Simulation of production systems, optimalization, genetic algorithms,

Peter ŠKORÍK[*], Milan GREGOR[**], Andrej ŠTEFÁNIK[***]

ARTIFICIAL INTELLIGENCE TOOLS IN SIMULATION AND OPTIMIZATION OF PRODUCTION SYSTEMS

Abstract

This article deals with solution developed as a cooperation of Industrial engineering department, University of Žilina and Central European Institute of Technology (CEIT SK). Proposed solution involves simulation with support of virtual reality for searching of engineer-accepted manufacturing system state (so called "optimal"). Article includes basic information about evolution methods and genetic algorithms. Authors own algorithm, which is based on use of genetic algorithm, is used for optimization. Outcomes compares the speed of convergence of chosen Witness optimization algorithms and authors-developed algorithm. Comparison is presented on project from industrial praxis.

1 MODELING AND SIMULATION

Modeling and simulation have been known for quite long time period. Their success is connected mainly with modern informatics technologies. But in industrial praxis they are not recognized well and are connected only with huge enterprises and corporations. The consequences of wrong innovation or optimization decisions may persist for long time and current industries cannot afford that risk. Modeling and simulation are tools to verify functionality, capacity and main parameters of production system in advance of real production, check the logics of production during planning stage and allow to reduce bottlenecks which can cause the losses connected with additional system adaptation and preproduction period.

Simulation as a statistic-experimental tool is based on performing chosen experiments on the model of real production system. As the number of possible experiments is huge (due to the combinations of system factors) and that it is necessary to perform each experiment several times (statistical significance), the selection of optimization algorithm is crucial.

[*] Ing. Peter Škorík, PhD., University of Žilina, SjF, KPI, e-mail: peter.skorik@fstroj.uniza.sk
[**] Prof. Ing. Milan Gregor, PhD., ŽU, SjF, KPI, e-mail: milan.gregor@fstroj.uniza.sk
[***] Ing. Andrej Štefánik, PhD., CEIT SK, e-mail: andrej.stefanik@ceit.eu.sk

2 OPTIMIZATION IN SIMULATION

Although optimization and simulation (logistic and manufacturing systems) have their own histories, the implementation of optimization in simulation has been developing only in last decade. Two most important publications from the field of computer simulation [1],[6] (Law proclaims 70 000 worldwide) included optimization in the 2000 edition (chapter 12.4, 12.6). Simulation optimization gives structural approach to set the optimal value of the input factors, where the optimum is measured by the function of output variables (parameters) from simulation model [8]. As the optimal value of the optimization function cannot be measured directly, but as a result of simulation runs, it is necessary to have the simulation model built to use simulation optimization. Simulation model is function (which exact form is not known), which evaluates set inputs. That means, the time period to perform the simulation run is very time consuming in comparison with optimization itself (setting the input factors for the next run).

As Banks [1] states, even if the simulation run is quite simple, optimization could take long time to perform due to big amount and variety of types of input variables and due to the fact, that we know only a little about the objective function.

3 EVOLUTIONARY APPROACH - GENETIC ALGORITHMS

These facts aim the research in this area into use of heuristics based on genetic algorithms, that means optimization techniques which can perform optimization effectively and quick without further information about the objective function (reaching results which are close to the optimum, not the optimal solution [2]). Despite the fact, that the research in heuristics is not done, they are effectively used in praxis mainly for its reliability in searching for near-optimal results (mathematical, combinatorial tasks, transport problems, robot management).

Genetic algorithm belongs to the evolutionary algorithms set. The main application of GA is optimization, even GA is not optimizer (the optimal solution is not guaranteed). In general, solving a task means solving a problem. This problem has its own set of solutions, necessary to search in. In general we need the best solution, so it is possible to say, that we are optimizing.

```
BEGIN                                    /*genetic algorithm */
          generating the start population;
          fitness enumeration for each individual;

          WHILE NOT end DO
          BEGIN                          /* new population creation*/

                    FOR (population size/2) DO
                    BEGIN                /*reproduction cycle*/
                    two old-generation individual selection for matting;
                                                  /*best individual selection*/
                    recombination individuals into offspring;
                                                  /*operators application*/
                    fitness enumeration for the offspring;
                    putting offspring to the population ;
                    END
                    IF population is converging THEN
                    end:= TRUE;
          END
END
```

Fig. 1: Basic GA principle

GA search the space of possible solutions (combining direct and stochastic search), working on the set of possible solutions - population (input factors stored in unique form) in several cycles - generations (developing subset of solutions) using the nature selection methods: inheritance and natural selection, which helps individuals to evolve. Exact and particular setting of these methods set the affectivity and speed of GA optimization. Therefore also possibility, will and effort to use simulation not only for "what if" analysis, but also for real use of simulation for optimizing real production and logistics systems [6].

Advantages of GA in comparison with common used optimization algorithms are:

- possibility to parallel enumeration of objective function,
- easy implementation of simulation run results statistical comparison,
- easy implementation of simulation run results approximate estimation, without real need to perform simulation run,
- suitability for use in applications, where exact form of objective function is not known (discrete-event simulation).

According to Mitchell [7], it is appropriate to use GA if:

- searching space is huge,
- searching space is not smooth and unimodal,
- structure of searching space is complicated,

- it is not necessary to reach the optimal solution, but near-optimal solution in acceptable time period.

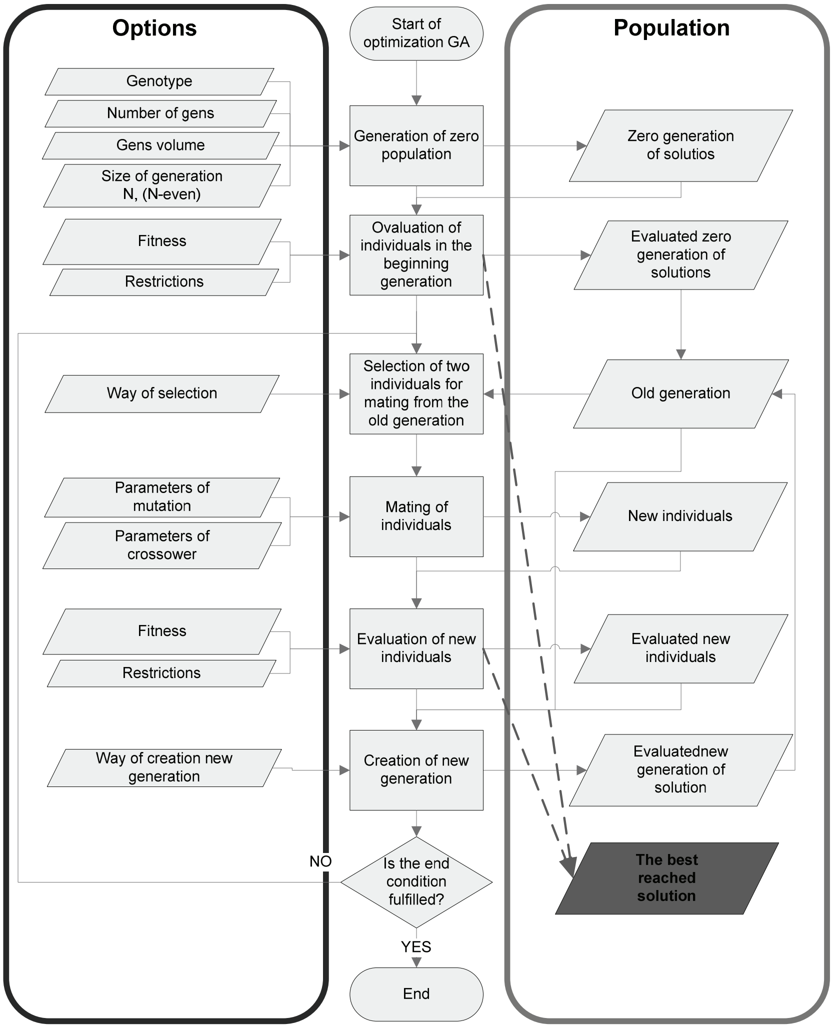
On the basis of the fact, that simulation fulfils all necessary assumptions and if we include advantages stated earlier, it is possible to consider the use of GA for optimization in simulation as an appropriate solution.

4 CURRENT SITUATION

Genetic algorithms are based on the idea of evolution by Darwin. Finding the optimal (or sufficient) solutions takes form of competition within the population of gradually evolved solutions. In order to determine which individuals are more likely to participate in forming new generation of solutions, it is necessary to define a way how to clearly evaluate this ability or its fitness value must be defined. Individuals with higher fitness value are more likely to survive and thus participate in forming new generation. Using the techniques of crossing and mutation a new generation of individuals is created, which partly takes the features after parents and partly the new features defined by a mutation in reproduction. Plenty reproductions of this evolutionary cycle later (tens to hundreds) there is population whose individuals have a high rating and will include a satisfactory (or optimal) solution to the problem. Each individual represents a unique solution to the problem. This solution is encoded by genes, which the individual is composed of and which are arranged linearly. For example, each gene represents one factor of production. Two individuals of the same type will have the i.th gene position of the same factor type (the number of machines, the size of the lot, ..).

On the contrary: by the composition of genes (factors) there is an individual (settings of the production system), the composition of individuals arises generation (more options of the system state) and a set of generations is the population (all states of manufacturing system used in a simulation optimization). The function of genetic algorithms is demonstrated in fig.2 and described by using the basic shape of Holland as follows:

- As the first step in GA we consider creation of starting (zero) generation of solutions. Starting generation is formed on the basis of information about genotype, the number of genes used to encode the state of the production system, scale factors (the value entered in the genes) and the size of the generation. The first generation of individuals is created randomly.
- Subsequently the individuals of starting generation are rated. The assessment will use a fitness function to assign each individual a clear assessment. Account must be taken of the restrictions associated with the real state of the system. Also the best solution achieved so far is defined. So the "old" generation is created.



Options
Start of optimization GA
Population
Genotype
Number of gens
Gens volume
Size of generation N, (N-even)
Generation of zero population
Zero generation of solutios
Fitness
Restrictions
Ovaluation of individuals in the beginning generation
Evaluated zero generation of solutions
Selection of two individuals for mating from the old generation
Way of selection
Old generation
Parameters of mutation
Parameters of crossower
Mating of individuals
New individuals
Fitness
Restrictions
Evaluation of new individuals
Evaluated new individuals
Way of creation new generation
Creation of new generation
Evaluatednew generation of solution
NO
Is the end condition fulfilled?
The best reached solution
YES
End

Fig. 2: Genetic algorithm principle

- The next phase is the evolution itself - creating a new generation. In this type of GA, we choose two individuals from the old generation, which will be designated for mating. The way of selecting individuals is performed by the "roulette wheel" rule. This rule says that every individual has a probability of selection for mating directly proportional to the evaluation degree - fitness (f_i). Then the probability of selecting any individual p_i, where N is the size of generation, is given by the relationship (1).

$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}; i \in \{1, \ldots, N\} \qquad (1)$$

- Selected individuals are mated and offspring is formed. This process consists of processes:
    - the crossover - the alternation of genetic material between individuals. We choose a gene from which genetic material is exchanged (single crossover). Two offspring arises, where each of them takes part of the genetic material from first parent and part from the second parent,
    - mutation – genes of individuals are changed at random (low probability), and thus premature convergence of generations is prevented.
- The offspring is then re-evaluated by the fitness function. If the offspring have a higher rating than the best solution achieved so far, they becomes the best solutions obtained so far.
- Based on the chosen method the new generation is created from the old one. The simplest way is that, in which the first N individuals (N - size of generations), with the largest value of fitness function are selected.
- The last step in GA is verification of the fulfillment of the end conditions. We have chosen the easiest option – termination after selected number of generations. If it is not reached the maximum number of generations, it is returned to step selection of individuals for mating, when the maximum number of generations is achieved, optimization is completed, and the best individual value achieved so far becomes optimum.

5 GA APPLICATION FOR OPTIMIZATION IN SIMULATION

On the example in previous part authors tried to show how the GA works. Therefore the example was as simple as possible. In simulation of manufacturing and logistics systems the previous simple example has only limited effectiveness, so it was necessary to use methods described in this chapter.

- Coding - significant part of GA types works with information stored in binary code. In simulation we do not recommend to use binary code, because there may be problem with coding information to and from binary code. As acceptable solution authors recommend coding with integer values, as the values of variables in simulation are mostly integer. This change allowed us not only to make the coding simpler, but also have a more natural outlook on the evolution of the results. The problem with more factors in simulation was solved with the use of genes. Each factor value was stored in separate gene, the individual is created form set of genes, so the individual stores all necessary data for the simulation run.
- fitness function - as we are using GA in simulation, the fitness function is substituted with simulation model. The values of factors the individual stores are used as an input for fitness enumeration, the value of fitness equals the value of the objective function

(it is enumerated at the end of simulation run from the values of output parameters) in the simulation model. The fitness enumeration is in fact the simulation run progress.

According to the fact, that simulation optimization (performing the simulation run) is very computer-performance demanding, authors consider parallel enumeration of fitness function on independent computers as a crucial aspect of the GA optimization. It is possible to speed up the optimization several times with the use of parallelization.

Tab. 1: Individuals selection methods comparison

Individual	**J_1**	**J_2**	**J_3**	**J_4**
Enumeration	12	3	2	1
Probability p_i	66,7%	16,7%	11,1%	5,5%
Order	4	3	2	1
Probability p_i	40%	30%	20%	10%

- Individuals selection method – experiments performed with GA showed fact, that the population of individuals used to converge. The reason was, that one superb individual was used on most matting actions over the population, so due to his "matting-strength" the genetic material got degraded. In this cases we recommend:
    - "rank selection" method – individuals are sorted according to their score. This sorted list determines whether individual is used for mating. Even if the diversity of score is huge, this method flatten the score. Mating probability can be determined from formula (2), where N is the generation size, the comparison of common and rank selection method is shown in tab.1. Graphic representation of probability for both methods is shown on fig. 3.

$$p_i = \frac{i}{\sum_{j=1}^{N} j} = \frac{2i}{N \times (N+1)}; i \in \{1,\dots,N\} \tag{2}$$

    - „tournament selection“ method – this method randomly choose set of individuals (k is the number of individuals in the set, k>1). Individuals in this set are compared by their fitness. Individual with the best score is chosen to further mating. The higher the "k" value is, the bigger "selection pressure" is made, therefore we recommend use k=2.

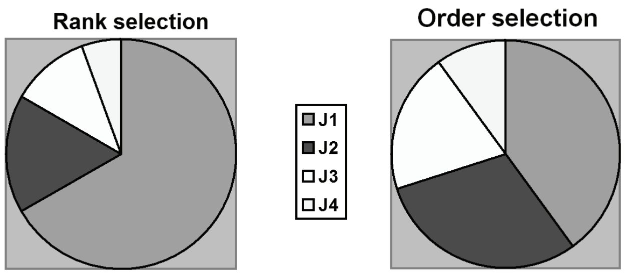


Fig. 3: Individuals selection methods comparison

- Genetic operators - to experiment with efficiency of different GA settings, we recommend to take into account also possible changes in the genetic operators. We also recommend to vary the above-mentioned single point crossover operator with the operator of multi-point crossover (for the crossing, it is used more points at which individuals can exchange genetic material). As in the simulation, we propose to use particularly coding using integers (real numbers), we also recommend to include to cross-over genetic operators as follows :
    - the arithmetic crossover - a new gene is formed by the arithmetic mean of parental genes,
    - the geometric intersection - a new gene arises with extraction of conjunction of parents genes,
    - extending the crossover - the difference between the parents values are added to a larger one, subtracted from the lesser.

  By analogy, we propose to vary mutation operator as follows:
    - the random replacement - replacing the original gene with a random value,
    - the additive change - adding a randomly generated value to the original value of the gene,
    - the multiplicative change - multiplying the original value with randomly generated number.
- Restrictions - we consider it as a very important element of GA. They reduces the scanned area, also excludes the solutions, which are not suitable for real production system. Although there are many approaches to incorporate constraints into the GA, if the nature of the simulation is considered (long duration of fitness evaluation), we recommend the use of methods which excludes insufficient results from search space (there is no need for simulation runs, which simulates a poor solutions). We recommend to solve the exclusion with the penalty method, where poor solutions are assigned with low fitness value. With this penalization, it is guaranteed for the poor solution not to participate in the creation of a new generation. In special cases, we consider it expedient to create a decoder which will transform unsuitable solutions into suitable according to specific algorithm. Given the complexity of the second approach we recommend it to be used only in cases where the method of penalties is not sufficient.

6 SOFTWARE FOR SIMULATION OPTIMIZATION

As mentioned earlier, present commercial simulation software includes also an optimization module, which is always designed "user friendly". But this fact creates the paradox, when the optimization packages are easy to use, but practically it is impossible to adjust the various parts of the algorithm itself - it loses the possibility of "upgrading". Also it should be noted, that the vast majority of optimization software is built either as a totally enclosed (OptQuest - black box, we do not know how it performs optimization. We know only what methods can be used, we do not know specifically which method in which case), or partially closed (we know exactly which method is used, we cannot change the core of that algorithm to improve it).

7 CREATING SOFTWARE FOR OPTIMIZATION USING GA

For solving the problem, authors created new software that is capable of communicating with two types of simulation software, is able to use genetic algorithms to vary factors for simulation and allows to set the parameters of genetic algorithms according to the needs of simulation optimization.

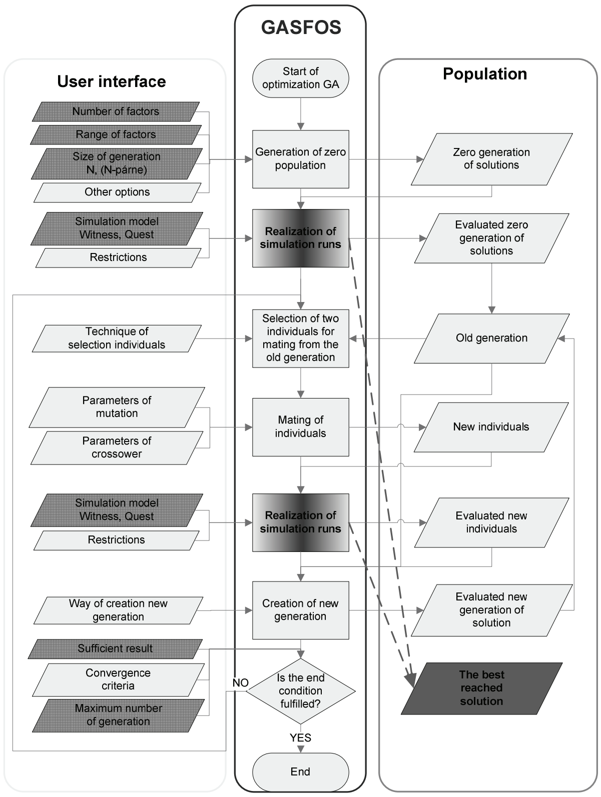

GASFOS
User interface
Population
Start of optimization GA
Number of factors
Range of factors
Size of generation N, (N-párne)
Other options
Generation of zero population
Zero generation of solutions
Simulation model Witness, Quest
Restrictions
Realization of simulation runs
Evaluated zero generation of solutions
Technique of selection individuals
Selection of two individuals for mating from the old generation
Old generation
Parameters of mutation
Parameters of crossower
Mating of individuals
New individuals
Simulation model Witness, Quest
Restrictions
Realization of simulation runs
Evaluated new individuals
Way of creation new generation
Creation of new generation
Evaluated new generation of solution
Sufficient result
Convergence criteria
Maximum number of generation
NO
Is the end condition fulfilled?
YES
End
The best reached solution

Fig. 4: GAsfoS software structure

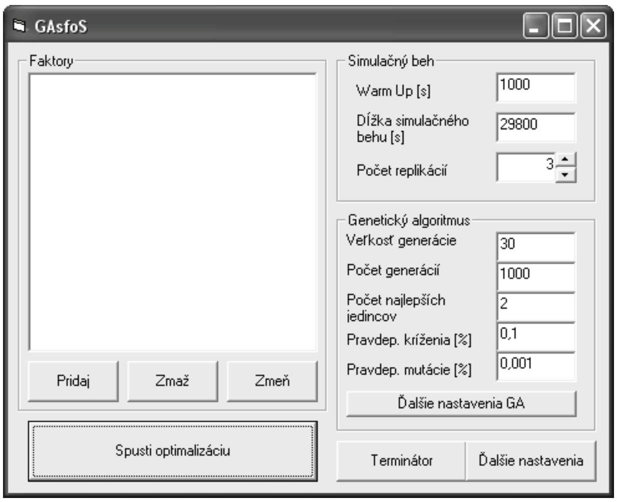
Software is called GAsfoS, it is standalone application, which allows to vary settings of genetic algorithms, input factors and simulation runtime. The structure of created software is shown on fig.4.

Software is logically divided into three main parts:

- user interface – fig.5, provides the collection of user settings,
- communication interface – provides communication between GAsfoS and simulation software,
- GA core – performing the optimization using GA.

In the user interface it can be easily entered necessary data (fig. 4, blue charts):

- input factors and their spectrum,
- simulation run settings,
- genetic algorithm settings.



Simulačný beh

Warm Up [s]	1000
Dĺžka simulačného behu [s]	29800
Počet replikácií	3

Genetický algoritmus

Veľkosť generácie	30
Počet generácií	1000
Počet najlepších jedincov	2
Pravdep. kríženia [%]	0,1
Pravdep. mutácie [%]	0,001

Fig. 5: GAsfoS user interface

Other values are default, and if necessary, they can be changed in subordinate windows.

Communication interface had to be created according to the need of passing factors into the simulation model, controlling its running and collecting of necessary data back to GAsfoS for the optimization. Communication interface has been also used for communication between the simulation and the GA core.

Communication is based on Socket (Quest), ActiveX (Witness) interface, whereby developers of simulation software allowed their control by external programs. To control the simulation, GAsfoS uses BCL (Quest), WCL (Witness) language commands.

User interface and communication interface has been created by using the programming language Visual Basic 6.0.

GA core was created by GALib. GALib is a library that supports work with genetic algorithms, programmed in C++. Its author is Matthew Wall (MIT). For academic purposes is freely available.

GALib is modular and therefore it is possible to compile genetic algorithm with the desired behavior. Due to the fact that we set up the modules properly and used already pre-programmed tools of GALib, it is enough to enter the input data, to determine the fitness function (target function) and evolving is performed automatically.

In fitness function, GAsfoS creates a command to communication interface to perform a simulation run (according to selected settings) with the values of the factors, that was determined by GA. As the value of fitness function the value of target function, calculated at the end of a simulation run, is returned to GALib.

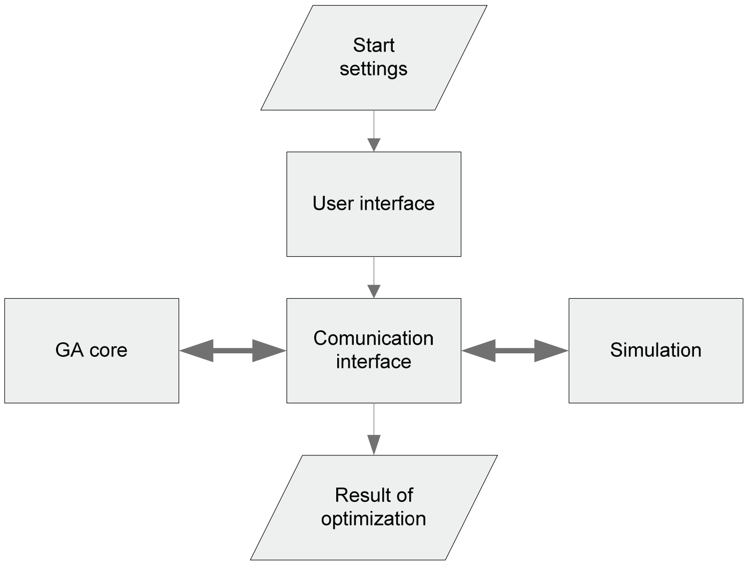


Fig. 6: Principle of GAsfoS software

How GAsfoS works is shown on fig.6. After entering required data in the user interface, optimization starts. For proper functioning it is necessary to run the required simulation software, which has already loaded the required simulation model. GA core creates starting generation. This generation is gradually evaluated in simulation software. Based on results of all individuals evaluations in the generation, pair of new individuals is created by using roulette wheel rule and genetic operators. This pair of new individuals is put into the old generation instead of a pair of the worst evaluated individuals. After each new generation the statistics are recorded and evolution continues until the end condition is fulfilled - sufficient value is reached, population converged or the maximum number of generations is exceeded.

8 SIMULATION OPTIMIZATION BASIS

Due to the confidentiality agreements it is not possible to show the real values of the simulation factors. Only general information are discussed below:

- The simulation model:
  - 7 different factors (26 800 different combinations):
    1. size of the transport batch of the problematic product - the gear box,
    2. -- 5. signal levels in selected stores in the workplace problem,
    6. workplace 9 function logic (the inclusion of the conveyor, respectively his disablement),
    7. number of workers for material supplement.
  - The management of production according to forecast 1 (advantageous and reliable production plan).
  - The objective function includes the costs associated with the number of workers (30% weight factor), production rate (30% weight factor) and the number of stocks (40% weighting coefficient).
  - The simulation time: 480 hours (work week).
  - The warm-up time: 4 hours (the time needed to stabilize the values of monitored factors).
  - The number of replications: 5.
- Genetic algorithms:
  - The roulette wheel rule.
  - Using the encoding with integer values.
  - The maximum number of generations - 400.

Tab. 2: Experiments variants

	Option 1	**Option 2**	**Option 3**	**Option 4**	**Option 5**	**Option 6**
Mutation probability	0.001	0.001	0.001	0.001	0.1	0.1
Crossover probability	0.9	0.9	0.9	0.9	0.9	0.9
Number of generation	50	50	50	50	50	50
Population size / generation	10	10	30	30	30	30
Number of new individuals / generation	2	4	2	4	2	4

9 RESULTS OF SIMULATION OPTIMIZATION

To compare the performance of genetic algorithms, we have conducted several experiments. Settings of individual experiments are listed in table 2.

Option 1 was the basic variant, from which we derived the changes under interim results. We have experimented mainly with the change of the number of individuals within one generation, the number of new individuals within one generation and the probability of mutation. Results of individual experiments are shown on fig.7. It is possible to see, that the best results are brought by the genetic algorithm set according to 2nd option.

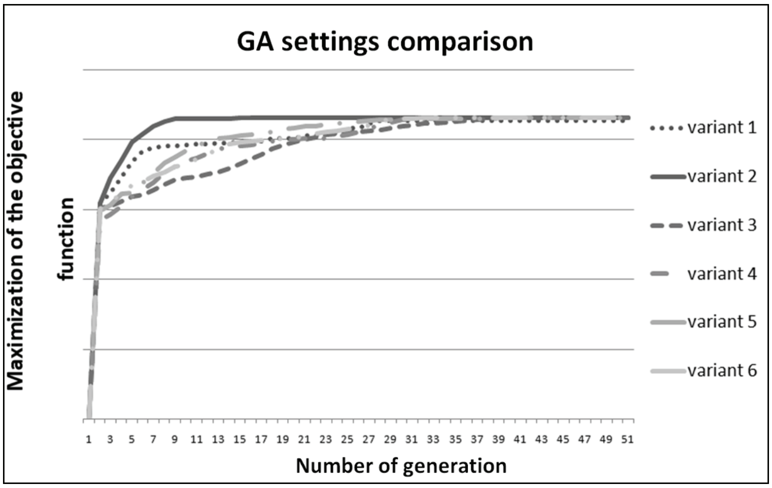


Fig. 7: GA efficiency comparison graph

We have also performed experiment to obtain the external performance of the best settings option (2nd option) of our optimization tool with a commercial tool Optimizer. We compared the result achieved and the number of necessary simulation runs to achieve this result. On fig.8 we are specifying the maximum values obtained by Optimizer software and the average value of the generation of solutions obtained by GAsfoS in the course of simulation runs. One simulation run was considered as one generation in Optimizer, so it was necessary to approximate the curves to make them comparable (GAsfoS performed 4 simulation runs within one generation). As we can see, in this case the results of GAsfoS and Optimizer are comparable, GAsfoS converges to the semi-optimal solution evenly.

Since genetic algorithms works on the base of probability and estimation, the chance to find solution with high-value of objective function relatively quick is high.

To compare the speed of reaching the semi-optimal solution (Optimizer, GAsfoS), another graph was made. As we can see on fig.9, GAsfoS converges to the optimum evenly, from the second simulation run even performs better than Optimizer.

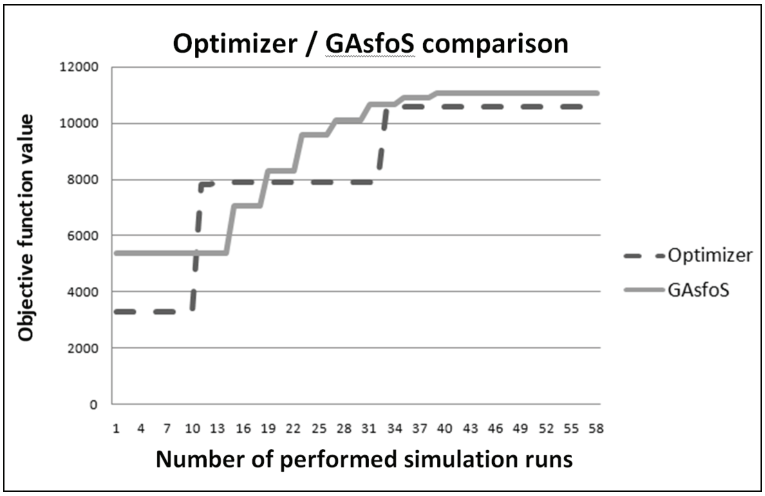


Fig. 8: OptQuest vs. GAsfoS comparison graph

10 GASFOS PERFORMANCE EVALUATION

The results of the experiments show that our genetic algorithms optimization software GAsfoS is without problem ready to use in simulation optimization.
Since the early stage of development, it is remarkable that the software reaches high values of objective function so quickly in the beginning of optimization. We find this feature as a crucial for simulation, since the goal for simulation optimization is to perform minimum number of simulation runs, which are computationally most challenging component of simulation optimization.

Weakness which is known so far is mainly the early convergence of GAsfoS, which deny looking up for the higher values of objective function. In the simulation, however, it is required a satisfactory outcome, not an optimal one, so this weakness is only relative disadvantage.
The maximal known value of the objective function is 22,300, the maximum value reached by GAsfoS was 21617. The authors believe that in the course of software development of GAsfoS, the problem of premature convergence will be successfully solved and GAsfoS will be valuable and reliable optimization tool that will be usable for variety of simulation software.

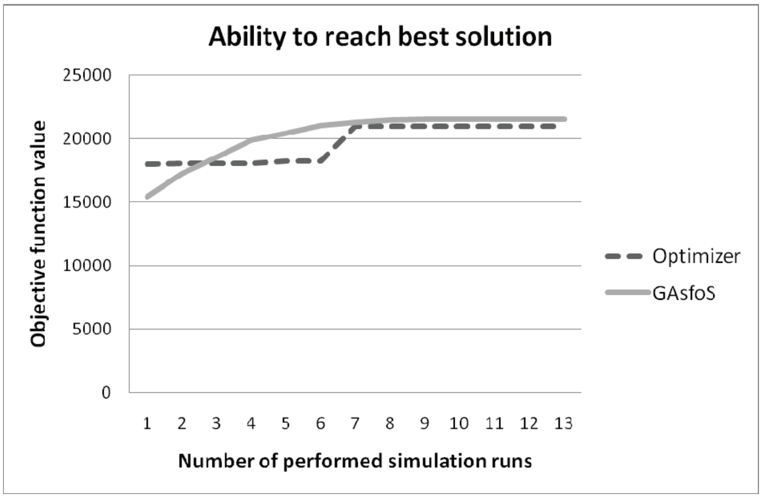


Fig. 9: Speed of search for the best solution

11 CONCLUSION

The area, which significantly affects the use of simulation in practice is optimization. It is not used often in the simulation, mainly due to weaknesses in the optimization algorithms. The possibility for the implementation of new techniques is in particular in the use of genetic algorithms, for which there is not an open simulation platform, allowing "tuning" them for their higher efficiency. Commercially available tools are designed as a "black box" and do not give room for use them in other areas of PLM.

Therefore, as soon as possible, the research activities should be focused in the area of creating an open platform for developing, debugging and application of genetic algorithms in simulation of manufacturing systems. The open platform is also important to use genetic algorithms as a part of the research tasks and projects run by Department of Industrial Engineering and research institutes within the University of Žilina (Central European Institute of Technology - CEIT SK).

This system should be designed to determine the impact of adjustments to the performance of genetic algorithms for optimization in the simulation of manufacturing and logistics systems. It offers an effective simulation optimization method for improving the productivity indicators of industrial enterprises and economic development in Slovakia. With appropriate parameters, the optimization will be more efficient, will reduce the time required to optimize and thus will improve the possibility of using optimization in practice.

This research was supported by Slovak Research and Development Agency under contract no. APVV-0594-07.

Literature

[1] BANKS, J. – CARSON, J.S. – NELSON, B.L. – NICOL, D.M. : Discrete Event Systems Simulation. 3rd edition. Englewood Cliffs, 2000.

[2] FU, C., M.: Optimization for Simulation: Theory vs. Practice In: INFORMS Vol. 14, No. 3, Summer 2002 pp. 192–215.

[3] FURMANN, R.: Modern methods of industrial engineering. In: 12th. National productivity conference – conference handbook, Slovak productivity center, Žilina, 2009, s.60-66, ISBN 978-80-89333-14-1.

[4] FURMANN, R. – KRAJČOVIČ, M.: Interactive 3D Design of Production Systems. In: Digital Factors 2009 – Workshop Handbook, SLCP, Žilina, 2009, s. 28, ISBN 978-80-89333-08-0.

[5] CHUNG, Ch. A.: Simulation Modeling Handbook. A Practical Approach. New York, CRC Press, 2004, ISBN 0-8493-1241-8.

[6] LAW, A. – KELTON, D. : Simulation, modeling and analysis, MGraw-Hill, New York, 2000, ISBN 0-07-100803-9.

[7] MITCHELL, M.: An introduction to genetic algorithms. Cambridge, MA: MIT Press, 1996.

[8] SWISHER, J.R. – JACOBSON, S.H. – HYDEN, P.D. – SCHRUBEN, L.W.: A survey of simulation optimization techniques and procedures. In: Proceedings of the Winter Simulation Conference, Orlando, USA, 2000.

[9] GREGOR, M. – ŠTEFÁNIK, A. – FURMANN, R. – ŠKORÍK, P.: Virtual manufacturing in research and industry. In: 9th IFAC Workshop on Intelligent Manufacturing Systems, Szczecin,2008.

[10] ŠTEFÁNIK, A. – GREGOR, M.: Simulation of production systems with support of virtual reality, In: CO-MA-TECH 2006, Trnava 2006, ISBN: 80-227-2472-6.