I Krajowa Konferencja
Algorytmy Ewolucyjne
Murzasichle, 12-15.06.1996

THE SEARCH FOR THE BEST DECISION VARIABLES VALUES OF THE EVOLUTION PROGRAM

Marek Pawlak
Faculty of Industrial Engineering
Technical University of Lublin
36, Nadbystrzycka St.; 20-618 Lublin; Poland
E-mail: pawlak@archimedes.pol.lublin.pl

Abstract

This paper presents an evolution program implemented to optimisation of resource requirements in area of process planing and the search for the best values of its decision variables. The optimisation is being carried through on assumption of minimum process duration. The program has been applied to a practical case: the process of a track-laying vehicle repair. Owing to the search for the best values of decision variables, the results produced by the evolution program are considerable better in comparison with the results obtained from heuristic algorithms.

1. Introduction

In recent years genetic algorithms and evolution programs are more and more frequently applied to solving numerous optimisation problems. Among others, such problems as travelling salesman problem, non linear transportation problem, production scheduling, drawing a directed graph [12], [15], [16] have become their subject of interest. The characteristic feature of the mentioned problems is the fact, that effective methods of finding an optimum solution for real-world size tasks of this kind in an acceptable time have not been found yet.

Resource demand optimisation in network scheduling is a very well known issue. The problem consists in taking into consideration not only the technological sequence but also the necessity of optimum restricted resources utilisation, when setting the start and finish time for each activity. The publications on this subject were most numerous in the sixties (e.g. [5], [6], [7], [17]).

The problem of resource demand optimisation may be expressed in two ways:

1) Minimise the process duration, assuming a specified resource demand level;
2) Optimise the resource demand level, assuming a specified (minimum) process duration.

In these cases the restricted resources may be, for example: workers, machinery, energy.

2. Problem formulation

Assuming the specified (minimum) process duration, the activities should be arranged in such way that the resource demand would be, in a way, optimised (taking the technological sequence into account). The assumption is made, that for each activity i, its duration is known and, that the resource demand level is constant.

The concept of the optimum resource demand for such problem formulation needs to be explained herein. To simplify, we assume that only one kind of resource is required for all the activities. The optimum criterion may be formulated in two ways [13]:

Criterion 1

Calculate the average value of the required resources in the following way:

$$R_m = \frac{1}{T}\sum_i r_i \cdot d_i \qquad i = 1,\ldots,n \qquad (1)$$

Where: R_m - Average resource demand per a unit of the process duration time.

T - Process duration.

r_i - Resource demand of the activity i.
d_i - The activity i duration.
n - Number of activities in the process.
i - The activity number.

As a measure of the resource demand $R(t)$ irregularity we can assume the mean square deviation of the resource demand at the point of time t from the average resource demand per a time unit R_m.

$$\frac{1}{T}\int_0^T [R(t)- R_m]^2 dt$$

The optimum plan is defined as a plan that minimises the above value.

To simplify, minimisation of the above function can be replaced by minimisation of the following function:

$$\int_0^T R^2(t)dt. \tag{2}$$

Criterion 2.

Second idea of optimum resource demand can be obtained by assuming its measure as the maximum demand during the process:

$$\max_{t\in[0,T]} R(t) \tag{3}$$

The optimum plan is defined as a plan that minimises the above value.

In the further consideration both presented criteria are going to be applied. Criterion 1, because it takes into consideration level of the resource demand in each time unit of the process duration and it allows capturing the differences between various plan variants to the greatest extent. This is of huge importance for the chromosome estimation later on. Criterion 2 is also going to be used for the reason that it is demonstrative and practical. It allows, even "at a glance," to tell the differences in the resource utilisation in different plans, and also define the maximum number of the resources permanently involved in the process realisation.

3. Time constraints in process planning

While designing a process plan you need to take a series of constraints into consideration. They result from a fixed sequence of activities (each of which has got its set of predecessors) and from each activity's duration. In the case of networks, in which the activities are represented by the network nodes and the sequence conditions are represented by arrows, the constraints can be defined as follows:

$$t_{jr} \geq t_{ir} + d_i; \qquad i \in \Gamma_j^1; \qquad j= 1,...,n; \tag{5}$$

$$t_{jz} \leq t_{iz} - d_i; \qquad i \in \Gamma_j^{-1}; \qquad j= 1,...,n; \tag{6}$$

where: t_{jr} - Activity j start time.
d_i - Activity i duration.
Γ_j^1 - Set of activity j direct predecessors.
Γ_j^{-1} - Set of activity j direct successors.
t_{jz} - Activity j finish time.

Moreover, the start and the finish times must be within certain intervals:

$$t_{jr}^w \leq t_{jr} \leq t_{jr}^p \tag{7}$$

$$t_{jz}^w \leq t_{jz} \leq t_{jz}^p \tag{8}$$

where:

$$t_{jr}^w = \max_{i\in\Gamma_j^1}\{t_{ir}^w + d_i; 0\}; \qquad j = 1,...,n \tag{9}$$

$$t_{jz}^{w} = t_{jr}^{w} + d_j; \qquad j = 1,...,n \tag{10}$$

$$t_{jz}^{p} = \min_{i \in \Gamma_j^{-1}} \{t_{iz}^{p} - d_i; T\}; \qquad j = 1,...,n \tag{11}$$

$$t_{jr}^{p} = t_{jz}^{p} - d_j; \qquad j = 1,...,n \tag{12}$$

$$T = \max_{j} \{t_{jz}^{w}\}; \qquad j = 1,...,n \tag{13}$$

where: t_{jr}^{w} - The earliest possible activity j start time.

t_{jz}^{w} - The earliest possible activity j finish time.

t_{jr}^{p} - The latest acceptable activity j start time.

t_{jz}^{p} - The latest acceptable activity j finish time.

4. The evolution program

The structure of the evolution program presented below is no different from the classical one. The representation has a direct reference to the variant representation in its natural form. Genetic operators of mutation and crossover that base themselves on the introduced representation are utilised herein [8].

4.1. Representation

It seems that from the representation point of view the process planning is either a simplification or a particular case of the classical production scheduling problem [14]. Whereas in the classical production scheduling problem there must be respective information for each machine, for example: the machine m_i, performs the operation o_{jk}, upon the part p_k, in time from t_1 to t_2, in the case of process planning and one kind of the resource, for each activity the sufficient information is: activity a_i starts at the moment t_j. This particular representation is applied in [2].

activity i_0	activity i_1	.....	activity i_n
start i_0	start i_1	.....	start i_n

Fig. 1. Activity/Start representation [2]

In the instance of the presented herein evolution program the direct representation has been used in the following shape:

$$t_{1r}, t_{2r}, ..., t_{ir}, ..., t_{nr}$$

where: t_{ir} - is activity i starting time expressed as an natural number.

This is a simplification in relation to the quoted representation described in [2]. However, the introduced notation allows including all the solution elements as the operation's number is identified by the number's position in the string and the starting time is defined by the number's value.

4.2. Population Initialisation

In the discussed case the random method of creating the initial population has been chosen, however, the one that guarantees meeting constraints (7) and (8). This is achieved in the following way:

1. Calculate t_{jr}^{w} according to formula (9) for $j = 1,..., n$.
2. Calculate t_{jr}^{p} according to formula (12) for $j = 1,..., n$.
3. Calculate $t_{jrk} = t_{jr}^{w} + random(t_{jr}^{p} - t_{jr}^{w})$ for $j=1,...,n$ and $k=1,...,m$

where : j - Activity number.

k - Chromosome number.

It is worth noticing that the population initialised in such way meets the conditions (7) and (8) but may not meet the conditions (5) and (6), and the repair algorithm should be used.

4.3. Estimation Function

For minimisation problems, which is our case, it is necessary to apply such transformation that the individuals with lower estimation function would have higher fitness function value, and the other way round. There are many ways of meeting that requirement. Example solutions can be found in [3], [1], [9]. In our case, the γ transformation has been applied according to the following formula [2]:

$$g(x)=\frac{\max\limits_{x} f(x)- f(x)+ \gamma}{\max\limits_{x} f(x)- \min\limits_{x} f(x)+ \gamma} \tag{14}$$

where:

$g(x)$ - Fitness function value.

$\max\limits_{x} f(x)$ - Maximum value of the estimation function in a given generation.

$f(x)$ - Estimation function value for the individual x.

$\min\limits_{x} f(x)$ - Minimum value of the estimation function in a given generation.

γ - Coefficient.

The γ coefficient may have two functions. If its value is adequately low - from the (0,1) interval, then it prevents the situation when the denominator of the formula equals zero. However, it may have a different purpose, which is the fitness function calibration and toning down the differences between the population individuals. In such case the γ value must be adequately higher and it is a system parameter.

The population elements are being estimated from their fitness point of view in each step of the evolution program. Evolution programs usually operate in such way that fitter elements are more probable to move into the next generation or a fitter element is more probable to participate in crossover than a less fit element. In the presented evolution program fitter elements are more probable to move into the next generation and the parents' selection for crossover operation is entirely random. Function $f(x)$ is defined according to Criterion 1 (formula (2)).

4.4. Genetic Operators

Crossover

In the discussed example of evolution program, search for the best crossover operator has been given up and the simplest one-point crossover operator has been chosen. This lets us think that application of a more sophisticated operator is going to allow improvement of the solutions the system yields. The crossover process goes as follows:

1) Random parent 1 selection.
2) Random parent 2 selection.
3) Random crossover point selection.
4) Crossover.
5) Offspring replace parents.

This can be described as follows:

$t_{1rk}, t_{2rk}, \ldots, t_{prk}, t_{p+1rk}, \ldots, t_{nrk}$ - Parent 1

$t_{1rl}, t_{2rl}, \ldots, t_{prl}, t_{p+1rl}, \ldots, t_{nrl}$ - Parent 2

$t_{1rk}, t_{2rk}, \ldots, t_{prk}, t_{p+1rl}, \ldots, t_{nrl}$ - Offspring 1

$t_{1rl}, t_{2rl}, \ldots, t_{prl}, t_{p+1rk}, \ldots, t_{nrk}$ - Offspring 2

Unfortunately, this way of crossover does not guarantee that the results are acceptable, i.e. they may not meet conditions (5) and (6). In the case of the presented evolution program we pre-select the maximum number of R crossover repetitions that can be performed in a given

generation. The actual number of crossover repetitions is a random number from the interval $<0,R>$.

Mutation

As to the problem discussed herein, mutation is performed for every schedule activity and consists in hastening or postponing its start time. It is performed as follows:

1) We assume the mutation probability e.g. $m=0.01$.
2) All the genes in all the chromosomes are examined successively.
3) For each gene a number is selected at random from the interval $<0,1)$.
4) If the selected number is less or equal to m then:
5) We choose at random whether the mutation will be based on moving the activities in the schedule left or right. Both alternatives are equally probable = 0.5.
6) If the activity displacement does not disturb conditions (7) and (8), then it is carried through.

Two mutation variants have been examined in the system. The first variant consists in activity's displacement one time unit right or left in the schedule, i.e.:

$$t_{ikr} := t_{ikr} + 1 \qquad \text{or}$$
$$t_{ikr} := t_{ikr} - 1$$

The second variant consists in activity's displacement by a random number of time units right or left in the schedule, i.e.:

$$t_{ikr} := random(t_{irk}, t_{ir}^{p}) \text{ or}$$
$$t_{ikr} := random(t_{ir}^{w}, t_{irk})$$

This method of mutation realisation does not guarantee meeting conditions (5) and (6). Moreover, we can notice that the further the evolution process progresses, the less likely the mutation is in the latter evolution phases. This is a result of the fact that in the latter evolution phases it is relatively easier to break conditions (7) and (8). In such case, according to the algorithm, the mutation does not take place.

4.5. Repair of the solutions that do not comply with constraints

If the application of genetic operators of crossover and mutation results in solutions that do not comply with the constraints, then a penalty function is applied or a solution is repaired. In the discussed case chromosome repair has been applied i.e. they are corrected so that conditions (5) and (6) are met.

Two alternative ways of chromosome repair have been applied, and both of them have the same realisation probability $=0.5$.

The first method consists in checking condition (5) for each activity (gene in the population). If it is not met, then:

$$t_{jr} := \max_{i \in \Gamma_j^1} \{t_{ir} + d_i; 0\}$$

The second method is based on checking condition (6) for each activity (gene in the population). If it is not met, then:

$$t_{jr} := \min_{i \in \Gamma_j^{-1}} \{t_{ir}; T\} - d_j$$

Chromosome repair realisation is an iterative process carried through with regard to the selected variant, until no more chromosomes disturbing the constraints can be found. Each of the two methods produces different results, but they both guarantee producing solutions that meet conditions (5) and (6).

4.6. Evolution Process

The population initialisation is the preliminary step of the evolution program. It is followed by a sequence of steps containing genetic operators of crossover and mutation

mentioned before. In the discussed case the evolution process is carried through in the following steps:

1) Population initialisation.
2) Chromosomes estimation.
3) Storing the best chromosome.
4) Chromosomes selection (population actualisation).
5) Crossover.
6) Mutation.
7) Chromosomes repair.
8) The best chromosome introduction.
9) Chromosomes estimation.
10) Storing the best chromosome.

Steps 1 to 3 are the initial steps, and they are executed only once. Steps 4 to 10 are executed many times, depending on the formulation of the condition that stops the evolution process. The process uses the strategy of selecting the best chromosome in a generation, and next replacing the best, the worst chromosome or a chromosome chosen at random with the best one. This approach takes pattern by DeJong's elitist model. As Goldberg writes, the difference is that in the DeJong's model "if the best element from the i^{th} generation occurs not in the i+1 generation as a result of the selection process, then it is added to the i+1 generation as N+1 element, where N is population size".

We need to explain the actualisation procedure. It is optional. If it is not executed, the only selection is made by means of the procedure that transfers the best chromosome to the next generation. We have chosen the "roulette wheel selection" procedure [10, p. 36].

5. The method application results.

The discussed genetic program has been applied to optimise the resource demand in a repair process of a military vehicle. The process is described by a network of 266 activities and 499 relations. The network is structured according to the convention that the graph nodes represent the activities and the graph arrows represent the succession relations. The resources to be optimised are workers who carry on particular operations. Fig. 2. shows the diagram of resource demand before optimisation, assuming the earliest possible activities start times. Maximum demand for workers is 20 (estimation according to Criterion 2).

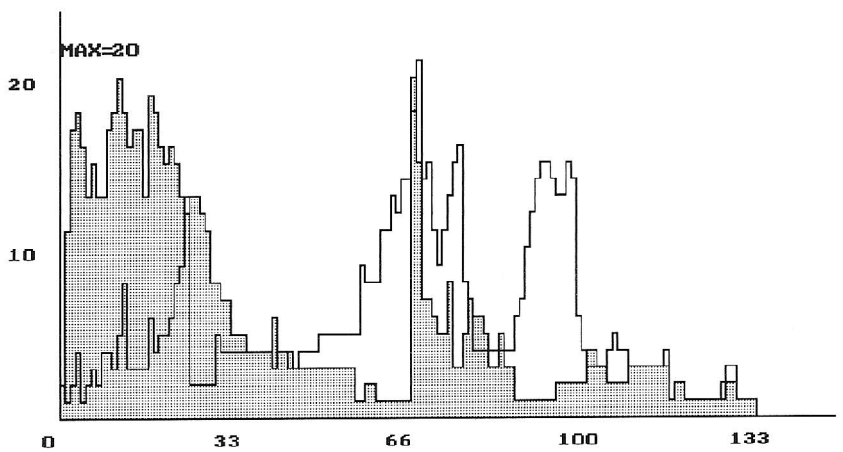

Fig. 2. The diagram of resource demand before optimisation, assuming the earliest possible activity start times.

Since for this particular network, the solution of the resource demand optimisation problem, that might serve as a reference is not known, then results of optimisation through heuristic algorithms were used as a reference. Two algorithms have been utilised herein. The

first one based upon book [13] and the other based upon paper [10]. Results of their application are shown in Table 1.

Method	Realisation Duration	Result according to Criterion 1	Result according to Criterion 2
Heuristic algorithm based upon [13]	40 seconds	5759	9
Heuristic algorithm developed by the author [10]	30,5 minutes	5439	8

Table 1. Results of heuristic algorithms application.
(All the calculation durations are for IBM PC 486 DX2 66 MHz class computer)

The research with evolution program have been carried through in such way that function (2) has been permanently applied as an estimation function. The evolution process has been interrupted when in the population there occurred even one individual that from the point of view of Criterion 2 (formula (3)) achieved the best value obtained from heuristic methods, that is value 8.

Following values have been distinguished as decision variables of the system:

1) Kind of mutation (two basic variants have been analysed herein).
2) Mutation probability.
3) Number of individuals submitted to crossover in a generation.
4) Way of introducing an element from the old generation into the new generation.
5) Selection realisation or no selection in the process.
6) Coefficient γ value in the fitness function.
7) Population size.
8) Conditions for process termination.

Fig. 3. shows a diagram presenting mentioned values.

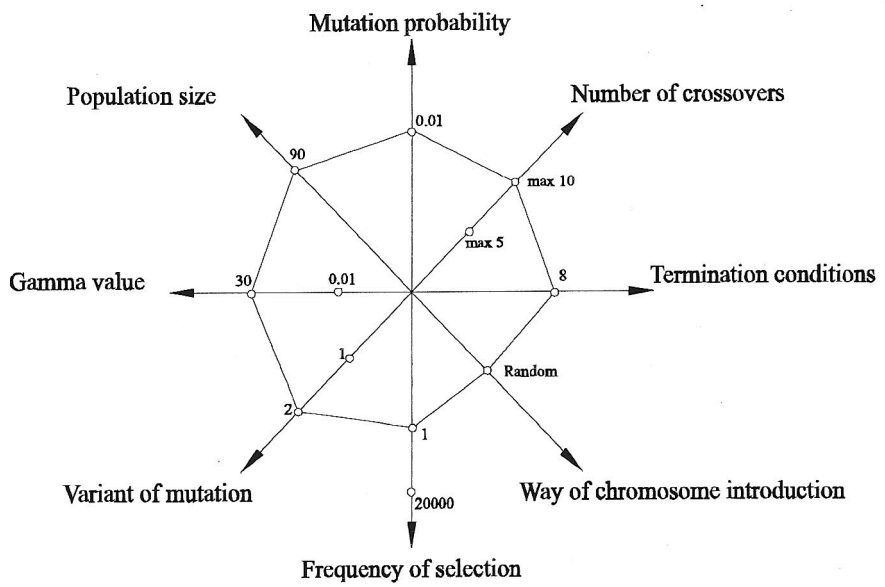

Fig. 3. System of the evolution process decision variables.

The first series of experiments has been carried out for the following decision variables values:

- Population size = 90 chromosomes.
- First way of mutation.
- Mutation probability = 0.02.
- Maximum number of crossovers in a generation = 5.
- The process carried through without selection.

- The best chromosome was transferred into the new generation and replaced a chromosome chosen at random.
- γ value = 0.01.
- Process interrupted when in the population there occurred an individual with estimation 8 according to Criterion 2.

The best result achieved after the first series of experiment is presented in Table 2 and Fig. 4 (compare also [11]).

Method	Number of generations	Realisation duration in hours	Result according to Criterion 1	Result according to Criterion 2
Evolution program	1063	1,06	5203	8

Table 2. The best result obtained after the first series of experiment using an evolution program.

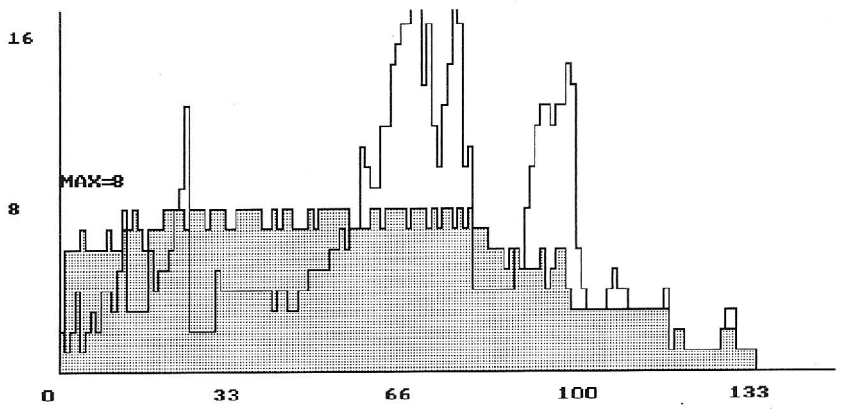

Fig. 4. Results of using an evolution program.

As it can be seen the obtained result matches the heuristic algorithms from the point of view of the Criterion 2, it is better from the point of view of Criterion 1 and worse from the calculation duration point of view.

In the next series of experiments second variant of mutation has been applied and the most advantageous values of mutation probability and number of crossovers in a generation have been sought, with constant application of the same process interruption conditions.

Average number of generations before the process termination has become a basis for estimation of quality of the mutation probability and crossovers number values system variant. Consecutive steps of parameters change together with average results obtained are shown in Fig. 5.

On the grounds of this series of experiments it is possible to state that the best results have been obtained for the mutation probability = 0.01 and maximum crossovers number equal to 10 (maximum 10 couples take part in crossover). As Fig. 5 shows, even a slight change of mutation probability as well as number of crossovers in relation to the most advantageous ones results in relatively considerable result deterioration.

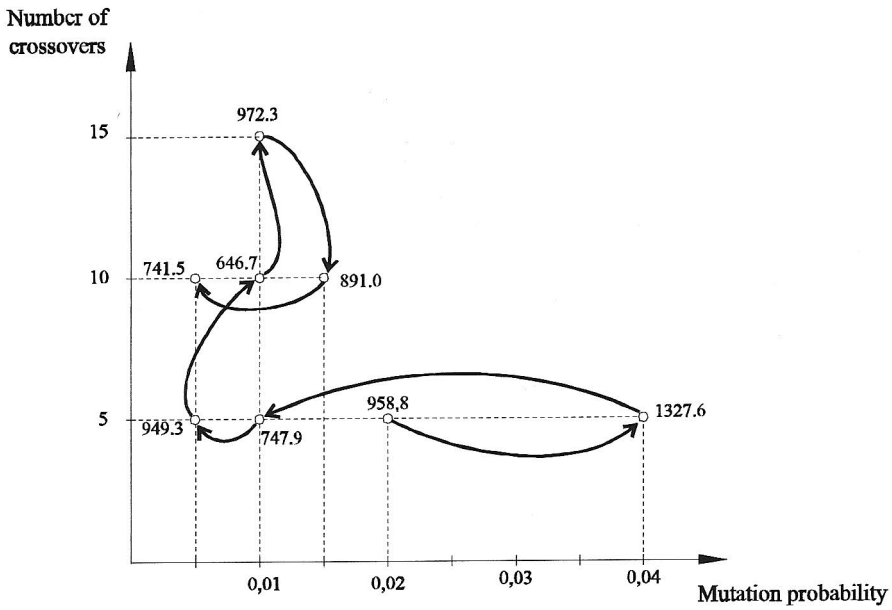

Fig. 5. Relation between the result and number of crossovers and mutation probability

In the next stage of research the simulation has been carried through with the application of selection; and the γ value influence upon the evolution program performance result has been analysed. Results of this analysis are presented in Table 4 and Fig. 6. Ten Evolution processes have been carried out in each cycle. Table 4 contains average generations numbers before the process termination.

Cycle number	γ value	Average number of generations	Average duration [hours]
1	0.01	531.2	0.47
2	10	531.5	0.47
3	30	390.5	0.35
4	50	500.1	0.45
5	70	426.2	0.38
6	90	430.5	0.38
7	110	474.9	0.42
8	130	552.9	0.49
9	200	582.3	0.52
10	250	516.2	0.46
11	300	628.0	0.56

Table 4. Influence of γ value on the evolution program execution research results

However this relation is not explicit, one might notice that introduction of selection allowed improving the average result (maintaining replacement of a chromosome chosen at random with the best one) and the best result has been achieved for the value γ=30. The result improved both from simulations number point of view and from the point of view of average duration required for obtaining value 8, according to Criterion 2. It is possible to state that average evolution program results are better in comparison with the heuristic algorithm. A disadvantage of the approach presented herein is that it uses processing duration as a value for comparison of the methods; since processing duration is influenced not only by the algorithm quality but also computer program implementation method.

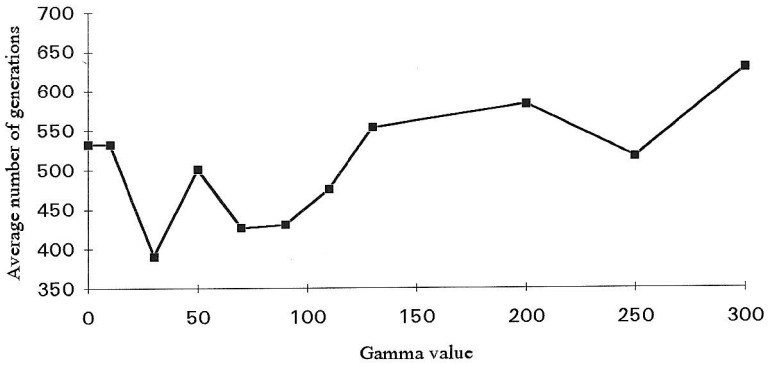

Fig. 6. Diagram of γ value influence on the evolution program execution

At the next stage of the research, result improvement from Criterion 2 point of view has been attempted. To achieve that the best hitherto decision variables system; that is mutation probability equal to 0.01, maximum number of crossovers equal to 10, replacing a chromosome chosen at random with the best one, selection in each generation, γ=30. Achieving value 7 from Criterion 2 point of view has been chosen as the criterion of process interruption.

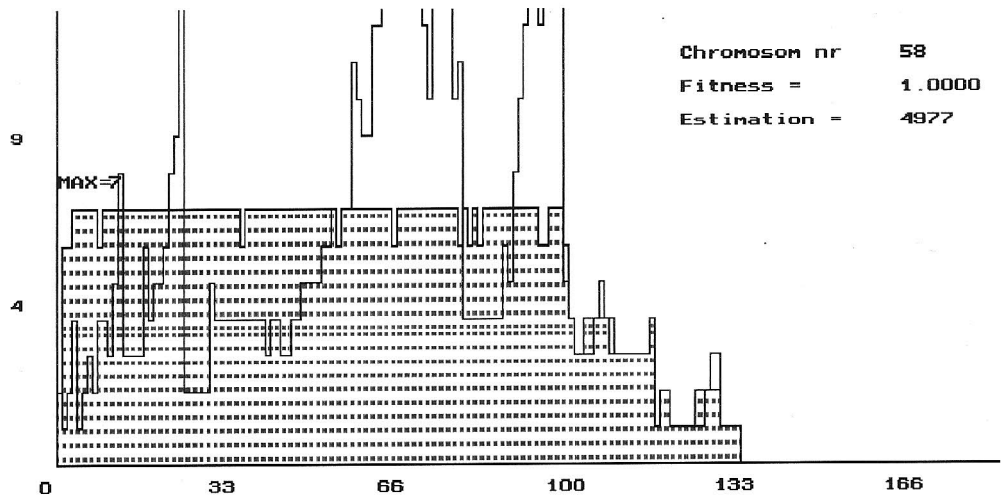

Fig. 7. The best result obtained from the evolution program.

The best result achieved after conducting a series of experiments is shown in Fig. 7 and Table 5.

Method	Number of generations	Realisation duration in hours	Result according to Criterion 1	Result according to Criterion 2
Evolution program	5951	5.31	4977	7

Table 5. The best result obtained from the evolution program.

This result has been achieved in 5951 generations. Estimation from Criterion 1 point of view is 4977, from Criterion 2 point of view it is 7. This is a considerable quality improvement in comparison with the results obtained from heuristic algorithms.

6. Conclusions and closing remarks.

This paper presents an evolution program for restricted resources demand optimisation in process planning. Results of applying the program for a specified process case are also

included. It is possible to state that the results offered by the evolution program are significantly better than results obtained from heuristic algorithms. Time required to obtain the same results, as well as estimation values according to the two criteria, have been used as the criterion of comparison of the evolution program with heuristic methods.

There an attempt has been made to select the most advantageous values of such variables as: mutation probability, number of crossovers, γ coefficient value. The course of selection of the most advantageous values of these parameters may be depicted as follows - Fig. 8.

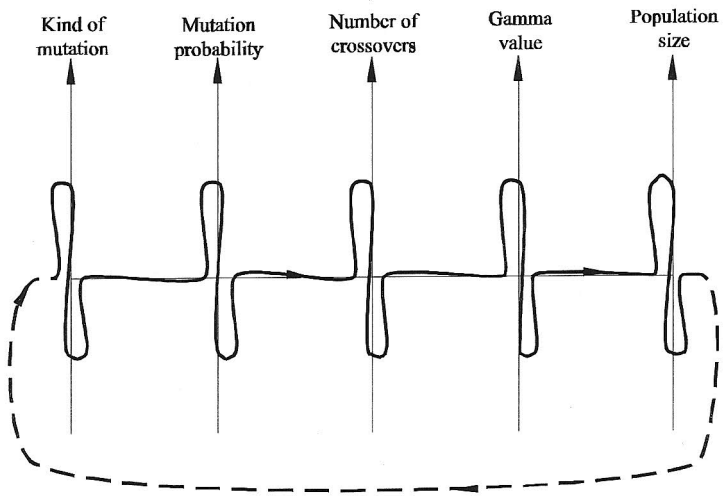

Fig. 8. Selection of the most advantageous decision values in the process.

Values for which the results are the most advantageous from the point of view of the assumed evolution program performance criterion have been analysed with regard to each of them. Population size has been included in the chart as an additional value that hadn't been examined. The loop indicating research repetition (dashed line) has been also marked; since it is impossible to state that fixing the most advantageous γ value will not, for example, result in necessity of changing the mutation probability. Therefore, the whole process should be an iterative one.

Does such iterative approach guarantee finding the most advantageous decision variables? There are no grounds to answer positively to such question. Such approach allows only finding local optimum. In order to find the global optimum another approach may be suggested; it consists in treating the decision values string as a chromosome, building a population, and searching for the most advantageous values of its parameters. We would then deal with a meta-population and be solving a meta-problem.

Of course the results and conclusions presented in this paper stand only for the problem and the example considered herein. Nevertheless, there will not be even a bit of coquetry in the statement that one may be fascinated with the results offered by this relatively new, and constantly searching for new application areas, method of evolution programs.

References

[1] Chuen-Lung Chen, Venkateswara S. Vempati, Nasser Aljaber, „An application of genetic algorithms for flow shop problems", *European Journal of Operational Research* 80 (1995) 389-396.

[2] Cheng R., Gen M., „Evolution program for resource constrained project scheduling problem", *Proceedings of the first IEEE Conference on evolutionary Computation.* June 27 - June 29 (1994) 736-741.

[3] Goldberg D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

[4] Heitkoetter, J., Beasley, D., „Guide to Evolutionary Computation", http://www.aic.nrl.navy.mil/galist, (1995).

[5] Kelley, J.E., „The critical path method: Resource planning and scheduling", in J.F. Muth and G.L. Thompson (eds.) *Industrial Scheduling*, Prentice-Hall, Englewood Cliffs, NJ, (1963) 347 - 365.

[6] Leifman L.J., „The main principles for optimal allocation of capacities", in Lombares H.J.M. (eds.), *Project planning by Network Analysis* - North-Holland Publ. Cy., Amsterdam (1969) 235-242.

[7] Lombares H.J.M. (eds.), *Project Planning by Network Analysis*, North-Holland Publ. Cy., (1969).

[8] Michalewicz Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, 1992.

[9] Murata T., Ishibuchi H., „Performance Evaluation of Genetic Algorithms for Flowshop Scheduling Problems", *Proceedings of the first IEEE Conference on evolutionary Computation*, June 27 - June 29, 1994 812-817

[10] Pawlak M., „Algorytmy suboptymalizacji zapotrzebowania na środki w sieciach MPM". I Krajowa Konferencja *Komputerowe wspomaganie w kształceniu technicznym* Politechnika Lubelska 1994, 109-113.

[11] Pawlak M., „Application of Evolution Program to Resource Demand Optimisation in Project Planning", *Proceedings of the IEEE International Conference on Evolutionary Computing*. Perth, Western Australia, 29 November - 1 December 1995, 435-438.

[12] Shen Ch., Pao Y., Yip P., „Scheduling Multiple Job with Guided Evolutionary Simulated Annealing Approach". *Proceedings of the first IEEE Conference on evolutionary Computation* 1994, 702-706.

[13] Suchowizki S. I., Radtschik I. A., *Mathematische Methoden der Netzplantechnik*, BSB B. G. Teubner Verlagsgesellschaft, Leipzik 1969.

[14] Syswerda G., „Schedule Optimization Using Genetic Algorithms", in L.Davis (eds.), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York (1991).

[15] Uckun S., Bagchi S., Kawamura K., Miyabe Y., „Managing Genetic Search in Job Shop Scheduling". *IEEE Expert* 1993 v. 8 n 5 Oct, 15-24

[16] Yamada T., Rosen B. E., Nakano R., „A Simulated Annealing Approach to Job Shop Scheduling using Critical Block Transition Operators", *Proceedings of IEEE International Conference on Neural Networks*. June 28 - July 2. 1994, 4687-4692.

[17] Zimmermann, H.J., *Netzplantechnik*, Walter de Gruyter. Berlin, New York, 1971.