Acta Scientiarum

http://www.uem.br/acta
ISSN printed: 1806-2563
ISSN on-line: 1807-8664
Doi: 10.4025/actascitechnol.v37i3.25986

Development of an adaptive genetic algorithm for simulation optimization

Rafael de Carvalho Miranda*, **José Arnaldo Barra Montevechi and Alexandre Ferreira de Pinho**

*Instituto de Engenharia de Produção e Gestão, Universidade Federal de Itajubá, Av. BPS, 1303, 37500-903, Pinheirinho, Itajubá, Minas Gerais, Brazil. *Author for correspondence. E-mail: mirandaprod@yahoo.com.br*

ABSTRACT. Optimization methods in discrete-event simulation have become widespread in numerous applications. However, the methods´ performance falls sharply in terms of computational time when more than one decision variable is handled. Current assay develops an adaptive genetic algorithm for the simulation optimization capable of achieving satisfactory results in time efficiency and response quality when compared to optimization software packages on the market. A series of experiments was elaborated to define the algorithm's most significant parameters and to propose adaptations. According to the results, the most significant parameters are population size and number of generations. Further, adaptive strategies were proposed for these parameters which enabled the algorithm to obtain good results in response quality and time necessary to converge when compared to a commercial software package.

Keywords: discrete-event simulation, meta-heuristic, optimization methods, computational time.

Desenvolvimento de um algoritmo genético adaptativo para otimização via simulação

RESUMO. Métodos de otimização em simulação a eventos discretos se tornaram comuns em diversas aplicações. No entanto, o desempenho desses métodos diminui significativamente, em termos de tempo computacional, na presença de mais de uma variável de decisão. Dessa forma, o objetivo deste trabalho é desenvolver um algoritmo genético adaptativo para a otimização via simulação, capaz de alcançar resultados satisfatórios em termos de tempo e qualidade de resposta, quando comparado a pacotes comerciais de otimização. De modo a cumprir com este objetivo, o planejamento de experimentos foi utilizado para definir os parâmetros mais importantes do algoritmo genético e propor adaptações. Os resultados mostraram que os parâmetros tamanho da população e número de gerações foram os mais significativos. Com isso, foram propostas estratégias de adaptação para estes parâmetros, o que permitiu que o algoritmo obtivesse bons resultados em termos de qualidade de resposta e tempo necessário para convergência, quando comparado com um pacote comercial de otimização.

Palavras-chave: simulação a eventos discretos; metaheurística; métodos de otimização; tempo computacional.

Introduction

Computational simulation is increasingly used to aid in decision-making (BANKS et al., 2009; LAW, 2007). It has already been indicated as one of the most common research techniques in many areas, due to its versatility, flexibility and power of analysis (JAHANGIRIAN et al., 2010; RYAN; HEAVEY, 2006).

According to Law (2007), one of simulation's disadvantages is that it is not an optimization technique in itself, but rather a means to test a single scenario. Without the optimization software and in search of an optimal solution, simulation practitioners are forced to reconfigure their models to find that which presents the best system performance. This reality has changed due to accelerated computational capacity and improved heuristic optimization search techniques. Banks et al. (2009) and Fu (2002) report that the use of optimization with simulation has been growing continuously due to simulation software packages with integrated optimization routines.

According to Hillier and Lieberman (2010), the methodology for employing simulation to identify optimal system configuration in the light of a specific response variable, is known as simulation optimization (SO). The same authors emphasize that this field has shown growing interest in the field of operational research.

According to Azadeh et al. (2009), SO is one of the most important computational developments in recent years. These authors highlight that methodologies prior to this integration demanded complex and difficult changes within simulation models that were often economically unviable, especially when a large number of decision variables had to be taken into account.

For Figueira and Almada-Lobo (2014), these methods have dominated the optimization routines in the discrete-event simulation software because of their flexibility of meta-heuristics to tackle almost any type of search space and ability to achieve quickly good quality solutions.

Among the various software modules for integrated optimization, the available commercial software packages include: AutoStat®, OptQuest®, Optimiz®, Optimizer® and SimRunner® (FU, 2002; LAW, 2007), which use different search methods, such as Evolutionary Algorithms, Genetic Algorithms, Scatter Search, Taboo Search, Neural Networks and Simulated Annealing.

Fu (2002) registers a predominance of Evolutionary Algorithms, such as Genetic Algorithms (GA), when using the meta-heuristics optimization routines. GA are present in many commercial optimization packages such as SimRunner® from ProModel® and AutoState® from AutoMod® (LAW, 2007).

In spite of advances in optimization software for simulation models, a frequent criticism is that the processing time becomes extremely long when handling more than one input variable (HARREL et al., 2004).

Kleijnen et al. (2010) allege that SO problems are difficult to solve, and state that disadvantages exist in their utilization due to the fact that simulation model outputs are products of implicit functions exposed to noise. The authors assert that, depending on the number of inputs in the simulation model, the adaptation process may become inefficient or expensive due to computational demands and time involved.

According to Steponavičė et al. (2014), SO´s greatest limitation is the number of decision variables to be evaluated, since the software's performance is considerably reduced in models with large numbers of variables. Tyni and Ylinen (2006) assert that convergence time is the most significant restriction to reaching computational efficiency for an optimization algorithm.

In fact, the optimization softwares for simulation models on the market today are true black box data. Only a small percentage about their real structure or about the supporting optimization algorithm is known. It turns out to be practically impossible to develop a wider discussion about these optimization methods.

Current research, therefore, develops an adaptive genetic algorithm (AGA) for the optimization of discrete-event simulation models. In fact, it is able to attain good results in terms of efficiency (speed) and response quality when compared to a commercial optimization package.

Current paper broadly presents the steps that led to the development of an optimization algorithm, the discussion of its main configuration parameters and the adjustments to the algorithm structure that made it capable of simulating model optimization. The method is applied to discrete-event simulation models, whose decision variables are: discrete, deterministic and integer. According to Pinho et al. (2012), most papers in the literature on optimization via simulation use these boundary conditions.

The paper is divided into four sections. The second section (Materials and methods) is an analysis of the GA parameters and presents the proposed AGA. The third section (Results and discussion) shows the tests used with AGA in the simulation optimization models, while the last section (Conclusions) offers conclusions based on the findings in current study.

Material and methods

Genetic algorithm parameter analysis with DOE

Due to the lack of consensus in the literature with regard to the definition of GA parameters, a design of experiments (DOE) determined which parameters and possible interactions were significant, impacted the algorithm's convergence time and attained high quality solutions in SO problems.

The four principal GA parameters, namely, population size, number of generations, crossover rate and mutation rate were analyzed. These parameters were chosen due to the fact that the literature treats them as direct influences on the necessary time to reach a response, as well as the response's quality (AZADEH; TARVEDIAN, 2007; YANG et al., 2007).

Three simulation models developed for manufacturing environments were used for optimization and analysis of GA parameters (population size, number of generations, crossover rate and mutation rate). In fact, GA was employed in its classic form to conduct these experiments, following Holland (1975).

In this experiment, the researchers chose to work with two levels for each factor. Each level and its variations are presented in Table 1. The choice of the values for each of the levels was carried following Paul and Chanev (1998) and Pinho et al. (2012).

Table 1. Factors, levels and variations for design of experiments.

Factors		Levels	
		-	+
A	Population size	20	80
B	Number of generations	5	30
C	Crossover rate	50%	90%
D	Mutation rate	1%	10%

A full-factorial design was utilized in the study since full-factorial (2^4 experiments – 4 factors with 2 levels) design allows for the estimation of main effects on the analyzed factors and any possible interactions in any order among the factors (MONTGOMERY, 2005).

In the case of the project under analysis, five replicas were defined for each experiment, totaling 80 experiments (2^4x5) for each simulation model. As three simulation models will be used in the study, at the end of each stage, there were a total of 240 experiments. All data collected were stored and analyzed statistically.

Two response variables will be analyzed for this experimental design. The first will verify the response quality presented by the classic GA in the optimization of the simulation models, while the second will analyze the time necessary for the algorithm to arrive at the solution.

Table 2 presents a summarized form of the principal conclusions of the variance analyses (ANOVA) conducted for the three simulation models. The GA parameters and the interactions that proved to be significant for the variables are presented for each case. The solution, the algorithm convergence time, and the effect that each parameter and interaction show on the variable under analysis are provided.

ANOVA verified the parameter population size (A), which is often pointed out in the literature. It deserves the greatest attention as it proved to be significant in all the variance analyses for the time necessary for convergence and for the response quality.

However, a hurdle exists in the case of the parameter. The larger the population size, the better the response found by the algorithm will be. Likewise, the larger the population size, the greater the amount of time necessary will be for the algorithm to arrive at this solution. Consequently, users may opt for high-quality solutions that take a greater amount of computational time, or a more rapid solution but a lower quality solution, when working with classic GA.

The number of generations (B) exhibited a behavior which was similar to the factor population size (A) for the response variable Time, the most significant for the algorithm convergence time. In the case of this factor, the greater the number of generations, the more time will be taken to arrive at a solution. The double interaction between the factors population size and number of generations (A*B) also showed itself to be significant for all three cases analyzed for the variable Time, as well as for double and triple interactions (number of generations, crossover rate and mutation rate). However, the latter had reduced effects when compared with that interaction.

Table 2. GA Parameter significance analysis.

Experiment	Variable	Factor	Effect	T	P-Value
First Experiment	Solution	A	5240	3.77	0.000
		B	4820	3.47	0.001
	Time	A	5897.78	152.05	0.000
		B	7018.57	180.95	0.000
		A*B	4130.12	106.48	0.000
Second Experiment	Solution	A	700	6.02	0.000
	Time	A	4418.48	279.37	0.000
		B	5303.72	335.35	0.000
		A*B	3173.17	200.64	0.000
		C*D	66.07	4.18	0.000
		B*C*D	58.38	3.69	0.000
Third Experiment	Solution	A	375.6	2.25	0.028
	Time	A	4595.88	144.69	0.000
		B	5452.12	171.65	0.000
		A*B	3193.32	100.53	0.000
		C*D	-79.83	-2.51	0.014

One important conclusion from the variance analyses was that the parameters: crossover rate (C) and mutation rate (D), when analyzed separately, were not significant in any of the three experiments for the confidence level adopted in the study (95%), although this does not mean that these parameters may be excluded.

The genetic operators bring population diversity to the GA. They were close to being significant, as the mutation rate for solution quality in the second experiment, and were significant in second and third order interactions, like the interaction between the crossover rate and mutation rate for algorithm convergence time in the second and third experiments.

Proposed optimization method

Necessary adaptations for the optimization algorithm parameters

The analysis carried out with DOE reveals that two GA parameters significantly influence the response quality and the algorithm convergence time.

The parameter population size was the most significant to obtain a high quality solution and was also significant for the time necessary for convergence. For the latter, the number of generations also had a significant impact. Thus, some adaptations will be proposed for the parameter population size and a stop criterion will be defined for the GA, albeit not based on number of generations, since it takes a long time to be processed.

In the case of the parameters crossover rate and mutation rate, which failed to show themselves significant in any of the experiments, the rates will be discussed for these parameters based on the experimentation carried out and also on a review of the literature.

Adaptations to the population size parameter

In current investigation, each individual in a population is represented by a binary solution. Taking into consideration the conditions of this research (discrete, deterministic and integer variables), it becomes necessary to initially determine the quantity of bits necessary to represent each possible optimization problem solution.

Therefore, the equation proposed by Mitchell (1996) was used, in which the quantity of bits necessary to represent a determined individual is given by Equation 1.

$$\mathrm{k} = \log_2\left(\frac{\mathrm{upp_i} - \mathrm{low_i}}{\mathrm{precision}} + 1\right) \quad (1)$$

where:

- k – number of bits;
- precision – desired precision to represent a solution;
- $\mathrm{low_i}$, $\mathrm{upp_i}$ – lower and upper bound for the operation range (variation).

For the models optimized in this investigation, with an upper bound equal to 9 and a lower bound equal to 1 and a precision equal to 1, Equation 1 needs a total of 4 bits to represent each decision variable for these problems. Consequently, if a problem has three decision variables, each solution will consist of 12 bits.

By defining the number of bits necessary to represent each solution, one should determine the number of individuals to be generated by the algorithm to compose the initial population. This number, as previously presented, should be chosen with care, as it will directly influence the response quality found by the algorithm as well as the time spent to find the solution.

In their proposal, Reeves and Rowe (2002) relate the number of population individuals with an individual size (number of bits) and the probability that the population generated may, with a crossover operator, generate any possible solution for the problem. The proposal is shown in Equation 2.

$$\text{Population Size} = \left[1 + \log(-k/\ln P_2)/\log 2\right] \quad (2)$$

where:

- k – number of bits (individual size);
- P_2 – probability of a generated population (represented by a binary codification) may generate all possible optimization solutions by reproduction through the crossover operator only.

Although Reeves and Rowe´s (2002) proposal may be extended to other types of genetic representations beyond the binary, these go beyond the scope of current investigation. According to the authors, a population size equal to 30 is able to resolve a great variety of optimization problems which use binary representation. Thus, it was decided that the formulation for the initial population calculation proposed by Reeves and Rowe (2002) would be used to define the initial AGA population.

The researchers also opted to increment the value of the population size with each generation, seeing that the population size calculated by Reeves and Rowe (2002) is the smallest population size capable of representing all the possible solutions for a determined problem. Moreover, as aforementioned, the larger the population size, the better the response quality will be. Due to this fact and to results provided by Kaveh and Shahrouzi (2007), Ma and Zhang (2008) and Pinho et al. (2012), the researchers opted to increase the initial population calculated by Reeves and Rowe (2002) by 50% in each new algorithm generation.

A 50% increase for each new generation is justified by the fact that the initial population always starts with a small rate, when compared to the most commonly used values in optimization problems using GA.

Alteration of the number of generations' parameter

For the AGA proposed in current assay, a stop criterion based on the response quality between successive generations was chosen. Essentially, upon terminating the fourth generation, the AGA will compare the best solution found with the best solution found in the previous generation. If the difference found between these solutions is inferior to 5%, the algorithm will converge and present the best result found. This criterion may be seen in Equation 3.

$$SC = \frac{y_i - y_{i-1}}{y_{i-1}} \qquad (3)$$

where:

- SC – Stop criterion;
- y_i – best solution found in generation i;
- y_{i-1} – best solution found in generation i – 1;

Upon finishing a generation, the AGA carries out the following test (starting from the fourth generation):

- If SC $\geq$ 0.05, the algorithm conducts a new generation;
- If SC $<$ 0.05, the algorithm converges and forwards the best solution found in generation i.

So that the occurrence of a premature convergence in the first generations could be avoided (as these are the smallest populations of the AGA), a criterion was added which stipulated that at least three generations must be executed. It should be noted that y_i is always larger or, in the worst hypothesis, equal to y_{i-1}, due to elitism, which always tends towards better solutions being found for the following generation in order not to lose a high quality solution during the algorithm processing.

Crossover, mutation and elitism rate parameters

Reeves and Rowe (2002) reported that 30 individuals were more than enough to compose an initial population that used binary representation. Therefore, the above-mentioned author recommended a high crossover rate and a low mutation rate (or even no mutation rate at all). When the population size is less than 30 individuals, the AGA adopts rates of 99% for the crossover operator and 1% for the mutation operators. In cases in which the initial population is greater than 30, the AGA adopts rates of 90% for the crossover operator and 10% for the mutation operator. The crossover operator used in the optimization method development was the one-point crossover, whereas the simple binary mutation was used for the mutation operator.

Elitism rates were also defined with regard to the initial population size. When the population size calculated is inferior to 30 individuals, the AGA adopts a rate of 20%; on the other hand, the adopted rate is 10%. These rates gave good results in pre-tests carried out in the algorithm.

Proposed AGA structure

Thereby, the optimization method proposed takes on the optimization of a non-linear discrete-event simulation model in which the decision variables meet the following boundary conditions: discrete, deterministic and integer variables.

Upon initiating the first generation, the algorithm calculates the number of scenarios that may be processed and then calculates the number of bits necessary to represent an individual of the population (MITCHELL, 1996). The initial population size is calculated according to Reeves and Rowe (2002) and the initial AGA population is generated.

Depending on the population size generated, a particular set of crossover, mutation and elitism parameters is adopted for the algorithm. If the initial population has less than 30 individuals, the AGA adopts the rates of 99% for the crossover operator; 1% for the mutation operator and 20% for elitism operator. If there are more than 30 individuals, the AGA adopts rates of 90% for the crossover operator, 10% for the mutation operator and 10% for the elitism operator.

The previous sequence of steps is only executed in the first AGA generation. Starting with the second generation, the population size is increased by 50% for each new algorithm iteration. These new individuals are generated randomly and added to those already generated by means of reproduction, thus composing a new generation.

Following the algorithm's steps, each possible problem solution (generated individual) is evaluated by a discrete-event simulator.

For the development of the proposed method, a computational tool was developed that manipulated its parameters and enabled communication between the computational tool and the discrete-event simulator. This communication occurs through an object called ProActiveX provided by the simulator manufacturer.

For optimization, the computational tool sends individuals of the population to the ProActiveX object. These individuals represent the input variables of the simulation model. The object then enters those rates into the simulator, waits for the simulation to run and retrieves the results of the simulated model. Results will evaluate each existing individual genetic algorithm.

All individuals generated in the first three generations of the algorithm are evaluated without being assessed by the stop criterion. For the first three generations after the evaluation, the roulette reproduction method and the crossover and mutation operators select the individuals according to rates defined by the initial population size.

These new individuals compose a new population that is increased by 50% for new, randomly generated individuals. Thus, for each generation after the second one, the populations are formed by individuals generated by means of reproduction using genetic operators (crossover and mutation) and for individuals generated randomly, thus guaranteeing greater population diversity in the proposed algorithm.

Starting with the fourth generation, the best individual is compared with the best individual in the previous generation, according to the adopted stop criterion. If the difference between the individuals is not significant (that is, less than 5%) the stop criterion is considered satisfactory and the algorithm converges, presenting the best solution. Otherwise, the algorithm initiates the composition of a new generation and the whole process starts again.

The proposed method has been implemented in VB.NET programming language on a computer with Intel processor (Core 2 Duo) 1.58 GHz, 2GB of RAM and Windows operating system, 64-bit platform. The flowchart representing the structure of the proposed optimization method may be obtained from the authors.

Results and discussion

The optimization method proposed in current investigation was tested on the optimization of three discrete-event simulation models. Every single model was checked and then statistically validated (SARGENT, 2013).

The simulation models used are related to the allocation of resources in the manufacturing area, where the simulation is widely used to support decision-making (JAHANGIRIAN et al., 2010). For Law (2007), such problems involving the definition of the number of people and equipment for performance increase in manufacturing, are an important class of problems for discrete-event simulation.

SimRunner® was the commercial software for comparing methods. The software has three profiles (Aggressive, Moderate, Cautious), herein denoted as 1, 2 and 3. The commercial optimization software profile reflects the number of possible solutions the optimizer will examine. Profile 3 (Cautious) considers a great number of possible solutions and is the most complete search. Its thorough nature, however, considerably extends the time necessary for processing when compared to that of other profiles. Profile 1 (Aggressive) works with a small population which allows for a quick convergence on a solution; however, the demands in time limit the response's quality when compared to that of the other two profiles. Finally, Profile 2 (Moderate) presents a balance between the former two profiles.

Optimization aims to find the best combination of decision variables to maximize profits in the three models. The optimization of the three simulations is then presented and analyzed individually. The results obtained will be compared to a commercial software package as well as to the proposed optimization method and its responses, the time necessary for the methods to converge and the number of experiments run.

Optimization of the first study object

The decision variables in current study were defined as the number of workers (types 1, 2 and 3) (x_1, x_2, x_3) and the number of machines (types 1 and 2) (x_4 and x_5). The decision variables are integers with lower and upper bounds of 1 and 9, respectively. Similar to the previous case, there were 59,049 possible scenarios.

In the optimization of the first simulation model, the proposed method reached a better solution than that of all the commercial software profiles and also showed to be more efficient in terms of time than other profiles. In fact, it arrived at a better solution in less time than the others. In relation to the number of experiments, the proposed method conducted just 3 experiments more than profile 1 of the commercial software. Table 3 shows the results reached by the tests.

Table 3 demonstrates a positive result for efficiency (+ξ) and indicates how much "better" (in terms of response quality, time and number of experiments) the commercial software was in relation to the proposed method. A negative result (-ξ) indicates how much "worse" (using the same basis for comparison) the commercial software was in relation to the proposed method. This interpretation is also valid for the other study objects.

Table 3. Optimization results for the first study object.

Optimizer	Time (sec)		Profit		Experiments conducted	
	Value	ξ	Value	ξ	Value	Ξ
Profile 1	2376	-15%	1102218.57	-1.01%	160	+2%
Profile 2	3714	-79%	1105897.67	-0.68%	246	-51%
Profile 3	7983	-286%	1110532.16	-0.27%	543	-233%
Proposed method	2070		1113505.00		163	

Optimization of the second study object

The variable decisions for this study object were defined as the number of operators responsible for quality control operations (x_1, x_2, x_3) and the number of workbenches for tests 1, 2 and 3 (x_4, x_5, x_6). All variables were defined as integers, with lower and upper bounds of 1 and 9, respectively. There were 531,441 possible scenarios for the problem under analysis.

In the optimization of the second simulation model, the proposed method reached the best result among all the commercial software profiles, with less time to reach the solution than profiles 2 and 3, while conducting a smaller number of experiments than these profiles. In relation to the time and number of experiments carried out, the proposed method was only inferior to profile 1 of the commercial software; however, the latter had a low response quality. The results for the tests may be seen in Table 4.

Table 4. Optimization results for the second study object.

Optimizer	Time (sec)		Profit		Experiments conducted	
	Value	ξ	Value	ξ	Value	Ξ
Profile 1	1630	+6%	5061936.67	-0.05%	135	+17%
Profile 2	2883	-66%	5061936.67	-0.05%	243	-49%
Profile 3	3090	-78%	5062776.67	-0.03%	255	-56%
Proposed Method	1733		5064232.00		163	

Optimization of the third study object

The decision variables for this study object were defined as the number of grinding operators (x_1, x_2), the number of lapidating operators (x_3, x_4) and the number of grinding machines (x_5, x_6). All variables were defined as integers, with a lower bound of 1 and an upper bound of 9, with 531,441 possible scenarios.

In the optimization of the last simulation model, the proposed method reached the best response quality among all the commercial software profiles. However, in relation to the number of experiments conducted, the proposed method achieved fewer experiments than profile 3 of the commercial software. In relation to the time of the proposed method, it was more efficient than profiles 2 and 3 of the optimizer. Results of the tests may be seen in Table 5.

Table 5. Optimization results for the third study object.

Optimizer	Time (sec)		Profit		Experiments conducted	
	Value	ξ	Value	ξ	Value	Ξ
Profile 1	24900	+13%	511861.00	-0.57%	197	+46%
Profile 2	43980	-55%	511904.00	-0.56%	348	+5%
Profile 3	53640	-88%	513390.00	-0.27%	424	-16%
Proposed method	28464		514804.00		367	

Conclusion

AGA proposed for the optimization of discrete-event simulation models showed good results in terms of response quality and efficiency (time and number of experiments carried out).

Thus, the method herein developed has proved to be an alternative to existing commercial optimization packages, being capable of conducting simulation model optimization quickly (low computational time) and dependably, while still offering high quality solutions.

Further research along these lines may include similar tests with other simulation models.

Acknowledgements

The authors would like to thank CAPES, CNPq and FAPEMIG for funding current research and the anonymous referees for their suggestions that contributed towards substantial improvements in the paper.

References

AZADEH, A.; TABATABAEE, M.; MAGHSOUDI, A. Design of intelligent simulation software with capability of optimization. **Australian Journal of Basic and Applied Sciences**, v. 3, n. 4, p. 4478-4483, 2009.

AZADEH, A.; TARVERDIAN, S. Integration of genetic algorithm, computer simulation and design of experiments for forecasting electrical energy consumption. **Energy Policy**, v. 35, n. 10, p. 5229-5241, 2007.

BANKS, J.; CARSON II, J. S.; NELSON, B. L.; NICOL, D. M. **Discrete-event Simulation**. 5th ed. New Jersey: Prentice-Hall, 2009.

FIGUEIRA, G.; ALMADA-LOBO, B. Hybrid simulation-optimization methods: A taxonomy and discussion. **Simulation Modelling Practice and Theory**, v. 46, p. 118-134, 2014.

FU, M. C. Optimization for simulation: theory vs. Practice. **Journal on Computing**, v. 14, n. 3, p. 192-215, 2002.

HARREL, C. R.; GHOSH, B. K.; BOWDEN, R. **Simulation using promodel**. New York: McGraw-Hill, 2004.

HILLIER, F. S.; LIEBERMAN, G. J. **Introduction to Operations Research**. 9th ed. New York: McGraw-Hill, 2010.

HOLLAND, J. H. **Adaptation in natural and artificial systems**. Ann Arbor: University of Michigan Press, 1975.

JAHANGIRIAN, M.; ELDABI, T.; NASEER, A.; STERGIOULAS, L. K.; YOUNG, T. Simulation in manufacturing and business: A review. **European Journal of Operational Research**, v. 203, n. 1, p. 1-13, 2010.

KAVEH, A.; SHAHROUZI, M. A hybrid ant strategy and genetic algorithm to tune the population size for efficient structural optimization. **Engineering Computations**, v. 24, n. 3, p. 237-254, 2007.

KLEIJNEN, J. P. C.; VAN BEERS, W.; VAN NIEUWENHUYSE, I. Constrained optimization in simulation: A novel approach. **European Journal of Operational Research**, v. 202, n. 1, p. 164-174, 2010.

LAW, A. M. **Simulation modeling and analysis**, 4th ed. New York: McGraw-Hill, 2007.

MA, Y.; ZHANG, C. Quick convergence of genetic algorithm for QoS-driven web service selection.

Computer Networks, v. 52, n. 5, p. 1093-1104, 2008.

MITCHELL, M. **An Introduction a genetic algorithm**. Cambridge: MIT Press, 1996.

MONTGOMERY, D. C. **Design and analysis of experiments**, 6th ed. New York: Wiley, 2005.

PAUL, R. J.; CHANEV, T. S. Simulation optimisation using a genetic algorithm. **Simulation Practice and Theory**, v. 6, n. 6, p. 601-611, 1998.

PINHO, A. F.; MONTEVECHI, J. A. B.; MARINS, F. A. S.; COSTA, R. F. S.; MIRANDA, R. C.; FRIEND, J. D. Evaluation of a proposed optimization method for discrete-event simulation models. **Pesquisa Operacional**, v. 32, n. 3, p. 543-559, 2012.

REEVES, C. R.; ROWE, J. E. **Genetic algorithms**: principles and perspectives. New York: Kluwer Academic Publishers, 2002.

RYAN, J.; HEAVEY, C. Process modeling for simulation. **Computers in Industry**, v. 57, n. 5, p. 437-450, 2006.

SARGENT, R. G. Verification and validation of simulation models. **Journal of Simulation**, v. 7, n. 1, p. 12-24, 2013.

STEPONAVIČĖ, I.; RUUSKA, S.; MIETTINEN, K. A solution process for simulation-based multi objective design optimization with an application in the paper industry. **Computer-Aided Design**, v. 47, n. 2, p. 45-58, 2014.

TYNI, T.; YLINEN, J. Evolutionary bi-objective optimization in the elevator car routing problem. **European Journal of Operational Research**, v. 169, n. 3, p. 960-977, 2006.

YANG, T.; KUO, Y.; CHO, C. A genetic algorithms simulation approach for the multi-attribute combinatorial dispatching decision problem. **European Journal of Operational Research**, v. 176, n. 3, p. 1859-1873, 2007.

Received on December 8, 2014.
Accepted on April 27, 2015.