Genetic Optimization for Transportation Problem

DIANA BETINA MIRSU[1] GABRIELA PROSTEAN[1] RADU MIRSU[2]
Faculty of Management in Production and Transportation[1]
Faculty of Electronics and Telecomunications[2]
Politehnica University of Timisoara
Timisoara, str. Remus, nr.14
ROMANIA
betinaiovan@yahoo.com gabriela.prostean@mpt.upt.ro radu.mirsu@etc.upt.ro

Abstract: - This paper presents a problem of transportation planning that is able to achieve higher performance by means of genetic optimization [2]. The paper introduces a model for simulating and evaluating the costs of a transportation system. It also describes a new way to use genetic optimization for a problem that requires adaptive chromosome length [1].

Key-Words: - Genetic optimization, transportation, management, adaptive chromosome length

1 Introduction

The paper presents a problem of transportation planning. It is similar in nature to the classical traveling salesman problem with new elements that add specific costs frequently found in industry. Section 2 introduces a simulation model that allows evaluating the costs of a specific transportation route. This model can serve as fitness function for the genetic optimization solver. Section 3 presents how the chromosomes are coded, how cross-over and mutation is performed and why an adaptive chromosome length is needed for this application. Section 4 presents the results of the optimization.

2 The Simulation Model

The model is developed in MATLAB and simulates the distribution of goods between several factories. The factories are served by one truck which has to transport products between them. The factories are distributed in a two dimensional space so distances between them can be computed. The coordinates of each factory can either be selected by the user or be randomly generated by the computer. Given a set of products $P_k, k = \overline{1...N}$, each factory must convert an input product $P_i \in P_k$ into an output product $P_j \in P_k, i \neq j$ at each simulation time step. The truck must transport products between factories. Distances are converted into integers so at each simulation time step the truck covers one distance unit. Figure 1 illustrates the simulation flow an individual factory for one simulation time step.

If the truck is in the factory it is searched for the demanded product P_i. Note that each factory accepts only one type of product. If product P_i is found it is transferred (truck unload) from the truck into the factory input product stock S_{input}. If the product P_i is not found in the truck the input stock remains unchanged. Next, the entire factory output product stock S_{output} is transferred into the truck (truck load). For this application the capacity of the truck is not limited. At the same time with the truck load/unload activity there is also factory production activity. Production occurs only if the input product stock is not empty. It is done by transferring an item from S_{input} to S_{output}. This equivalent to converting one item of product P_i into one item of product P_j. Last, the costs are computed. Each factory has two types of costs: storage cost and production halt cost. The storage cost C_s is proportional to the output product stock S_{output}. This means that any factory would rather not store its output products P_j and will produce high cost if rarely visited by a truck. The production halt cost C_{PH} is computed with formula (1), where HSL is the halt safety level and C_k is a scaling constant. This cost comes from the idea that any situation when the production halts due to input stock insufficiency should be penalized.

$$C_{PH} = \max\left[\left(1 - \frac{S_{input}}{HSL}\right) * C_K, 0\right] \quad (1)$$

Normally, this cost should not depend on the stock level because production halts only when stock is

zero. However, it is often useful to also penalize any situation that potentially leads to a production halt. If the input stock level S_{input} is above the halt safety level HSL the cost C is zero. If stock level is below HSL, the cost linearly increases with gradient $\frac{C_k}{HSL}$. If production halts, the maximum penalty is given ($C_{PH} = C_k$).

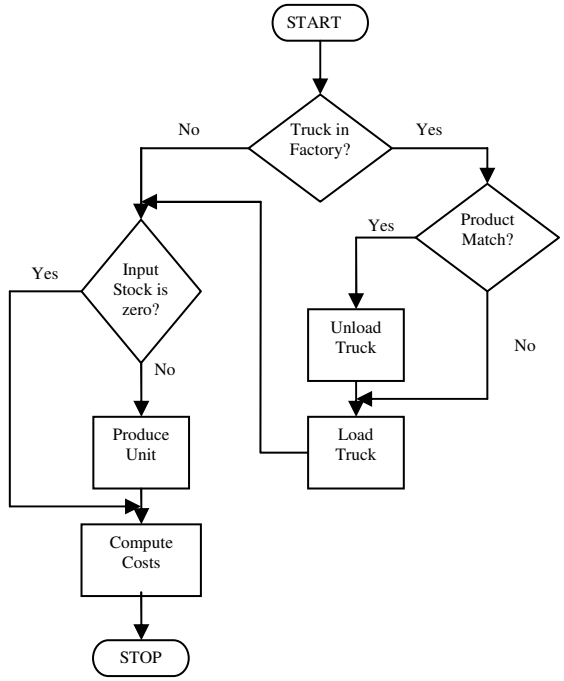

Figure 1. Factory Model Simulation Flow

Figure 2 illustrates the simulation flow for a truck model during one simulation time step. If the truck is inside a factory it waits for the factory to perform the load/unload functions as presented in figure 1. Next, the truck starts toward the next destination as instructed by the genetic optimization solver. The distance to the next destination is also computed at this point.

If the truck is not inside a factory (is traveling) it decrements the remaining distance and computes the traveling costs. Travel costs C_T are constant per unit of distance. Note that figures 1 and 2 present the steps performed during a single simulation time step. The variables that are updated during these steps (factory stocks, truck stock, distances to travel, and costs) are global simulation state variables that store their values from one simulation time step to the next. An overall cost is computed as an accumulation of all costs during the simulation. This cost will serve as performance criteria to the genetic optimization solver.

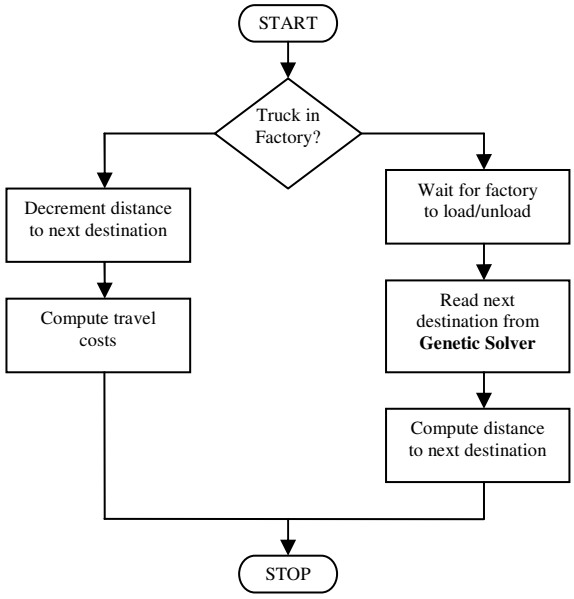

Figure 2. Truck Model Simulation Flow

3 Applying Genetic Optimization

Section 2 introduced a MATLAB routine that implements the model of a transportation problem. Because the routine estimates the general cost of a proposed route it can be used as fitness function for a genetic optimization tool. The results presented in this paper are obtained by using the MATLAB genetic optimization toolbox.

3.1. Chromosome Encoding

Each chromosome is represented by a string of positive integers representing a specific truck route. Each integer in the string ranges from 1 to N, where N is the number of factories. Figure 3 presents an example where the truck route is: factory 3, factory 1, factory 5 and so on.

3	1	5	2	6	4

Figure 3. Chromosome Encoding

The MATLAB genetic optimization solver only accepts chromosomes of types ‘double’ and ‘bit-strings’. For these two data types all functions that are necessary to perform the optimization are provided by the toolbox. An additional type ‘custom’ is also accepted by the solver. This type allows using any encoding for the chromosomes. However, when the ‘custom’ data-type is selected, the user has to provide the following external functions: “create initial population”, “create crossover” and “create mutation”. These three functions, and also the fitness function have to be

compatible with the new 'custom' data-type. For this paper, because the chromosome contains positive integers from a bounded interval and not doubles or bits, the 'custom' data-type is selected.

- The "create initial population" function generates a cell array *P* that contains several vectors, each one corresponding to a single individual of the population. If for example N=10 factories and the chromosome length L = 30 one individual will be a vector of 30 elements containing a concatenation of three random permutations of numbers 1 to 10. This is equivalent of the truck passing three times via each factory. The first element in the vector (the first factory) can be imposed by the program allowing the user to have a pre-determined start point for the truck route. This is useful because it allows several optimizations to be performed serially, with each one starting where the previous one left off.

- The "create cross over" function operates on a single parent and creates a single child. The cross-over copies the chromosome of the parent to the child but reverses the order of the string between two indexes that are randomly generated. Figure 4 illustrates a cross-over operation.

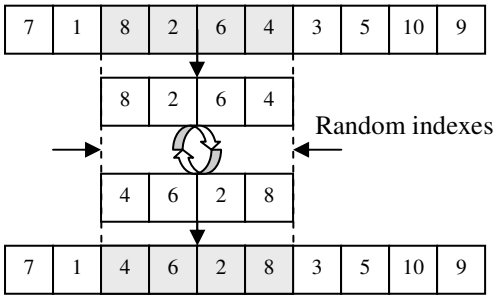

Figure 4. Cross-over operation

- The "create mutation" function also generates one child from one parent. It does this by inter-changing two entries of the parent chromosome. The entries are chosen randomly. Figure 5 illustrates a mutation operation.

3.2. Consistency Problem. Uniform Simulation Time versus Uniform Chromosome Length.
The simulating routine presented in section 2 states that a truck will cover one unit of distance for every simulation time-step. This means that for two distinct individuals (same chromosome length) the simulation time will vary [1] since two different routes do not have the same overall length.

If the problem would only take into account truck costs it would be reduced to the traveling salesman problem. In that case, different simulation durations, thus different route lengths would not be a problem since finding the shorter route is the only performance criteria.

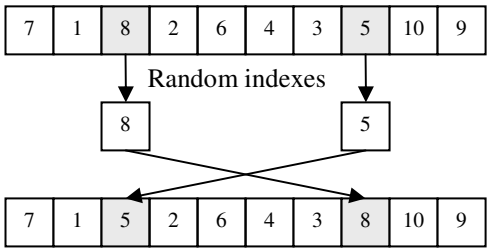

Figure 5. Mutation Operation.

In our case, because production halt costs are taken into account, the shorter routes are not always better since they might poorly satisfy the product match between factories (causing significant halt costs). Therefore, a longer route that suits well the factory's demands is likely to be better. Unfortunately, comparing the halt cost of a route that takes more time with the halt cost of a route that takes less time is not fair leading to an in-consistency problem. In order for the comparison to be relevant the halt cost needs to have accumulated over the same amount of time and so the simulation durations for two routes have to be the same. This raises an issue because having equal simulation durations means unequal chromosome lengths.

In order to fix this problem the fitness function (section 2) was modified such that it always keep a record of the smallest simulation duration corresponding to the current population. When evaluating a new route, the simulator dynamically compares the current simulation time with the smallest simulation duration recorded until then. If that time is reached the simulation ends without reaching all the destinations of the route. This is equivalent to having a non-uniform chromosome length but without explicitly specifying it to the genetic solver (the rest of the chromosome is still there but is not used). This concept is illustrated in figure 6. All routes have six destinations. The destinations that lie in wider rectangles are those that the truck needs more time to reach. It can be noticed that all scores are evaluated after the same amount of time, assuring that halt costs are fairly compared.

Now it is interesting to see if this modification has not lead to the same problem when comparing the truck costs. Since scores are evaluated after the same amount of time truck costs will be the same for all routes. Has this destroyed the merit of shorter routes? The advantage of having a shorter route is hidden in the fact that more destinations are visited in the same amount of time. If these

ISBN: 978-1-61804-023-7

destinations match the factory's product demand it will represent a significant advantage.

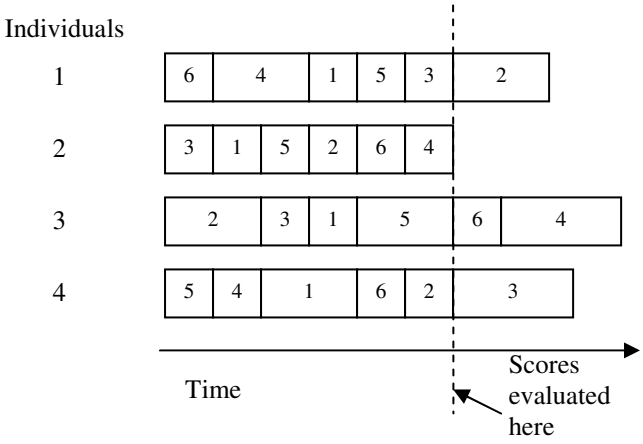

Figure 6. Adjusting the chromosome length

4 Results

The genetic optimization uses the following parameters: population size = 100 individuals, elite children = 2 individuals, cross-over children = 80% of population (except the elite children), mutation children = 20% of population, maximum number of generations = 100. Figure 6 illustrates the evolution of the overall cost obtained by the best individual in the population (best route for each generation). The trace does not illustrate the absolute value of the cost. It presents a relative cost computed as the ratio of the general cost and the simulation duration. As the population evolves, the simulation duration adapts (it is always equal to the smallest simulation duration of the current population set). Because the simulation duration changes the absolute value of the general cost does not reflect the improvement of the algorithm (mostly because the simulation duration tends to increase as short routes are rarely better than long routes). In turn, we use a relative cost that reflects the average performance per unit of simulated time.

Normally, a genetic algorithm that uses at least one elite child should have a monotonous cost function over the generations. This is because passing elite children from one generation to the next guarantees that the best performance of the next generation will be at least just as good as the previous one. It can be noticed that this is not the case for the trace in figure 7. This is the effect of having a chromosome of changing size. For example, one individual is considered to be elite because it has the best cost. However, most often, the chromosome of the individual does not participate with all its vector entries to the simulation (as explained in section 3).

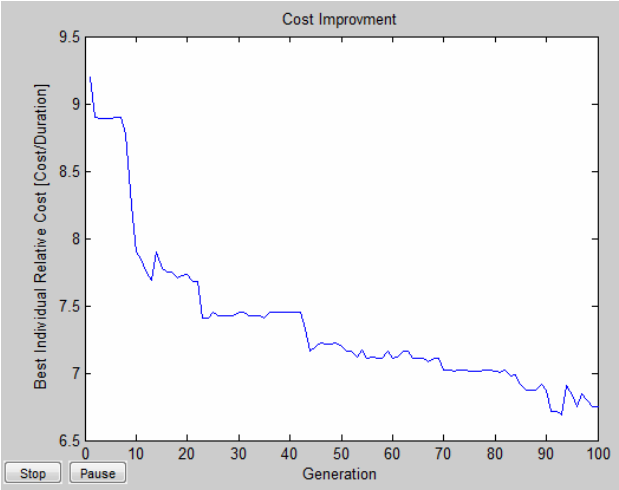

Figure 7. Cost Improvement by Genetic Optimization

If the simulation duration increases for the next generation more of the chromosomes entries might become active. If these new entries are not fortunate choices the individual can have a higher cost per unit of simulation time compared to the previous generation even though the individual is the same. Even so, it can be seen that the cost has a clear tendency to decrease and does so significantly over the 100 generations.

5 Conclusions and Future Work

This paper presents a new approach to implementing a genetic optimization on a transportation problem. Our current research aims towards modeling and optimizing a company from a management perspective. This model can be used as a sub-model of the transportation department and it can be integrated into the company model in order to perform a global optimization.

Acknowledgement: This work was partially supported by the strategic grant POSDRU 107/1.5/S/77265, inside POSDRU Romania 2007-2013 co-financed by the European Social Fund – Investing in People.

References:

[1] Chang Wook Ahn, Ramakrishna, R.S, A genetic algorithm for shortest path routing problem and the sizing of populations, *IEEE Transactions on Evolutionary Computation*, Vol.6, No.6, 2002, pp. 566-579.

[2] J. E. Beasley, P. C. Chu, A genetic algorithm for the set covering problem, *European Journal of Operational Research*, Vol.94, No.2, 1996, pp. 392-404.