Multiobjective Optimization of a Free Radical Polymerization
Genetic Algorithms

FLORIN LEON[1], SILVIA CURTEANU[2]*

[1] Technical University "Gh. Asachi" Iasi, Department of Computer Science and Engineering, 71A D. Mangeron Blvd., 700050, Iasi, Romania

[2] Technical University "Gh. Asachi" Iasi,, Department of Chemical Engineering, 71A D. Mangeron Blvd. 700050, Iasi, Romania

The present paper continues a previous article [1] where multiobjective optimization of a polymerization process was approached with a classical method – sequential quadratic programming. In this article, an optimization method based on genetic algorithms is used for the free radical polymerization of methyl methacrylate. The influence on the optimization results of the main parameters (the size of the initial population, the number of generations, recombination rate, mutation rate), as well as that of different variants of genetic algorithms (different ways of recombination, mutation and selection) is studied. The main conclusion of the article is that these values and methods depend on the studied process, but also inter-condition each other, such that the optimization results are rather more correlated with the overall set of values considered. Thus, we try to establish some directions to guide the search for optimal values.

Keywords: genetic algorithms, optimization, free radical polymerization

Most real-word engineering optimization problems are multiobjective in nature, since they normally have several (possibly conflicting) objectives that must be satisfied at the same time.

It is well known that batch free radical polymerization is inherently a multivariable control design problem. In order to obtain a polymeric material with pre-specified molecular weight and other important properties, one should manipulate at least two variables: initiator or monomer addition policy and reactor temperature. If only one property is controlled by only one manipulated variable, other properties might deviate from their desired values during the reaction. This undesirable picture certainly will impair the final end use of the manufactures polymer [2].

For multi-objective optimization, the objective function can be formulated as a vector function whose elements represent the objectives functions [2-4] or as a single scalar objective function that combines all identifiable performance measures with appropriate weighting factors [5,6].

Computer-aided optimization methods have been widely employed in chemical process industries. Traditional optimization methods can be classified into two distinct groups: direct and gradient-based methods. Deb [7] emphasizes some common difficulties with most of the traditional direct and gradient-based techniques: 1) the convergence to an optimal solution depends on the chosen initial solution; 2) most algorithms tend to get stuck to a sub-optimal solution; 3) an algorithm efficient in solving one optimization problem may be not efficient in solving a different optimization problem.

In recent years, there is a growing interest in optimization techniques based on evolutionary algorithms, particularly genetic algorithms. Because of their flexibility, ease of operation, minimal requirements and global perspective, these algorithms have been successfully used in a wide variety of multiobjective problems [8]. Multiobjective optimization of the polymerization processes is an example of their applications [3-5, 8-10]. These techniques do not need any initial guesses and converge, in the most cases, to the global optimum even when there are several local optima present. In addition, genetic algorithms use information about the objective function and not its derivatives (such traditional optimization techniques), nor does they require any other auxiliary knowledge about the process [4]. Different types of genetic algorithm and their applications in chemical reaction engineering, including polymerization processes, are described in some review works [11-14].

In the first part of the cycle entitled "Multiobjective optimization of a free radical polymerization" [1], the optimization problem of methyl methacrylate free radical polymerization is solved using a traditional method, sequential quadratic programming, respectively, implemented in *Matlab* with pre-defined function *fmincon*.

In the present paper, which is the second part of the study, the influence of the parameters of a standard genetic algorithm on the optimization results of the methyl methacrylate (MMA) polymerization process is studied. Also, different types of genetic algorithms are developed, with different variants for the algorithm phases, in order to design the most suitable optimization methodology for the studied process. The article especially addresses the establishment of a general strategy for solving a multiobjective optimization problem using genetic algorithms.

Genetic algorithms

Genetic Algorithms (GA) are a family of computational models inspired by natural evolution, in which the fittest species survive and propagate while the less successful tend to disappear. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination operators to these structures so as to preserve critical information.

The chromosomes in GA are similar to biological chromosomes, as their genes reflect different aspects of the solution. Chromosomes consist of genes, blocks of

* *email: scurtean@ch.tuiasi.ro*

DNA. Each gene will produce a particular protein that will shape a certain trait. Possible variations of a gene are called alleles. Each gene has its own position in the chromosome; this position is called a locus. The complete set of genetic material (all chromosomes) is called the genome.

An implementation of a genetic algorithm begins with a population of (typically random) chromosomes. One then evaluates these structures and allocates reproductive opportunities in such a way that those chromosomes which represent a better solution to the target problem are given more chances to "reproduce" than those chromosomes which are poorer solutions. According to the evolutionary theory, only the most suited elements of a population can survive and generate offspring, thus transmitting their characteristics to new generations. The heredity is enclosed in the chromosomes of individuals represented in an optimization problem by a specific numerical (often binary) code. The suitability of each element according to the optimization problem under consideration is evaluated via a fitness value directly derived from the objective function.

In a broader usage of the term, a genetic algorithm is any population-based model that uses selection and recombination operators to generate new sample points in a search space. Unlike other methods, evolution is not a directed process, but a heuristic one, in which the purpose of the individuals is to compete in order to propagate their genetic material to the next generation. In the biological case, the fitness of an individual results from its interaction with the environment. Genetic algorithms use a fitness function instead to compute how close a potential solution is to the desired solution.

The three fundamental procedures in a typical genetic algorithm are selection, crossover and mutation. The cycle of evolution is generally repeated until a predefined number of generations is reached.

The *selection* establishes the way in which parents will be chosen for the offspring that will form the next generation. In this phase, the fitness of all the individuals in the population is evaluated. The individuals with higher fitness must have more chances to reproduce. For each individual to be created in the next generation, two parents are thus selected.

Crossover is the operation that ensures the genetic diversity of the population. After two parents have been selected, their chromosomes are combined to produce an offspring. In nature, crossover occurs when corresponding chromosomes of a parent exchange genetic material by breaking and reuniting of DNA molecules. In this case, each parent has two strings of chromosomes (the double helix), and one string from a parent is combined with one string from the other. In GA, an individual has only one set of chromosomes. Thus, the chromosome of the offspring is built by taking different parts of the parents' chromosomes and binding them together. There are many variants for performing this operation, which largely depend on the problem.

After crossover, a small change in the chromosome of the offspring can be applied. The importance of this operation - *mutation* - is still a matter of debate. It is believed that its role is to get the system out of local extremes or to accelerate convergence, although genetic algorithms do not need differential functions and gradient descendent methods for convergence.

The first generation is randomly generated. Then, using the above operations, a new population is created. The old population is abandoned and the subsequent generation is produced using the new population. There is no theoretical reason for this clear distinction between generations. This restriction is only an implementation model that simplifies the computation. In order not to lose good solution during the search, sometimes an *elitism* procedure is employed, i.e. the best individuals from the old population are directly copied into the new one. That ensures that the overall solution of the GA will not get worse.

The process is repeated until a convenient solution is found. Normally, the best (fittest) individual of its generation represents the solution given by the genetic algorithm at a certain moment.

Figure 1 summarizes the stages of a GA procedure.

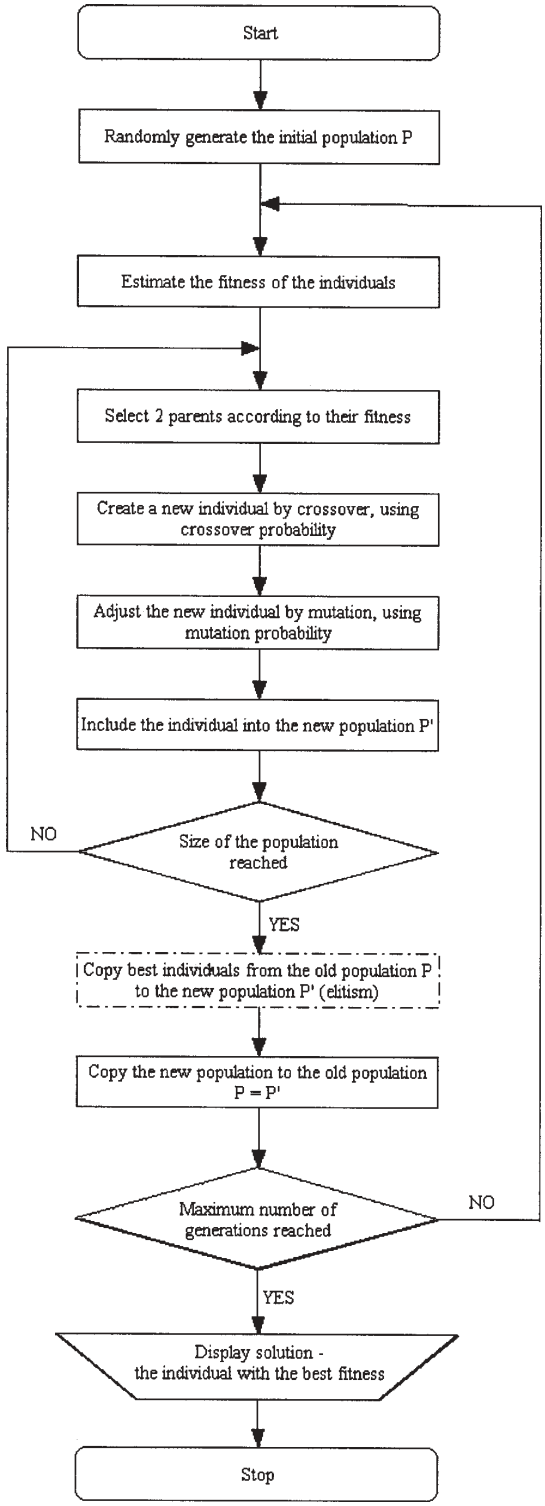

Fig. 1. General flowchart of a genetic algorithm

Although GA's highly rely on stochastic processes (selection, crossover, and mutation are performed with certain probability rates), they are not random searches. The evolutionary mechanisms have definitely better results than random exploration, and these results are achieved with faster convergence.

Description of the GA model

In our GA model, we used real value encoding for the chromosomes. There are other approaches for MMA polymerization using binary solution representation [9], as it is the simplest type of encoding, in which chromosomes are composed only of 1's and 0's. Even the number of alleles is thus rather small (two), this encoding is very common, because it is very easy to use. However, value encoding is more general, because genes are real numbers. Some experiments [15] have shown that real value encoding is more time efficient, with better precision of the solutions.

Our optimization tests are based on some variants of GA resulted from different methods for selection, crossover and mutation.

Selection

Variant 1: Roulette wheel selection

One of the most common selection types is the roulette wheel selection. In this strategy, the parents are selected proportionally to their fitness. The probability of an individual *i* to be chosen is:

$$P_i = \frac{F_i}{\sum_{j=1}^{n} F_j} \quad (1)$$

where is the fitness of individual *i* and *n* is the number of individual in the population. Since roulette wheel is basically a stochastic process, there is a good chance that the individual with best fitness is selected both as mother and father. Thus, in order to diminish the loss of genetic diversity, one can impose that the two parents be different individuals.

Variant 2: Rank selection

The roulette method of selection will have problems when the fitnesses differ greatly. For example, if the best chromosome fitness is 90 % of the entire roulette wheel then the other chromosomes will have a slim chance of being selected. Rank selection first ranks the population and then every chromosome receives fitness from this ranking. The worst will have fitness 1, second worst 2 etc. and the best will have fitness *n* (number of chromosomes in population).

The probability of an individual *i* to be chosen is now:

$$P_i = \frac{R_i}{\sum_{j=1}^{n} R_j} \quad (2)$$

where R_i is the fitness rank of individual *i* in the population.

Variant 3: Tournament selection

Both the roulette wheel and the rank selection have the disadvantage of being computationally expensive (the population must be sorted in some way in order to obtain their ranks and some linear complexity $O(n)$ processing must be made to compute the sum of fitnesses or ranks). In general tournament selection, *m* individuals are selected at random from the population and the fittest of them is selected. The most common type of tournament selection is binary tournament selection, where just two individuals are selected to produce a child.

Crossover

Arithmetic real value crossover produces a linear combination of the parents. Given a uniform random number $r \in [0,1]$,

$$C = r \cdot M + (1 - r) \cdot F \quad (3)$$

or

$$C = r \cdot F + (1 - r) \cdot M \quad (4)$$

where *C* is the real value chromosome of the child, and *M* and *F* are the chromosomes of the parents.

Variant 1: crossover with the same point for all genes

If we imagine the individuals as point in an *n*-dimensional space of the features produced by the genes, a crossover with the same point generates a new individual on the line segment that links the two parent points. This means that the *r* above is the same for all genes. Therefore, the offspring will be like its parents to the same extent for all its features.

Variant 2: crossover with different points.

In this case, the new individual will no longer be on the line segment that links its parents. The *r* value is different for every gene. The offspring will look more like one parent regarding a feature and less regarding another.

Mutation

Variant 1: fine tuning

After a new individual has been created, a mutation is performed on it. Given the chosen solution encoding, a uniform mutation can be employed, that randomly changes a gene to a uniform random value from an interval: $x'_i = U(\min_i, \max_i)$. The interval we used was $[0.95x_i, 1.05x_i]$ where x_i represents the current value of the gene. We did not use absolute boundaries for the interval in order not to constrain this genetic operator.

In this way, the individual is "shaken" a little; after the mutation it receives a gene value close to the one obtained by crossover. Thus catastrophic mutations are avoided, that totally change a gene value. This method performs a "fine tuning" of the gene value, and thus an individual close to the optimal solution will not be taken out of that area in the solution space. Each gene is altered by a maximum of 10 % of its initial value.

Variant 2: resetting

This method is the conceptual opposite of the previous variant. A gene value is reset to a random value in its search interval. The purpose is to refresh the search process, in case when the genetic diversity of the population decreases (so no longer converges to the solution) or the algorithm has converged into a local optimum. Each gene is independently considered, and mutation gives it a new random value in the initialization interval. Only some genes change (possibly all, but unlikely).

In this paper, we develop software for the different genetic operators described above. The optimization results obtained for each situation were compared in order to choose the best variant according to the process under study.

Formulation of the optimization problem

The mass balance equations give the following set of ordinary differential equations:

$$\frac{dz}{dt} = f(z,u,t) \qquad z(t_0) = z_0 \qquad (5)$$

where $z(t)$ is the state variable vector defined, for bulk polymerization, by

$$z = [I, x, \lambda_0, \lambda_1, \lambda_2, \mu_0, \mu_1, \mu_2] \qquad (6)$$

In equation (6), I represents the concentration of the initiator, x is the monomer conversion and λ_k and μ_k (k = 0,1,2) are moments of chain length for radicals and dead polymer, respectively.

The control variable vector is $u(t)$, with temperature and initial concentration of the initiator as components:

$$u(t) = [T_1, T_2, T_3, I_0] \qquad (7)$$

An admissible control input $u^*(t)$ should be formed in such a way that the performance index J, defined by the following equations, is minimized:

$$Min\ J[u(t)] = w_t \cdot t_f + w_Q \cdot Q_f + w_x \cdot (1 - x_f) + w_{DPn} \cdot \left(1 - \frac{DP_{nf}}{DP_{nd}}\right)^2 \qquad (8)$$

subject to:

$$dz/dt = f(z,u,t) \qquad (9)$$

$$u_{min} \le u(t) \le u_{max.} \qquad (10)$$

where number average polymerization degree is defined as:

$$DP_n = \frac{\lambda_1 + \mu_1}{\lambda_0 + \mu_0} \qquad (11)$$

$$x_f = x(t_f) \quad \text{and} \quad DP_{nf} = DP_n(t_f) \qquad (12)$$

In the above equation, J is the objective function to be minimized, w are weighting factors, Q is the polydispersity index, x_d and DP_{nd} are desired values of monomer conversion and number average chain length at $t = t_f$; x_f and DP_{nf} are the actual values corresponding to the final reaction time t_f.

An important objective for the polymerization system is the minimization of the final reaction time, which leads to higher productivity. The other objective included in the same function is the minimization of the polydispersity index of the polymer product. This ensures good physical properties of the polymer manufactured. The maximization of monomer conversion forces the amount of unreacted monomer to be small, and hence keeps post-reactor separation and recycling costs low. The endpoint requirement on DP_n leads to the production of polymer having desired properties, because several physical properties of polymers are related to their values of DP_n.

Many authors suggest the use of vectorial objective functions, which leads to obtaining more solutions, each "specialized" on a certain criterion. While this approach has the benefit of emphasizing the equilibrium regions of the decision space, eventually it is the user who must decide to choose an appropriate solution, if the problem requires a single result. In our case, the user must take into account several technological criteria. Using a scalar function with user-chosen objective weights proves to be a more simple approach, better suited for our GA based investigation.

Results and discussions

Population size, number of generations, crossover probability and mutation probability are known as the control parameters of genetic algorithms. The values of these parameters must be specified before the execution of GA and they depend on the nature of the objective function.

There is no general termination criteria for GA. Predetermined number of generations or time or comparison of the best solutions to average fitness may be taken as stopping criterion. In our work, the number of generations is established before running.

Table 1

THE INFLUENCE OF THE INITIAL POPULATION DIMENSION UPON OPTIMIZATION RESULTS

No.	Io (mol/ m³)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**15.6**	**61.4**	0.46	117	**2.41**	3961	8.22662	**dim=20**
		85	0.85	121				
		36.5	**0.90**	400				
2	**18.8**	**45.9**	0.40	382	**2.81**	10391	9.97335	**dim=50**
		63	0.80	395				
		62.3	**0.85**	400				
3	**34.5**	**43.5**	0.41	388	**2.69**	8238	10.0584	**dim=100**
		62.4	0.78	400				
		72.2	**0.80**	400				

Table 2

THE INFLUENCE OF THE NUMBER OF GENERATIONS UPON OPTIMIZATION RESULTS

No.	Io (mol/ m³)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**16.9**	**62**	0.43	115	**2.45**	3872	8.5214	dim=20
		79	0.80	119				**gen=15**
		35.2	**0.88**	400				
2	**15.6**	**61.4**	0.46	117	**2.41**	3961	8.22662	dim=20
		85	0.85	121				**gen=30**
		36.5	**0.90**	400				
3	**26**	**93.6**	0.33	8	**2.62**	375	8.30637	dim=20
		111	0.92	13				**gen=50**
		74.2	**0.93**	44				
4	**20.5**	**42.6**	0.23	376	**3.36**	3984	11.5804	dim=20
		73.2	0.75	399				**gen=100**
		81.2	**0.85**	400				

A good process model is a necessary prerequisite for application of the optimal control strategy. Consequently, the kinetic model has been validated by experimental runs of bulk polymerization in a wide range of operating conditions. Our previous works [17, 18] present good agreement between simulation results and experimental data for the kinetic model developed for batch MMA polymerization.

The polymerization process is conducted in a fixed time of 400 s, in a perfect mix batch reactor. Limit ranges for the reaction temperature, T, and the initiator concentration feed, I_0, are established based on experimental data: 40°C ≤ T ≤ 90°C and 10 mol/m^3 ≤ I_0 ≤ 50 mol/m^3.

The GA based optimization technique is implemented in *Matlab* with original software, as specific functions were programmed for each phase of the genetic algorithm. Figure 1 presents an outline of the algorithm. At each step of the optimization procedure, model equations are integrated using a special function for solving stiff differential equation, *ode15* in *Matlab 7.0*. Integration leads to conversion, number and weight average molecular weight histories for $t_{init} \leq t \leq t_f$.

In order to watch the influence of the GA parameters on the optimization results, in the objective function (8) we consider $w_t = 0$ and $W_{DPn} = 0$, i.e. we try to maximize conversion and minimize the polydispersity index. By decreasing the number of objectives which should be simultaneously accomplished, we focus on the relationship between the values of parameters of the genetic algorithm and the optimization results.

The tables that show the results of the optimizations contain the following columns: the current number, the values of decision variables, I_0 and T (three values for temperature, T_1, T_2, T_3), conversion, x, time, t, in minutes, polydispersity index, Q, number average polymerization degree, DP_n (all these resulting from solving the model in the conditions I_0, T established through the optimization procedure), the value of the objective function, J, and an observation column where the GA parameters or methods are mentioned. In an optimization, three values are shown for conversion and time. The first two represent the values corresponding to the intermediate steps of temperature, and the last – the value obtained at the end of the reaction, when using optimal parameters. On each row (in each optimization), the final results are marked by bold characters, representing the objectives of the optimization and the optimal values of decisions variables (T, I_0). Also, particular cases of the objective function are included in the tables, by assigning zero values to the weights of some objectives.

The weights used in optimization, according to the discussions and tests in [1], are: $w_x = 10$, $w_Q = 3$.

Tables 1 and 2 contain optimizations made with different values of the GA basic parameters, i.e. the dimension of the initial population (*dim*) and the number of generations (*gen*). The selection method is the rank selection, and for crossover and mutation, the variants marked as 1 in the section that describes the GA are used. Other values being used are: the crossover rate (*cross*) = 0.8, the mutation rate (*mut*) = 0.03, and the number of generations (*gen*) = 15.

GA research showed that the solution improves as the number of individuals in the population increases, but only up to a point. Beyond that, a larger population decreases the convergence speed of the algorithm, without leading to an improvement of the solution. In table 1, the value of 20 for the initial population size leads to the smallest value of the objective function and to acceptable values of the partial objectives, while trying to achieve a big conversion and a small polydispersity index. Thus, in optimization marked as 1 in table 1, with I_0 = 15.6 mol/m^3 and T = 61.4, 85, 36.5°C ($T_1 < T_2 > T_3$), a final conversion of 0.90 and a polydispersity index of 2.41 are achieved.

With the increase in the number of generations, the execution time increases. Since the GA is an iterative procedure, the quality of the solution should increase with the number of generations, especially if elitism is used, which guarantees the fact that the solution will not worsen over time. But for each parameter and process there is a limit beyond which there are no more improvements of the results. As table 2 shows, for MMA polymerization, a number of 30 generations (*gen=30*) is sufficient to provide acceptable results. In other words, the increase of the *gen* parameter beyond this limit does not improve the solution, but surely increases considerably the solving time.

Table 3

THE INFLUENCE OF THE CROSSOVER RATE UPON OPTIMIZATION RESULTS

No.	Io (mol/ m^3)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**18.8**	**45.9** **63** **62.3**	0.40 0.80 **0.85**	382 395 400	**2.81**	10391	9.97335	dim=20 gen=30 **cross =0.6**
2	**19.8**	**71.9** **94.7** **59**	0.50 0.83 **0.92**	53 55 107	**2.45**	1826	8.21527	dim=20 gen=30 **cross=0.8**
3	**14.2**	**62.8** **84.8** **35.6**	0.46 0.84 **0.90**	110 113 400	**2.40**	4152	8.244491	dim=20 gen=30 **cross =1**

Table 4

THE USE OF DIFFERENT CROSSOVER VARIANTS IN THE OPTIMIZATION PROCEDURE

No.	Io (mol/ m^3)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**19.8**	**71.9** **94.7** **59**	0.50 0.83 **0.92**	53 55 107	**2.45**	1826	8.21527	dim=20 gen=30 cross=0.8 crossover variant 1
2	**27.6**	**101.3** **57.8** **48**	0.88 0.95 **0.97**	11 400 400	**3.19**	443	9.86025	dim=20 gen=30 rata=0.8 crossover variant 2

The crossover rate (*cross*) represents the probability with which from two parents a new individual is generated. If the rate is small, there are high chances that one of the parents to be directly copied into the new population. Since crossover is the basis of the search process, a rate close to 1 should increase the speed of finding a solution. Copying a parent into the new population is beneficial only when it has a high fitness value (the elitism achieves this objective in order not to lose the best solutions). Table 3 motivates the choice of the 0.8 rate, based both on partial objectives, and on the minimum value of the objective function.

Table 4 uses the established values for the GA parameters and tests the two crossover variants, numbered 1 and 2, described in the previous section. From the objective function point of view, variant 1 leads to a lower value, therefore mathematically the results can be considered better. Regarding the proposed objectives, a better value for the polydispersity index is obtained with variant 1 (2.45 compared to 3.19), and a grater value for conversion is obtained with variant 2 (0.97 compared to 0.92). Variant 2 has a different thermal regime from those presented in previous tables ($T_1 < T_2 > T_3$), i.e. a thermal regime formed of increasing temperature steps ($T_1 < T_2 < T_3$). Comparing the results of the two optimizations in table 4 emphasizes the role of the user in choosing the solution based on technological reasons, depending on the objective which is considered with priority.

The next tests (table 5) contain different values for the mutation rate (*mut*). In addition, in some simulations the weight of the conversion was increased from $w_x = 10$ to $w_x = 20$, in order to force the obtaining a higher conversion.

Increasing the mutation rate to 0.05 (optimization 5) and using the variant 1 for mutation yielded the best values for the proposed objectives, i.e. a final conversion close to 1 and a rather low polydispersity index of 2.65. The conditions provided by the values of the decision variables correspond to a thermal regime formed of high temperatures, with increasing values, and a lower value of the initiator concentration. One can notice the short reaction time (32 minutes) and a small value of the polymerization degree, due to high values of the temperature. The objective function has the lowest value as well, compared to previous optimizations. In improving the optimization results, the increase of the mutation rate had a considerable influence.

Comparisons between the two mutation variants marked 1 and 2 are given in table 6. The optimizations are compared two by two, i.e. 1 with 1', 2 with 2' etc. One can notice that the general principle of multiobjective optimization (with more contradictory objectives) is respected: improving an objective leads to the worsening of another. Thus, the increase of conversion leads to the increase of polydispersity index. In addition, one must take into account the influence of weights, which determine the priority of achieving a certain objective by increasing their corresponding values. Variant 1 for mutation better balances the two objectives, by associating high conversions with rather small values of the polydispersity index.

The selection operator that decides which of the individuals of a population will be able to participate in forming the next population has an important role within a genetic algorithm. The goal of selection is to ensure more chance of reproduction to the fittest individuals and thus maximizing the performance of the new individuals, and eventually of the whole population.

The simulations presented in the previous tables were made using the rank selection. With the best values of the GA previously established, different selection methods are tried. In table 7 one can see better optimization results using rank selection.

In the following, we consider the objective function (8), where $w_{DPn} = 50$, $w_t = 10^{-5}$, and $DP_{nd} = 1800$. This means more objectives to be met simultaneously. Table 8 presents optimizations made with a GA where the mutation and selection methods are changed. From the mathematical point of view (the minimization of the objective function), optimization 1 is the best. The user's decision can take into account a certain preferred objective. For example, the best value for conversion is achieved in optimization 4, the smallest polydispersity index in optimization 1, DP_n closest to DP_{nd} in optimization 3, and the shortest reaction time in optimization 2.

Table 5
THE INFLUENCE OF THE MUTATION RATE UPON OPTIMIZATION RESULTS

No.	Io (mol/ m³)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**15.6**	**61.4** **85** **36.5**	0.46 0.85 **0.90**	117 122 400	**2.41**	3961	8.226622	dim=20 gen=30 w_x=10 cross=0.8 **mut=0.01**
2	**24.7**	**55.8** **77.1** **56.8**	0.38 0.81 **0.91**	146 154 209	**2.74**	3546	9.139093	dim=20 gen=30 w_x=10 cross=0.8 **mut=0.03**
3	**20.5**	**62.7** **86.2** **37.7**	0.42 0.85 **0.91**	94 99 400	**2.43**	2674	10.123701	dim=20 gen=30 w_x=20 cross=0.8 **mut=0.03**
4	**17.3**	**65.1** **87.5** **46.9**	0.41 0.81 **0.91**	84 89 143	**2.58**	2587	9.512288	dim=20 gen=30 w_x=20 cross=0.8 **mut=0.04**
5	**12**	**94.6** **96.1** **114**	0.33 0.41 **0.997**	12 14 **32**	**2.65**	608	8.017003	dim=20 gen=30 w_x=20 cross=0.8 **mut=0.05**

Table 6

THE USE OF DIFFERENT MUTATION VARIANTS IN THE OPTIMIZATION PROCEDURE

No.	Io (mol/ m³)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**15.6**	**61.4** **85** **36.5**	0.46 0.85 **0.90**	117 122 400	**2.41**	3961	8.226622	dim=20 gen=30 w_x=10 cross=0.8 mut=0.01 mutation variant 1
1'	**10.7**	**57.7** **80.8** **50.6**	0.43 0.82 **0.89**	174 179 229	**2.42**	6184	8.40132	dim=20 gen=30 w_x=10 cross=0.8 mut=0.01 mutation variant 2
2	**24.7**	**55.8** **77.1** **56.8**	0.38 0.81 **0.91**	146 154 209	**2.74**	3546	9.139093	dim=20 gen=30 w_x=10 cross=0.8 mut=0.03 mutation variant 1
2'	**40.7**	**59.1** **80.6** **51.5**	0.47 0.78 **0.89**	100 103 162	**2.58**	2683	8.833824	dim=20 gen=30 w_x=10 cross=0.8 mut=0.03 mutation variant 2
3	**35.2**	**74.1** **91** **39.8**	0.43 0.85 **0.92**	34 38 400	**2.46**	1145	9.073308	dim=20 gen=30 w_x=20 cross=0.8 mut=0.03 mutation variant 1
3'	**46.7**	**88.6** **49.3** **54.5**	0.89 0.89 **0.95**	17 400 400	**3.34**	580	11.04941	dim=20 gen=30 w_x=20 cross=0.8 mut=0.03 mutation variant 2
4	**17.3**	**65.1** **87.5** **46.9**	0.41 0.81 **0.91**	84 89 143	**2.58**	2587	9.512288	dim=20 gen=30 w_x=20 cross=0.8 mut=0.04 mutation variant 1
4'	**46.7**	**83** **48.1** **51.7**	0.86 0.86 **0.94**	22 400 400	**3.60**	793	11.9728	dim=20 gen=30 w_x=20 cross=0.8 mut=0.04 mutation variant 2
5	**12**	**94.6** **96.1** **114**	0.33 0.41 **0.997**	12 14 **32**	**2.65**	608	8.017003	dim=20 gen=30 w_x=20 cross=0.8 mut=0.05 mutation variant 1
5'	**36.8**	**64.6** **83.5** **54.3**	0.47 0.82 **0.90**	69 72 126	**2.48**	2054	9.47213	dim=20 gen=30 w_x=20 cross=0.8 mut=0.05 mutation variant 2

The examples presented in this paper aim at describing an optimization approach based on genetic algorithms, more precisely, a methodology of searching for the optimum by means of the most suitable GA parameters and variants.

Our discussions were focus not the absolute results, but rather the steps of the GA based optimization strategy. The user decision in choosing the weights of the objective function and the optimization results and the technological

Table 7
THE USE OF DIFFERENT SELECTION METHODS IN THE OPTIMIZATION PROCEDURE

No.	Io (mol/ m³)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**12**	**94.6** **96.1** **114**	0.33 0.41 **0.997**	12 14 **32**	**2.65**	608	8.017003	dim=20 gen=30 w_x=20 cross=0.8 mut=0.05 mutation variant 1 **rank selection**
1'	**12.9**	**60.7** **80.9** **46.2**	0.42 0.81 **0.87**	129 135 184	**2.45**	4961	9.929125	dim=20 gen=30 w_x=20 cross=0.8 mut=0.05 mutation variant 1 **roulette selection**
1''	**29.9**	**76.9** **82.5** **55.2**	0.46 0.85 **0.90**	31 36 83	**3.14**	1454	11.391942	dim=20 gen=30 w_x=20 cross=0.8 mut=0.05 mutation variant 1 **tournament select.**

Table 8
EXAMPLES OF OPTIMIZATION REALIZED WITH A COMPLEX OBJECTIVE FUNCTION

No.	Io (mol/ m³)	T (°C)	x	t (min)	Q	DP_n	Objective function	Observations
1	**27.7**	**78.4** **51.3** **65.1**	0.85 0.94 **0.95**	35 400 **400**	**4.27**	**1364**	14.474469	dim=20 gen=30 w_x=20; w_Q=3; $w_t=10^{-5}$; DP_{nd}=1500 cross=0.8 mut=0.05 mutation variant 1 rank selection
2	**38.3**	**59.6** **67.8** **70**	0.36 0.44 **0.95**	88 94 **151**	**5.81**	**1522**	18.5963	dim=20 gen=30 w_x=20; w_Q=3; $w_t=10^{-5}$; DP_{nd}=1500 cross=0.8 mut=0.05 mutation variant 2 rank selection
3	**41.9**	**42.8** **76.1** **76.6**	0.37 0.50 **0.93**	374 379 **400**	**5.28**	**1680**	18.145888	dim=20 gen=30 w_x=20; w_Q=3; $w_t=10^{-5}$; DP_{nd}=1500 cross=0.8 mut=0.05 mutation variant 2 roulette selection
4	**14.4**	**40.7** **59.7** **72.4**	0.12 0.40 **0.97**	279 362 **400**	**6.295**	**1660**	20.310328	dim=20 gen=30 w_x=20; w_Q=3; $w_t=10^{-5}$; DP_{nd}=1500 cross=0.8 mut=0.05 mutation variant 2 tournament select.

criteria are important elements in an optimization procedure.

Conclusions

The optimization methods based on genetic algorithms are flexible, robust, easy to use and usually lead to globally optimal solutions. More, they do not use initial guesses nor derivatives of the objective function.

In the present paper, a step by step optimization strategy is presented, in which the most appropriate values of the GA parameters are found for the studied polymerization process. From this point of view, one can underline the influence of the overall set of parameters (rather than individual ones) upon the optimal solution. Also, the parameter values and GA variants can be correlated to either the value of the objective function or the values reached by some objectives to be minimized, even if the objective function has not the minimum value.

The main goal of the study is thus to establish an optimization methodology based on genetic algorithms,

designed in different variants. This strategy is quite general and could be applied to other chemical processes, with high probability to obtain accurate results.

References

1.CURTEANU, S., Bulletin of IPI, Inst. Politehnic Ia°i L(LIV) fascicola 1-2, secþiunea chimie °i inginerie chimicã, 2006, p. 77-90
2.MIERQUIOR, D.M., FONTOURA, J.M.R., PINTO, J.C., LIMA, E.L., Latin Am. Appl. Research, **31**, 2001, p.513
3.GARG, S., GUPTA, S.K., Macromol. Theory Simul., **8**, 1999, p.46
4.ZHOU, F., GUPTA, S.K., RAY, A.K., J. Appl. Polym. Sci., **78**, 2000, p.1439
5.GARG, S., GUPTA, S.K., SARAF, D.N., J. Appl. Polym. Sci., **71**, 1999, p.2101
6.AHN, S.M., CHANG, S., RHEE, H.K., J. Appl. Polym. Sci., **69**, 1998, p.59.
7.DEB, K., Optimization for engineering design: algorithms and examples, New Delhi, India, Prentice Hall, 1995
8.SILVA, C.M., BISCAIA, Jr.E.C., Comp. Chem. Eng., **27**, 2003, p.1329
9.CHAKRAVARTHY, S.S.S., SARAF, D.N., GUPTA, S.K., J. Appl. Polym. Sci., **63**, 1997, p.529
10.GUPTA, R.R., Gupta, S.K., J. Appl. Polym. Sci.,**73**, 1998, p.729
11.BHASKAR, V., GUPTA, S.K., Ray, A.K., Rev. Chem. Eng., **16**, 2000, p.1
12.DEB, K., Multiobjective Optimization using Evolutionary Algorithms, Chichester, UK, Wiley, 2001
13.DEB, K., PRATAP, A., AGARWAL, S., MEYARIVAN, T.A., IEEE Trans. Evol. Comut., **6**, 2002, p.182
14.NANDASANA, A.D., RAY, A.K., GUPTA, S.K., Int. J. Chem. React. Eng., **1**, 2003, p.1
15.MICHALEWICZ, Z., Genetic Algorithms + Data Structures = Evolution Programs, Springer Verlag, Berlin, 1996.
16.CURTEANU, S., BULACOVSCHI, V., Roum. Chem. Quart. Rev., 7, 1999, p.281
17.CURTEANU, S., BULACOVSCHI, V., CONSTANTINESCU, M., Hung. J. Ind. Chem., 27, 1999, p.287

Manuscript received: 9.02.2007