A Good Nodes Set Evolution Strategy for Constrained Optimization

Chixin Xiao, Zixing Cai, *Senior Member ,IEEE* and Yong Wang

***Abstract*—Good Nodes Set(GNS) is a concept in number theory. To overcome the deficiency of orthogonal design to handle constrained optimization problems(COPs), this paper presents a method that incorporate GNS principle to enhance the crossover operator of the evolution strategy (ES) can make the resulting evolutionary algorithm more robust and statically sound. In order to gain the rapid and stable rate of converging to the feasible region, traditional crossover operator is split into two steps. GNS initialization methods is applied to ensure the initial population span evenly in relatively large search space and reliably locate the good points for further exploration in subsequent iterations. The proposed method achieves the same sound results just as the orthogonal method does, but its precision is not confined by the dimension of the space. The simplex selected and diversity mechanism similar to Carlos's SMES is used to enrich the exploration and exploitation abilities of the approach proposed. Experiment results on a set of benchmark problems show the efficiency of our methods.**

I. Introduction

In many constrained optimization problems, the challengeable issue is to avoid be trapped in the local optima of objective function especially when the dimension is high and there are numerous local optima (e.g.,see[3-6]).

Many researchers treat the crossover operator sampling the genes from the parents to produce some potential offspring as a sampling experiment. They incorporate orthogonal design [7-9] into crossover operator and gain many sound results. In evolution strategy (ES), considered the rising dimension and the precision of objective function, the orthogonal table which designed for crossover operator must be revised at every turn. However, constructing an orthogonal table for high dimension isn't an easy thing. In fact, the higher dimensions of object function are, the more distinct deficiency of orthogonal method is.

In this paper, we introduce an evolutionary algorithm using Good Nodes Set (GNS) [1][2] principle for constrained optimization. Zhang [2] has applied the GNS techniques to crossover operator of genetic algorithm (GA) in binary code. The precision of this method has nothing to do with the space dimension and the GNS idea can make the points set more evenly distributed than random points. As we known, one of the important techniques to construct GNS is the method of good lattice points (GLP). The method of good lattice points also called Korobov lattice rules is a central technique from the fields of Monte Carlo (MC) and quasi-Monte Carlo (QMC) methods. Good lattice points are classical node sets for QMC integration, defined by the Russian mathematician Korobov [10-12]. But we attempt to construct the GNS based on Hua and Wang[1]. These motivated the changes to real code evolutionary algorithms are introduced in this paper.

This paper is organized as follows. In Section 2 we define the global nonlinear optimization problem that we aim to solve. After that, in Section 3 the detailed features of our algorithm are introduced. Section 4 presents the experimental results on a set of benchmark problems. Comparisons with other evolutionary algorithms are also included in this section. Finally, Section 5 concludes with a brief summary of the paper.

Chixin Xiao is with the College of Information Science and Engineering, , Central South University, Changsha, Hunan, China and Information Engineering College, Xiangtan University ,Xiangtan Hunan, P.R. China(e-mail: chixinxiao@hotmail.com)

Zixing Cai and Yong Wang are with the College of Information Science and Engineering,Central South University, Changsha, Hunan, China

This work was supported by National Basic Scientific Research Founds under Grant A1420060159, National Natural Science Foundation of China under Grant 60234030, 60404021 and Hunan Science Foundation under Grant 06IJY3035

II. Statement of the Problem

We are interested in the general nonlinear programming problem in which we want to:

Find $\bar{x}$ which optimizes $f(\bar{x})$ (1)

Subject to

$g_i(\bar{x})\le 0, \quad i=1,\ldots,n$ (2)

$h_i(\bar{x})\le 0, \quad j=1,\ldots,p$ (3)

Where $\bar{x}$ is the vector of solutions $\bar{x}=[x_1,x_2,\ldots,x_r]^T$, n is the number of inequality constrains, and p is the number of equality constrains (in both cases, constraints could be linear or nonlinear).

If we denote with F to the feasible region and with S to the whole search space, then it should be clear that $F\subseteq S$.

For an inequality constraint that satisfies $g_i(\bar{x})=0$, then we will say that is active at $\bar{x}$. All equality constraints h_j (regardless of the value of $\bar{x}$ used) are considered active at all points of F.

III. Our Mechanisms and Algorithms

Our new approach is based on the similar frame of an $(\mu+\lambda)$-ES that discussed in [11]. We put the emphases on modification of crossover operator. The detailed features of our algorithm are the following.

1-4244-1340-0/07$25.00 ©2007 IEEE

A. Background on Good Nodes set (GNS):[1][2]

1. Let G_S is S-dimensional unit cube in Euclidean space. If $\bar{x}\in G_S, \bar{x}=(x_1,x_2,...,x_S)$, then $0\le x_i\le 1$, for all $i=1,2,...,s$.
2. If a set of n points $P_n(k)\subseteq G_S$, then $P_n(k)=\{(x_1^{\{n\}}(k),...,x_S^{\{n\}}(k))$, $1\le k\le n$, $0\le x_i^{(n)}(k)\le 1$, $1\le i\le s\}$.
3. Let $\bar{r}\in G_S$, if $P_n(k)=\{(\{r_1^{\{n\}}*k\},...,\{r_i^{\{n\}}*k\},...,\{r_S^{\{n\}}*k\})\}$, $\{r_i*k\}$ is the decimal fraction of r_i*k, $k=1,2,...,n$, has discrepancy $\varphi(n)$, $\varphi(n)=C(r,\varepsilon)n^{(-1+\varepsilon)}$, where $C(r,\varepsilon)$ is constant only relate with $r,\varepsilon(\varepsilon>0)$, then $P_n(k)$ is called Good Nodes Set and $\bar{r}$ is called Good Node.
4. If let $r_i=2\cos\frac{2\pi i}{p}$, $1\le i\le s$, where p is the minimum prime number content with $\frac{p-3}{2}\ge s$, then $\bar{r}$ is a Good Node. If let $r_i=e^i$, $1\le i\le s$, $\bar{r}$ is a Good Node also.

When we want to estimate the integral of a function f over the s-dimensional unit hypercube G_S, namely $u=\int_{x\in G_S} f(x)dx$, by the average value of f over any point sets $P_n(k)(1\le k\le n)$, $Q_n=\sum\frac{f(p_n(i))}{n}$, the integration error $E_n=\mu-Q_n$ is the smallest when $P_n(k)(1\le k\le n)$ consist of Good Node, i.e., if we estimate the integral by any node set which comprises nodes in equal quantity , the Good Nodes are distributed more evenly than any other nodes

We construct two node sets shown as Fig.1, one comprises 100 Good Nodes in 2-dimensional unit space, the other is 100 random nodes in 2-dimensional unit space. We can easily find that Good Nodes are distributed more evenly than random nodes.

This is a good idea for high dimension approximate computation. In other words, the idea of GNS make the point set distributed more evenly than random points. When we construct the real coded crossover operator of ES by means of GNS, obviously, the cross points are even and representative. This is the same effect as the orthogonal crossover method does. However, the precision of the latter is confined by the dimension of the search space and the size of the orthogonal table, the former is not.

B. Crossover operator based on GNS

If we select two individuals from parents at randomly, let $X_1=(x_1^1,x_2^1,...,x_n^1)$, $X_2=(x_1^2,x_2^2,...,x_n^2)$, then define a local search space $T:\left[\underline{x}X_1,X_2,\bar{x}X_1,X_2\right]$,with

$$\begin{cases}\underline{x}X_1,X_2=(\min(x_1^1,x_1^2),\min(x_2^1,x_2^2),...,\min(x_n^1,x_n^2))\\ \bar{x}X_1,X_2=(\max(x_1^1,x_1^2),\max(x_2^1,x_2^2),...,\max(x_n^1,x_n^2))\end{cases} \quad (5)$$

In order to measure the degree of the difference between two decision variables at same dimension respectively. We define $d(x_i^1,x_i^2)\equiv\left|x_i^1-x_i^2\right|\le\delta$, with $1\le i\le n$, where δ is a more small positive constant than the scope of the decision variables (in this paper $\delta=0.01$). If at i-dimension of the two selected individuals the decision variables content with the former inequation, then we call two individuals i-dimensional $\delta-similar$. We define a not $\delta-similar$ decision variables set by

$$J=\{i|x_i^1 \text{ not } \delta-similar \text{ with } x_i^2, 1\le i\le n\}.$$

Assume individual X_1 and individual X_2 are not $\delta-similar$ among the first t-dimensional decision variables, but the last $n-t$-dimensional decision variables are $\delta-similar$.

In the t-dimensional local search space we can find the Good Nodes as follows.

In t-dimensional space H the so-called Good Nodes Set (GNS) which contain n points is defined by

$$P_n(k)=\{(\{r_1*k\},\{r_2*k\},...,\{r_t*k\}),k=1,2,...,n\},$$

with $r_i=2\cos\frac{2\pi i}{p}$, $1\le i\le t$,where p is the minimum prime

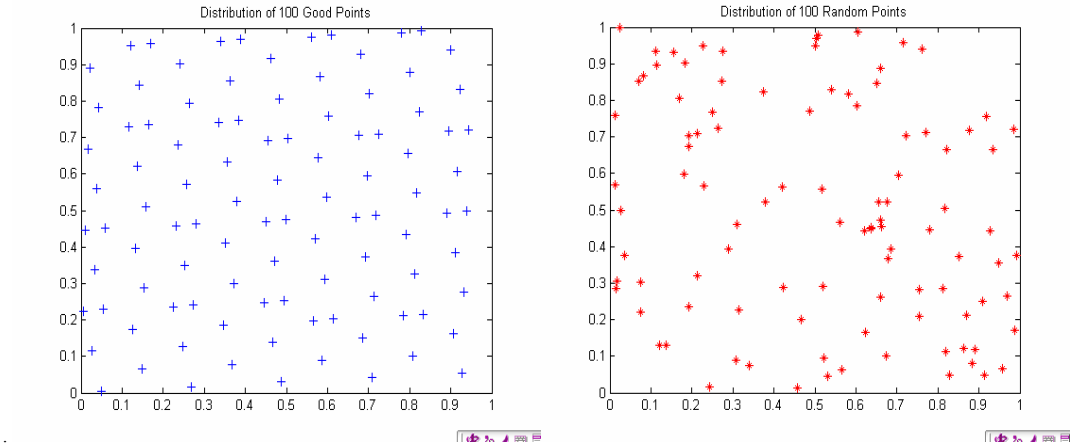

Fig.1. Comparison of distribution between 100 Good Points and 100 Random Points in 2-dimensional unit space

number which content with $p \geq 2t+3$ and $\{r_i * k\}$ is the decimal fraction of $r_i * k$ (or with $r_i = e^i$, $1 \leq i \leq t$).

Good Nodes Set principle is based on unit hypercube or hypersphere, in order to mapping n Good Nodes from unit space $H:[0,1]^t$ to the search space $T:\left[\underline{x}_{L,M}, \bar{x}_{L,M}\right]^t$ of the questions, we define the function $f: H \to T$ by

$$f(r_i * k) = \min(x_i^1, x_i^2) + \{r_i * k\} * (\max(x_i^1, x_i^2) - \min(x_i^1, x_i^2)) \quad (6)$$

with $1 \leq i \leq t$, $1 \leq k \leq n$, where $\{r_i * k\}$ is the same as the former defined.

function GNS-crossover()

Input two individuals $X_1 = \{x_1^1, x_2^1, \ldots, x_n^1\}, X_2 = \{x_1^2, x_2^2, \ldots, x_n^2\}$ which are the output from the temporary crossover operator

For $j=1$ **to** λ / μ **Do**

For $i=1$ **to** n (DIMENSION_OF_INDIVIDUALS) **Do**

If($i \notin J$ **)**

If(flip(50%) **)**

$x_{ji}^{new} = x_i^1;$

Else

$x_{ji}^{new} = x_i^2;$

End If

Else

$x_{ji}^{new} = f(r_i * j);$

End If

End For

End For

End

Fig.2. Pseudocode of the GNS crossover operator used by our approach. J is a set defined as former; $\mathrm{flip}(P)$ is a function that returns TRUE with probability P。.

Now, we employ an example to illustrate the above process. Suppose that two individuals are selected from parent population with the following vector values of X_1, X_2 and four offspring are to be generated through the GNS crossover operator.

$X_1 = (9.5, 11.3, 45, 30.5, 55.4)$, $X_2 = (2.5, 84, 23, 30, 55)$

Since $d(30.5, 30) \equiv |30.5 - 30| \leq \delta$, $d(55.4, 55) \equiv |55.4 - 55| \leq \delta$, where $\delta = 0.01$, obviously, the first 3-dimensiomal decision variables between two individuals are not δ–*similar* and the last 2-dimensional decision variables are δ–*similar*.

Let P_4 be 3-dimensional Good Nodes set (GNS) which contains four Good Nodes

$P_4 = \{(\{r_1 * k\}, \{r_2 * k\}, \{r_3 * k\}), k = 1, 2, \ldots, 4\}$,

where $r_i = 2\cos\frac{2\pi i}{p}$, $1 \leq i \leq 3$, since p is minimum primer number and $p \geq 2*4+3$, we have $p = 11$.

The GNS P_4 is given by

$$\begin{Bmatrix} (0.682507, 0.830830, 0.284629) \\ (0.365014, 0.661660, 0.569259) \\ (0.047521, 0.492490, 0.853889) \\ (0.730038, 0.323320, 0.138518) \end{Bmatrix}$$

Now through function $f: H \to T$ we can get four new offspring individuals in the search space given by

$$\begin{Bmatrix} (7.277549, 71.701341, 29.261838, 30, 55) \\ (5.055098, 59.402682, 35.523698, 30.5, 55.4) \\ (2.832647, 47.104023, 41.78558, 30.5, 55) \\ (7.610266, 34.805364, 26.047396, 30, 55.4) \end{Bmatrix}$$

The pseudocode of the GNS crossover operator is shown in Fig.2. For every couple of the input individuals, they generate λ / μ offspring. If the decision variables between the parent are δ–*similar* ($i \notin J$), the offspring would inherit any of the parent decision variables at the same dimension with 50% probability. The rest would be generated by Good Nodes Set(GNS) method.

From the above results, one can conclude that the GNS crossover operator can generate new offspring in search space more evenly than random method or orthogonal method. And its precision isn't confined by the space dimension. The numerical results in the Section.4 show the efficiency of the proposed method.

But when come to such problems that scope of decision variables are very large and the number of the points generated by the GNS crossover operator is very small, that is, the points is distributed sparsely in the search space.

function Temporary-crossover()

Select randomly $P_1 = \{p_1^1, p_2^1, \ldots, p_n^1\}, P_2 = \{p_1^2, p_2^2, \ldots, p_n^2\}$ from μ parents population

For $i=1$ **to** n (DIMENSION_OF_INDIVIDUALS) **Do**

If(flip(50%) **)**

$x_i^1 = p_i^1;$

Else

$x_i^1 = p_i^2;$

End If

$x_i^2 = r * p_i^1 + (1 - r) * p_i^2;$

End For

Return $X_1 = \{x_1^1, x_2^1, \ldots, x_n^1\}$, $X_2 = \{x_1^2, x_2^2, \ldots, x_n^2\}$

End

Fig.3. Pseudocode of the temporary crossover operator before GNS crossover operator used by our approach. $\mathrm{flip}(P)$ is a function that returns TRUE with probability P. Let $r = 0.5$.

Evidently, the best effect can be reached when the points in unit search space keep a certain probability. In our proposed method the number of offspring generated by each parent is a constant (λ / μ), however, the search spaces of different problems are not same. If we want to get the content probability of the Good Nodes distributed in search space as quickly as possible, one simple effective way is to decrease the search space dramatically. So we devise a temporary crossover operator before the GNS crossover operator which input two random parent individuals

$$P_1 = \{p_1^1, p_2^1, \ldots, p_n^1\}, \; P_2 = \{p_1^2, p_2^2, \ldots, p_n^2\}$$

and generate two temporary individuals

$$X_1=\{x_1^1,x_2^1,\ldots,x_n^1\},\ X_2=\{x_1^2,x_2^2,\ldots,x_n^2\}$$

with the same probability. The pseudocode is shown in Fig.2. When we select two individuals from the parent population, we get a local search space. For every dimension of the local search space, we select two points, one is p_i^1 or p_i^2 with the 50% probability, the other is $0.5*(p_i^1+p_i^2)$, that is, it is a middle point of the dimension. Through this process the original local search space is reduced into 1/2. Obviously, with the same quantity Good Nodes the distributing probability of the Good Nodes rise rapidly.

From the above results, one can conclude that the GNS crossover operator can generate new offspring in search space more evenly than random method or orthogonal method. And its precision isn't confined by the space dimension. With the help of the temporary crossover operator, the offspring population can converge rapidly towards the feasible region.

Table 1Comparison of five algorithms (SMES [14], ASCHEA[15], SR[13], CW[17] and GNSES) on 13 benchmark functions. NA = not available.

Fcn/ optimal	status	methods				
		SMES[14]	**ASCHEA** [15]	**SR** [13]	**CW**[17]	**GNSES**
g01/ -15.000	**best**	**-15.000**	**-15.0000**	**-15.000**	**-15.000**	**-15.000**
	mean	-15.000	-14.8400	-15.000	-15.000	-15.000
	worst	-15.000	NA	-15.000	-15.000	-15.000
	st. dev	0		2.0E-06	1.3E-14	0
g02/ -0.803619	**best**	**-0.803601**	-0.785	-0.803515	**-0.803619**	**-0.803611**
	mean	-0.785238	-0.788950	-0.724886	-0.803220	-0.78484
	worst	-0.751322	NA	-0.590908	-0.792608	-0.754106
	st. dev	1.7E-02		7.0E-02	2.0E-03	0.016254
g03/ -1.000	**best**	**-1.000**	**-1.000000**	**-1.000**	**-1.000**	1.000923
	mean	-1.000	-0.999970	-0.789	-1.000	1.001031
	worst	-1.000	NA	-0.640	-1.000	1.001051
	st. dev	2.1E-04		1.1E-01	2.8E-16	3.29E-05
g04/ -30665.539	**best**	**-30665.539**	-30665.50	**-30665.539**	**-30665.539**	**-30665.53867**
	mean	-30665.539	-30665.500	-30665.539	-30665.539	-30665.53867
	worst	-30665.539		-30665.539	-30665.539	-30665.53867
	st. dev	0		0	8.0E-12	2.62E-11
g05/ 5126.498	**best**	5126.599	5126.500	**5126.497**	**5126.498**	5126.534
	mean	5174.492	5126.530	5207.411	5126.498	5222.371
	worst	5304.167	NA	5327.391	5126.498	5470.453
	st. dev	5.0E+01		6.9E+01	1.513E-12	96.45524
g06/ -6961.814	**best**	**-6961.814**	-6961.81	**-6961.814**	**-6961.814**	**-6961.814**
	mean	-6961.284	-6961.810	-6961.814	-6961.814	-6961.57
	worst	-6952.482	NA	-6961.814	-6961.814	-6954.38
	st. dev	1.9E+00		0	1.8E-12	1.357006
g07/ 24.306	**best**	24.327	24.33230	**24.307**	**24.306**	**24.31246**
	mean	24.475	24.66360	24.306	24.306	24.48019
	worst	24.843		24.306	24.306	24.85859
	st. dev	1.3E-01		1.0E-06	5.7E-12	0.147817
g08/ -0.095825	**best**	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**	**0.095825**
	mean	-0.095825	-0.095825	-0.095825	-0.095825	0.095825
	worst	-0.095825	NA	-0.095825	-0.095825	0.095825
	st. dev	0		0	3.2E-17	5.646E-17
g09/ 680.630	**best**	680.632	680.632	**680.630**	**680.630**	**680.6309**
	mean	680.643	680.6410	680.630	680.630	680.6447
	worst	680.719	NA	680.630	680.630	680.7146
	st. dev	1.6E-02		0	4.7E-13	0.021526
g10/ 7049.248	**best**	**7051.903**	7061.13	7054.316	**7049.248**	**7050.814**
	mean	7253.047	7615.200	7049.248	7049.248	7252.625
	worst	7638.366	NA	7049.248	7049.248	7408.321
	st. dev	1.4E+02		1.7E-04	4.0E-09	90.9047
g11/ 0.75	**best**	**0.75**	**0.750**	**0.750**	**0.750**	**0.749267**
	mean	0.75	0.750000	0.758	0.750	0.749858
	worst	0.75	NA	0.796	0.750	0.749864
	st. dev	1.5E-04		1.7E-02	0.0E+00	9.279E-5
g12/ -1.000	**best**	**-1.000**	NA	**-1.000**	**-1.000**	**-1.000**
	mean	-1.000	NA	-1.000	-1.000	-1.000
	worst	-1.000	NA	-1.000	-1.000	-1.000
	st. dev	0		0	0.0E+00	0
g13/ 0.0539498	**best**	0.053986	NA	**0.053957**	**0.0539498**	**0.05395**
	mean	0.166385	NA	0.288324	0.0539498	0.25081
	worst	0.468294	NA	0.392100	0.0539498	0.482565
	st. dev	1.8E-01		1.7E-01	6.5E-17	0.2101

The numerical results in the Section.4 show the efficiency of the proposed method.

C. Mutation operator

In our proposed algorithm Gaussian mutation operator [13][14]is chosen and does not introduce any specialized constraint-handing variation operators. We would like to show that specialized and complex variation operators for constrained optimization problem are unnecessary, although they may be quit useful for particular types of problems. The parameters of our Gaussian mutation operator are given as follows.

$$\begin{cases} \sigma_i^{k+1}(j)=\sigma_i^k(j)\exp\left\{\tau'*N(0,1)+\tau*N_j(0,1)\right\}, j=0,1,...,n-1; \\ x_i^{k+1}(j)=x_i^k(j)+\sigma_i^{k+1}(j)*N_j(0,1), j=0,1,...,n-1; \end{cases} \quad (7)$$

with $\sigma_i^0(0)=0.4*(\frac{\Delta x_i}{\sqrt{n}})$, $\tau'=1/\sqrt{2\sqrt{n}}$, $\tau=1/\sqrt{2n}$. $N(0,1)$ is a normally distributed one-dimensional random variable with an expectation of 0 and variance 1.

D. The self-adaptive mechanism and a comparison mechanism based on the following criteria

1. Between two feasible solutions, the one with the highest fitness value wins.

2. If one solution is feasible and the other one is infeasible, the feasible solution wins.

3. If both solutions are infeasible, the one with the lowest sum of constraint violation is preferred.

E.GNS initialization

The uniform distribution of the initial population in the search space can guarantee evenly to scan the search space and easily to locate the good points for further exploration in subsequent iterations. But to a relatively large search space with a limited quantity of population, for example in g10 that depicted in Section 4, the efficiency of traditional random methods can not catch up with the GNS methods. In this paper, we apply the approach to initial the population just like (6) equation discussed in Section3. In Section4, the experimental results show the validity of our approaches.

F. Select next generation [14]

The proposed method allow infeasible individual with the best value of the objective function and with the lowest amount of constraint violation to survive to the next generation. These solutions can be chosen either from the parents or offspring population, with 50% probability. And all these solutions only have 3% probability to survive, that is, 97% probability is to select the best individuals based on comparison mechanism from the union of the parents and offspring population.

IV. Experiments and Results

Summarizing, our approach works over a simple multimembered evolution strategy (SMES): $(\mu+\lambda)-ES$. We use a set of benchmark problems G01 to G13 for testing the performance of the proposed algorithm. These problems are proposed in [13],[14], [15]. And for all test problems, the parameters are fixed to as following: μ=100 , λ=300 ,Number of generations: T = 800

To deal with equality constraints, a dynamic mechanism originally proposed in ASCHEA [15] and used in [14] is adopted. The tolerance value ε is decreased with respect to the current generation using the following expression: $\varepsilon_j(t+1)=\varepsilon_j(t)/1.00195$ (the initial ε_0 was set to

Table 2 Comparison of $\delta = 0.01$ and respectively apply two such methods **Only GNS, Non-GLP-initial** as experiment approaches

Fcn/ optimal	status	Methods($\delta = 0.01$)	
		Only GNS	**Non-GLP-initial**
g01/ -15.000	**best**	**-15.000**	**-15.000**
	mean	**-14.8438**	**-15.000**
	worst	**-13.8281**	**-15.000**
	st. dev	**0.405171**	**0**
g02/ -0.803619	**best**	0.803603	0.803605
	mean	0.763166	0.785355
	worst	0.676795	0.734671
	st. dev	0.040161	0.017558
g03/ -1.000	**best**	1.000985	1.001045
	mean	1.001021	1.000899
	worst	1.00104	0.997094
	st. dev	1.35E-05	0.000719
g04/ -30665.539	**best**	-30665.5	-30665.5
	mean	-30665.5	-30665.5
	worst	-30664.1	-30665.5
	st. dev	0.254221	7.4E-12
g05/ 5126.498	**best**	**5126.518**	**5126.526**
	mean	**5262.464**	**5229.237**
	worst	**5721.154**	**5784.043**
	st. dev	**170.7444**	**133.5752**
g06/ -6961.814	**best**	-6961.81	-6961.81
	mean	-6961.81	-6961.8
	worst	-6961.81	-6961.27
	st. dev	9.25E-13	0.09984
g07/ 24.306	**best**	24.3307	24.32305
	mean	24.44566	24.48925
	worst	24.69179	24.79215
	st. dev	0.094495	0.112864
g08/ -0.095825	**best**	0.095826	0.095826
	mean	0.095826	0.095826
	worst	0.095826	0.095826
	st. dev	5.65E-17	5.65E-17
g09/ 680.630	**best**	680.6313	680.6322
	mean	680.64	680.6436
	worst	680.6786	680.6904
	st. dev	0.00857	0.012593
g10/ 7049.248	**best**	**7070.376**	**7084.676**
	mean	**7243.251**	**7288.971**
	worst	**7551.211**	**8031.552**
	st. dev	**120.1833**	**186.9797**
g11/ 0.75	**best**	0.749547	0.749312
	mean	0.749777	0.74984
	worst	0.749813	0.751709
	st. dev	5.07E-05	0.00038
g12/ -1.000	**best**	1.000	1.000
	mean	1.000	1.000
	worst	1.000	1.000
	st. dev	0	0
g13/ 0.0539498	**best**	0.054443	0.054417
	mean	0.144786	0.165779
	worst	0.444297	0.474938
	st. dev	0.167715	0.182812

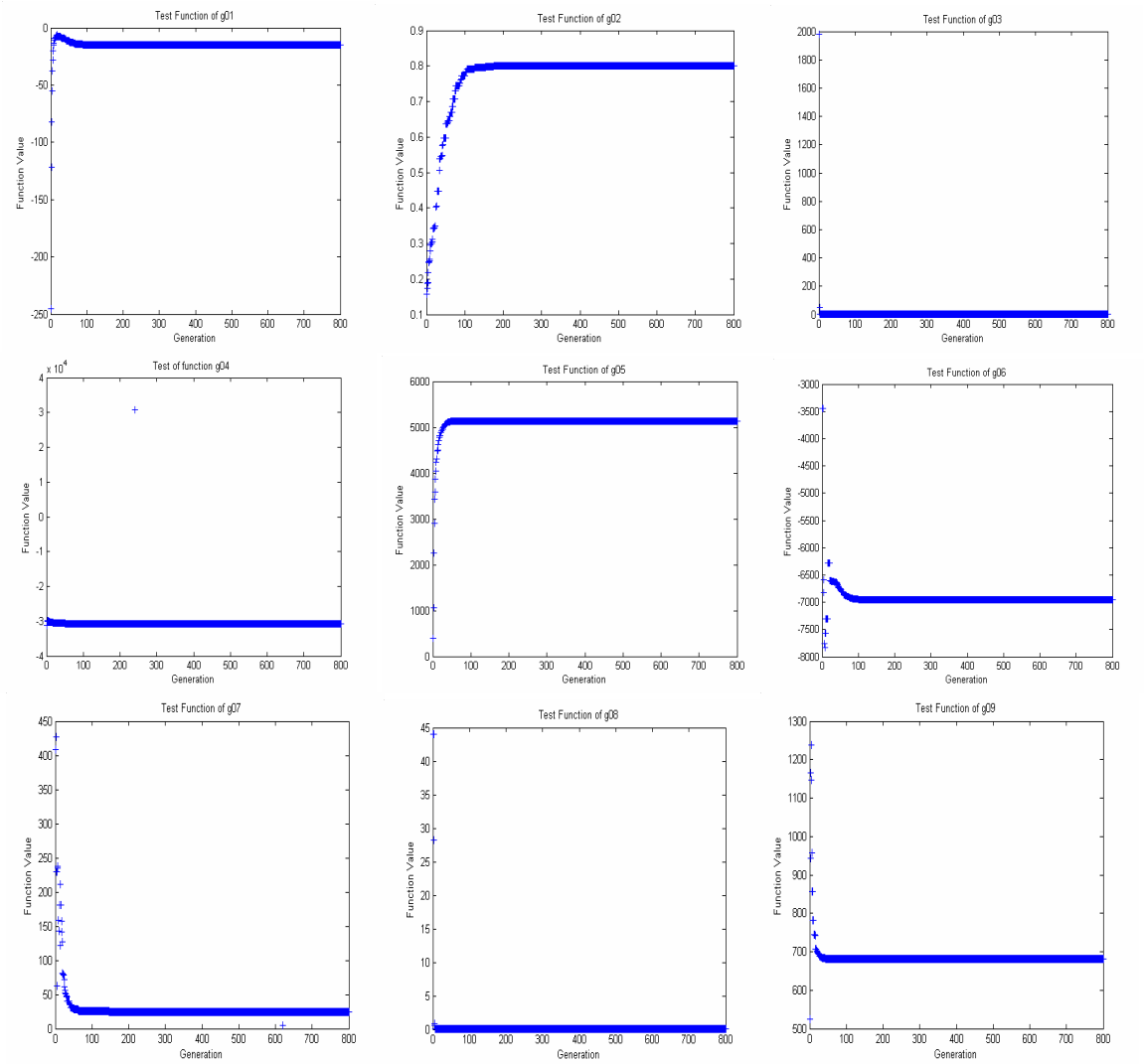

Fig 4. Convergence results of the test problems g01-g09.

0.001).

General performance of the proposed algorithm

Table 1 summarizes the experimental results using the above parameters. These parameters were selected just similar to [13][14][16]. The table shows the known "optimal" solutions for each problem, and records the best, median, mean, worst, and standard deviations of the objective function values found over the 30 independent runs.

Table 1 summarizes the comparison of results with our (indicated by GNSES) algorithms, Mezura-Montes and Coello's (indicated by SMES [14])algorithms , Runarsson and Xin Yao's (indicated by SR [13])algorithms ,Hamida and Schoenauer's (indicated by ASCHEA [15])algorithms, Cai and Wang's(indicated by CW[17]) algorithms.

One of the first observations is that for each problem, the best solution is almost equivalent to the optimal solution. For problems g01, g03, g08, g11and g12, the optimal solutions are consistently found in all 30 runs. For problems g02, g04, g06, g07, g09 and g13, the close-to-optimal solutions are found in all 30 runs. For problem g05, the optimal solution is not consistently found; the primary feature of this function is that it contains three equality constraints. Another problem whose optimum is not found consistently is g10; the main characteristics of this problem are its relatively large search space and three active constraints at the optimum.

On the other hand, note that the standard deviations over 30 runs for all the problems except g05 and g10 are relatively small. These results validate that GNSES has the substantial capability in handling various COPs and its solution quality is quite stable.

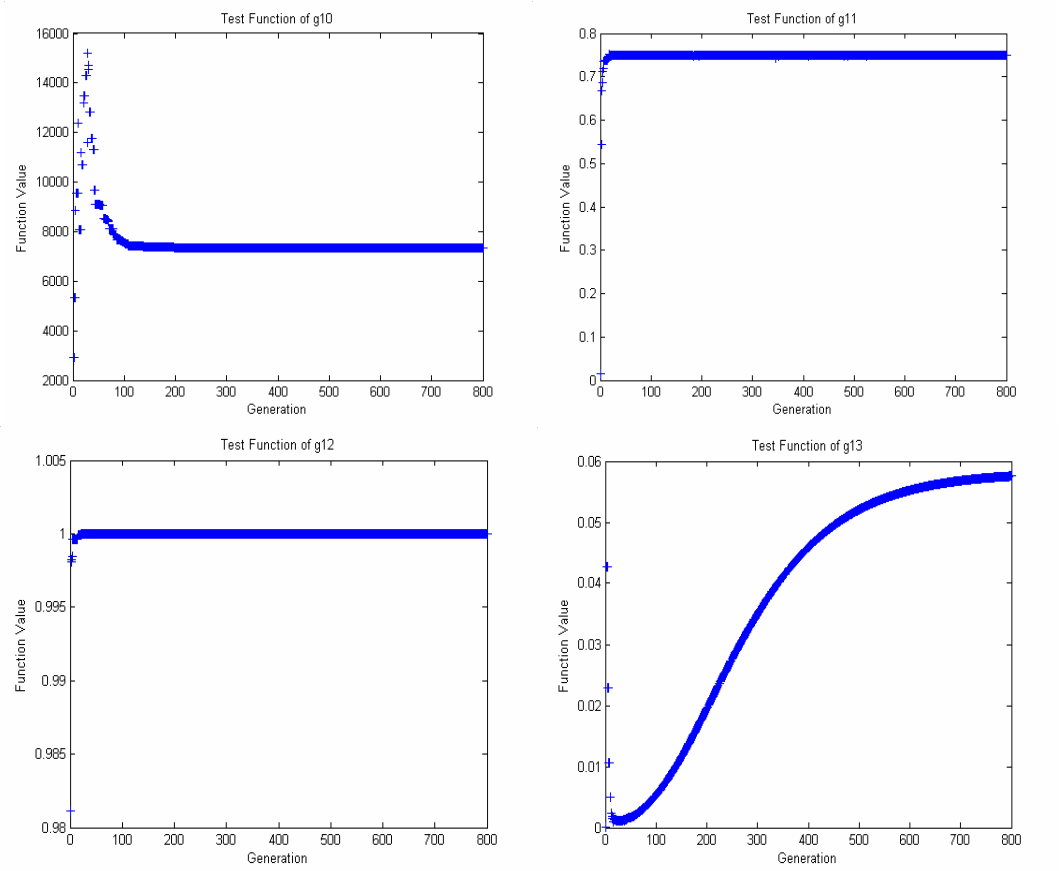

Fig5. Convergence results of the test problems g10-g13.

Table2 summarizes the comparison of results respectively apply two such methods indicated by Only GNS and Non-GLP-initial as experiment approaches with δ=0.01 . "Only GNS" represents the proposed algorithms only use GNS crossover operator (i.e. not apply the temporary crossover operator), one can easily find that for all problems except g05 and g10, the optimal or close-to-optimal solutions are consistently found in all 30 runs. For g01, the optimal solution can be found in all 30 runs, but its standard deviation rises evidently. Besides analysis as before, the main characteristics of these problems are their scope discrepancy of each dimension. For GNS principle is based on the hyper unit sphere, the scope of each dimension is equal to the others. As discussed in Section 3, we add the temporary crossover operator before the GNS one and gain the sounds results shown in "Non-GNS-intial" column with boldface except for g10.

"Non-GNS-initial" represents our proposed methods apply two steps crossover operator as discussed in Section3, but not use GNS initial the population. When the GNS initialization adopted, we get the final results shown in Table1. The problem g10 not only gain the better solution, but also gain the lower standard deviation.

Convergence dynamics of the proposed GNSES

Furthermore, feasible solutions are continuously found for all the test problems in 30 runs. This is largely credited by the two special operations of GNSES as discussed. Indeed, the rate at which the algorithm lies within the feasible region is very fast during the evolutionary process based on the observations. After discussing the quality and robustness of the proposed approach, we want to verify the rate at which the algorithm is able to reach the optimal or close-to-optimal solutions, i.e. the convergence dynamics of the approach. The convergence result for each problem is depicted in Fig.4 and 5 so that the efficiency of GNSES can be shown more explicitly.

As can be seen, for all problems other than g13, GNSES can converge to a stable value not more than 100 generations. Moreover, GNSES can achieve the convergence at the final stage for the remaining problems.

These results can be attributed to the two merits of GNSES. Firstly, the uniform distribution of the initial population in the search space can be guaranteed by the initialization procedure. Second, GNS can evenly scan the search space once to locate the good points for further exploration in

subsequent iterations.

From the tables, one may find that the results of our algorithm are better not only in terms of objective value but also in terms of solution violations.

In summary, our results illustrate the algorithm proposed can obtain the ideal solutions without the restriction of the search space dimensions. This improved character is more superior to that of orthogonal methods.

V. CONCLUSION

This paper is a continuation of the study devoted to SMES[14], a new operator for ES introduced in this work. By introducing with the GNS design, the evolutionary algorithm based on GNS can solve constrained optimization problems efficiently. The validity of the proposed algorithm is tested on a set of benchmark problems and the experimental results are very promising. From these results, we may conclude that the proposed algorithm seems to be a useful candidate for ES.

REFERENCES

1. Luo-geng Hua, Yuan Wang, The Application of Number Theory in Approximate Analysis, Science Press, Beijing, 1978,10:83-87 (in Chinese).
2. Ling Zhang, Bo Zhang, Good point set based genetic algorithm, Chinese Journal of Computer, 2001,9, Vol.24, No.9(in Chinese).
3. W.W. Hager,W. Hearn, and P. M. Pardalos, Eds., Large Scale Optimization:State of the Art. Norwell, MA: Kluwer, 1994.
4. C. A. Floudas and P. M. Pardalos, Eds., State of the Art in Global Optimization:Computational Methods and Applications. Norwell, MA: Kluwer, 1996.
5. M. A. Stybinski and T. S. Tang, "Experiments in nonconvex optimization:Stochastic approximation and function smoothing and simulated annealing," Neural Networks, vol. 3, pp. 467–483, 1990.
6. P. Siarry, G. Berthiau, F. Durbin, and J. Haussy, "Enhanced simulated annealing for globally minimizing functions of many-continuous variables," ACM Trans. Math. Software, vol. 23, pp. 209–228, June 1997.
7. D. C. Montgomery, Design and Analysis of Experiments, 3rd ed. New York: Wiley, 1991.
8. Yiu-Wing Leung, An Orthogonal Genetic Algorithm with Quantization for Global Numerical Optimization, 2001,02,Vol.5,No.1.
9. C. R. Hicks, Fundamental Concepts in the Design of Experiments,4th ed. TX: Saunders College Publishing, 1993.
10. N. Korobov, The approximate calculation of multiple integrals, Dokl. Akad. Nauk SSSR 124 (1959) 1207–1210(in Russian).
11. N.Korobov, Properties and calculation of optimal coefficients, Dokl.Akad. Nauk SSSR 132 (1960) 1009–1012 (in Russian)(English transl.: Soviet Math. Dokl. 1, 696–700).
12. N. Korobov, Number-Theoretic Methods in Approximate Analysis, Fizmatgiz, Moscow, 1963 (in Russian).
13. Runarsson, T.P., Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization, IEEE Trans. on Evolutionary Computation, Vol.4, No.3 (2000)
14. Mezura-Montes E, Coello Coello CA. A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Trans. Evol. Comput, 2005,9(1):1–17.
15. Hamida, S.B., Schoenauer, M.: ASCHEA: New results using adaptive segregational constraint handling. Proc. of the 2002 Congress on Evolutionary Computation (2002)
16. Efr´en Mezura-Montes and Carloa A. Coello Coello: Adding a Diversity Mechanism to a Simple Evolution Strategy to Solve Constrained Optimization Problems. In Proceedings of the Congress on Evolutionary Computation 2003 (CEC'2003), volume 1, pages 6–13, Piscataway, New Jersey, December 2003. Canberra, Australia, IEEE Service Center.
17. Cai Z, Wang Y. A multiobjective optimization based evolutionary algorithm for constrained optimization. IEEE Trans. Evol. Comput, 2006,10(6).