, performance of the set number of iterations, or also a finding of an individual with rather high value of fitness function. Each pass of a basic cycle frequently is called a generation and is considered as his one iteration of basic cycle, and a condition of a population B (t) at the moment of time t is also called a generation.

Let's consider in more detail concepts of coding of individuals and genetic operations. The circuit of the algorithm offered by Holland, abstracts from the certain way of coding of individuals and operations’ performance of crossover and a mutation, however the given algorithm is most known at use of the binary coding. In the given version a chromosome is a n-dimensional vector of genes, each of which is zero or a one (fig. 3.1).

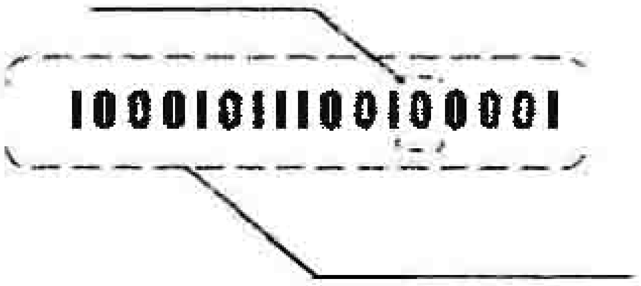

fig. 3.1. A chromosome with the binary coding

Mutation of an individual is the operator, made above each gene separately. It’s based on a probability p_m, which value is a parameter of algorithm. If mutation was made, then a value of a gene on opposite.

In turn, two parental individuals take participation in crossover. The course of this operation imitates process of changes in DNA, observable in the nature, an event, during the generative reproduction.

Parental chromosomes are separated on two fragments. Further, the first fragment of the first chromosome sticks together with the second fragment of the second chromosome, similarly first fragment of the second chromosome sticks together with the second fragment of the first chromosome. The chains of genes received thus form hereditary individuals (fig. 3.2).

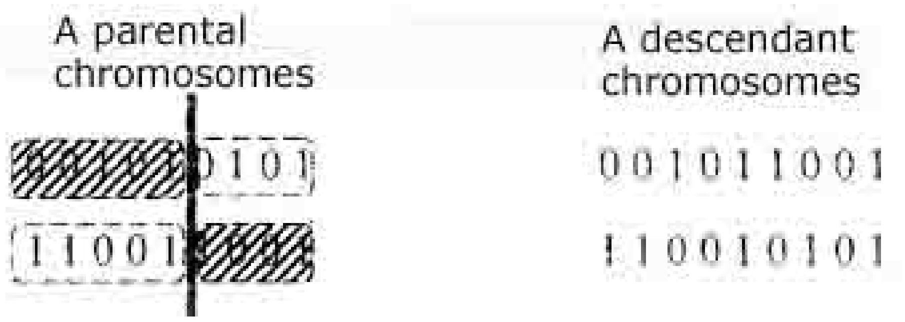

fig. 3.2. Crossover scheme in the binary coding

As all chromosomes contain identical number of genes, division should occurs in the same place for both parental chromosomes. The division place is chosen casually with regular distribution.

The operator of inversion applied to a separate chromosome, changes value of each of genes on opposite.

One of the major aspects of genetic algorithm is the mechanism of a choice of the most fitted individuals for the further reproduction. The population B (1) will consist of individuals among which prevail the most fitted. For selection of individuals for crossover, G. Holland has offered the algorithm named a proportional reproduction or a roulette reproduction. It’s described by probability on set of elements of a population B (t) with distribution

$$p_i(X) = \frac{\Phi(X_i)}{\sum_{j=1}^{N} \Phi(Y_j)}$$

where *X* designates an individual,

Φ {X} — value of its fitness function,

N — amount of picked items.

For definition of the individual for a reproduction, we make a casual choice (with regular distribution) number *a* from an interval (0.1). The individual Xi. for which following condition satisfies is reproduced:

$$P_r(x^{i-1})<a<P_r(X^i).$$

It is necessary to note, that average value of individuals’ fitness function in population B (1) is more than average value of fitness in base population B (0). It could be expected, since the algorithm is constructed thus, that more adapted individuals are translated in a temporary population. However the price that is necessary to pay for it, the reduction of heterogeneity B (0) in relation to B (1) is. Genetic operators return heterogeneity in B (1) not guaranteeing, however, improvement of fitnes’s average value.

When to stop genetic algorithm? The question about stop conditions of algorithm is very important, and the answer to it has not been received in above resulted reasoning. The best (though and not very practical) the answer will be...

never, as the genetic algorithm is casual process and it is impossible to guarantee achievements of result for final number of generations.

3.2.2. Some aspects of algorithm's realization.

Within the framework of a solved problem the binary chromosome is convenient way of coding. It can be examined as a vector of the subjects packed into a knapsack.

For transformation of chromosomes we shall use the genetic operators determined above, except for the operator of inversion perhaps.

Calculation of criterion function we shall make so that the fittest appeared the sets having the maximal total cost and which total weight would be less than limit of a knapsack:

$$\Phi\,(X)=\begin{cases}\sum c_i x_{i,}\ if \sum a_i x_i \le p;\\ 0, otherwise\end{cases}$$

The proportional reproduction will be used. But thus it is required, that values of function of fitness were positive, that, generally speaking, is not true for the accepted kind of function. For overcoming it, we apply, a method of fitness scaling, at which the probability of a reproduction is described by a ratio:

$$p_i(X)=\frac{\Phi(X_i)-\Phi_{\min}}{\sum_{j=1}^{N}\Phi(Y_j)\Phi_{\min}}$$

where $\Phi_{\min}$ is, value of fitness function of the worst individual in population B(0):

$$\Phi_{\min} = \min \Phi(Y)$$

Due to this we avoid danger, reception of 'negative probability'.

The algorithm stops, if during 100 consecutive generations there hasn’t reached improvement of result.

4. RESULTS OF TESTING

Testing for a casual set of things which results have been tabulated (tab. 4.1.) has been has been carried out

Table 4.1.

Item No	Item’s value	Item’s weight	1)	2)
1	32	9	0	0
2	29	6	1	0
3	99	3	1	3
4	23	1	1	4
5	36	6	1	0
6	74	10	1	0
7	6	9	1	0
8	44	6	1	0
9	57	3	1	0
10	19	4	1	0
11	99	1	1	7
12	30	2	0	2
13	35	1	1	4
14	2	3	0	0
15	50	3	0	1
16	39	4	0	0
17	19	6	0	0
18	12	9	0	0
19	54	2	0	2
20	39	9	0	0
21	100	7	0	0
22	81	6	0	0
23	1	1	0	0
24	27	9	0	0
25	21	4	0	0
26	80	7	0	0
27	49	4	0	0
28	78	3	0	5
29	20	3	0	0
30	56	6	0	0
Sum			521	1830
Freespace			0	
Quantity of iterations				

Results of comparison of efficiency of two algorithms

1) – How many items of this type was selected with Monte-Carlo method?
2) – How many items of this type was selected Simple Genetic Algorithm?

According to performance of computing process schedules on performance of each of algorithms (fig. 4.1) have been constructed.

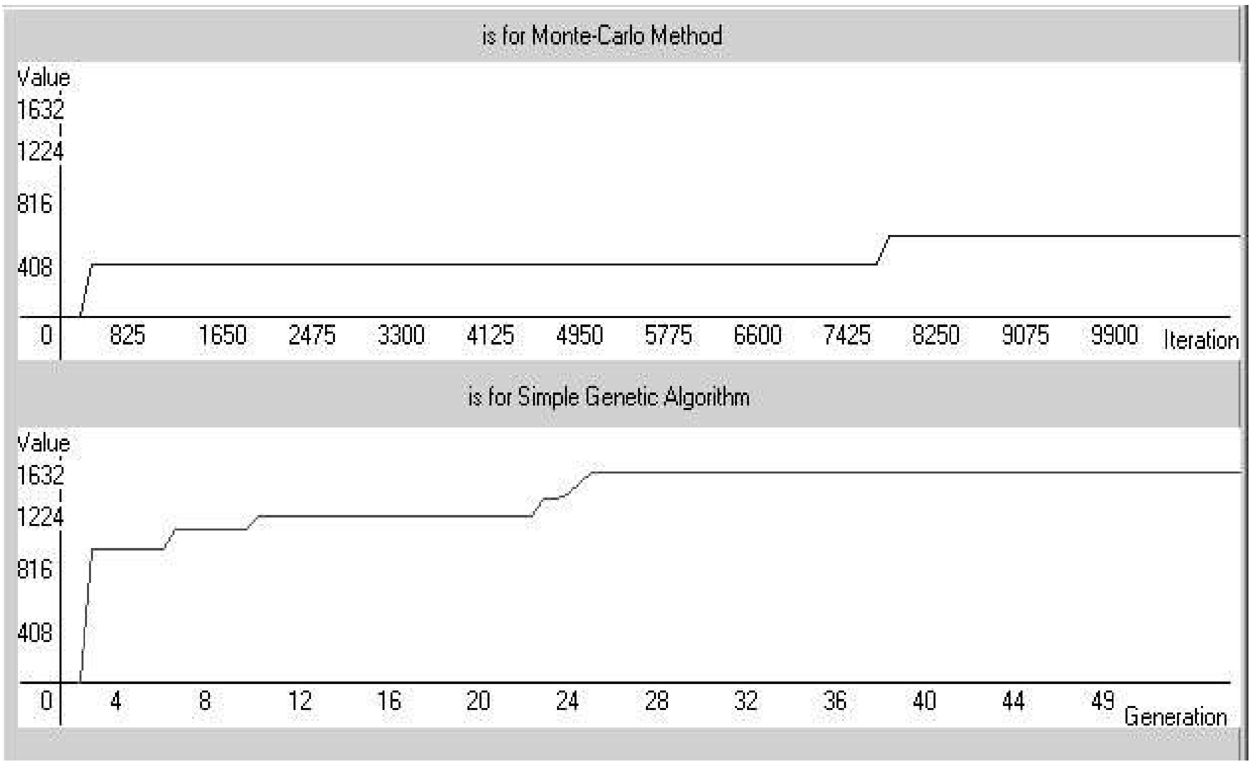

fig. 4.1. Dynamics of computing process on each of algorithms

On the schedule it is well shown, that in simple genetic algorithm the method of elite populations has been used: the schedule has not decreasing kind. It occurs because the fittest individuals enter into each subsequent population from the previous population. Thus, fitness of each subsequent population turns out not less, than fitness of the previous populations.

CONCLUSIONS

During performance of the given work the following methods of the decision of problems of discrete programming have been considered: simple genetic algorithm and a method of Monte Carlo. The simple genetic algorithm allows finding the best value, in comparison with a method of Monte Carlo. But the method of Monte Carlo prospects for decisions much faster. However, the given method not always gives the decision equal to the decision, received as a result of functioning simple genetic algorithm.

It’s possible to increase an accuracy of the decision with the help of simple genetic algorithm, having increased the size of populations. In our case, the size of a population twice is more than amount of subjects accessible to the tourist. At increase in number of populations, time of the decision of a problem grows, but the found decision is the best, in most cases, than the decision found at smaller number of a population.

Analyzing the results of testing above mentioned, it is possible to draw a conclusion, that the simple genetic algorithm is better than a method of Monte Carlo. It is obvious, as if in the second algorithm there is a casual choice of elements to the certain probability in the first algorithm occurs purposeful choice, leading to increasing the result that has been saved up at the previous stages.

THE LITERATURE

1. Воронковский Г.К., Махотило К.В., Петрашев С.Н., Сергеев С.А. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности – Харьков: ”Основа”, 1997.