Part II

Multi-Objective Optimization Developments

Multi-Objective Optimization in Chemical Engineering: Developments and Applications, First Edition.
Edited by Gade Pandu Rangaiah and Adrián Bonilla-Petriciolet.

4

Performance Comparison of Jumping Gene Adaptations of the Elitist Non-dominated Sorting Genetic Algorithm

Shivom Sharma[1], Seyed Reza Nabavi[2] and Gade Pandu Rangaiah[1]
[1]Department of Chemical and Biomolecular Engineering, National University of Singapore, Singapore
[2]Faculty of Chemistry, University of Mazandaran, Iran

4.1 Introduction

Many applications of optimization involve more than one objective function. To solve such multi-objective optimization (MOO) problems, Deb *et al.* (2002) have developed the elitist non-dominated sorting genetic algorithm (NSGA-II), which has found many applications in chemical engineering. In order to improve the performance of the binary-coded NSGA-II algorithm, Kasat and Gupta (2003) have included the jumping gene (JG) operator in it. Following this, several variants of JG adaptations have been developed and applied to solve a number of application problems. Guria *et al.* (2005a) have developed one variant of JG adaptation, referred to as mJG, for problems having optimal solutions near to bounds on decision variables. Bhat *et al.* (2006) have proposed aJG variant, which was later used with NSGA-II in Bhat (2007). Agarwal and Gupta (2008a) have suggested two new variants of JG adaptations, namely, sJG and saJG, and studied them with binary-coded NSGA-II. NSGA-II-saJG can only be applied if the number of binaries used for representing each

Multi-Objective Optimization in Chemical Engineering: Developments and Applications, First Edition.
Edited by Gade Pandu Rangaiah and Adrián Bonilla-Petriciolet.

decision variable is the same, whereas NSGA-II-sJG completely replaces part of the chromosome associated with a particular decision variable and so can be used even if the number of binaries used is not the same for different variables. Agarwal and Gupta (2008a) also compared four variants of JG adaptations (namely, JG, aJG, sJG and saJG) on three unconstrained test functions. Set convergence ratio, spacing and maximum spread are used as performance indicators, and it was found that performance of NSGA-II-aJG, NSGA-II-sJG and NSGA-II-saJG is comparable, whereas NSGA-II-JG is outperformed by the other three.

Ramteke and Gupta (2009a) have discussed and evaluated five variants of JG adaptations, namely, NSGA-II-JG/mJG/aJG/saJG/sJG, on three unconstrained test functions. Recently, two more variants of JG adaptations, namely, Alt-NSGA-II-aJG (Ramteke and Gupta, 2009b) and biogenetic-NSGA-II-aJG (Ramteke and Gupta, 2009c), were proposed. Alt-NSGA-II-aJG mimics biological altruism from the honey bee to solve MOO problems. In biogenetic-NSGA-II-aJG, information/solution from an earlier optimization problem is used to solve the modified/new optimization problem. This strategy can be used with other JG variants too, and is relevant for modifications in industrial optimization problems, such as an increase in the number of objectives, decision variables, and/or ranges of decision variables.

In order to speed up the convergence, Ripon *et al.* (2007) and Furtuna *et al.* (2011) applied the concept of jumping gene in real coded NSGA-II. In the work of Ripon *et al.* (2007), part of the chromosome (i.e., transposon) is copied/cut-and-pasted into the same or a different chromosome. Its performance has been compared with seven MOO algorithms on five test functions, using convergence metric, spacing, spread, and hyper volume as performance metrics. Mostly, the proposed JG adaptation performed better than other algorithms, in terms of diversity of non-dominated solutions and convergence to the known Pareto-optimal front. Furtuna *et al.* (2011) adapted the JG proposed by Kasat and Gupta (2003) for the real coded NSGA-II. The JG and aJG variants (of binary coded NSGA-II) are also used with multi-objective simulated annealing (Sankararao and Gupta, 2006, 2007a, 2007b). So, the jumping gene concept has potential for use with other MOO algorithms.

In summary, a number of JG adaptations have been proposed and applied to chemical engineering problems since the early 2000s, and some of these have been compared on a limited number of problems (Agarwal and Gupta, 2008a; Ramteke and Gupta, 2009a). However, there has been no comprehensive and systematic evaluation of these adaptations. Further, NSGA-II has been popular for solving application problems. Hence, this chapter analyzes and compares the performance of four variants of JG adaptations, namely, NSGA-II-aJG, NSGA-II-saJG, NSGA-II-sJG and Alt-NSGA-II-aJG, for bi-objective optimization problems. As in the earlier studies, which proposed these adaptations, binary coding is used for representing variables. In this work, NSGA-II-mJG and biogenetic-NSGA-II-aJG are not considered because the former's applicability is for a specific type of optimization problem and the latter is similar to NSGA-II-aJG except for the difference in the approach, which can be used with other adaptations/algorithms as well.

Many application problems have constraints; hence, performance of the above four JG adaptations is compared on four constrained and five unconstrained test functions. Furthermore, several accepted and/or applied performance metrics are used for performance comparison, at intermediate generations and also at the end of the search. Search termination at the right time improves efficiency of the algorithm; hence, a termination criterion, based on the improvement in non-dominated solutions obtained in different generations, is also

tested with the four selected JG adaptations of NSGA-II. A true (i.e., known) Pareto-optimal front, which is to be found for a new application problem, is not used in the development of this termination criterion.

The rest of this chapter is organized as follows. The next section of this chapter briefly discusses different variants of JG adaptations and their applications. Section 4.3 describes performance metrics and the termination criterion for MOO algorithms. Details on constraint handling and program implementation including values of algorithm parameters are given in section 4.4. Section 4.5 compares the performance of selected JG adaptations on many test functions. Finally, useful findings of this work are summarized in the last section of this chapter.

4.2 Jumping Gene Adaptations

A multi-objective optimization problem for k different objectives: $f_1, f_2, \ldots, f_k$, can be stated mathematically as follows:

$$\text{Minimize} \quad \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots f_k(\mathbf{x})\} \tag{4.1}$$

$$\text{Subject to} \quad x^L \leq \mathbf{x} \leq \mathbf{x}^U \tag{4.2}$$

$$\mathbf{g}(\mathbf{x}) \leq \mathbf{b} \tag{4.3}$$

Here, $\mathbf{x}$ is the vector of decision variables with lower (i.e., $\mathbf{x}^L$) and upper (i.e, $\mathbf{x}^U$) bounds; $\mathbf{g}$ is the set of inequality constraints where $\mathbf{b}$ is the vector of constants. If an optimization problem has equality constraints, then those can be converted into inequality constraints by relaxation.

A detailed flowchart of NSGA-II with JG adaptation for MOO is given in Figure 4.1. More details on NSGA-II can be found in Deb (2001). NSGA-II can be implemented using binary or real coding for the values of decision variables. Here, binary coding is used, wherein a number of bits is used to represent each and every decision variable of a (trial) individual (i.e., solution), also known as a chromosome. JG adaptations are mostly developed for binary coded NSGA-II, and programs for different JG adaptations are readily available (www.iitk.ac.in/che/skg.htm). Hence, binary coding is selected in this study. The number of binaries, l_{string} used for each variable may or may not be the same depending on the accuracy required and variable type (i.e., integer or continuous variable). For simplicity, only the common termination using the maximum number of generations (MNG) is shown in the flowchart in Figure 4.1.

Different JG adaptations are explained below and also shown schematically in Figure 4.2. For this, a chromosome is assumed to have four decision variables with 6, 7 or 8 bits used for each variable value. In total, the length of the chromosome, n_{chr} is 28.

- In NSGA-II-JG (Kasat and Gupta, 2003), part of the chromosome is randomly replaced based on JG probability (p_{JG}). For this, two random positions between 1 and n_{chr} (p_1 and p_2 in Figure 4.2a) are selected, and bits between these positions are randomly replaced.
- In NSGA-II-mJG (Guria *et al.*, 2005a), each chromosome undergoes mJG adaptation as per specified JG probability. In this, all bits of a randomly selected decision variable (e.g., third decision variable in Figure 4.2b) of a chromosome are changed to either zeros or

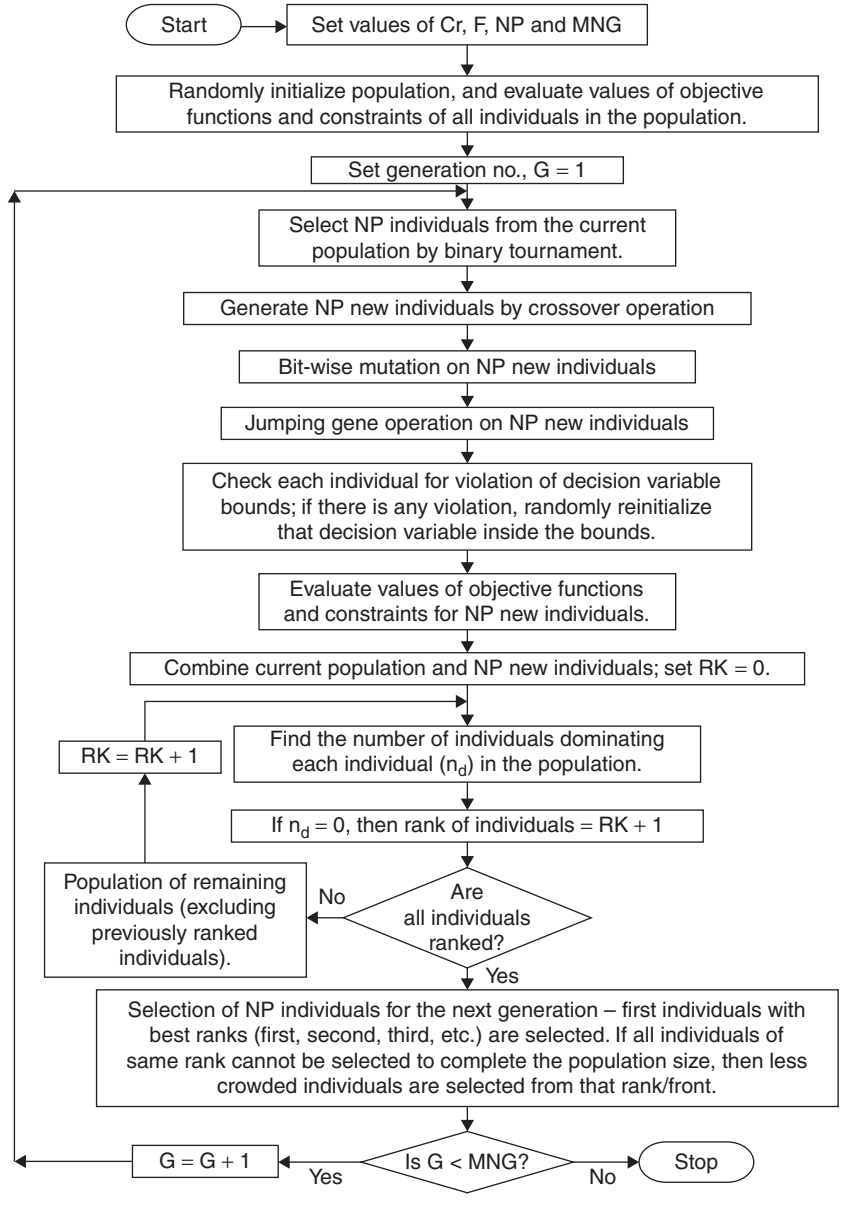

Figure 4.1 *A detailed flowchart of NSGA-II with JG adaptation for MOO.*

ones with equal probability. This replacement of bits is different from other adaptations discussed, where bits are replaced by zeros and ones randomly.

- In NSGA-II-aJG (Bhat *et al.*, 2006; Bhat, 2007), the fixed length of the chromosome is randomly changed; JG probability is used to make the decision on this partial replacement. One random position (p_1) between 1 and ($n_{chr} - f_b$) is selected to replace f_b number

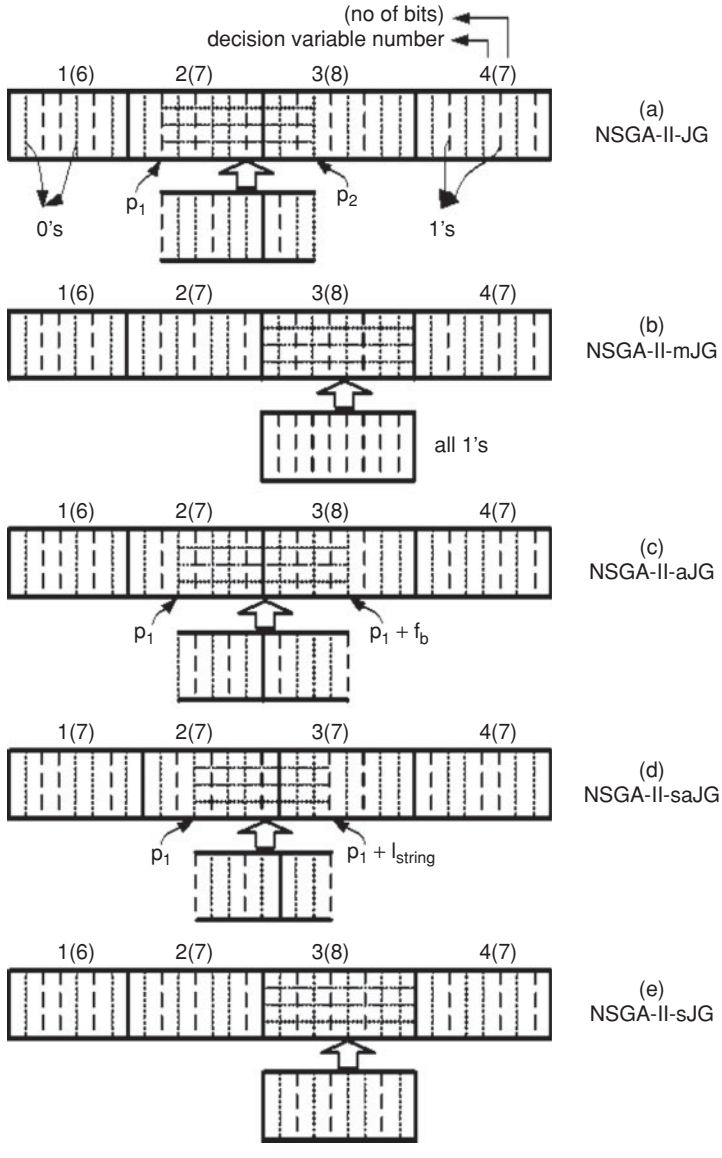

Figure 4.2 *Schematic of JG adaptations.*

of bits, where f_b (= 10 in Figure 4.2c) is any arbitrary number between 1 and n_{chr}. Thus, f_b is another parameter of this algorithm, to be specified by the user.

- NSGA-II-saJG (Agarwal and Gupta, 2008a) requires an equal number of bits (l_{string}) for each decision variable. If the probability allows saJG adaptation of a chromosome, then one random position (p_1) between 1 and ($n_{chr} - l_{string}$) is chosen to replace l_{string} (= 7 in Figure 4.2d) bits from p_1, randomly.

- In NSGA-II-sJG (Agarwal and Gupta, 2008a), all bits of a selected decision variable are changed randomly. JG probability is used to decide sJG adaptation of each chromosome, and the selection of a particular decision variable (e.g., the third variable in Figure 4.2e) is random. Except for the random replacement of a particular decision variable, sJG is similar to mJG.
- In Alt-NSGA-II-aJG (Ramateke and Gupta, 2009b), the jumping gene operator is the same as that in NSGA-II-aJG, but the selection of individuals for reproduction operation is not random. It mimics biological altruism in hymenopterans species (mostly honey bees) to solve MOO problems. In order to get the maximum benefit of the queen in the optimization, more than one queen (e.g., ten) and two- or three-mate crossover strategies are used in Alt-NSGAII-aJG. These queens are selected in the beginning of each generation based on their crowding distance indices. In the altruistic adaptation, all genetic operations (crossover, mutation and aJG) are carried out only between the one of the queen chromosomes and one of the worker-bee chromosomes. In non-altruistic adaptations, these operators are carried out for randomly selected chromosomes based on their probability.

Several researchers have successfully used one or more variants of JG adaptations to solve different application problems; these applications are summarized in Table 4.1. Mathematical functions tested in these studies are also included in this table. Although many applications have been studied using JG adaptations, their evaluation using mathematical functions is limited, as can be seen in Table 4.1.

4.3 Termination Criterion

In this chapter, a performance-based termination criterion is employed, instead of MNG, to compare the performance of JG adaptations. It monitors the improvement in the non-dominated solutions obtained in recent generations using selected performance metrics, and decides the search termination using statistical tests. Several performance metrics have been proposed in MOO literature to evaluate the quality of the obtained optimal solutions compared to the known Pareto-optimal front; these include generational distance (GD) (Van Veldhuizen and Lamont, 1998), spread (SP) (Deb *et al.*, 2000) and hyper volume (HV) (Zitzler and Thiele, 1998). These have been used mainly to evaluate the quality of the obtained non-dominated solutions by an algorithm with respect to the known Pareto-optimal front, after MNG. However, their usage for search termination is very limited.

The true/known Pareto-optimal front is yet to be found and so is unavailable in advance for new application problems; hence, original performance metrics cannot be used to monitor the search progress. Wagner *et al.* (2009) have developed a termination criterion using different performance metrics (HV, R2 indicator and ε_+ indicator). Sharma (2013) has modified GD and SP for the non-dominated solutions obtained in consecutive generations, and then used them to develop a search termination criterion.

Generational distance is calculated between the non-dominated solutions obtained in the current generation and its previous generation, using the following equation:

$$\mathrm{GD} = \frac{1}{\mathrm{NP}} \sqrt{\sum_{i=1}^{\mathrm{NP}} d_i^2} \tag{4.4}$$

Table 4.1 *Use of different JG adaptations to solve application problems.*

Application	Test functions	Algorithm(s)	Reference(s)
Fluidized-bed catalytic cracker	Schaffer, ZDT4	NSGA-II and NSGA-II-JG	Kasat and Gupta (2003)
Flotation circuits	–	NSGA-II (binary & real coded) and NSGA-II-mJG	Guria *et al.* (2005a and 2006)
Reverse osmosis desalination units	–	NSGA-II, NSGA-II-JG and aJG	Guria *et al.* (2005b)
Industrial steam reformer	–	MOSA, MOSA-JG, MOSA-aJG, NSGA-II, NSGA-II-JG, NSGA-II-aJG	Sankararao and Gupta (2006)
Low density poly-ethylene reactor	CONSTR, SRN, TNK, WATER	NSGA-II, NSGA-II-JG and aJG	Agrawal *et al.* (2006 and 2007)
Fuel-oil blending	–	NSGA-II, NSGA-II-JG and aJG	Khosla *et al.* (2007)
Pressure swing adsorbers for air separation	–	MOSA-aJG	Sankararao and Gupta (2007a)
Fluidized-bed catalytic cracker	ZDT4, two more test functions	NSGA-II, NSGA-II-JG, NSGA-II-aJG, MOSA-JG, MOSA-aJG	Sankararao and Gupta (2007b)
Shell and tube heat exchangers	ZDT2, ZDT3, ZDT4	NSGA-II-JG, aJG, sJG and saJG	Agarwal and Gupta (2008a and 2008b)
Industrial phthalic anhydride (PA) reactor	–	NSGA-II-aJG and guided NSGA-II-aJG	Bhat and Gupta (2008)
Nylon-6 semi batch reactor	–	NSGA-II, NSGA-II-aJG, MOSA-aJG	Mitra *et al.* (1998), Ramteke and Gupta (2008)
Industrial PA reactor system, and simulation of cancer	ZDT2, ZDT3, ZDT4	Alt-NSGA-II-aJG	Ramteke and Gupta (2009b)
Industrial PA reactor system; nylon-6 polymerization	Two modified test functions	NSGA-II-aJG, B-NSGA-II-aJG	Ramteke and Gupta (2009c)
Liquefied petroleum gas (LPG) cracker	–	NSGA-II-aJG	Nabavi *et al.* (2009)
Synthesis of polymeric nano-particles		NSGA-II (real coded), NSGA-RJG	Furtuna *et al.* (2011)
Fixed bed maleic anhydride reactor	–	NSGA-II-aJG, Alt-NSGA-II-aJG	Chaudhari and Gupta (2012)

Here, NP is the number of non-dominated solutions obtained in the previous generation, and d_i is the Euclidean distance of each of these solutions to its nearest non-dominated solution in the current generation.

Spread was introduced by Deb *et al.* (2000) for bi-objective optimization problems to measure the distribution of non-dominated solutions. Later, Zhou *et al.* (2006) extended the spread indicator to more than two objectives. The modified SP calculation requires the Euclidean distance between two neighboring, non-dominated solutions in the current generation.

$$SP = \frac{\sum_{j=1}^{NP} |d_j - \bar{d}|}{NP\bar{d}} \tag{4.5}$$

where

$$d_j = \min \parallel S_j - S_k \parallel^2 \text{ with respect to } k = 1, 2, \ldots, NP \text{ (except } k = j) \tag{4.6}$$

Here, S is the set of NP non-dominated solutions obtained in the current generation, d_j is the Euclidean distance of solution S_j to solution S_k, and $\bar{d}$ is the average of d_j for all solutions in the set S.

For the development of the termination criterion, GD and SP values obtained for λ number of recent generations are used for χ^2-test (i.e., to test their variation), as follows.

$$\text{Chi (PI)} = \frac{\text{Variance}\,[PI_1, PI_2, \ldots, PI_\lambda]\,(\lambda - 1)}{\delta_{PI}^2} \tag{4.7}$$

$$p(PI) = \chi^2[\text{Chi (PI)}, (\lambda - 1)] \tag{4.8}$$

Here, PI is the performance metric, which can be GD or SP, and δ_{PI} is the threshold value for the standard deviation of PI values. In Equation (4.8), p is the probability that χ^2-test supports the hypothesis that variance in PI values obtained in λ number of recent generations is lower than the threshold value (δ_{PI}). If this probability is more than 99% for GD and also for SP simultaneously, then global search is terminated. To avoid indefinite looping, termination criterion based on MNG is also included in the program.

4.4 Constraint Handling and Implementation of Programs

Penalty function and feasibility criterion are the two popular approaches for handling constraints within evolutionary algorithms. In this work, penalty function approach is used to handle inequality constraints. In this approach, objective functions are penalized (i.e., modified) by adding a penalty term to each of the original objective functions, as follows.

$$F_i(\mathbf{x}) = f_i(\mathbf{x}) + \sum_{j=1}^{n_c} R_j \max\left[0, G_j(\mathbf{x})\right] \tag{4.9}$$

Here, F_i and f_i are i^{th} modified and original objective function respectively, $G_j(\mathbf{x})$ is j^{th} inequality constraint (defined in Equation 4.10 below), R_j is the user-defined penalty parameter for j^{th} inequality constraint and n_c is the number of inequality constraints. In

order to use a single-penalty parameter for all inequality constraints in Equation 4.3, they are normalized using the following transformation.

$$G_j(x) = g_j(x)/b_j - 1 \leq 0 \tag{4.10}$$

Qualities of the non-dominated solutions obtained at different generations are assessed using GD^t, SP^t and IGD^t along with the known Pareto-optimal front. Note that GD^t and SP^t for this purpose are slightly different from those used for search termination, which do not require the known Pareto-optimal front. GD^t is calculated based on Equation (4.4) using the non-dominated solutions from the best front obtained in the current generation and the known Pareto-optimal front. SP is calculated using the following equation (Zhou *et al.*, 2006):

$$SP^t = \frac{\sum_{m=1}^{M} d(e_m, S) + \sum_{j=1}^{NP} |d_j - \bar{d}|}{\sum_{m=1}^{M} d(e_m, S) + NP\bar{d}} \tag{4.11}$$

Here, M is the number of objective functions, $\{e_1, e_2, \ldots, e_M\}$ are M boundary solutions from the known Pareto-optimal front, and $d(e_m, S)$ is the Euclidean distance between the extreme solution of m^{th} objective in the known Pareto-optimal front to its nearest non-dominated solution obtained. The remaining symbols used in Equation 4.11 are defined in the previous section.

IGD^t, similar to GD^t, is calculated between the non-dominated solutions from the best front obtained in the current generation and the known Pareto-optimal front:

$$IGD^t = \frac{1}{NT} \sum_{n=1}^{NT} d_n \tag{4.12}$$

Here, NT is the number of solutions in the known Pareto-optimal front, and d_n is the Euclidean distance of each solution in the known Pareto-optimal front to its nearest solution in the obtained Pareto-optimal front. Like GD^t, IGD^t also determines the closeness of the non-dominated solutions obtained to the known Pareto-optimal front but it calculates the closeness in the opposite direction from the known Pareto-optimal front. IGD can measure both convergence and diversity of the obtained non-dominated solutions (Zhang *et al.*, 2009).

FORTRAN programs for NSGA-II-aJG and Alt-NSGA-II-aJG have been taken from www.iitk.ac.in/che/skg.htm (accessed November 26, 2012), and then modified for NSGA-II-sJG and NSGA-II-saJG. All these programs have been amended to include GD and SP calculations, and to implement the χ^2-test for termination, as discussed in section 4.3. Moreover, GD^t, SP^t and IGD^t calculations are also implemented to compare the obtained Pareto-optimal front with the known Pareto-optimal front at intermediate generations and also at the end of search.

Parameters used in the termination criterion are: $\lambda = 10$, $\delta_{GD} = 0.0002$ and $\delta_{SP} = 0.03$. A large, fixed value of the penalty parameter ($R = 10^9$) is used for all constrained problems. As stated earlier, this chapter analyzes and compares the performance of four JG adaptations: NSGA-II-aJG, NSGA-II-saJG, NSGA-II-sJG and Alt-NSGA-II-aJG. Values of the parameters in these algorithms used in this study are chosen based on the values used and/or recommendations in Agarwal and Gupta (2008a) and Ramteke and Gupta (2009b), and these are given in Table 4.2.

Table 4.2 *Values of parameters in JG adaptations of NSGA-II used in this study.*

Parameter	aJG	saJG	sJG	Alt-aJG
NP (population size)	200	200	200	200
MNG (maximum number of generations)	1500	1500	1500	1500
p_c (crossover probability)	0.9	0.9	0.9	0.9
p_m (mutation probability)	0.001	0.001	0.001	0.001
p_{JG} (JG probability)	0.5	0.5	0.5	0.5
f_b (arbitrary number used in aJG operator)	25 or 10[a]	–	–	25 or 10[a]
l_{string} (no. of bits for each decision variable)	30	30	30	30

[a]For constrained problems

4.5 Performance Comparison

Performance of the selected JG adaptations of NSGA-II is compared on five bi-objective unconstrained test functions: ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 (Zitzler *et al.*, 2000) and on four bi-objective constrained test functions: Osyczka, CONSTR, SRN and TNK (Coello Coello *et al.*, 2007). The unconstrained test functions have different characteristics like continuous or discontinuous objective functions, multi-modality and convexity of search space. ZDT1 and ZDT2 test functions have convex and non-convex Pareto-optimal front respectively. ZDT3 has several non-continuous convex parts in the Pareto-optimal front. ZDT4 is multi-modal in nature and has 99 local optimal fronts (Sindhya *et al.*, 2011), and ZDT6 problem has non-uniform density of solutions. The constrained test functions: Osyczka, CONSTR, SRN, and TNK are considered for testing because many applications involve constraints. Main details of unconstrained and constrained test functions are given in Tables 4.3 and 4.4 respectively.

All the non-dominated solutions obtained in the current generation are used to calculate values of GD^t, SP^t and IGD^t for assessing their quality. For the termination criterion, all the non-dominated solutions obtained in the current and its previous generations are used to calculate GD and SP values. The maximum number of non-dominated solutions possible in a generation is NP, population size.

4.5.1 Performance Comparison on Unconstrained Test Functions

Values of GD^t and IGD^t vary significantly with generations and also for different algorithms, whereas SP^t does not change much. Hence, for ease of comparison among different algorithms, GD^t and IGD^t values are normalized using certain GD^t_{max} and IGD^t_{max}. GD^t and IGD^t values are larger in the beginning of search, and decrease slowly with the progress of search. Further, there can be some fluctuations in GD^t and IGD^t values at the start of search due to change in the number of non-dominated solutions in the best Pareto-optimal front obtained. However, GD^t and IGD^t vary smoothly after some generations (e.g., 100). Hence, maximum values of GD^t and IGD^t obtained after 100 generations using four different algorithms are considered as GD^t_{max} and IGD^t_{max}. Table 4.5 presents GD^t_{max} and IGD^t_{max} for different unconstrained and constrained test problems. As mentioned earlier

Table 4.3 *Unconstrained test functions used in this work; DVs—decision variables.*

Test function	No of DVs	Range of DVs	Min. $f_1(x)$	Min. $f_2(x, g(x))$	$g(x)$
ZDT1	30	$x_i \in [0, 1]$	x_1	$g(\mathbf{x})\left[1 - \sqrt{\frac{f_1}{g(x)}}\right]$	$1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
ZDT2	30	$x_i \in [0, 1]$	x_1	$g(\mathbf{x})\left[1 - \left(\frac{f_1}{g(x)}\right)^2\right]$	$1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
ZDT3	30	$x_i \in [0, 1]$	x_1	$g(\mathbf{x})\left[1 - \sqrt{\frac{f_1}{g(x)}} - \frac{f_1}{g(x)}\sin(10\pi f_1)\right]$	$1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$
ZDT4	10	$x_1 \in [0, 1]$, $x_i \in [-5, 5]$	x_1	$g(\mathbf{x})\left[1 - \sqrt{\frac{f_1}{g(x)}}\right]$	$1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i))$
ZDT6	10	$x_i \in [0, 1]$	$1 - \exp(-4x_1)\sin^6(6\pi x_1)$	$g(\mathbf{x})\left[1 - \left(\frac{f_1}{g(x)}\right)^2\right]$	$1 + 9\left[\frac{\sum_{i=2}^{n} x_i}{9}\right]^{0.25}$

Table 4.4 *Constrained test functions studied in this work.*

Test function	Range of decision variables	Objectives (minimize)	Constraints
Osyczka	$(x_1, x_2, x_6) \in [0, 10]$, $(x_3, x_5) \in [1, 5]$, $x_4 \in [0, 6]$,	$f_1(\mathbf{x}) = -[25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 + (x_4 - 4)^2 + (x_5 - 1)^2]$ $f_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$	$x_1 + x_2 - 2 \geq 0$ $-x_1 - x_2 + 6 \geq 0$ $x_1 - x_2 + 2 \geq 0$ $-x_1 + 3x_2 + 2 \geq 0$ $-(x_3 - 3)^2 - x_4 + 4 \geq 0$ $(x_5 - 3)^2 + x_6 - 4 \geq 0$
CONSTR	$x_1 \in [0.1, 1]$, $x_2 \in [0, 5]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = (1 + x_2)/x_1$	$9x_1 + x_2 - 6 \geq 0$ $9x_1 - x_2 - 1 \geq 0$
SRN	$(x_1, x_2) \in [-20, 20]$	$f_1(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2 + 2$ $f_2(\mathbf{x}) = 9x_1 - (x_2 - 1)^2$	$-x_1^2 - x_2^2 + 225 \geq 0$ $-x_1 + 3x_2 - 10 \geq 0$
TNK	$(x_1, x_2) \in [0, \pi]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = x_2$	$x_1^2 + x_2^2 - 1 - 0.1 \times \cos(16 \arctan(x_1/x_2)) \geq 0$ $-(x_1 - 0.5)^2 - (x_2 - 0.5)^2 + 0.5 \geq 0$

in section 4, GD^t, SP^t and IGD^t are calculated for original values of objectives, and hence significant variation can be observed from problem to problem (Table 4.5).

4.5.1.1 Comparison of JG Adaptations using the Termination Criterion

Table 4.6 presents GD^t/GD^t_{max}, SP^t and IGD^t/IGD^t_{max} for unconstrained test problems using four JG adaptations: NSGA-II-aJG, NSGA-II-saJG, NSGA-II-sJG and Alt-NSGA-II-aJG. Here, the search is stopped using the termination criterion discussed in section 4.3; the termination generation (G_T) of each algorithm for each problems is also given in Table 4.6. These performance metrics values are the average of ten runs with different random number seed values (i.e., 0.05, 0.15, 0.25, 0.35, 0.45, 0.55, 0.65, 0.75, 0.85, 0.95), for each problem with each algorithm. Note that random number seed value affects the series of random numbers generated, which in turn can affect performance of stochastic algorithms. Same set of random seed values and procedure are employed for testing JG adaptations on constrained problems. The best values obtained for a problem by different JG adaptations are identified in bold in Table 4.6 and subsequent tables.

It can be seen in Table 4.6 that Alt-NSGA-II-aJG gives smaller values of GD^t/GD^t_{max} and IGD^t/IGD^t_{max} for the unconstrained functions tested, compared to other three JG

Table 4.5 *Maximum values of GD^t and IGD^t obtained after 100 generations, using four different algorithms.*

PM	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	OSY	CONSTR	SRN	TNK
GD^t_{max}	0.0046	0.0043	0.0038	0.0118	0.0343	1.5156	0.0008	0.0173	0.0008
IGD^t_{max}	0.1155	0.1468	0.1133	0.2631	0.7386	9.1184	0.0341	0.5498	0.0212

Table 4.6 *GD^t/GD^t_{max}, SP^t and IGD^t/IGD^t_{max} for unconstrained test functions obtained by four JG adaptations using termination criterion; these values are the average of ten runs, each with a different random number seed value.*

	Algorithm	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	Total
GD^t/GD^t_{max}	NSGA-II-aJG	0.598	0.860	0.675	0.361	0.275	2.768
	NSGA-II-saJG	0.707	0.866	0.778	0.601	0.244	3.196
	NSGA-II-sJG	0.604	0.720	0.651	0.439	0.215	2.629
	Alt-NSGA-II-aJG	**0.030**	**0.020**	**0.131**	**0.029**	**0.003**	**0.214**
SP^t	NSGA-II-aJG	0.829	1.079	0.978	0.797	1.028	4.711
	NSGA-II-saJG	0.855	1.107	0.963	0.885	1.019	4.828
	NSGA-II-sJG	**0.802**	1.061	**0.904**	0.814	**1.002**	**4.584**
	Alt-NSGA-II-aJG	0.970	**0.983**	1.177	**0.771**	1.004	4.906
IGD^t/IGD^t_{max}	NSGA-II-aJG	0.592	0.733	0.738	0.467	0.287	2.817
	NSGA-II-saJG	0.637	0.763	0.777	0.564	0.280	3.021
	NSGA-II-sJG	0.599	0.627	0.720	0.446	0.223	2.616
	Alt-NSGA-II-aJG	**0.026**	**0.026**	**0.099**	**0.031**	**0.002**	**0.183**
G_T	NSGA-II-aJG	165	136	163	**202**	466	1132
	NSGA-II-saJG	**156**	**135**	163	297	514	1265
	NSGA-II-sJG	181	159	195	251	522	1308
	Alt-NSGA-II-aJG	407	139	**116**	227	**222**	**1111**

adaptations. NSGA-II-sJG and Alt-NSGA-II-aJG gives better values of SP^t on different unconstrained test problems. The Alt-NSGA-II-aJG algorithm performs well on ZDT3 and ZDT6 problems, based on the closeness of the non-dominated solutions obtained for the known Pareto-optimal front (i.e., smaller GD^t/GD^t_{max} and IGD^t/IGD^t_{max}), and it takes the smallest G_T (i.e., 116 and 222). For ZDT2 and ZDT4 problems, Alt-NSGA-II-aJG is superior to the other three adaptations, based on all three performance metrics; here, generations used (i.e., 139 and 227) are also comparable to the smallest G_T for these problems (i.e., 135 and 202). For ZDT1 problem, Alt-NSGA-II-aJG gives significantly smaller values of GD^t/GD^t_{max} and IGD^t/IGD^t_{max} but it takes larger number of generations (i.e., 407 compared to the smallest G_T of 156). NSGA-II-aJG is the second best algorithm for solving ZDT1 problem, which gives smaller GD^t/GD^t_{max}, smaller IGD^t/IGD^t_{max} and comparable SP^t than those obtained by NSGA-II-saJG and NSGA-II-sJG; here, the required number of generations (i.e., 165) is also comparable to those used by NSGA-II-saJG and NSGA-II-sJG algorithms (i.e., 156 and 181 respectively).

The total values of GD^t/GD^t_{max}, SP^t, IGD^t/IGD^t_{max} and G_T for four JG adaptations on the unconstrained functions tested are shown in the last column of Table 4.6. Overall, Alt-NSGA-II-aJG is the best, based on GD^t/GD^t_{max}, IGD^t/IGD^t_{max} and G_T, among the adaptations tested. NSGA-II-sJG is better than other adaptations based on SP^t; Alt-NSGA-II-aJG gives smallest SP^t for ZDT2 and ZDT4 but relatively larger SP^t for ZDT1 and ZDT3 test functions. The NSGA-II-saJG algorithm performs worse than others tested,

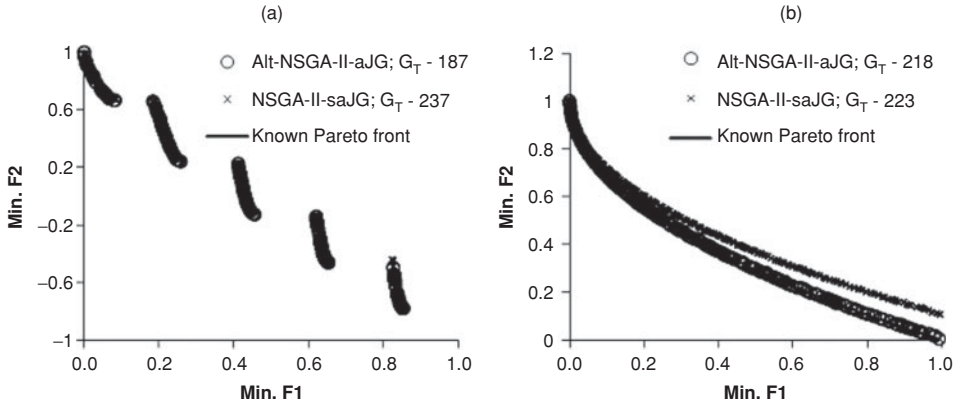

Figure 4.3 *Non-dominated solutions obtained by Alt-NSGA-II-aJG and NSGA-II-saJG algorithms using random seed of 0.05: (a) ZDT3 and (b) ZDT4.*

based on all performance metrics. Figure 4.3 shows the non-dominated solutions obtained by Alt-NSGA-II-aJG (the best adaptation) and NSGA-II-saJG (the worst adaptation) for ZDT3 and ZDT4 test functions. For ZDT3 function, the non-dominated solutions obtained by both these adaptations are closer to the known Pareto-optimal front. The obtained non-dominated solutions by NSGA-II-saJG for ZDT4 function are away from the known Pareto-optimal front, indicating premature convergence to a local Pareto-optimal front (as ZDT4 has 100 distinct Pareto-optimal fronts). However, the global Pareto-optimal front of the ZDT4 problem can be found by NSGA-II-saJG using a larger number of generations, which means that this algorithm is able to escape from the local Pareto-optimal front after some generations of stagnation. On the other hand, Alt-NSGA-II-aJG gives converged solutions closer to the global Pareto-optimal front in fewer generations (see Figure 4.3).

In order to analyze the working of the termination criterion, modified GD and SP values obtained in different generations, for two selected problems (i.e., ZDT3 having a discontinuous Pareto-optimal front and ZDT4 having a multi-modal Pareto-optimal front) using two JG adaptations (best and worst) are shown in Figure 4.4. Figures 4.4(a) and 4.4(b) show respectively variations in GD and SP with generations using Alt-NSGA-II-aJG on ZDT3 test function. G_T based on GD and SP alone are marked in the respective figures with dotted vertical lines. In Figure 4.4(a), search can terminate very early based on GD (i.e., generation 28), but it continues until both modified GD and SP values in the termination criterion satisfy test statistics individually (i.e., $G_T = 187$). Variations in modified GD and SP with generations for ZDT4 function using NSGA-II-saJG are shown in Figure 4.4(c) and 4.4(d); here, search is terminated at 223 generation, using the improvement based termination criterion.

4.5.1.2 Comparison of JG Adaptations at Intermediate Generations

Figure 4.5 shows variations of GD^t/GD^t_{max} and SP^t with generations for different unconstrained problems using the four JG adaptations considered. The IGD^t/IGD^t_{max} variation is not shown in this figure and Figure 4.6 as it is similar to GD^t/GD^t_{max}.

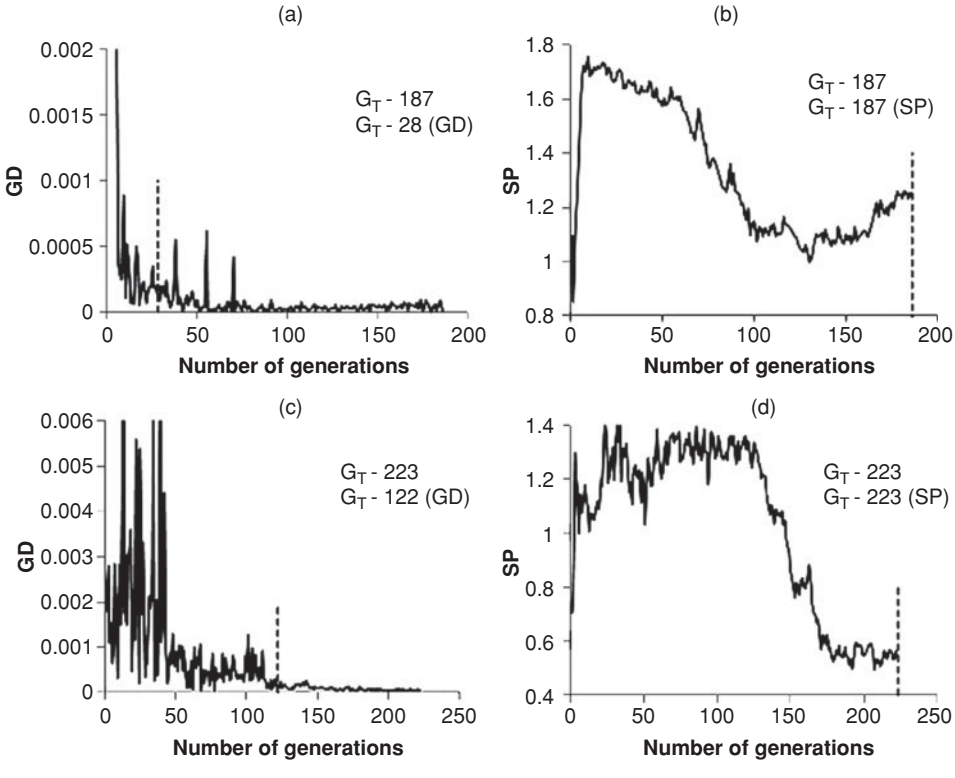

Figure 4.4 *Variations in GD and SP values with generations for: (a) and (b) ZDT3 using Alt-NSGA-II-aJG (random seed = 0.05), and (c) & (d) ZDT4 using NSGA-II-saJG (random seed = 0.05).*

Profiles in Figure 4.5 confirm that, based on GD^t/GD^t_{max}, Alt-NSGA-II-aJG is the best algorithm on all the unconstrained problems studied; the non-dominated solutions obtained by Alt-NSGA-II-aJG are closer to the known Pareto-optimal front, compared to those by other JG adaptations at all generations shown. If all the non-dominated solutions obtained are equally spaced and also contain the extreme solutions, then SP^t should be zero. The performance of NSGA-II-aJG, NSGA-II-saJG and NSGA-II-sJG, based on SP^t, is nearly comparable on the tested unconstrained problems; SP^t decreases initially except for ZDT6, and it is nearly constant towards the end of the search. This is due to variations in the number of non-dominated solutions (during the initial stage of the search) and in the number of duplicate non-dominated solutions with generations (illustrated in Table 4.7).

There are many duplicate non-dominated solutions present in the best Pareto front obtained at different generations, and this number increases with generations. Table 4.7 presents the number of non-dominated solutions and also the number of unique non-dominated solutions obtained for ZDT1 function using NSGA-II-sJG with random seed = 0.05. NSGA-II and NSGA-II-sJG give a comparable number of unique non-dominated solutions until about 500 generations. However, NSGA-II-sJG gives more unique

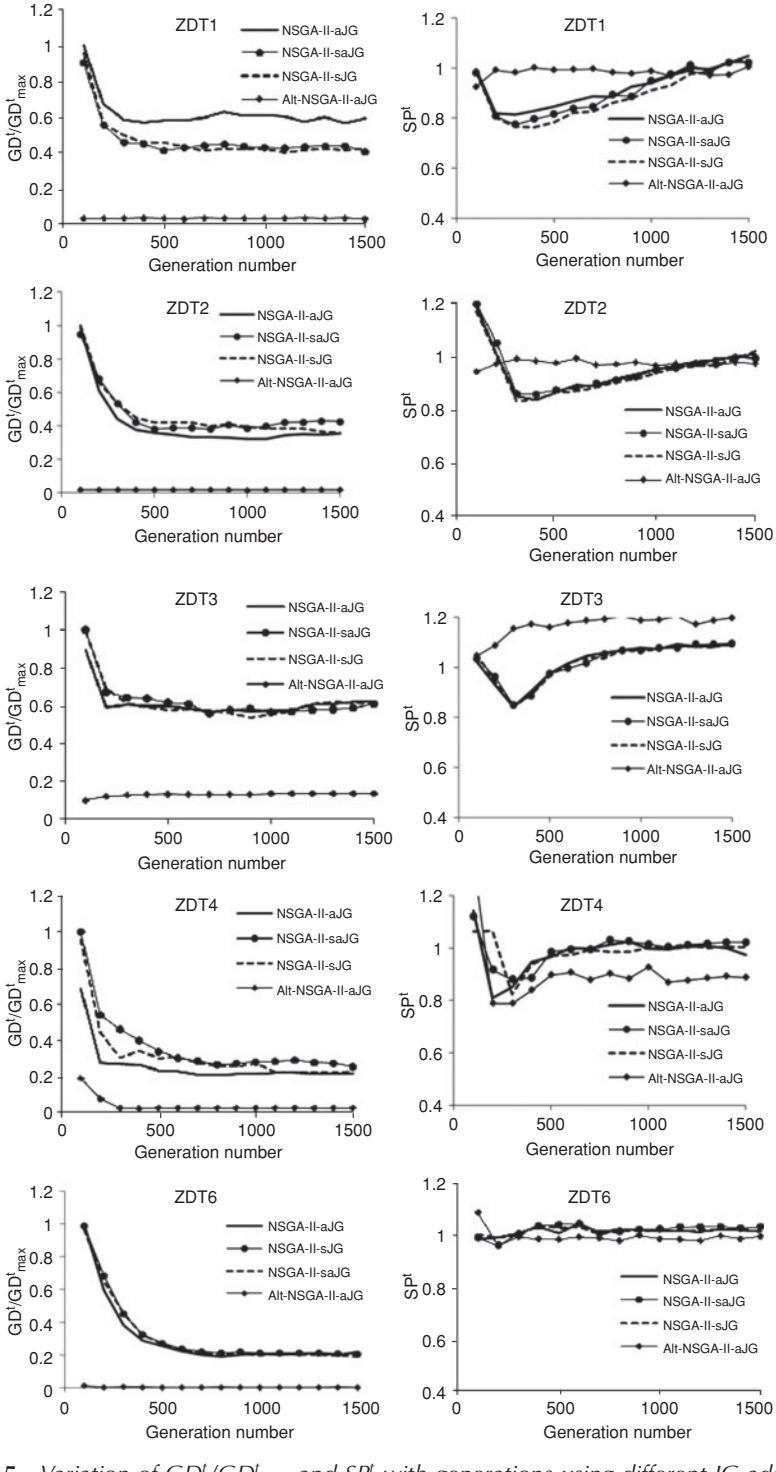

Figure 4.5 *Variation of GD^t/GD^t_{max} and SP^t with generations using different JG adaptations for unconstrained test functions; these profiles are based on an average of 10 runs by each algorithm for each test function.*

Table 4.7 *Variation in the number of unique non-dominated solutions obtained with generations (NSGA-II with and without sJG; test function − ZDT1; random seed = 0.05).*

Generation no.	No. of non-dominated solutions	No. of unique non-dominated solutions using NSGA-II-sJG	No. of unique non-dominated solutions using NSGA-II	SP^t using NSGA-II-sJG
100	160/157*	148	149	1.004
200	200	190	195	0.718
300	200	191	194	0.670
400	200	188	186	0.694
500	200	188	171	0.733
1000	200	172	145	0.923
1500	200	157	143	1.051

* These numbers are respectively in case of NSGA-II-sJG and NSGA-II (i.e., with zero JG probability).

non-dominated solutions after 500 generations, showing that the adaptation of sJG can improve the performance in this aspect. Duan *et al.* (2010) have mentioned that duplicate non-dominated solutions are common in NSGA-II. Fewer unique non-dominated solutions lead to increase in SP^t in later generations (see Table 4.7).

Variation in SP^t, in the event of Alt-NSGA-II-aJG for ZDT1, ZDT2 and ZDT3 problems, follows a slightly different trend than the other JG adaptations (see Figure 4.5). Compared to other JG adaptations tested, Alt-NSGA-II-aJG provides non-dominated solutions with lower SP^t for ZDT4 but higher SP^t for ZDT3.

4.5.2 Performance Comparison on Constrained Test Functions

4.5.2.1 Comparison of JG Adaptations using the Termination Criterion

Table 4.8 shows values of GD^t/GD^t_{max}, SP^t, IGD^t/IGD^t_{max} and G_T for constrained test problems by four JG adaptations using the termination criterion. As mentioned in section 4.5.1.1, these results are the average of ten runs with different random seed values. Alt-NSGA-II-aJG gives the smallest GD^t/GD^t_{max} values for the OSY, CONSTR and TNK problems, and close to the smallest value of GD^t/GD^t_{max} for the SRN problem. It requires the smallest G_T for the OSY, CONSTR and SRN problems, and its G_T (= 115) for the TNK problem is comparable to the best G_T value of 105. All JG adaptations of NSGA-II are comparable based on SP^t for different problems tested; the variation in the total value of SP^t for the different JG adaptations is less than 2%. NSGA-II-aJG gives the smallest values of IGD^t/IGD^t_{max} for OSY, CONSTR and TNK problems, and close to the smallest value of IGD^t/IGD^t_{max} for the SRN problem. Alt-NSGA-II-aJG is computationally more efficient than the other adaptations, while number of generations used by NSGA-II-aJG is significantly more than the best G_T required for solving these problems.

The total of GD^t/GD^t_{max}, SP^t, IGD^t/IGD^t_{max} and G_T by four JG adaptations on constrained functions tested are in the last column of Table 4.8. Overall, Alt-NSGA-II-aJG and

Table 4.8 *GD^t/GD^t_{max}, SP^t and IGD^t/IGD^t_{max} for constrained test functions obtained by four JG adaptations using the termination criterion; these values are average of ten runs, each with a different random number seed value.*

	Algorithm	OSY	CONSTR	SRN	TNK	Total
GD^t/GD^t_{max}	NSGA-II-aJG	0.547	0.946	**0.930**	0.556	2.979
	NSGA-II-saJG	1.129	0.974	1.108	0.917	4.128
	NSGA-II-sJG	1.248	1.070	1.014	0.627	3.960
	Alt-NSGA-II-aJG	**0.233**	**0.927**	0.941	**0.168**	**2.269**
SP^t	NSGA-II-aJG	0.921	0.988	**0.466**	1.001	3.375
	NSGA-II-saJG	0.888	1.009	0.500	1.021	3.418
	NSGA-II-sJG	**0.872**	**0.975**	0.484	1.034	**3.366**
	Alt-NSGA-II-aJG	0.938	0.987	0.515	**0.971**	3.410
IGD^t/IGD^t_{max}	NSGA-II-aJG	**0.528**	**0.943**	0.976	**0.545**	**2.991**
	NSGA-II-saJG	0.824	1.061	1.001	0.706	3.593
	NSGA-II-sJG	0.754	0.920	**0.964**	0.595	3.233
	Alt-NSGA-II-aJG	0.583	0.987	0.965	0.998	3.533
G_T	NSGA-II-aJG	305	198	317	137	957
	NSGA-II-saJG	223	145	290	120	778
	NSGA-II-sJG	182	161	340	**105**	788
	Alt-NSGA-II-aJG	**173**	**95**	**106**	114	**488**

NSGA-II-aJG are the best based on GD^t/GD^t_{max} and IGD^t/IGD^t_{max} respectively. The former is computationally efficient too. NSGA-II-saJG performs worse than other adaptations, based on all performance metrics.

4.5.2.2 Comparison of JG Adaptations at Intermediate Generations

The variation in GD^t/GD^t_{max} and SP^t with generations for constrained problems tested using the four JG adaptations is shown in Figure 4.6. The performance of all four JG adaptations is comparable for CONSTR and SRN problems, based on both GD^t/GD^t_{max} and SP^t. Based on GD^t/GD^t_{max}, NSGA-II-sJG and Alt-NSGA-II-aJG perform better on the OSY problem, whereas NSGA-II-aJG and Alt-NSGA-II-aJG are better on the TNK problem at the beginning of the search (until 800 generations). Initially, Alt-NSGA-II-aJG gives a significantly smaller GD value as there are fewer non-dominated solutions in the best Pareto-optimal front obtained. The performance of different JG adaptations is nearly comparable based on the distribution of non-dominated solutions, except that Alt-NSGA-II-aJG gives marginally better SP^t than other adaptations for the TNK problem. Further, SP^t is nearly constant after 100 generations, for the JG adaptations tested.

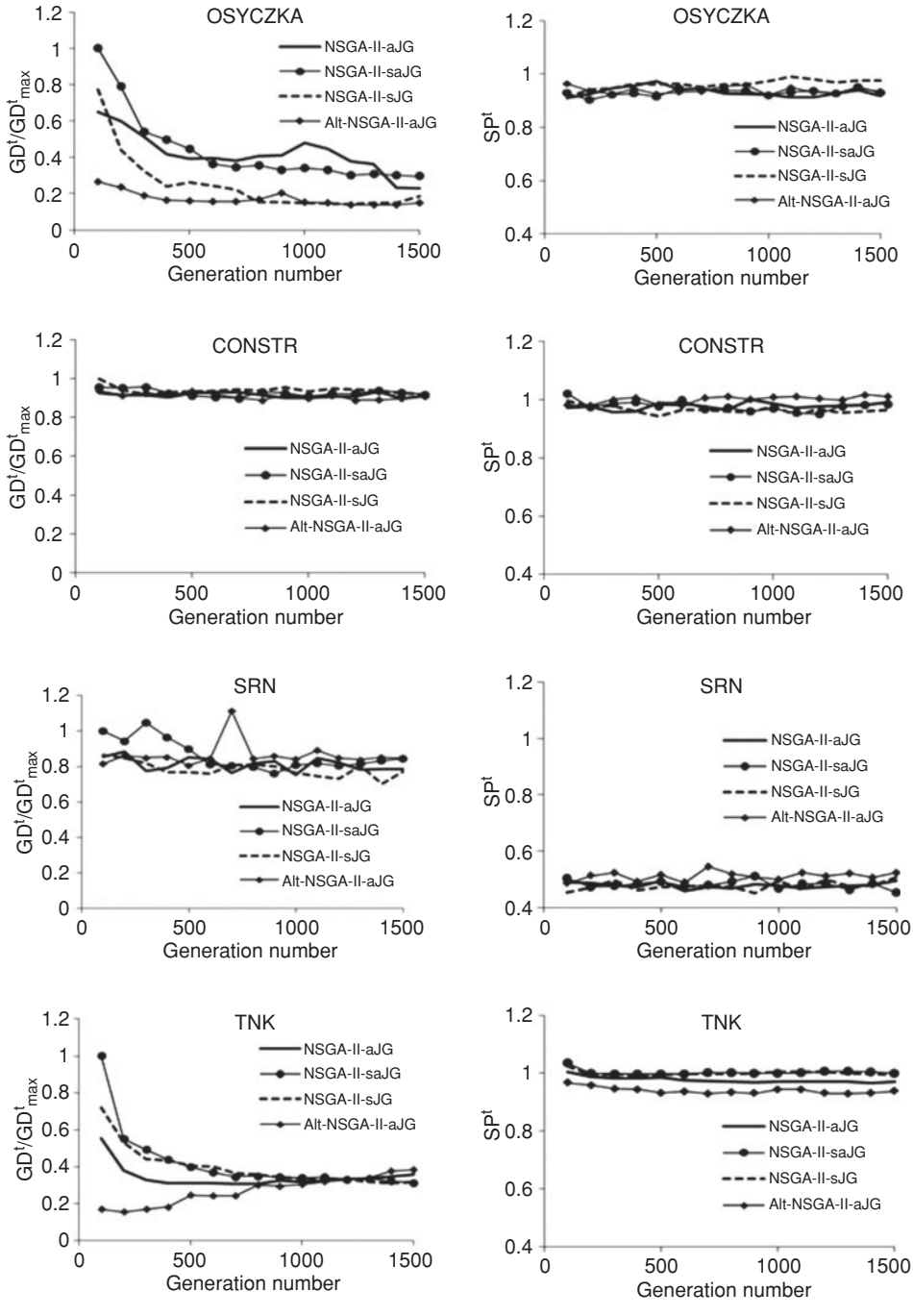

Figure 4.6 *Variation of GD^t/GD^t_{max} and SP^t with generations using different JG adaptations for constrained test functions; these profiles are based on average of ten runs by each algorithm for each test function.*

4.6 Conclusions

In this chapter, the performance of four jumping gene adaptations of NSGA-II was analyzed on bi-objective test problems; of these, five were unconstrained and four were constrained problems. This analysis considered the quality of non-dominated solutions (i.e., convergence to the known Pareto-optimal front measured by GD^t and IGD^t, and distribution of non-dominated solutions measured by SP^t) and also computational efficiency measured by the number of generations for satisfying the termination criterion (G_T). There is not much improvement in the non-dominated solutions after satisfying the termination criterion with further search (i.e., performance metrics are nearly constant). This confirms that the described termination criterion is able to terminate the search at the right time, and so it can avoid unnecessary computations. The termination criterion described will be useful for the comparative evaluation of MOO algorithms.

For the unconstrained problems tested, Alt-NSGA-II-aJG performs significantly better than other adaptations for convergence to the known Pareto-optimal front; it gives non-dominated solutions having lowest values of GD^t and IGD^t. Either NSGA-II-sJG or Alt-NSGA-II-aJG gives lowest SP^t depending on the problem. NSGA-II-sJG gives the lowest total SP^t whereas Alt-NSGA-II-aJG takes the least number of generations in total for the unconstrained problems tested. For the constrained problems tested, Alt-NSGA-II-aJG and NSGA-II-aJG respectively give better GD^t/GD^t_{max} and IGD^t/IGD^t_{max} than other adaptations. Further, the four JG adaptations tested give comparable SP^t, indicating similar distribution of non-dominated solutions obtained. Alt-NSGA-II-aJG with termination criterion takes the least number of generations in total for all constrained problems tested. Overall, Alt-NSGA-II-aJG is better than the other three JG adaptations for both unconstrained and constrained problems. As Alt-NSGA-II-aJG is better than NSGA-II-aJG, other operators such as sJG and saJG can be combined with the altruism approach in order to improve their performance.

Exercises

4.1. This study employs the algorithm parameters used/recommended in the literature (see Table 4.2). Study the effect of one or more of these parameter values on the JG adaptations tested in this chapter.

4.2. Study the performance of NSGA-II-mJG with the termination criterion for the test problems used in this chapter. For this, a slight modification in NSGA-II-sJG code is required.

4.3. Several unconstrained and constrained multi-objective test functions with the known Pareto-optimal fronts are available at: http://www.cs.cinvestav.mx/~emoobook/ (accessed November 27, 2012). Study the performance of the JG adaptations on these problems. Note that some of these test functions are challenging.

4.4. Study the performance of (a) sJG and (b) saJG in combination with the altruism approach. The Alt-NSGA-II-aJG code can be easily modified for Alt-NSGA-II-saJG and Alt-NSGA-II-sJG.

4.5. Test the performance of JG adaptations for chemical engineering application problems: Williams-Otto process, alkylation process and industrial ecosystem, described in Rangaiah (2009).

References

Agarwal, A. and Gupta, S.K. (2008a), Jumping gene adaptations of NSGA-II and their use in the multi-objective optimal design of shell and tube heat exchangers, *Chem. Eng. Res. Des*, 86, 132–139.

Agarwal, A. and Gupta, S.K. (2008b), Multiobjective optimal design of heat exchanger networks using new adaptations of the elitist non-dominated sorting genetic algorithm, NSGA-II, *Ind. Eng. Chem. Res.*, 47, 3489–3501.

Agrawal, N., Rangaiah, G.P., Ray, A.K. and Gupta, S.K. (2006), Multi-objective optimization of the operation of an industrial low-density polyethylene tubular reactor using genetic algorithm and its jumping gene adaptations, *Ind. Eng. Chem. Res.*, 45, 3182–3199.

Agrawal, N., Rangaiah, G.P., Ray, A.K. and Gupta, S.K. (2007), Design stage optimization of an industrial low-density polyethylene tubular reactor for multiple objectives using NSGA-II and its jumping gene adaptations, *Chem. Eng. Sc.*, 62, 2346–2365.

Bhat, S.A., Saraf, D.N., Gupta, S. and Gupta, S.K. (2006), On-line optimizing control of bulk free radical polymerization reactors under temporary loss of temperature regulation: experimental study on a 1-l batch reactor, *Ind. Eng. Chem. Res.*, 45, 7530–7539.

Bhat, S.A. (2007). On-line optimizing control of bulk free radical polymerization of methyl methacrylate in a batch reactor using virtual instrumentation, Ph.D. thesis, Indian Institute of Technology, Kanpur.

Bhat, G.R. and 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.

Chaudhari, P. and Gupta, S.K. (2012), Multi-objective optimization of a fixed bed maleic anhydride reactor using an improved biomimetic adaptation of NSGA-II, *Ind. Eng. Chem. Res.*, 51, 3279–3294.

Coello Coello, C.A., Lamont, G.B. and Van Veldhuizen, D.A. (2007), Evolutionary algorithms for solving multi-objective problems, Springer, Berlin/Heidelberg, 2nd edition.

Deb, K., Agarwal, S., Pratap, A. and Meyarivan, T. (2000), A fast and elitist multi-objective genetic algorithm: NSGA-II, Technical Report 2000001, IIT Kanpur, KanGAL, http://www.iitk.ac.in/kangal/reports.shtml#2000 (accessed 27 November, 2012).

Deb, K. (2001), *Multi-objective Optimization using Evolutionary Algorithm*, John Wiley & Sons, Chichester.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002), A fast and elitist multi-objective genetic algorithm: NSGA-II, *IEEE Trans. Evolutionary Computing*, 6, 182–197.

Duan, X., Liu, J., Zhang, L., and Jiao, L. (2010), Multi-objective job shop scheduling based on multi-agent evolutionary algorithm, *Lecture Notes in Comp. Sc.*, 6477, 543–552.

Furtuna, R., Curteanu, S., Racles, C. (2011), NSGA-II-RJG applied to multi-objective optimization of polymeric nanoparticles synthesis with silicone surfactants, *Cent. Eur. J. Chem.*, 9(6), 1080–1095.

Guria, C., Verma, M., Mehrotra, S.P. and Gupta, S.K. (2005a), Multi-objective optimal synthesis and design of froth flotation circuits for mineral processing, using the jumping gene adaptations of genetic algorithm, *Ind. Eng. Chem. Res.*, 44, 2621–2633.

Guria, C., Bhattacarya, K. and Gupta, S.K. (2005b), Multi-objective optimization of reverse osmosis desalination units using different adaptations of the non-dominated sorting genetic algorithm (NSGA), *Comp. and Chem. Eng.*, 29, 1977–1995.

Guria, C., Verma, M., Mehrotra, S.P. and Gupta, S.K. (2006), Simultaneous optimization of the performance of flotation circuits and their simplification using the jumping gene adaptations of genetic algorithm-II: more complex problems, *Int. J. Mineral Processing*, 79, 149–166.

Kasat, R.B. and 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, *Comp. and Chem. Eng.*, 27, 1785–1800.

Khosla, D.K., Gupta, S.K. and Saraf, D.N. (2007), Multi-objective optimization of fuel oil blending using the jumping gene adaptations of genetic algorithm, *Fuel Processing Technology*, 88, 51–63.

Mitra, K., Deb, K. and Gupta, S.K. (1998), Multi-objective dynamic optimization of an industrial nylon 6 semi batch reactor using genetic algorithm, *J. Appl. Poly. Sci.*, 69, 69–87.

Nabavi, S.R., Rangaiah, G.P., Niaei, A. and Salari, D. (2009), Multi-objective optimization of an industrial LPG thermal cracker using a first principles model, *Ind. Eng. Chem. Res.*, 48, 9523–9533.

Ramteke, M. and Gupta, S.K. (2008), Multi-objective optimization of an industrial nylon-6 semi batch reactor using the a-jumping gene adaptations of genetic algorithm and simulated annealing, *Polymer Engineering and Science*, 48(11), 2198–2215.

Ramteke, M. and Gupta, S.K. (2009a), Multi-objective genetic algorithm and simulated annealing with jumping gene adaptations, In Rangaiah, G.P. (editor), *Multi-objective Optimization: Techniques and Applications in Chemical Engineering*, World Scientific, Singapore.

Ramteke, M. and Gupta, S.K. (2009b), Biomimicking altruistic behavior of honey bees in multi-objective genetic algorithm, *Ind. Eng. Chem. Res.*, 48, 9671–9685.

Ramteke, M. and Gupta, S.K. (2009c), Biomimetic adaptations of the evolutionary algorithm, NSGA-II-aJG, using the biogenetic law of embryology for intelligent optimization, *Ind. Eng. Chem. Res.*, 48, 8054–8067.

Rangaiah, G.P. (2009), *Multi-objective Optimization: Techniques and Applications in Chemical Engineering*, World Scientific, Singapore.

Ripon, K.S.N., Kwong, S., Man, K.F. (2007), A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization, *Information Science*, 117, 632–654.

Sankararao, B. and 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-Pacific J. Chemical Eng.* 1, 21–31.

Sankararao, B. and Gupta, S.K. (2007a), Multi-objective optimization of pressure swing adsorbers for air separation, *Ind. Eng. Chem. Res.*, 46, 3751–3765.

Sankararao, B. and Gupta, S.K. (2007b), Multi-objective optimization of an industrial fluidized-bed catalytic cracking unit (FCCU) using two jumping gene adaptations of simulated annealing, *Comp. and Chem. Eng.*, 31, 1282–1295.

Sharma, S. (2013), Multi-objective differential evolution: modifications and applications to chemical processes, Ph.D. thesis, National University of Singapore, Singapore.

Sindhya, K., Deb, K. and Miettinen, K. (2011), Improving convergence of evolutionary multi-objective optimization with local search: a concurrent-hybrid algorithm, *Nat. Comput.*, 10, 1407–1430.

Van Veldhuizen, D.A. and Lamont, G.B. (1998), Evolutionary computation and convergence to a Pareto front, citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.7224 (accessed December 2012).

Wagner, T., Trautmann, H. and Naujoks, B. (2009), OCD: online convergence detection for evolutionary multi-objective algorithms based on statistical testing, *Lecture Notes in Comp. Sc.*, 5467, 198–215.

Zhang, Q., Zhou, A., Zhano, S., Suganthan, P.N., Liu, W. and Tiwari, S. (2009), Multi-objective optimization test instances for the CEC 2009 special session and competition, CEC Special Session on the Performance Assessment of Multi-objective Optimization Algorithms. Conference organized by IEEE, Trondheim, Norway, 18–21 May.

Zitzler, E., Deb, K. and Thiele, L. (2000), Comparison of multi-objective evolutionary algorithms: empirical results, *IEEE Trans. Evolutionary Computation*, 8, 173–195.

Zitzler, E. and Thiele, L. (1998), Multi-objective optimization using evolutionary algorithms: a comparative case study, *Parallel Problem Solving from Nature*, 292–301.

Zhou, A., Jin, Y. and Zhang, Q. (2006), Combining model-based and generics-based offspring generation for multi-objective optimization using a convergence criterion, *Congress on Evolutionary Computation*, 3234–3241.