JGBL paradigm: A Novel Strategy to Enhance the Exploration Ability of NSGA-II

Ke Li
Department of Computer Science
City University of Hong Kong
Hong Kong, China
keli.genius@gmail.com

Sam Kwong
Department of Computer Science
City University of Hong Kong
Hong Kong, China
cssmak@cityu.edu.hk

Kim-Fung Man
Department of Electronic Engineering
City University of Hong Kong
Hong Kong, China
eekman@cityu.edu.hk

ABSTRACT

NSGA-II is one of the most efficient multi-objective evolutionary algorithms (MOEAs) for solving multi-objective optimization problems (MOPs). In this paper, a *Jumping Gene Based Learning* (JGBL) paradigm is proposed to enhance the exploration ability of NSGA-II. JGBL paradigm simulates the natural behavior of maize and is incorporated into the framework of the original NSGA-II. It only operates on the non-dominated solutions which are eliminated in the environmental selection procedure due to the low quality of crowded distance. The activation of JGBL operation is entirely adapted online according to the search status of evolutionary process to give a needed fuel when the population evolves slowly with inherent variation operators. Extensive comparative studies are performed on different benchmark problems and significant algorithmic improvements in terms of proximity, uniformity and diversity are obtained with the incorporation of the proposed JGBL paradigm into NSGA-II process.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search − *Heuristic methods.*

General Terms

Algorithms, Performance, Reliability, Experimentation.

Keywords

Multi-objective optimization, jumping gene based learning, NSGA-II

1. INTRODUCTION

Evolutionary Algorithms (EA) are stochastic search techniques inspired from the nature phenomenon of biological evolution, such as nature selection and genetic operation. They have been successfully applied to solve numerous optimization problems in diverse fields, due to their suitability for tackling highly sophisticated and non-linear problems. Not only for single-objective optimization problems (SOPs), but also suitable for solving sophisticated MOPs due to their population based approach which, unlike classical optimization methods, obtains a set of Pareto optimal solutions simultaneously in a single run.

Ever since the pioneering study of Schaffer [1], tremendous state-of-the-art MOEAs have been suggested over the past decades [2]-[5]. Among the great variety of variants, NSGA-II is one of the most efficient MOEAs using elitist approach. Based on the non-domination order relationship, individuals in the hybrid population, formed by parents and offspring, are sorted in different fronts. Then, the best individuals in terms of non-domination and diversity are chosen for the genetic variation.

Exploration and exploitation are two important parts of the search dynamics of MOEAs. Generally speaking, exploitation corresponds to the mechanism that related to the selection of high fitness individuals from the current population, while exploration represents the function that concerning with the generation of offspring from parental population by means of genetic variation. However, most of the existing studies about MOEAs mainly focus on fitness assignment, diversity preservation, or environmental selection, despite the equal importance of exploitation and exploration contributing to the performance of MOEAs. On the contrary, the exploration part commonly follows traditional routine of genetic variation such as crossover and mutation.

Nevertheless, it is a gratification to see that some recent MOEAs with non-traditional variation operators gradually appear and have shown their competitive capabilities for solving MOPs. The characteristic feature of these MOEAs is the combination of novel heuristic exploration techniques and the search dynamics of evolutionary multi-objective optimization (EMO). For instance, Abbass [6] proposed a Pareto-based differential evolution algorithm that self-adapts the crossover and mutation rates simultaneously. Tan et al. [7] proposed an adaptive variation operator to balance the dilemma of exploration and exploitation. Zhang et al. [8] studied the regularity of the Pareto set in the decision space, and designed a model-based multi-objective estimation of distribution algorithm: RM-MEDA. Nebro et al. [9] applied the scatter search technique to solve MOPs, while Chan et al. [10] integrated the “jumping gene paradigm” drawn from the mechanism of transposon in maize into the framework of MOEA.

In this paper, our contribution is proposing Jumping Gene Based Learning (JGBL) paradigm to acclimate to the framework of the state-of-the-art NSGA-II, in order to enhance its exploration ability. The original jumping gene paradigm have been proposed in our previous work about JGGA [10][11]. The differences between our JGBL paradigm and original JGGA are two folds. On the one hand, the jumping gene operators in JGBL paradigm are completely adapt to the continuous search space. On the other hand, JGBL paradigm is no longer playing as an alternative variation operator which operates on the entire population. The fundamental motivation is to explore the non-dominated individuals which are eliminated by the environmental selection procedure of NSGA-II, so as to inject needed energy when the inherent genetic variations meet some difficulties to propel the population forward.

The outline of this paper is as follows. The underlying mechanism of JGBL paradigm is given in Section 2. Next, the quintessence of how to embed the JGBL paradigm into the framework of NSGA-II is illustrated in Section 3. The effectiveness of our idea is examined in Section 4. Finally, some conclusions and future directions are drawn in Section 5.

2. JUMPING GENE BASED LEARNING PARADIGM

Early in the year 1950, discovered from the maize, Barbara McClintock firstly reported the phenomenon of jumping gene (also called transposon) in [12][13]. According to her experimental studies, jumping gene facilitated two different ways to transfer gene around the genome. The one is called *cut and paste* and the other is *copy and paste*, which make the gene transmission in a horizontal way. In recent years, jumping gene phenomenon in natural science has been transplanted into the framework of EA to solve optimization problems [10][11][14]-[19]. Most of these variants are designed for discrete search space, and the genes are represented in binary strings, except for the RJGGA proposed in [11]. Nevertheless, instead of a well-designed routine to mimic the mechanism of *cut and paste* and *copy and paste*, RJGGA employed the existing polynomial mutation [26] and simulated binary crossover [25], which are prevalent in the current MOEAs, to simulate the operations of single and double chromosomes' jumping gene respectively. There are several difficulties and drawbacks encountered when using binary coded strings for solving problems in real world continuous search space. Such as the inability to achieve arbitrary precision, the high computation overload, the side effect incurred by the Hamming cliff, etc. However, these aforementioned difficulties could be tackled by the application of real coding scheme to solve the continuous optimization problems straightforwardly. Thus, it is a urgent task to remodel the original binary coded jumping gene paradigm to acclimate the characteristic of continuous search space without losing the characteristic of the original mechanism of transposon. Nonetheless, there are two formidable problems ahead of us when designing the real-coded jumping gene paradigm. The first one is how to represent the individual so that the characteristic of the mechanism of transposon could be properly preserved, and the other one is how to deal with the boundary violation problem caused by the jumping gene operation itself [11]. In the following paragraph, the corresponding solutions would be given step by step.

2.1 Individual Representation

The mechanisms of transposons, namely *cut and paste* and *copy and paste*, are originally operated on the DNA sequences. Thus, a binary coded chromosome would be a straightforward way to mimic these behaviors within the framework of EA. As for continuous search space, instead of discrete "0-1" strings, an individual is usually made up of several real variables. Nevertheless, an individual could also be represented by a chromosome whose genes are encoded by these variables, without losing the generality. A simple example is given in Figure 1, an individual is composed of eight real numbers, which are corresponding to the gene positions with their ranges above them.

2.2 Remedy Strategy For Boundary Violation

Referring back to Figure 1, variables of this individual possess different ranges. Therefore, arbitrary switches of the information contained in genes are usually not permitted due to the boundary violation problem discussed in [11]. Considering a simple example, the range of the variable in the gene position of x_1 is [3, 4] while that of x_4 is [0, 1], as shown in Figure 1. Thus x_1 would be invalid in the gene position of x_4 if they exchange gene positions, vice versa. Hence, some kind of remedy techniques should be considered in order to keep the variables in valid ranges after information exchange.

DEFINITION 1 (VARIABLE INFORMATION) *The percentage of a real number that occupies in its range is defined as the variable information of it maintained in this range. It could be calculated as the following formulation:*

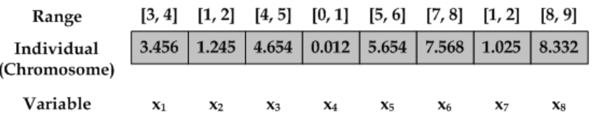
$$Information(x) = \frac{x - lower(x)}{upper(x) - lower(x)} \qquad (1)$$

where Information(x) *means the variable information maintained by* x, *and upper*(x) *and lower*(x) *represent the upper and lower bound of the range that* x *resides in, respectively*.

Considering a simple example, as the individual illustrated in Figure 1, the variable information of x_1 is 0.456 according to formulation (1). In order to solve the problem caused by boundary violation, we assume that the variable information of a variable in its original position would be remained, no matter where it is switched to. Thus, the value after its transfer to a new position could be restored as the following formulation:

$$Transfer(x) = lower(y) + Information(x) \times [upper(y) - lower(y)] \quad (2)$$

where *Transfer*(x) indicates the value after x is transferred to a new gene position y, while *lower*(y) and *upper*(y) are the lower and upper bound of the range in the gene position y, respectively. Concerning back to the former example, we assume that x_1 is transferred to another gene position, say the position of x_4. The direct switch is invalid because x_1 is out of the range of x_4. Based on the previous provided strategy, we keep a record of the variable information of x_1, and the new value after x_1 being switched to the position of x_4 could be restored to 0.456. Therefore, the restored value is appropriate for this new gene position.



Range	[3, 4]	[1, 2]	[4, 5]	[0, 1]	[5, 6]	[7, 8]	[1, 2]	[8, 9]
Individual (Chromosome)	3.456	1.245	4.654	0.012	5.654	7.568	1.025	8.332
Variable	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8

Figure 1. Representation of a solution

2.3 Computational Formation of Jumping Gene Operator

As introduced in the previous paragraph, the operation of jumping gene is basically upon the mechanism of transposon. In general, transposon is selected from the genes of one chromosome in a random manner. The components of a transposon could be made up by more than one gene. And for a particular chromosome, there is also no specific limitation on the number of transposons. Moreover, the settle positions of the transposons are also chosen randomly. And the contents of the transposons transferred in a horizontal way, which is a type of a lateral movement of genes within the same chromosome or even to another individual in the population pool. Transposons themselves are assigned to one of two classes according to their mechanism of transposition, which can be described as either *cut and paste* or *copy and paste*. The implementation of the *cut and paste* operation is that the transposon is cut from the original position and insert into a new site. In the case of *copy and paste* operation, the selected genes replicate themselves to form the transposon, and then this copy is used to replace the genes in the selected positions of the target chromosome. Figure 2 gives an intuitive illustration of these two operations. The upper figure illustrates the operation of *cut and paste*, while the lower one represents the mechanism of *copy and paste*.

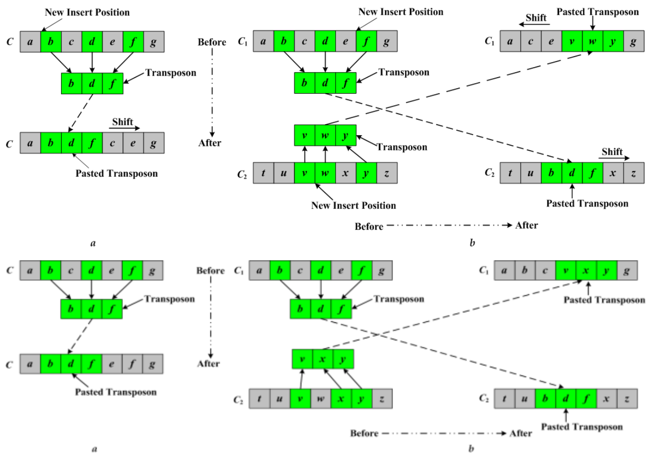
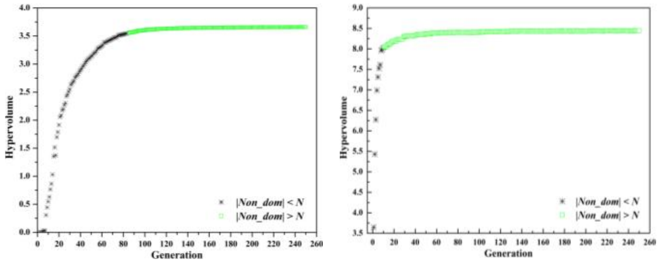


Figure 2. Transposon operation: (a) single and (b) double

3. JGBL PARADIGM FOR NSGA-II

Different from the SOP, which has determined guidelines for discriminating the superiority of individuals by their fitness values, there often exist incommensurable relationships among individuals in multi-objective optimization due to the conflicting nature of multiple criteria. Therefore, rather than a single optimal point, a set of tradeoff solutions which are so-called non-dominated with each other in the population is preferred when solving MOPs. Usually, the number of non-dominated individuals varies with the process of evolution and also largely reflects the evolutionary status of a MOEA. Specifically, the number of non-dominated individuals is relatively small at the early stage of the evolution process. Nevertheless, they also play a crucial role in guiding the exploration towards the desirable direction. Afterwards, the size of the non-dominated individuals would gradually grow up with the proceeds of evolution. However, some possible side effects caused by the explosion of the size of non-dominated individuals might be the stagnation of evolution. This circumstance often happens when the solutions set already approaches the Pareto optimal front or the search process has been trapped by some local optima. Both of these two scenarios could impede the population further evolve towards the final Pareto optimal front. Figure 3 gives two examples to illustrate the relationship between the performance of NSGA-II on Hypervolume indicator and the number of non-dominated solutions during the process of evolution.

In Figure 3, the black fork indicates the stages that the number of non-dominated solutions in the population (denoted as |*Non_dom*|) is smaller than or equal to a predefined threshold *N* (here *N* is set as the population size in NSGA-II), while the green square represents the opposite scenario. It could be observed from these two sub-figures that the evolution processes have been stumped in varying degrees in case that the number of non-dominated solutions exceeds *N*. Specifically, Figure 3(a) illustrates the evolutionary trajectory of the Hypervolume indicator for NSGA-II on test problem ZDT 1. The sharp rise of the trajectory tends to be moderate with the increase of |*Non_dom*|, and then stagnate when |*Non_dom*| surpass *N*. This scenario could be interpreted as the approximated solutions set has already approached the Pareto optimal front so that it is difficult for NSGA-II to do further exploration. As for the test problem WFG 9, it features significant bias, parameter dependency and multimodality which could influence the effect of NSGA-II. As shown in Figure 3(b), the Hypervolume indicator values surge up to a high level at the early stage of evolution when |*Non_dom*| is lower than *N*. But from about 10th generation on, the portion of non-dominated solutions expand to a large amount, and the increase of Hypervolume indicator values goes into a stable status.



(a) ZDT 1 **(b) WFG 9**

Figure 3. Evolutionary trajectories of Hypervolume indicator

Here we propose to incorporate the JGBL paradigm into the framework of NSGA-II, in order to enhance the exploration ability of NSGA-II to better approach and spread around the Pareto optimal front. Specifically, JGBL operations are activated only when the cardinality of the non-dominated solutions population is larger than that of the parent population. In other words, considering back to Figure 3, JGBL operations only acts on the stages that are marked as green squares. After the environmental selection procedure of NSGA-II, some of the non-dominated solutions are eliminated due to the poor performance on crowding distance metric. Nonetheless, the JGBL paradigm is just operating on these moderate solutions.

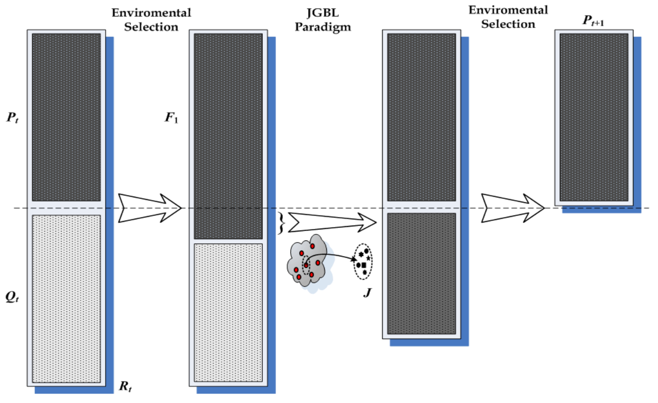


Figure 4. Procedure of JGBL paradigm in NSGA-II

Figure 4 gives an intuitive illustration of the procedure of JGBL paradigm in NSGA-II. At first, a combined population $R_t = P_t \cup Q_t$ is formed. The size of R_t is $2N$. Then the environmental selection procedure is called to sort R_t according to non-domination. We assume that the number of non-dominated solutions is larger than N at the current generation t. That is to say, the size of the best non-dominated set F_1 surpasses N. In order to choose exactly N population members to form the next generation parent population, NSGA-II sorts the solutions in F_1 according the crowding distance metric in descending order to fill all population slots. Instead of eliminating the solutions that have small crowding distance values in NSGA-II directly, JGBL paradigm aims at mining some useful information through exploring this kind of solutions. Two advantages encourage the application of JGBL paradigm in this way:

1. This kind of solutions possess satisfactory convergence property since no other solutions dominate them in the hybrid population R_t. Applying JGBL operators act on them might have more chance to generate better mutants compared to those dominated ones.
2. In general, they are eliminated during the environmental selection procedure due to the poor performance on crowding distance metric. In other words, they inhabit in crowded areas. The JGBL paradigm could be also regarded as a local searcher which might be helpful to explore some unknown region in a sense. Thus the overall diversity of population might be enhanced at last by the addition of some favorable solutions.

One of the prominent features of nature is the opportunistic manner for dealing with everything in the world. That is to say, there is no deterministic rule to sequence the happen of events and you also could not foresight the situation in the future. Evolutionary algorithm is derived from nature so the variation operators (i.e., crossover and mutation) are also neither streamlined nor can be planned in advance. Similar to the behavior of other variation operators, the operation of JGBL paradigm is also based on the opportunity. However, different from the traditional variation operators, the activation of JGBL paradigm is not only decided by a predefined probability, but also the status of the current evolution. It is worth noting that all four different jumping gene operators are utilized whenever the JGBL paradigm is activated. In this way, a solution would be able to generate six possible mutants in one time, so that sufficient opportunity could be given to this solution to explore some unknown regions. To the best of our knowledge, few reports on this topic could be found in this literature.

4. Experiments

In this paper, ZDT test suite [20], DTLZ test suite [21] and WFG test suite [22] are chosen to form the benchmark problems. As illustrated in Section 1, the primary motivation of JGBL paradigm is not to replace the classical variation operators in the MOEA realm. It is used to be embedded into the framework of MOEA to enhance the exploration ability of the baseline MOEA. Thus, the performance comparison is carried out between JGBL paradigm based NSGA-II (denoted as NSGA-II+JGBL) and its baseline NSGA-II, in order to verify the efficiency of JGBL paradigm.

4.1 Performance Assessment Tools

In the field of multi-objective optimization, the strengths and weaknesses of any algorithm are based on the qualities of the finally evolved solutions such as proximity to the reference set, spread and uniformity of the non-dominated solutions in the objective space, etc. Several performance assessment tools, which independently explore different features of an algorithm, have been proposed and available to evaluate the performance of a MOEA. In our experimental section, three different metrics, namely generation distance (GD) [23], spacing (SP) [24] and hypervolume (HV) [4], are chosen to evaluate three different aspects of a MOEA: convergence, spread of solutions and comprehensive performance. Detailed illustrations of these three metrics could be found in corresponding references.

4.2 General Parameter Settings

In this paper, variables are encoded as real numbers while the individual representation of our NSGA-II+JGBL has been introduced in Section 2.1. Two additional control parameters, namely jumping percentage and jumping rate are set as 0.4 and 0.6 for all benchmark problems. As for variation operators, the classical simulated binary crossover operator (SBX) [25] is used for crossover and the polynomial mutation [26] is utilized as the mutation operator in NSGA-II. The crossover rate p_c=0.9 and the crossover index η_c=10 for WFG test suite and η_c=20 for the others. The mutation rate p_m=1/*nreal* and the mutation index η_m=50 for WFG test suite and η_m=20 for the others.

The initial populations are randomly generated among the range of each decision variables. The size of population is set as 100 for all the test cases. 250 generations and 300 generations are evolved for two-dimensional and three-dimensional benchmark problems, respectively. All numerical experiments are conducted on Intel(R) Core(TM) 2 Duo CPU P8400 @ 2.26GHz, 2 GB RAM computer running the Microsoft Windows 7. The reference points used for evaluating the HV metric are given in Table 1.

Table 1. Reference points settings for different test problems

Test Problems	Reference Points
ZDT test suite	(2.0, 2.0)
DTLZ-1	(1.0, 1.0, 1.0)
DLTZ-2 to DTLZ-6	(2.0, 2.0, 2.0)
DTLZ-7	(2.0, 2.0, 7.0)
WFG test suite	(3.0, 5.0)

4.3 Experimental results and discussion

50 independent runs are implemented on each test problems. The statistical values of GD, SP and HV metrics for 21 benchmark test problems obtained by NSGA-II+JGBL and its baseline NSGA-II are recorded in Table 2. Besides, the boxes with the best results are highlighted in grey boldface. Following the central limit theorem, we assume that the sample means are normally distributed. Therefore, paired t-test statistical test at 95% is adopted to compare the significance between two competing algorithms. Furthermore, the symbol † indicating that NSGA-II+JGBL is significant better than its baseline NSGA-II, and ‡ representing that the baseline NSGA-II significantly outperforms our proposed method. The last row of Table 2 gives the ratio of domination for each MOEA in every three performance metrics. In other words, 4/21 means a MOEA shows significant better

performance on 4 out of 21 benchmark problems for a given metric, comparing to its competitor.

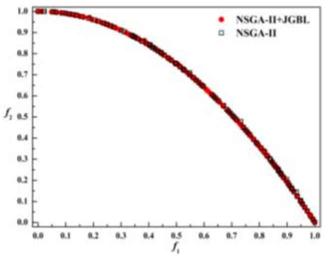
As shown in Table 2, NSGA-II+JGBL performs significant better than its competing baseline NSGA-II. Ratios of domination for GD, SP and HV are 18/21, 15/21 and 21/21, respectively. It is worth noting that NSGA-II+JGBL obtains significant better results on the comprehensive performance metric HV for all benchmark problems. Moreover, the scales of differences on *HV* values vary in different benchmark problems. In other words, the improvements of HV values are slight for some benchmark problems, which means the promotions brought by JGBL paradigm are relatively limited. However for the other cases, the HV values have been significantly promoted after the application of JGBL paradigm, like DTLZ3, DTLZ 6, WFG 1, WFG 5 and WFG 8. In order to illustrate the superiority of NSGA-II+JGBL in an intelligible way, we show some comparisons of final solutions obtained by NSGA-II and NSGA-II+JGBL on several representative problems in Figure 5. It is interesting to see that there is a large gap between the extreme solution and the other solutions obtained by NSGA-II+JGBL, refer to Figure 5(d). This could explain the inferior performance of NSGA-II+JGBL on *SP* value, compared to its baseline MOEA. Nevertheless, our proposed mechanism actually makes the final solutions better approaching the Pareto optimal front with a more diverse and wider spread. Same fact could be found on WFG 8 as shown in Figure 5(g). Consider Figure 5(a) to Figure 5(c) and Figure 5(f), both of these two MOEAs converge well to the Pareto optimal front, however our NSGA-II+JGBL obtains a more uniform solutions set as there are some obvious gaps exist in the solutions set obtained by the baseline NSGA-II. Although the *SP* metric value shows that our NSGA-II+JGBL is inferior to its competing baseline MOEA on WFG 5 according to Table 2, Figure 5(e) gives us a clear illustration that the solutions set obtained by NSGA-II+JGBL converge to the Pareto front with a wider spread, comparing to NSGA-II. In view of Figure 5(h), the performance of our NSGA-II+JGBL is obviously superior to its baseline NSGA-II. The solutions set obtained by NSGA-II are not only scatter, but also not convergent, comparing to the well spread and converged line obtained by our method.

Table 2. Performance comparison between NSGA-II and NSGA-II+JGBL

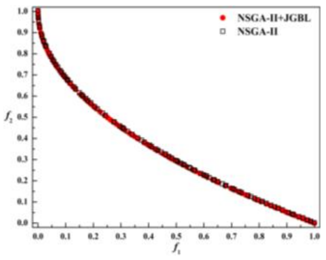
	GD		**SP**		**HV**	
Problems	NSGA-II	NSGA-II+JGBL	NSGA-II	NSGA-II+JGBL	NSGA-II	NSGA-II+JGBL
ZDT 1	1.807E-4(6.75E-5)†	**1.357E-4(1.66E-4)**	7.206E-3(5.92E-4)†	**6.643E-3(1.136E-3)**	3.65882(3.99E-4)†	**3.66051(1.95E-4)**
ZDT 2	1.529E-4(2.99E-5)†	**1.354E-4(1.47E-4)**	7.402E-3(6.57E-4)†	**6.892E-3(8.69E-4)**	3.32438(6.07E-4)†	**3.32713(2.26E-4)**
ZDT 3	9.672E-5(4.70E-5)†	**7.852E-5(7.69E-5)**	7.911E-3(7.19E-4)†	**7.165E-3(7.57E-4)**	4.81271(4.02E-4)†	**4.81454(9.86E-5)**
ZDT 4	3.690E-4(1.79E-4)†	**2.375E-5(1.99E-5)**	7.539E-3(7.62E-4)†	**6.734E-3(5.39E-4)**	3.65136(5.52E-3)†	**3.66102(1.62E-4)**
ZDT 6	7.255E-4(7.63E-5)†	**3.186E-4(4.87E-5)**	5.614E-3(4.75E-4)†	**5.227E-3(4.81E-4)**	3.01962(2.66E-3)†	**3.03282(1.42E-3)**
DTLZ 1	**6.743E-2(2.25E-1)**‡	6.815E-3(2.07E-2)	3.679E-1(1.66E+0)†	**6.968E-2(1.84E-1)**	0.94723(6.01E-2)†	**0.96731(1.32E-3)**
DTLZ 2	**1.330E-3(1.72E-4)**‡	2.599E-3(1.09E-3)	**5.769E-2(5.00E-3)**‡	6.058E-2(6.31E-3)	7.32607(2.55E-2)	**7.32705(2.05E-2)**
DTLZ 3	9.603E-1(1.09E+0)†	**8.931E-2(3.17E-1)**	3.935E+0(6.54E+0†	**5.027E-1(1.74E+0)**	0.49633(1.29E+0)†	**7.34293(1.91E-2)**
DTLZ 4	8.362E-3(6.09E-3)†	**5.722E-3(5.14E-3)**	8.297E-2(4.01E-2)†	**6.873E-2(1.65E-2)**	6.87040(6.30E-1)†	**7.22939(5.10E-1)**
DTLZ 5	2.168E-4(7.22E-5)†	**9.198E-5(3.33E-5)**	9.809E-3(7.31E-4)†	**8.557E-3(7.35E-4)**	6.09980(1.12E-3)†	**6.10214(1.70E-3)**
DTLZ 6	8.784E-2(1.08E-2)†	**1.008E-5(3.94E-5)**	9.399E-2(2.84E-2)†	**1.128E-2(7.40E-4)**	3.73857(2.98E-1)†	**6.10134(3.13E-3)**
DTLZ 7	3.702E-3(1.08E-3)†	**1.755E-3(1.08E-2)**	**7.150E-2(9.56E-3)**‡	8.949E-2(2.13E-2)	13.06274(9.65E-2)†	**13.15044(8.14E-2)**
WFG 1	3.425E-2(8.71E-3)†	**2.234E-2(1.20E-4)**	**2.109E-2(2.30E-2)**‡	2.568E-2(1.59E-2)	8.57315(6.62E-1)†	**10.04920(3.53E-1)**
WFG 2	**1.053E-3(9.07E-4)**	1.331E-3(8.84E-4)	**1.541E-2(2.12E-3)**‡	1.637E-2(1.41E-2)	11.02166(4.16E-1)†	**11.16917(3.92E-1)**
WFG 3	7.000E-4(8.48E-5)†	**5.812E-4(6.40E-5)**	2.009E-2(1.66E-3)†	**1.861E-2(1.77E-3)**	10.93235(4.76E-3)†	**10.94084(3.11E-3)**
WFG 4	1.435E-4(1.66E-4)†	**1.343E-3(1.52E-4)**	2.175E-2(1.98E-3)†	**2.003E-2(2.23E-3)**	8.66203(6.46E-3)†	**8.67612(3.01E-3)**
WFG 5	6.501E-3(4.84E-5)†	**2.788E-3(1.07E-3)**	**2.156E-2(2.24E-3)**	2.191E-2(2.59E-3)	8.15226(3.00E-2)†	**8.62545(1.42E-1)**
WFG 6	2.571E-3(2.52E-3)†	**1.759E-3(2.36E-3)**	2.209E-2(2.33E-3)†	**2.023E-2(1.83E-3)**	8.52492(1.60E-1)	**8.57356(1.49E-1)**
WFG 7	9.656E-4(6.09E-5)†	**8.677E-4(8.02E-5)**	2.210E-2(1.98E-3)†	**2.020E-2(1.73E-3)**	8.66620(2.61E-3)†	**8.67168(2.69E-3)**
WFG 8	2.549E-2(7.94E-3)†	**1.245E-2(6.41E-3)**	**2.233E-2(6.29E-3)**	2.559E-2(2.52E-2)	7.13858(4.93E-1)†	**7.95717(4.25E-1)**
WFG 9	1.033E-3(2.59E-4)†	**9.880E-4(1.71E-4)**	2.093E-2(2.01E-3)†	**2.000E-2(2.15E-3)**	8.42788(2.43E-2)	**8.42920(1.40E-2)**
Ratio	**3/21**	**18/21**	**6/21**	**15/21**	**0/21**	**21/21**

Furthermore, in order to better understand the performance during the whole evolutionary process, we also plot the evolutionary trajectories of HV for the JGBL based NSGA-II and its baseline algorithms on four selected test problems in Figure 6 (all these four selected test problems are difficult for regular MOEAs to tackle). From these four subfigures, it is clear to see that *HV* values have been substantively improved after the application of JGBL paradigm. Referring to Figure 6(a) and Figure 6(b), NSGA-II+JGBL experiences a placid stage in the early of evolution and suddenly surge up to a top high afterwards (150th generation in DTLZ 3 and 100th generation in DTLZ 6). As for WFG 1, as shown in Figure 6(c), all the trajectories increase with mild slops, nevertheless the advantage of JGBL based NSGA-II is still evident. Its trajectory goes up higher than that of the baseline

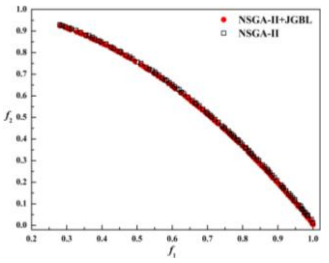
NSGA-II in the early stage of evolution, and it still increase with a sharp slope when the slope of the trajectory of NSGA-II becomes gentle after 130th generation. Considering the Figure 6(d), all the trajectories surge up high in the early 15 generations, and they transfer into a stable stage afterwards. But we could also find that the slope of ascending trend for NSGA-II+JGBL is larger than that of its baseline algorithm all the time.



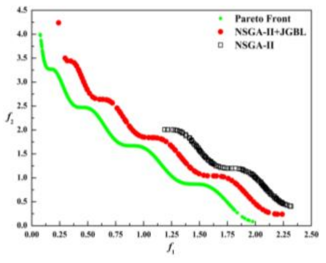
a. ZDT 2



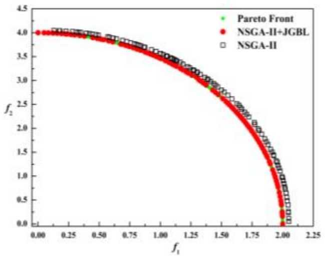
b. ZDT 4



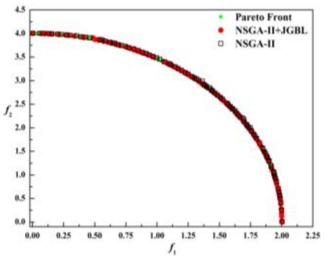
c. ZDT 6



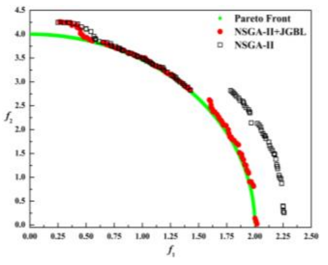
d. WFG 1



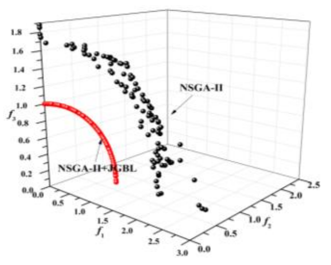
e. WFG 5



f. WFG 6



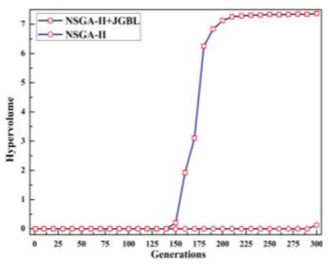
g. WFG 8



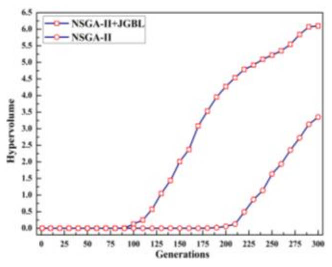
h. DTLZ 6

Figure 5. Final solutions obtained by NSGA-II+JGBL and NSGA-II

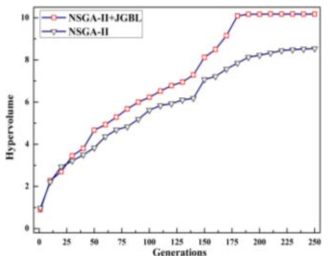
From these results observed in this section, we could find that NSGA-II with the incorporation of JGBL paradigm performs evidently better than its original version. The improvement of performance is comprehensive, not only in terms of convergence but also in diversity and spread. The most important reason contributes to this promotion might be the application of JGBL operation on those worse-distributed non-dominated solutions, thereby more readily to provide promising offspring.



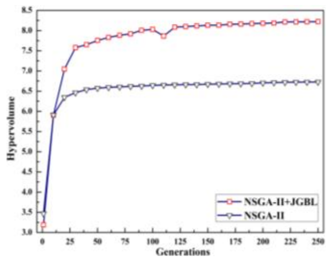
a. DTLZ 3



b. DTLZ 6



c. WFG 1



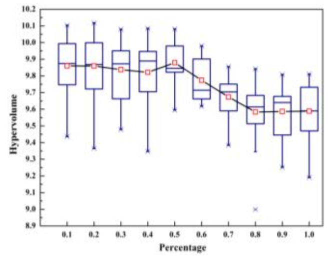
d. WFG 8

Figure 6. Evolutionary trajectories on selected benchmark problems

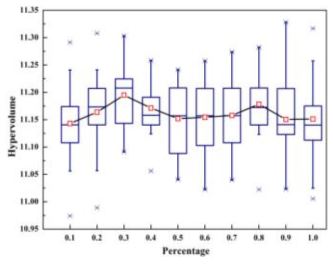
In the jumping gene paradigm, two important control parameters, namely jumping percentage and jumping rate, are introduced to control the number of genes forming the transposon and the probability to implement the jumping gene operations. Although these two parameters are fixed for the previous experiments, it is also interesting to see how the performance of JGBL paradigm varies with different parameter settings. So that a proper value (or range) of them would be provided for common users.

In order to study the sensitivity of jumping percentage, same experiments are conducted with the jumping percentage increased from 0.1 to 1.0 with the step size of 0.1. In addition to this, the jumping rate is adjusted from 0.05 to 1.0 with the step size of 0.05 for every investigated jumping percentage. To this end, we have 10 × 20 combinations for every benchmark problem. Due to the page limitations, only the experimental results on WFG test suites are given in Figure 7, while similar trends could be found in the other benchmark problems. Hypervolume indicator is employed here to illustrate the performance of each combination. In Figure 7, each boxplot indicates the statistical results of HV values within the variation of jumping rate in a specific jumping percentage. The red square in a boxplot represents the mean value. By carefully observing these subfigures, we could conclude that the performance would gradually deteriorate with the excessive increase of jumping percentage, except for WFG 6. In addition, [0.3, 0.6] would be a proper range where NSGA-II+JGBL could obtain acceptable results. In a word, a large number of gene transpositions would result in a deterioration for the performance of JGBL paradigm.

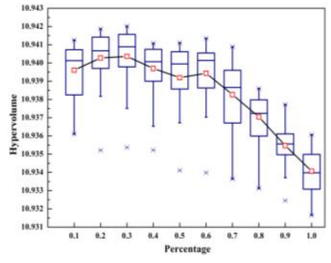
In order to understand how the effects of NSGA-II+JGBL are influenced by the jumping rate, the previous experiments are repeated except that the jumping percentage is fixed to 0.4. Analogous to the previous parameter sensitivity study, hypervolume indicator is also employed here to represent the performance for this particular jumping rate setting. Due to the limitation of pages, we also only show the hypervolume trajectories for WFG test suite here. According to these subfigures, we might find that HV values fluctuate a lot with different jumping rates. Nevertheless, this might be due to the rescale of the vertical axis in order to give a more intuitive illustration. However, a large jumping rate seem to be more probable to obtain satisfied results. [0.4, 0.8] is recommended here for unknown optimization problems.



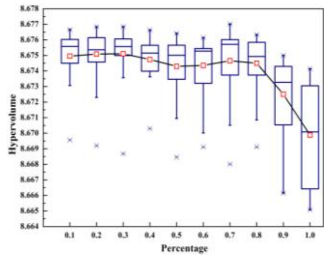
a. WFG 1



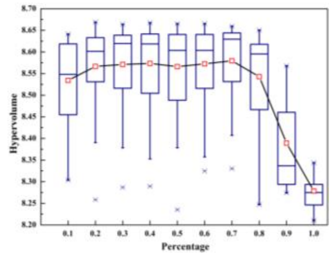
b. WFG 2



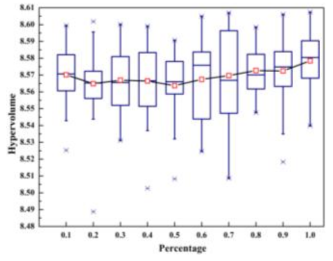
c. WFG 3



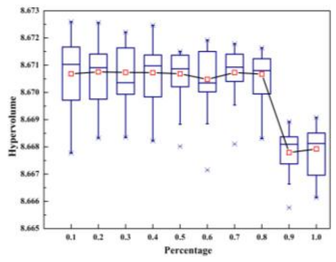
d. WFG 4



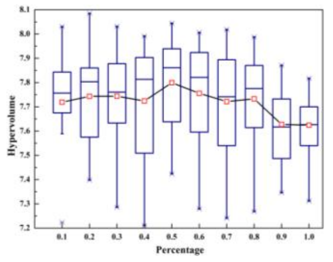
e. WFG 5



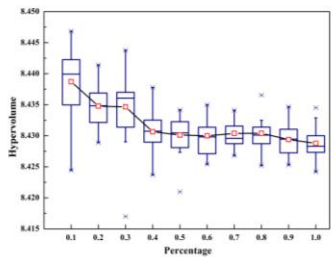
f. WFG 6



g. WFG 7



h. WFG 8



i. WFG 9

Figure 7. Sensitivity study of jumping percentage

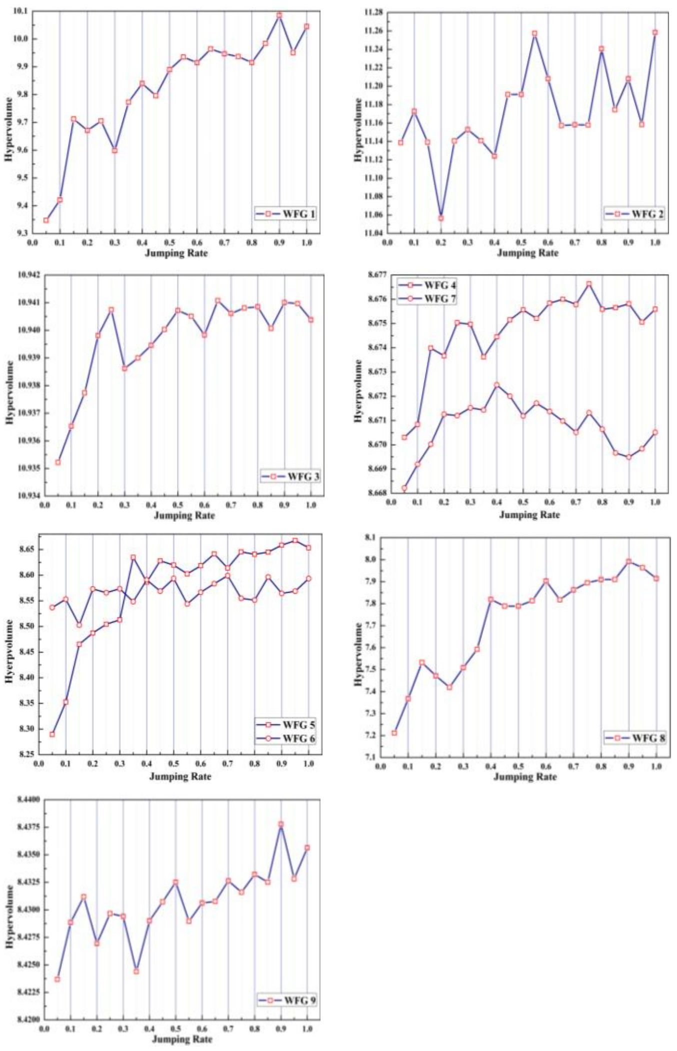


Figure 8. Sensitivity study of jumping rate

5. CONCLUSIONS AND FUTURE WORK

This paper provided a strategy, JGBL paradigm, which could be embedded into the framework of NSGA-II. JGBL paradigm is inspired from the mechanism of transposon in maize. Different from our previous work on jumping gene paradigm, JGBL paradigm is completely designed for solving problems in continuous search space. Furthermore, instead of alternating the state-of-the-art variation operators, the underlying motivation of JGBL paradigm is to explore some helpful information from those moderate solutions, which are non-dominated in current population, but eliminated by the environmental selection due to their poor performance on maintaining uniformity and diversity. In this way, needed energy could be injected to the search engine whenever the evolution gets trapped. Experimental results confirm the substantial improvements of the performance of baseline NSGA-II by the assistance of JGBL paradigm, in terms of convergence, diversity and uniformity at the same time. The proper ranges of two introduced parameters are also investigated for other unknown optimization problems.

Although the JGBL paradigm designed here is merely based on the framework of NSGA-II, we are sure that JGBL paradigm is simple and general enough, so that it could be embedded into any framework of MOEA and improve the exploration ability of the baseline algorithm. This work is left for our future discussion.

6. REFERENCES

[1] Schaffer, J. D. Multiple objective optimization with vector evaluated genetic algorithm. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pp: 93-100, Hillsdale, NJ, USA, 1985.

[2] Srinivas, N. and Deb, K. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3): 221-245, 1994.

[3] Deb, K., Pratap A., Agarwal S. and Meyarivan, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2): 182-197, 2002.

[4] Zitzler, E. and Thiele, L. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4): 257-271, 1999.

[5] Zitzler, E., Laumanns, M. and Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. TIK Report, 2001.

[6] Abbass, H. A. The self-adaptive Pareto differential evolution algorithm. In *Proceedings of the 2002 International Conference on Evolutionary Computation*, pp: 831-836, Piscataway, NJ, USA, 2002.

[7] Tan, K. C., Chiam, S. C., Mamun, A. A. and Goh, C. K. Balancing exploration and exploitation with adaptive variation for evolutionary Multi-objective optimization. *European Journal of Operational Research*, 197(2): 701-713, 2009.

[8] Zhang, Q., Zhou, A. and Jin, Y. RM-MEDA: A regularity model based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 12(1): 41-63, 2008.

[9] Nebro, A. J., Luna, F., Alba, E., Dorronsoro, B., Durillo, J. J. and Beham, A. AbYSS: Adapting scatter search to multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 12(4): 439-457, 2008.

[10] Chan, T. M., Man, K. F., Kwong, S. and Tang, K.S. A jumping gene paradigm for evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 12(2): 143-159, 2008.

[11] Ripon K. S. N., Kwong S. and Man, K.F. A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization. *Information Sciences*, 177(2): 632-654, 2007.

[12] McClintock, M. The origin and behavior of mutable loci in maize. In *Proceedings of National Academy of Sciences of the United States of America*, 36(6): 344-355, 1950.

[13] McClintock, M. Chromosome organization and gene expression. *Cold Spring Harbor Symposia on Quantitative Biology*, vol. 16, pp: 13-47, 1951.

[14] Simoes, A. B. and Costa, E. Enhancing transposition performance. In *Proceedings of the 1999 International Conference on Evolutionary Computation*, pp: 1434-1441, Piscataway, NJ, USA, 1999.

[15] Spirov, A. V. and Kazansky, A. B. Jumping Genes-mutators canrise efficacy of evolutionary search. In *Proceedings of the 2002 Genetic and Evolutionary Computation Conference*, pp: 561-568 2002.

[16] Kasat, R. B. and Gupta, S. K. Multi-objective optimization of an industrial fluidized-bed catalytic cracking unit (FCCU) using genetic algorithm with jumping gene operator. *Computer and Chemical Engineering*, 27(12): 1785-1800, 2003.

[17] Chan, T. M., Man, K. F., Tang, K. S. and Kwong, S. A jumping gene algorithm for multiobjective resource management in wideband CDMA systems. *Computer Journal*, 48(6): 749-768, 2005.

[18] Yeung, S. H., Man, K. F., Luk, K. M. and Chan, C. H. A trapeizform U-slot folded patch feed antenna design optimized with jumping genes evolutionary algorithm. *IEEE Transactions on Antennas Propagation*, 56(2): 571-577, 2008.

[19] Chan, T. M., Man, K. F., Tang, K. S. and Kwong, S. A jumping genes paradigm for optimizing factory WLAN networks. *IEEE Transactions on Industrial Informatics*, 3(1): 33-43, 2007.

[20] Zitzler, E., Deb, K. and Thiele, T. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2): 125-148, 2000.

[21] Deb, K., Thiele, T., Laumanns, M. and Zitzler, E. Scalable Test Problems for Evolutionary Multiobjective Optimization. In Ajith Abraham, Lakhmi Jain and Robert Goldberg (editors), Evolutionary Multiobjective Optimization. Theoretical Advances and Applications, pp:105-145, 2005.

[22] Huband, S., Hingston, P., Barone, L. and While, L. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477-506, 2006.

[23] Veldhuizen, D. A. V. and Lamont, G. B. Evolutionary computation and convergence to a Pareto front. In *Late Breaking Papers at the Genetic Programming Conference*, pp. 221–228, 1998.

[24] Schott, J. R. Fault tolerant design using single and multicriteria genetic algorithm optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1995.

[25] Deb, K. and Agrawal, R. B. Simulated Binary Crossover For Continuous Search Space. *Complex System*, 9(2): 115-148, 1985.

[26] Deb, K. and Goyal, M. A Combined Genetic Adaptive Search (GeneAS) For Engineering Design. *Computer Science and Informatics*, 26(4): 30-45, 1996.