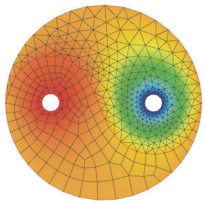
Publishing House
AKAPIT

COMPUTER METHODS IN MATERIALS SCIENCE

Informatyka w Technologii Materiałów

Vol. 15, 2015, No. 1



MULTI-OBJECTIVE OPTIMIZATION OF PHTHALIC ANHYDRIDE CATALYTIC REACTOR USING GENETIC ALGORITHM WITH SIMULATED BINARY JUMPING GENES OPERATOR

VIBHU TRIVEDI, MANOJKUMAR RAMTEKE*

Department of Chemical Engineering, Indian Institute of Technology Delhi, Hauz Khas, New Delhi 110 016, India
**Corresponding author: mcramteke@chemical.iitd.ac.in*

Abstract

Multi-objective optimization problems of chemical industry have been efficiently solved by evolutionary algorithms (EAs). However, due to high computational costs, different concepts are introduced in evolutionary framework for the improvement of convergence speed. One such concept is 'jumping genes' which has been adapted in binary-coded genetic algorithm and found to be improving the performance of the algorithm significantly. However, its adaptation in real-coded form lacked the similar success. In an attempt to fill this gap, a new jumping genes operator has been recently developed for real-coded elitist non-dominated sorting genetic algorithm (RNSGA-II), namely, simulated binary jumping genes (SBJG). This work aims at exploring the utility of SBJG for solving real-life industrial optimization problems using a case study of multi-objective optimization (MOO) of an industrial phthalic anhydride (PA) catalytic reactor. The results obtained are found to be converging faster than RNSGA-II and its other existing jumping genes adaptations for both two- and three-objective formulations.

Key words: evolution, genetic algorithm, multi-objective optimization, jumping genes

1. INTRODUCTION

Real-life optimization problems often involve multiple objectives, complex process models and several variables. Since last two decades, EAs and particularly genetic algorithm (GA) (Holland, 1975) has been used efficiently to solve these problems. GA is inspired from Darwinian principle of evolution of species. Originally, it is a binary-coded algorithm which requires the coding of problem variables in binary (0 or 1) form (i.e. each variable represents a set of binaries and corresponding binaries of all variables are arranged one after another in a string to constitute a chromosome) since these symbolize the actual genes closely and facilitate mimicking of genetic operators while their real values are required to evaluate the fitness of chromosomes. Thus, binaries are decoded to the real values, repeatedly. Additionally, this approach has few more limitations (especially, for the optimization problems with continuous variables) such as lack of absolute precision in the solutions obtained, fixed mapping of the problem variables, presence of Hamming cliff, etc. These limitations often make it difficult to achieve the convergence and lead to increase in time and cost of computation. Since real-life optimization problems often involve continuous variables, real-coded algorithms are better suited for these. RNSGA-II is one such popular real-coded algorithm which is a modification of original binary-coded NSGA-II (Deb et al., 2002). It simulates the effect of binary crossover and mutation in real-coded

ISSN 1641-8581

framework using simulated binary crossover (SBX) (Deb & Agrawal, 1995) and polynomial mutation (PM) (Deb & Agrawal, 1999). Further, these problems involve complex model equations and solving these requires inordinately large computational efforts. In RNSGA-II, these model equations are solved repeatedly for a fixed number of times (= population size × number of generation). Since computational resources are restricted, these problems are often attempted for a limited number of generations and population size. Thus, one has to rely on prematurely converged solutions and their quality depends on the convergence speed of the algorithm. Any improvement in convergence speed is thus highly recommended.

Recently, a new operator, SBJG (Ramteke et al., 2014), is developed to further improve the convergence speed of RNSGA-II. This operator simulates the concept of jumping genes (JG) (McClintock, 1987) from natural genetics and has been successfully applied to several benchmark problems (Ramteke et al., 2014). However, its potential for complex industrial problems is yet to be explored. In this study, two such complex MOO problems related to an important process of chemical industry i.e., phthalic anhydride catalytic reactor are solved using RNSGA-II-SBJG. The results obtained using RNSGA-II-SBJG are compared with that obtained using RNSGA and its other existing JG adaptations and are found to be quite promising with a considerable improvement in convergence speed.

2. SIMULATED BINARY JUMPING GENES (SBJG)

Kasat and Gupta (2003) introduced the concept of JG (or transposons) from natural genetics in the framework of binary-coded NSGA-II (NSGA-II-JG). Subsequently, several other researchers (Sankararao & Gupta, 2006; Bhat et al., 2006; Bhat, 2007; Ripon et al., 2007) proposed various JG adaptations of binary- and real-coded NSGA-II. In NSGA-II-JG, binary chromosomal strings, present in the offspring population, are selected randomly with a jumping gene probability, P_{JG}. The jumping gene sites on these chromosomal strings are then randomly identified. For this, a random number between 0 and 1 is generated and multiplied by l_{chr} (length of chromosomal string); the result rounded-off to an integer represents one jumping gene site. Similarly, the other site is identified and binaries between these sites are replaced by randomly generated new ones. This procedure involves a macro – macro mutation, and helps in increasing the genetic diversity. A drawback of this adaptation is that several variables get perturbed simultaneously, if the length of jumping gene is too large, compromising the fitness of the chromosome. To improve upon this, Bhat et al. (2006) and Bhat (2007) proposed another adaptation (NSGA-II-aJG) with fixed length jumping genes. In this, one jumping gene site is identified randomly whereas the other is selected f_{JG} (a predefined constant) binaries later in the same chromosome. This often restricts the perturbation up to maximum two variables. A similar JG adaptation (say RJG1) in real-coded framework for multi-objective simulated annealing (MOSA) has been proposed by Sankararao and Gupta (2006). Here, real-coded variables instead of binaries are replaced by randomly generated new ones following the similar procedure as in NSGA-II-JG and NSGA-II-aJG. This complete perturbation of variables often results in the loss of original information contained in variables and subsequently to their inferior fitness values. In another real-coded adaptation (Ripon et al., 2007) (say RJG2), variables are re-operated using existing operators, SBX and PM, with a probability P_{JG}. Thus, no new operation is added in this adaptation and existing operators are used for simulating the effect of jumping genes.

SBJG is an attempt towards improving the drawbacks of above given real-coded JG adaptations. It simulates the binary-coded JG operation for its implementation in RNSGA-II. In SBJG operation, a variable V_j $\left(j \in [1,n]\right)$ of a real-coded chromosome (i.e. the array of n problem variables (V)), selected randomly with a probability P_{JG}, is perturbed with following mathematical formulation:

$$V_j = V_j + \left(V_j^{High} - V_j^{Low}\right)\alpha_j \tag{1}$$

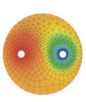
where, α is a distribution function given by:

$$\alpha_j = \begin{cases} \left[\left(2R_j\right)^{\frac{1}{\eta_{JG}+1}}\right] - 1 & \text{if } R_j < 0.5 \\ 1 - \left[2\left(1-R_j\right)\right]^{\frac{1}{\eta_{JG}+1}} & \text{if } R_j \geq 0.5 \end{cases} \tag{2}$$

where, R_j is a random number and η_{JG} (jumping gene index) is a integer random number (IR) between two jumping gene sites jg_1 and jg_2:

$$\eta_{JG} = \left(IR\right)_j \in [jg_1, jg_2] \tag{3}$$

The efficacy of SBJG in simulating the effect of binary JG operation is analyzed by Ramteke et al.



(2014) through a qualitative analysis. In this analysis, the distribution of variable perturbation is calculated for a large number of random instances. The pattern of distribution of local perturbation obtained using SBJG resembles to that of binary JG operation whereas the same using RJG1 (i.e. large perturbation) differs completely from that of binary JG operation. As RJG2 utilizes SBX and PM operators, the functioning of SBJG is also compared with that of SBX and PM by calculating the distribution functions of these operators (i.e., α, β and δ respectively) for a large number of random instances. It is found that values of α corresponding to same random numbers are distributed for different instances whereas the values of β and δ remain fixed. This uniqueness of SBJG is attributed to variable nature of η_{JG} which differs from fixed indices of SBX and PM. Thus, SBJG maintains higher genetic diversity while remaining in the local range of variable perturbation which is similar to binary JG operation.

3. PROBLEM FORMULATION

Phthalic anhydride (PA) is commonly used as a raw material in the manufacture of various polymers (i.e. polyester, plastics from vinyl chloride), insecticides, pesticides, dyes, etc. It is produced by gas-phase catalytic oxidation of o-xylene in a multitubular reactor with V_2O_5 and TiO_2 catalyst packed in many zones in each tube. The processed gas coming out of the reactor is treated in a pair of 'switch condensers' used alternately where the gas is cooled to separate the condensed PA. The treated gas is then scrubbed with water, or incinerated catalytically or thermally. Bhat and Gupta (2008) developed a mathematical model of this operation using a reaction network suggested by Skrzypek et al. (1985) (see figure 1a). They have assumed that the reactions take place in a reactor having several identical tubes with each tube comprising nine zones of catalyst and eight intermediate zones of inert packing (see figure 1b). Each of these inert packing zones acts as a cooling region for the high temperature processed gas coming from the previous catalyst zone. The coolant is continuously circulated in the outer side of the tubes to maintain the temperature within the acceptable limits. This model consists of nonlinear algebraic equations for steady state mass and energy balance between the bulk gas and the external surface of the impervious solid catalyst and differential equations for the steady state balance of the bulk gas phase over the length of a single tube. The nonlinear algebraic equations are solved using Powell's hybrid algorithm (DNEQNF program) whereas the differential equations are solved using Gear's algorithm (code DIVPAG) from IMSL library, simultaneously. The complete details of model such as model equations, rate expressions, adsorption and kinetic parameters are reported in Bhat and Gupta (2008).

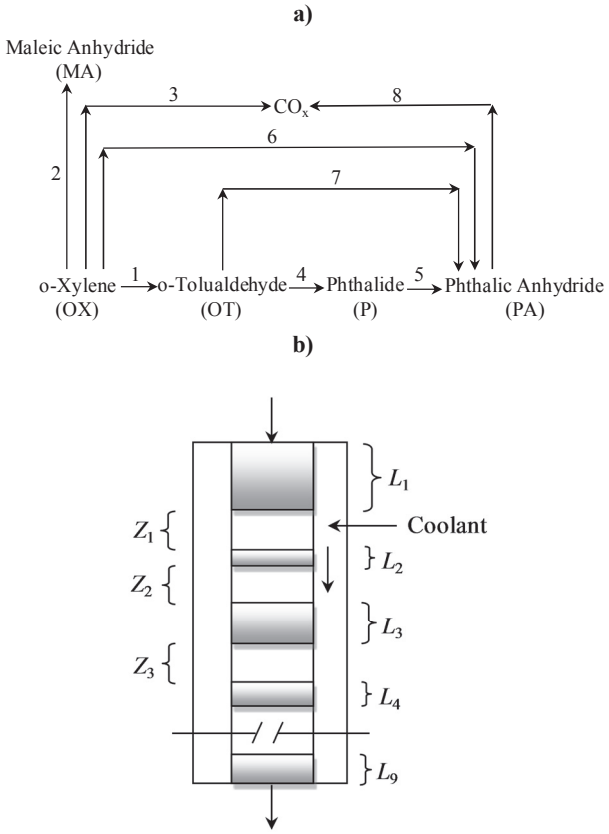


Fig. 1. *Reaction network for the production of PA (Skrzypek et al., 1985) (a) and PA reactor with 9 zones of catalyst (Bhat & Gupta, 2008) (b).*

Bhat and Gupta (2008) formulated a two-objective optimization problem involving maximization of the yield of PA which increases the productivity and minimization of the cumulative length of the catalyst beds which decreases the loading of expensive catalyst. The problem formulation involves 9 catalyst zones and 20 decision variables. The details are given as follows:

3.1. Problem 1

Objectives:

$$\max I_1(u) = X_{PA} + w_1\left[1-\frac{X_{PA}}{1.2}\right]^2 + w_2\left[1-\frac{L_{cat}}{3.6}\right]^2 + w_3 \tag{4}$$

$$\max I_2(u) = \left(\frac{1}{1+L_{cat}}\right) + w_1\left[1-\frac{X_{PA}}{1.2}\right]^2 + w_2\left[1-\frac{L_{cat}}{3.6}\right]^2 + w_3 \quad (5)$$

where, minimization of L_{cat} is converted (Deb, 2001) to maximization problem as $\left(\frac{1}{1+L_{cat}}\right)$

$$u = [c_{in}, T_{F,in}, T_{c,in}, \dot{m}, Z_1, Z_2, ..., Z_8, L_1, L_2, ..., L_8]^T$$

$$\text{and } L_{cat} = \sum_{i=1}^{9} L_i \quad (6)$$

Constraints:

$$T_{\max} \le 510\ ^{o}\text{C} \quad (7)$$

$$L_i \ge 0.01 \text{ m} \quad (8)$$

Total length of the reactor tube,

$$L_{tube} = 3.5 \text{ m} \quad (9)$$

$$L_9 = \left(3.5 - \sum_{i=1}^{8} L_i - \sum_{i=1}^{8} Z_i\right) \quad (10)$$

Bounds:

$$65 \le c_{in} \le 85 \text{ g OX/(m}^3\text{air at NTP)} \quad (11)$$

$$147\ ^{o}\text{C} \le T_{F,in} \le 287\ ^{o}\text{C} \quad (12)$$

$$337\ ^{o}\text{C} \le T_{c,in} \le 447\ ^{o}\text{C} \quad (13)$$

$$0.001 \le \dot{m} \le 0.005 \text{ (kg coolant)/s} \quad (14)$$

$$0.2 \le Z_i \le 0.45 \text{ m};\ i = 1,\ 2,...,\ 7 \quad (15)$$

$$0.1 \le Z_8 \le 0.45 \text{ m} \quad (16)$$

$$0.05 \le L_i \le 0.9 \text{ m} \quad (17)$$

$$0.01 \le L_i \le 0.2 \text{ m};\ i = 2,3,...,8 \quad (18)$$

Here, X_{PA} and L_{cat} are the yield of PA (i.e. mass of PA produced / mass OX consumed) and cumulative length of catalyst beds. The decision variables c_{in}, $T_{F,in}$, $T_{c,in}$, $\dot{m}$, Z_i and L_i represent concentration of OX in feed, inlet temperature of feed, inlet temperature of coolant, mass flow rate of coolant, length of i^{th} inert zone and length of i^{th} catalyst zone, respectively. In equations (4) and (5), the values of X_{PA} = 1.2 and L_{cat} = 3.6 m in the bracketed penalty functions are guessed to keep these variables in the range of real or expected values. The optimal values of $L_1 - L_8$ and $Z_1 - Z_8$ are chosen by the optimization algorithm whereas L_9 is calculated directly using length balance (see equations (9) – (10)). Equations (7) – (10) represent the constraints on temperature of gas and length of catalyst beds while equations (11) – (18) represent the bounds of the decision variables. The weighting functions for bracketed penalty terms used in equations (4) and (5) are given as:

$$\text{If } X_{PA} \le 1.1, w_1 = -500; \text{ else } w_1 = 0 \quad (19)$$

$$\text{If } L_9 \le 0 \text{ m}, w_2 = -3000; \text{ else } w_2 = 0 \quad (20)$$

In addition to this, hard penalties, w_3, are imposed in equations (4) and (5) for violation of constraint on maximum allowable temperature of gas given in equation (7). Depending on the location of catalyst bed where the constraint is violated different penalty weight is imposed as given below:

$$\text{If } T_{\max} \le 510\ ^{o}\text{C in bed } i;\ i = 1,\ 2,...,\ 9;$$

$$w_3 = 0; \text{ else } w_3 = -3000 + 250(i-1) \quad (21)$$

Also, the hard penalties are imposed to maintain the length of each catalyst bed above a given limit of 0.01 m (see equation (8)) as given below:

$$\text{If } L_i \ge 0.01 \text{ m};\ i = 1,\ 2,...,\ 9;\ w_3 = 0; \text{ else } w_3 = 300 \quad (22)$$

The diameter of each reactor tube used is 25 mm, mass flux G is 19.455 kg m^{-2} h^{-1} and diameter of catalyst particle used is 3 mm. More details can be obtained from Bhat and Gupta (2008).

Another critical factor in the production of PA is the formation of CO_x. Excess formation of CO_x causes a severe decrease in the PA yield, reduces the catalyst life and increases the downstream processing cost. Hence, the minimization of CO_x is also an important objective for the optimal operation of PA reactor. Considering this fact, a three-objective optimization problem (problem 2) is formulated for the first time in this study where the first two objectives are same as problem 1 and the third objective is the minimization of mole fraction of CO_x in the processed gas.

3.2. Problem 2

Objectives:

$$\max I_1(u) = X_{PA} + w_1\left[1-\frac{X_{PA}}{1.2}\right]^2 + w_2\left[1-\frac{L_{cat}}{3.6}\right]^2 + w_3 \quad (23)$$

$$\max I_2(u) = \left(\frac{1}{1+L_{cat}}\right) + w_1\left[1-\frac{X_{PA}}{1.2}\right]^2 + w_2\left[1-\frac{L_{cat}}{3.6}\right]^2 + w_3 \quad (24)$$

$$\max I_3(u) = \left(\frac{1}{1+\mathrm{CO_x}}\right) + w_1\left[1-\frac{X_{PA}}{1.2}\right]^2 + w_2\left[1-\frac{L_{cat}}{3.6}\right]^2 + w_3 \quad (25)$$

where, minimization of CO_x is converted (Deb, 2001) to maximization problem as $\left(\frac{1}{1+\mathrm{CO_x}}\right)$.

The constraints are defined by equations (7) – (10), bounds: equations (11) – (18). All other details are same as in the problem 1.

4. RESULTS AND DISCUSSION

The above explained MOO problems are solved using RNSGA-II-SBJG. The values of GA parameters: population size (N_P), crossover probability (P_{cros}), mutation probability (P_{mut}), jumping genes probability (P_{JG}), crossover index (η_{cros}), mutation index (η_{mut}) and jumping genes index (η_{JG}) are taken (Ramteke et al., 2014) as 100, 0.9, 0.5/total decision variables, 20, 20 and $IR \in [5, 20]$, respectively. The algorithms are executed on a desktop computer (Intel Xeon E3 – 1225 v3@3.20 GHz processor, 8 GB RAM and Windows 7 operating system).

and RNSGA-II-RJG2. Figure 2a shows that the results obtained using RNSGA-II-SBJG are superior to that obtained using RNSGA-II, RNSGA-II-RJG1 and RNSGA-II-RJG2. Also, the results with all JG adaptations are superior to that of RNSGA-II. This proves the usefulness of JG operation for real-life MOO problems. Among the JG adaptations, SBJG performs better than the other two adaptations. It is expected to perform better than RJG1 because it uses the distribution function for perturbing the variables instead of complete replacement used in the latter. The complete replacement in RJG1 causes the loss of useful improvement accumulated in the variables over the generations and in turn affects the results badly. Also, SBJG adaptation performs better than RJG2 since it adds a better perturbation process instead of repeating the existing operations of RNSGA-II. Also, the spread of Pareto optimal front for RNSGA-II-SBJG is better than that of other three algorithms. The range of Pareto optimal front, $\left[(X_{PA})_{\max}, (L_{cat})_{\min}\right]$, using RNSGA-II-SBJG is [1.171, 0.407] which is superior to [1.166, 0.543], [1.166, 0.512] and [1.169, 0.450] obtained using NSGA-II, RNSGA-II-RJG1 and RNSGA-II-RJG2, respectively. Further, a point A (1.168, 0.548) in figure 2a is identified on the Pareto optimal front obtained using RNSGA-II-SBJG beyond which the nature of the curve changes in such a way that marginal increase in yield has to be compromised with

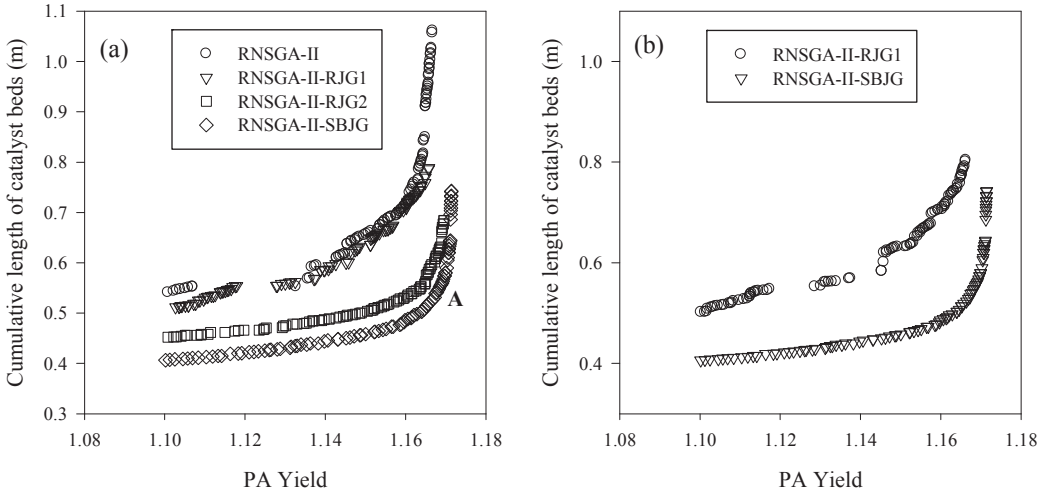


Fig. 2. Comparison of the optimal solutions of problem 1 obtained after 200 generations (a) and using RNSGA-II-RJG1 and RNSGA-II-SBJG for same CPU time (39 hrs) (b).

The Pareto optimal front for problem 1 obtained using RNSGA-II-SBJG for 200th generation is shown in figure 2a. The results are compared with that obtained using RNSGA-II, RNSGA-II-RJG1

significant increase in catalyst loading. Such point can be selected as an operating point. However, more accurate selection requires the in hand experi-

ence and the knowledge of cost factors associated with individual objectives.

The CPU time required for solving 200 generations using RNSGA-II, RNSGA-II-RJG1, RNSGA-II-RJG2 and RNSGA-II-SBJG for problem 1 is 42, 35, 40 and 39 hrs respectively. It shows that CPU time taken by RNSGA-II-SBJG is less than that of RNSGA-II and RNSGA-II-RJG2 and thus performs better in terms of CPU time in addition to convergence as explained above. However, it takes higher CPU time than RNSGA-II-RJG1. In order to compare fairly, both RNSGA-II-SBJG and RNSGA-II-RJG1 are executed for same CPU time of 39 hrs. Also for this case, the results of RNSGA-II-SBJG are found to be considerably better than that of RNSGA-II-RJG1 (see figure 2b).

optimal front shown by an ellipse in figure 3d. This is primarily due to the fact that in most of the instances, values of distribution function α for SBJG remain close to 0 maintaining the local perturbation. This local perturbation keeps the fitness of a perturbed solution under the acceptable limit. Thus, SBJG in RNSGA-II provides an effective exploration of search space unlike other JG adaptations. Further, the diversity of the solutions is maintained in all variants of RNSGA-II using the concept of crowding distance (Deb et al., 2002) in which the less crowded solutions in objective function space are assigned with higher fitness in the selection process.

For problem 2, the CPU time taken by RNSGA-II, RNSGA-II-RJG1, RNSGA-II-RJG2 and

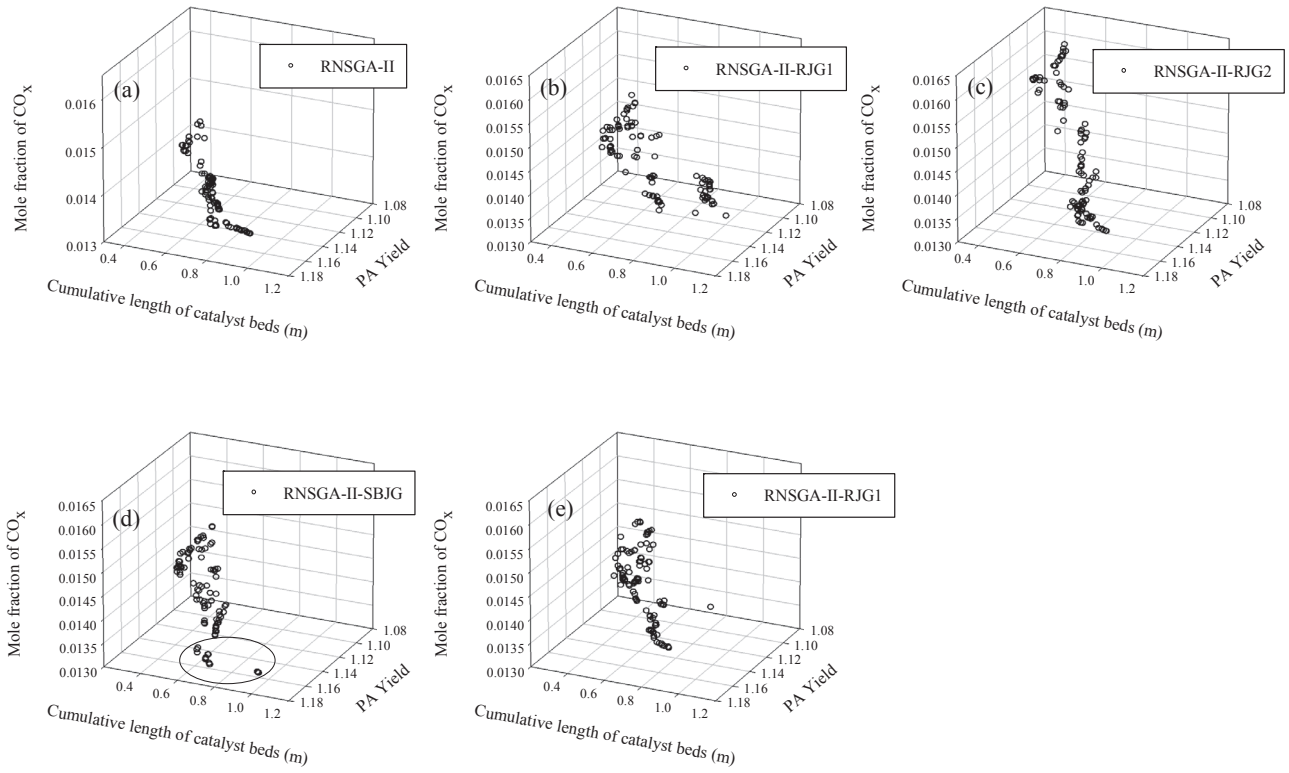


Fig. 3. *Comparison of the optimal solutions of problem 2 obtained after 200 generations using: RNSGA-II (a), RNSGA-II-RJG1 (b), RNSGA-II-RJG2 (c) and RNSGA-II-SBJG (d). The results of RNSGA-II-RJG1 for CPU time of 40 hrs (same as RNSGA-II-SBJG) (e).*

For problem 2, the Pareto optimal fronts using RNSGA-II, RNSGA-II-RJG1, RNSGA-II-RJG2 and RNSGA-II-SBJG are shown in figure 3a-d for 200th generation. Among all adaptations, the Pareto optimal front of RNSGA-II-SBJG is the best in terms of PA yield and cumulative length of catalyst beds whereas it is comparable with that of RNSGA-II and RNSGA-II-RJG1 in terms of the mole fraction of CO_x. Also, some additional points are captured by RNSGA-II-SBJG at the lower end of the Pareto

RNSGA-II-SBJG is 44, 36, 43 and 40 hrs, respectively. Again, the CPU time taken by RNSGA-II-SBJG is less than that of RNSGA-II and RNSGA-II-RJG2 whereas it is higher than that of RNSGA-II-RJG1. Thus, the results of RNSGA-II-SBJG are compared with that of RNSGA-II-RJG1 (see figure 3e) for same CPU time of 40 hrs. The results of RNSGA-II-SBJG show better convergence compared to that of RNSGA-II-RJG1 for same CPU time of 40 hrs. Thus, it can be inferred from the re-

sults of problems 1 and 2 that RNSGA-II-SBJG performs better than all other adaptations.

5. CONCLUSIONS

Two MOO problems related to PA reactor are solved by incorporating a recently developed jumping genes operator, SBJG, in RNSGA-II. The results are found to be better than RNSGA-II and its two JG adaptations i.e., RNSGA-II-RJG1 and RNSGA-II-RJG2. This indicates that SBJG operator could prove quite useful for solving other computationally intense MOO problems similar to those solved in this study. Moreover, the use of SBJG operator is not limited to RNSGA-II and it can be incorporated in other real-coded algorithms such as simulated annealing, particle swarm algorithm, etc. This could be a subject of future studies.

ACKNOWLEDGEMENTS

Partial financial support from the Department of Science and Technology, Government of India, New Delhi (through grant SERB/F/1510/2014-2015, dated June 5, 2014) is gratefully acknowledged.

REFERENCES

Bhat, G.R., Gupta, S.K., 2008, MO Optimization of Phthalic Anhydride Industrial Catalytic Reactors using Guided GA with the Adapted Jumping Gene Operator, *Chem Eng Res Des*, 86, 959–976.

Bhat, S.A., 2007, *On-Line Optimizing Control of Bulk Free Radical Polymerization of Methyl Methacrylate in a Batch Reactor using Virtual Instrumentation*, PhD thesis, Indian Institute of Technology, Kanpur.

Bhat, S.A., Gupta, S., Saraf, D.N., Gupta, S.K., 2006, On-Line Optimizing Control of Bulk Free Radical Polymerization Reactors under Temporary Loss of Temperature Regulation: An Experimental Study on a 1-Liter Batch Reactor, *Ind Eng Chem Res*, 45, 7530–7539.

Deb, K., 2001, *Multi-objective Optimization Using Evolutionary Algorithms*, Wiley, New York.

Deb, K., Agrawal, R.B., 1995, Simulated Binary Crossover for Continuous Search Space, *Complex Systems*, 9, 115–148.

Deb, K., Agrawal, S., 1999, A Niched-Penalty Approach for Constraint Handling in Genetic Algorithms, *Proc. Conf. Artificial Neural Networks and Genetic Algorithms ICANNGA'99*, eds, Dobnikar, A., Steele, N.C., Pearson, D.W., Albrecht, R.F., Portoroz, 235–243.

Deb, K., Agrawal, S., Pratap, A., Meyarivan, T., 2002, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, *IEEE T Evolut Comput*, 6, 182–197.

Holland, J.H., 1975, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge.

Kasat, R.B., Gupta, S.K., 2003, Multi-Objective Optimization of an Industrial Fluidized-Bed Catalytic Cracking Unit (FCCU) Using Genetic Algorithm (GA) with the Jumping Genes Operator, *Comput Chem Eng*, 27, 1785–1800.

McClintock, B., 1987, *The Collected Papers of Barbara McClintock*, Garland, New York.

Ramteke, M., Ghune, N., Trivedi, V., 2014, Simulated Binary Jumping Gene: A Step Towards Enhancing the Performance of Real-Coded Genetic Algorithm, submitted.

Ripon, K.S.N., Kwong, S., Man, K.F., 2007, A Real-Coding Jumping Gene Genetic Algorithm (RJGGA) for Multiobjective Optimization, *Inform Sciences*, 177, 632–654.

Sankararao, B., Gupta, S.K., 2006, Multi-Objective Optimization of the Dynamic Operation of an Industrial Steam Reformer Using the Jumping Gene Adaptations of Simulated Annealing, *Asia-Pac J Chem Eng*, 1, 21–31.

Skrzypek, J., Grzesik, M., Galantowicz, M., Solinski, J., 1985, Kinetics of Catalytic Air Oxidation of o-Xylene over a Commercial V_2O_5–TiO_2 Catalyst, *Chem Eng Sci*, 40, 611–620.

WIELOKRYTERIALNA OPTYMALIZACJA REAKTORA BEZWODNIKA FTALOWEGO KATALITYCZNEGO Z WYKORZYSTANIEM ALGORYTMU GENETYCZNEGO Z OPERATOREM PRZESUNIĘCIA GENÓW

Streszczenie

Wielokryterialna optymalizacja problemów przemysłu chemicznego rozwiązywana jest z powodzeniem z wykorzystaniem algorytmów ewolucyjnych (EAs). Jednak w związku z długimi czasami obliczeń, dotychczasowe algorytmy są modyfikowane dla poprawy szybkości zbieżności. Jedną z możliwości jest idea „przesuwania genów", zaadoptowana dla algorytmu genetycznego z kodowaniem binarnym, która znacząco poprawiła działanie algorytmu. Jednakże wprowadzenie tej modyfikacji do algorytmu z kodowaniem zmiennoprzecinkowym nie przyniosło spodziewanych efektów. Rozwiązaniem tego problemu było wprowadzenie nowego operatora przesunięcia genów do algorytmu RNSGA-II, a mianowicie symulowanego binarnego przesunięcia genów (SBJG). Celem pracy była ocena możliwości zastosowania algorytmu SBJG do rozwiązywania rzeczywistych problemów przemysłowych na przykładzie wielokryterialnej optymalizacji (MOO) reaktora bezwodnika ftalowego katalitycznego. Otrzymane wyniki pokazały lepszą zbieżność zastosowanego algorytmu w porównaniu z metodą RNSGA-II i innymi dotychczasowymi adaptacjami operatora przesunięcia genów zarówno dla dwu- jak i trój-kryterialnego sformułowania problemu.

Received: September 29, 2014
Received in a revised form: November 11, 2014
Accepted: November 20, 2014