Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 214814, 14 pages
http://dx.doi.org/10.1155/2013/214814

Research Article

A Mutual-Evaluation Genetic Algorithm for Numerical and Routing Optimization

Chih-Hao Lin and Jiun-De He

Department of Information Management, Chung Yuan Christian University, Jhongli 320, Taiwan

Correspondence should be addressed to Chih-Hao Lin; linch@cycu.edu.tw

Received 9 May 2013; Accepted 22 July 2013

Academic Editor: Anyi Chen

Many real-world problems can be formulated as numerical optimization with certain objective functions. However, these objective functions often contain numerous local optima, which could trap an algorithm from moving toward the desired global solution. To improve the search efficiency of traditional genetic algorithms, this paper presents a mutual-evaluation genetic algorithm (MEGA). A novel mutual-evaluation approach is employed so that the merit of selected genes in a chromosome can be determined by comparing the fitness changes before and after interchanging with those in the mating chromosome. According to the determined genome merit, a therapy crossover can generate effective schemata to explore the solution space efficiently. The computational experiments for twelve numerical problems show that the MEGA can find near optimal solutions in all test benchmarks and achieve solutions with higher accuracy than those obtained by eight existing algorithms. This study also uses the MEGA to find optimal flow-allocation strategies for multipath-routing problems. Experiments on quality-of-service routing scenarios show that the MEGA can deal with these constrained routing problems effectively and efficiently. Therefore, the MEGA not only can reduce the effort of function analysis but also can deal with a wide spectrum of real-world problems.

1. Introduction

Many engineering optimization issues can be formulated as global optimization problems with numerical functions. When solving a complex problem, the particular challenge is that algorithms may be trapped in local optima and fail in finding global optima. Recently, genetic algorithms (GAs) have received considerable attention for solving complex and unstructured problems [1, 2]. However, traditional genetic algorithm (TGA) often suffers from the drawbacks of premature convergence and weak exploitation capabilities [3].

To overcome the deficiencies of the TGA, this paper proposes a mutual-evaluation approach to incorporate with the TGA as a mutual-evaluation genetic algorithm (MEGA). The proposed therapy crossover can implicitly generate effective schemata to efficiently exploit the given search space without explicitly analyzing the solution space. The performance of the proposed MEGA is experimented on 12 well-known numerical functions and compared with four well-known evolutionary algorithms (EAs) and four existing modified GAs. The experimental results show that the proposed MEGA is able to increase the accuracy by several orders of magnitude in almost all the cases. That is, MEGA can effectively approach the global optimum without being trapped in many local optima.

Because of the simplification property of the mutual-evaluation approach, the MEGA is suitable to deal with a wide spectrum of real world problems. In this paper, the proposed MEGA is also realized to deal with multipath routing problems in a multicommodity network, where more than one routes will be connected for each origin-destination (OD) pair. The goal is to find an optimal flow-allocation strategy to minimize the total transmission cost and satisfy all quality-of-service (QoS) requirements at the same time. The performance of the proposed MEGA is experimented by optimizing several multipath routing problems. The experiment results show that the MEGA outperforms the TGA dramatically. Furthermore, the search ability of the MEGA is robust in obtaining consistent results.

The rest of the paper is organized as follows. Section 2 describes the proposed mutual-evaluation approach. The main operations of the MEGA are introduced in Section 3.

In Section 4, the experimental studies on 12 numerical optimization problems are described. The results are compared with eight existing optimization algorithms. In Section 5, the MEGA is realized to deal with a real-world problem: multipath routing problem. A novel representation of chromosomes for routing problems is also proposed. Finally, conclusions and contributions are offered in Section 6.

2. Mutual Evaluation Approach

To overcome the deficiencies of the TGA, modified GAs should keep evolutional population as diverse as possible to improve algorithms' exploration capability for discovering new solution area. In regard to evaluation approaches, existing GAs can be classified into three categories: (1) chromosome-oriented, (2) gene-oriented, and (3) schema-oriented.

2.1. Chromosome-Oriented Approaches. Chromosome-oriented GAs concentrate on the chromosome fitness in the evolutionary process. The TGA adopts chromosome fitness to evaluate the quality of whole individual chromosome [4, 5]. A number of researches recently focus on incorporating mathematical methods with the chromosome-oriented GAs to alleviate the deficiency of premature convergence [4]. For example, Yao et al. proposed fast evolutionary programming (FEP) with Cauchy mutation to solve the premature convergence deficiency [6]. To improve slow finishing deficiency, Tu and Lu developed a stochastic genetic algorithm (StGA) with mathematical methods, which not only can find global optima, but also can reduce the computational effort [7]. By merging niche techniques and Nelder-Mead's simplex method, Wei and Zhao proposed the niche hybrid genetic algorithm (NHGA) to alleviate premature convergence and weak exploitation deficiencies of TGA [3].

2.2. Gene-Oriented Approaches. In gene-oriented GAs, the compact genetic algorithm (cGA) is a common representative [8]. The cGA represents the population as a probability vector over the set of individuals to mimic the order-one behavior of TGA. The cGA manipulates the gene distribution and essentially evolves each gene individually [8]. To enhance cGA's performance, Ahn and Ramakrishna proposed a strong elitism version of cGA [9] and then Rimeharoen et al. introduced a moving average technique to update the probability vector [10]. Although the cGA reduces memory requirement and offers many advantages, its limitation is the assumption of the independency between individual genes.

2.3. Schema-Oriented Approaches. The schema-oriented GAs explore the exact schemata by borrowing from the schema theorem proposed by Holland [11]. A schema is a pattern within a chromosome defined by fixing the values of specific chromosome loci. The increase of effective schemata enables the efficient search within a solution space and guides the evolution of the population in approaching the global optimal solution [12]. Yen and Shyu proposed a statistical gene evaluation method that uses simple statistical quantities to investigate the individual gene influence which suggests better choices for a gene evolution [13]. Kubota et al. proposed the virus-evolutionary genetic algorithm (VEGA) [14] that simulates coevolution of a virus population and a host population. VEGA applies horizontal propagation and vertical inheritance in a population with virus infection operators and genetic operators [14]. In this paper, the proposed MEGA is a new schema-oriented GA in which an innovative mutual-evaluation approach is used to achieve highly efficient evolution with necessary robustness. The MEGA does not only evaluate a chromosome by its fitness but also analyze genome's merit to improve the population's quality.

2.4. The Proposed Mutual-Evaluation Approach. According to the biological concept of the genetic engineering, a gene splicing is a process that manipulates genes outside the traditional random reproductive process. The proposed MEGA integrates a mutual-evaluation approach in a novel therapy crossover operation to produce offspring by introducing isolation, manipulation, and reintroduction of gene splicing techniques to improve the chromosome's fitness. The algorithm randomly selects two parents from a mating pool of generation t. Let the parent with superior fitness be named as the good parent $\vec{x}_{\text{good}}(t)$ and the other one is the bad parent $\vec{x}_{\text{bad}}(t)$. The MEGA generates a therapy mask to indicate which gene loci in a chromosome are chosen for crossover points. For each bit in a chromosome, we uniformly generate a random number in interval $[0, 1]$ and compare the number with a predefined therapy rate p. If the random number is less than p, its mask bit of the corresponding locus is set to value 1, which means that this gene locus belongs to the therapy genome. Otherwise, the mask bit is 0 that means the gene in this locus will not change during crossover operation.

Step 1. According to Darwin's evolution theory, two crossover parents are combined to produce new offspring in the hope that the fitness of next generation may improve gradually. In this paper, the therapy crossover wants to preserve parents' advantage and enhance population's diversity at the same time. Thus, offspring inherits the majority of parents' properties from the good parent than the bad one. Each mating parent has a different therapy rate according to its fitness; for example, good parent $\vec{x}_{\text{good}}(t)$ has a lower therapy rate (e.g., $p_{\text{good}} = 0.45$ in this paper) than that of the bad one $\vec{x}_{\text{bad}}(t)$ (e.g., $p_{\text{bad}} = 0.9$). On average, 45% of genes in the good parent will be merged with those genes in the bad parent.

Example 1. A minimization function $f(x) = \vec{x}^T \times \vec{x}$ is adopted here to illustrate the rationale of the therapy crossover. Let two parents with five genes at generation t be $\vec{x}_{\text{good}}(t) = [1.0\ \ 1.0\ \ 1.0\ \ 1.0\ \ 1.0]^T$ and $\vec{x}_{\text{bad}}(t) = [2.0\ \ 2.0\ \ 2.0\ \ 2.0\ \ 2.0]^T$, where $\vec{x}_{\text{good}}(t)$ is the good parent with better fitness $f(\vec{x}_{\text{good}}(t)) = 5$ and $\vec{x}_{\text{bad}}(t)$ is the bad one with worst fitness $f(\vec{x}_{\text{bad}}(t)) = 20$. Two therapy masks for $\vec{x}_{\text{good}}(t)$ and $\vec{x}_{\text{bad}}(t)$ are randomly generated by comparing five random numbers with therapy rates $p_{\text{good}} = 0.45$ and $p_{\text{bad}} = 0.9$, respectively. Without loss of generality, we assume that these two masks are $\vec{m}_{\text{good}} = [0\ \ 1\ \ 0\ \ 1\ \ 0]^T$ and $\vec{m}_{\text{bad}} = [1\ \ 1\ \ 1\ \ 1\ \ 0]^T$. That is, the second and fourth genes of $\vec{x}_{\text{good}}(t)$ should be merged, and most of the genes in $\vec{x}_{\text{bad}}(t)$ should be merged except for the fifth one.

Step 2. For comparison purpose, two auxiliary chromosomes $\vec{s}_{\text{good}}$ and $\vec{s}_{\text{bad}}$ are generated for $\vec{x}_{\text{good}}(t)$ and $\vec{x}_{\text{bad}}(t)$, respectively. The auxiliary chromosome clones genes from the corresponding parent and replaces the selected therapy loci with those genes in the other parent. Thus, in (1), $\vec{s}_{\text{good}}$ copies all genes from $\vec{x}_{\text{good}}(t)$ and then replaces the genes (its locus marked by $\vec{m}_{\text{good}}$) by those genes in $\vec{x}_{\text{bad}}(t)$. On the other hand, (2) describes the value of each gene in $\vec{s}_b$ according to its therapy mask $\vec{m}_{\text{bad}}$:

$$\vec{s}_{\text{good}} = \left[s_{gi}\right], \quad \text{where } s_{gi} = x_{gi} \times \left(\neg m_{gi}\right) + x_{bi} \times m_{gi}, \quad (1)$$

$$\vec{s}_{\text{bad}} = \left[s_{bi}\right], \quad \text{where } s_{bi} = x_{bi} \times \left(\neg m_{bi}\right) + x_{gi} \times m_{bi}. \quad (2)$$

Notation ($\neg$) denotes logic negation (usually expressed by "NOT") which operates on one Boolean value and returns its complement as result.

Example 2. Thus, (1) guides $\vec{s}_{\text{good}}$ to copy the majority of genes (i.e., 1st, 3rd, and 5th) from $\vec{x}_{\text{good}}(t)$ and the minority of genes (i.e., 2nd and 4th) from $\vec{x}_{\text{bad}}(t)$, that is, $\vec{s}_{\text{good}} = [1.0\ \ 2.0\ \ 1.0\ \ 2.0\ \ 1.0]^T$, with respect to $\vec{m}_{\text{good}} = [0\ \ 1\ \ 0\ \ 1\ \ 0]^T$. For $\vec{x}_{\text{bad}}(t)$ in this example, $\vec{s}_{\text{bad}}$ also can be produced as $\vec{s}_{\text{bad}} = [1.0\ \ 1.0\ \ 1.0\ \ 1.0\ \ 2.0]^T$ by (2).

Step 3. Because it is difficult to determine the merit of a set of genes in a chromosome, a simple but effective method proposed in this paper is the mutual-evaluation approach, which measures the merit of two selected genomes by comparing the fitness changes before and after interchanging the genome with the other mating chromosome. Comparing the fitness change before and after the genome replacement (i.e., $f(\vec{x}_{\text{good}}(t))$ versus $f(\vec{s}_{\text{good}})$ and $f(\vec{x}_{\text{bad}}(t))$ versus $f(\vec{s}_{\text{bad}})$) can realize the substitution effect and can be used to represent the relative merit of these genomes.

Example 3. Because the fitness of $\vec{s}_{\text{good}}$ (i.e., $f(\vec{s}_{\text{good}}) = f([1.0\ \ \mathbf{2.0}\ \ 1.0\ \ \mathbf{2.0}\ \ 1.0]^T) = 11$) is worse than that of $\vec{x}_{\text{good}}(t)$ (i.e., $f(\vec{x}_{\text{good}}(t)) = f([1.0\ \ \mathbf{1.0}\ \ 1.0\ \ \mathbf{1.0}\ \ 1.0]^T) = 5$), we can intuitively deduce that genome $[^*\ \ 1.0\ \ ^*\ \ 1.0\ \ ^*]^T$ may perform better than genome $[^*\ \ 2.0\ \ ^*\ \ 2.0\ \ ^*]^T$ with respect to $[1.0\ \ ^*\ \ 1.0\ \ ^*\ \ 1.0]^T$. Therefore, the offspring $\vec{x}_{\text{good}}(t+1)$ inherits more genetic material from $[^*\ \ 1.0\ \ ^*\ \ 1.0\ \ ^*]^T$ than $[^*\ \ 2.0\ \ ^*\ \ 2.0\ \ ^*]^T$ for the second and forth genes. Comparing the fitness of $\vec{s}_{\text{bad}}$ (i.e., $f(\vec{s}_{\text{bad}}) = f([\mathbf{1.0}\ \ \mathbf{1.0}\ \ \mathbf{1.0}\ \ \mathbf{1.0}\ \ 2.0]^T) = 8$) and that of $\vec{x}_{\text{bad}}(t)$ (i.e., $f(\vec{x}_{\text{bad}}(t)) = f([\mathbf{2.0}\ \ \mathbf{2.0}\ \ \mathbf{2.0}\ \ \mathbf{2.0}\ \ 2.0]^T) = 20$), one can deduce that genome $[1.0\ \ 1.0\ \ 1.0\ \ 1.0\ \ ^*]^T$ performs better than $[2.0\ \ 2.0\ \ 2.0\ \ 2.0\ \ ^*]^T$ with respect to $[^*\ \ ^*\ \ ^*\ \ ^*\ \ 2.0]^T$.

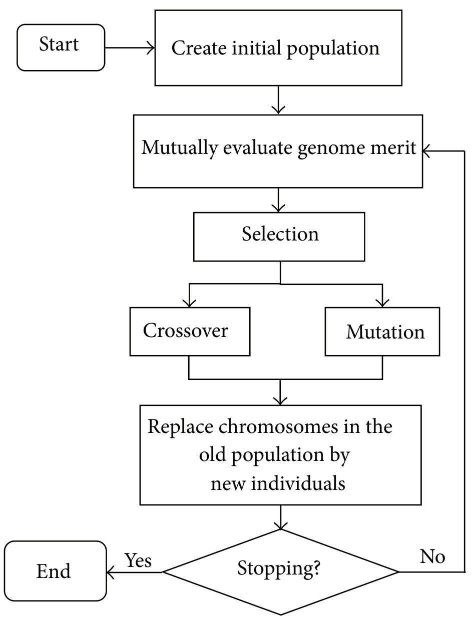

Start
Create initial population
Mutually evaluate genome merit
Selection
Crossover
Mutation
Replace chromosomes in the old population by new individuals
Stopping?
Yes
No
End

Figure 1: Flowchart of the proposed MEGA.

3. Mutual-Evaluation Genetic Algorithm (MEGA)

The main operations of the MEGA are initialization, mutual evaluation, selection, crossover, mutation, and replacement. We depict the flowchart of the MEGA in Figure 1 and describe their functionality in this section.

3.1. Encoding and Initialization. For illustration, the following minimization problem with fixed boundaries is considered:

$$\begin{aligned} &\text{Minimize} \quad f(\vec{x}) \\ &\text{subject to} \quad \vec{l} \le \vec{x} \le \vec{u}. \end{aligned} \quad (3)$$

Notation $\vec{x} = [x_1\ \ x_2\ \ \cdots\ \ x_N]^T \in \mathfrak{R}^N$ is the variable vector and $f(\vec{x})$ denotes the objective function. Because the lower bound $\vec{l} = [l_1\ \ l_2\ \ \cdots\ \ l_N]^T$ and the upper bound $\vec{u} = [u_1\ \ u_2\ \ \cdots\ \ u_N]^T$ define the feasible solution space, the domain of each x_i is denoted as interval $[l_i, u_i]$.

For numerical problems, each decision variable is treated as a gene and encoded by a floating-point number. Each chromosome representing a feasible solution is encoded as a vector of genes $\vec{x} = [x_1\ \ x_2\ \ \cdots\ \ x_N]^T$, where x_i denotes the value of the ith gene and N is total number of variables in an optimization problem. An initial population of M chromosomes is randomly generated within the feasible solution space $[\vec{l}, \vec{u}]$.

3.2. Selection Operation. Fitness of each chromosome represents the objective function value of this solution, denoted as $f_j = f(\vec{x}_j) = f([x_{j1}\ \ x_{j2}\ \ \cdots\ \ x_{jN}]^T)$ for the jth chromosome. The MEGA employs a traditional roulette selection method as a discriminator of the solution quality that uses chromosome's fitness to create a selective pressure towards global optimal solution. Those chromosomes with higher fitness should have a greater selection chance, thus creating a selective pressure towards high-fitness solutions [1].

3.2.1. Therapy Crossover. The proposed MEGA incorporates the mutual-evaluation approach with a therapy crossover to enhance the exploitation ability and speed up the convergence rate. According to the determined relative merit of genome, the parents linearly combine their genomes to generate a new genome for their offspring like (4) for $\vec{x}_{\text{good}}(t+1)$ and (5) for $\vec{x}_{\text{bad}}(t+1)$, respectively. If a mask bit in its therapy mask is 0 (e.g., $m_{g1}=0$), the gene at the locus of the good parent does not change (i.e., $x_{g1}=s_{g1}$). Otherwise, if a mask bit is 1 (e.g., $m_{g2}=1$), the locus in the crossover child (e.g., $\vec{x}_{\text{good}}(t+1)$) inherits from both genetic materials from two parents (i.e., $x_{g1}\times\text{coef}+s_{g1}\times(1-\text{coef})$). The principle of this linear combination is to retain the favorable schemata in the evolution process. Therefore, each gene of crossover child can be reproduced by the following equations:

$$\vec{x}_{\text{good}}(t+1) = \begin{cases} \vec{x}_{\text{good}}(t)\times\text{coef}+\vec{s}_{\text{good}}\times(1-\text{coef}), & \text{if } f\left(\vec{x}_{\text{good}}\right)\le f\left(\vec{s}_{\text{good}}\right) \\ \vec{x}_{\text{good}}(t)\times(1-\text{coef})+\vec{s}_{\text{good}}\times\text{coef}, & \text{if } f\left(\vec{x}_{\text{good}}\right)> f\left(\vec{s}_{\text{good}}\right), \end{cases} \quad (4)$$

$$\vec{x}_{\text{bad}}(t+1) = \begin{cases} \vec{x}_{\text{bad}}(t)\times\text{coef}+\vec{s}_{\text{bad}}\times(1-\text{coef}), & \text{if } f\left(\vec{x}_{\text{bad}}\right)\le f\left(\vec{s}_{\text{bad}}\right) \\ \vec{x}_{\text{bad}}(t)\times(1-\text{coef})+\vec{s}_{\text{bad}}\times\text{coef}, & \text{if } f\left(\vec{x}_{\text{bad}}\right)> f\left(\vec{s}_{\text{bad}}\right). \end{cases} \quad (5)$$

Coefficient $\text{coef}=0.8+(\text{rand}/2)$ is a random value in interval $[0.8, 1.3]$ to reduce the deficiency of premature convergence. The rand function returns a uniformly distributed pseudorandom number between 0 and 1.

Example 4. In this example, we assume the random coefficient $\text{coef}=1.2$. Because $f(\vec{x}_{\text{good}}(t))=5$ is better than $f(\vec{s}_{\text{good}})=11$, (4) guides us to get $\vec{x}_{\text{good}}(t+1)=\vec{x}_{\text{good}}(t)\times 1.2+\vec{s}_{\text{good}}\times(1-1.2)=\begin{bmatrix}1 & 0.8 & 1 & 0.8 & 1\end{bmatrix}^T$. Thus, this child's fitness $f(\vec{x}_{\text{good}}(t+1))=4.28$ is better than $f(\vec{x}_{\text{good}}(t))=5$. Similarly, (5) calculates $\vec{x}_{\text{bad}}(t+1)=\vec{x}_{\text{bad}}(t)\times(1-1.2)+\vec{s}_{\text{bad}}\times 1.2=\begin{bmatrix}0.8 & 0.8 & 0.8 & 0.8 & 2\end{bmatrix}^T$ because fitness $f(\vec{x}_{\text{bad}}(t))=20$ is worse than $f(\vec{s}_{\text{bad}})=8$. The child's fitness $f(\vec{x}_{\text{bad}}(t+1))=6.56$ is better than $f(\vec{x}_{\text{bad}}(t))=20$. Therefore, both children have better fitness values than that of their parents.

The exclusive features of the therapy crossover include that (1) the merit of each genome is evaluated individually; and (2) the gene merit facilitates the MEGA to perform an efficient search by adaptively shifting emphasis on significant genome without explicit functional analysis [15]. That is, the therapy crossover can avoid frequently throwing away potential schemata in inferior chromosomes and inherit the genetic advantages of superior chromosomes without loss of genetic diversity.

3.2.2. Partial-Gaussian Mutation. In this paper, a partial-Gaussian mutation used in the MEGA can increase population diversity to enhance its exploration ability. The original Gaussian mutation proposed by Hinterding in 1995 can converge to near-optimal solutions of some multimodal optimization problems [16]. Our partial-Gaussian mutation concentrates on exploiting potential optimal areas and speeds up the convergent effect. At the beginning of evolution, a high mutation rate is assigned to sample the search space extensively. And then, the mutation rate decreases with time to fine tune solutions and concentrates on exploiting potential optimal areas. Equation (6) calculates the mutation rate p_m as

$$p_m = 0.5\times\left(1-\frac{\text{Current Generation}}{\text{Maximal Generation}}\right), \quad (6)$$

where maximal generation is 3000. A random number in interval $[0, 1]$ is generated for each gene and then compared with the mutation rate p_m. If the mutation rate is greater than or equal to the random number, this gene value will be flipped by adding a unit Gaussian distributed random value to the chosen gene. Otherwise, no mutation occurs at this gene. This mutation operation can only be used for integer and float genes.

3.3. Reproduction Operation. The MEGA adopts a replacement-with-elitism method to monotonously enhance the solution quality. A number of the elite parents can survive into next generation for preventing good solutions from being lost through a nondeterministic selection operation. Successive population consists of three evolutionary sources: (1) the elite 10% chromosomes can survive to next generation; (2) 80% offsprings are produced by the crossover operation; and, (3) the rest 10% chromosomes are produced by the mutation operation.

4. Performance Analyses for Numerical Optimization Problems

4.1. Numerical Test Functions. Numerical experiments are conducted to demonstrate the robustness and reliability of the proposed MEGA. This work selects 12 well-known benchmark functions, which cover broad range functionality characteristics with two categories: unimodal functions (Functions f_1–f_6) and multimodal functions (Function f_7–f_{12}). Table 1 depicts these test functions with their formulation, problem dimension (N), prescribed search domain (D), and their global optimum function value ($f_{\min}$) in each column. The high-dimension unimodal functions are the Sphere function (f_1), the Schwefel's Problem 2.22 (f_2), the Schwefel's Problem 1.2 (f_3), the Schwefel's Problem 2.21 (f_4), a modified Rosenbrock function (f_5), and the noisy quadratic function (f_6). Unimodal functions are relatively easy to solve but the difficulty increases as the problem dimension goes high. The high-dimension multimodal functions are a generalized Schwefel's function 7 (f_7), the Rastrigin's function 6 (f_8),

Table 1: Twelve benchmark functions.

Benchmark functions	N	D	$f_{\min}$
$f_1(x) = \sum_{i=1}^{N} x_i^2$	30	$[-100, 100]$	0
$f_2(x) = \sum_{i=1}^{N} \lvert x_i \rvert + \prod_{i=1}^{N} \lvert x_i \rvert$	30	$[-10, 10]$	0
$f_3(x) = \sum_{i=1}^{N} \left(\sum_{j=1}^{i} x_j\right)^2$	30	$[-100, 100]$	0
$f_4(x) = \max\{\lvert x_i \rvert, i = 1, 2, \ldots, N\}$	30	$[-100, 100]$	0
$f_5(x) = \sum_{i=1}^{N-1} \left[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right]$	30	$[-5, 10]$	0
$f_6(x) = \sum_{i=1}^{N} x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]$	0
$f_7(x) = \sum_{i=1}^{N} \left(-x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)\right)$	30	$[-500, 500]$	-12596.5
$f_8(x) = \sum_{i=1}^{N} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	0
$f_9(x) = -20\exp\left(-0.2\sqrt{1/n\sum_{i=1}^{n} x_i^2}\right) - \exp\left((1/N)\sum_{i=1}^{N}\cos(2\pi x_i)\right) + 20 + \exp(1)$	30	$[-32, 32]$	0
$f_{10}(x) = (1/4000)\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N}\cos(x_i/\sqrt{i}) + 1$	30	$[-600, 600]$	0
$f_{11}(x) = \frac{\pi}{N}\left\{10\sin^2(\pi y_1) + (y_N - 1)^2 + \sum_{i=1}^{N-1}(y_i - 1)^2 \times [1 + 10\sin^2(\pi y_{i+1})]\right\} + \sum_{i=1}^{N} u(x_i, 10, 100, 4)$ where $y_i = 1 + (1/4)(x_i + 1)$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$f_{12} = (1/10)\left\{\sin^2(3\pi x_1) + (x_N - 1)^2[1 + \sin^2(2\pi x_N)] + \sum_{i=1}^{N-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})]\right\} + \sum_{i=1}^{N} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0

N: problem dimension; D: prescribed search domain; $f_{\min}$: global optimum value.

Table 2: Experimental results obtained by the MEGA over 50 independent trails.

Test function	$f_{\min}$	Mean best value	Standard deviation	Mean required generations	Computational effort (MNFE)
f_1	0	2.44×10^{-19}	1.33×10^{-18}	64	17,280
f_2	0	2.00×10^{-7}	4.27×10^{-7}	725	195,750
f_3	0	5.49×10^{-30}	2.74×10^{-29}	650	175,500
f_4	0	1.99×10^{-8}	2.42×10^{-8}	185	49,950
f_5	0	0.03762	0.02816	88	23,760
f_6	0	3.78×10^{-3}	1.84×10^{-3}	96	25,920
f_7	−12596.5	−12596.5	9.98×10^{-3}	132	35,640
f_8	0	0.0	0.0	155	41,850
f_9	0	2.78×10^{-8}	5.15×10^{-8}	400	108,000
f_{10}	0	0.0	0.0	23	6,210
f_{11}	0	7.85×10^{-7}	4.42×10^{-7}	215	58,050
f_{12}	0	1.11×10^{-5}	8.31×10^{-6}	70	18,900

Table 3: Comparison of the mean best value (and Std Dev) with other EAs.

Test function	PSO (Variance)	EO (Variance)	CEP	FEP	MEGA
f_1	11.175 (1.3208)	9.8808 (0.9444)	2.2×10^{-4} (5.9×10^{-4})	5.7×10^{-4} (1.3×10^{-4})	2.44×10^{-19} (1.33×10^{-18})
f_2	N/A	N/A	2.6×10^{-3} (1.710^{-4})	8.1×10^{-3} (7.7×10^{-4})	2.00×10^{-7} (4.27×10^{-7})
f_3	N/A	N/A	0.05 (0.066)	0.016 (0.014)	5.49×10^{-30} (2.74×10^{-29})
f_4	N/A	N/A	2.0 (1.2)	0.3 (0.5)	1.99×10^{-8} (2.42×10^{-8})
f_5	1911.598 (374.2935)	1610.39 (293.5783)	6.17 (13.61)	5.06 (5.87)	0.03762 (0.02816)
f_6	N/A	N/A	0.018 (6.4×10^{-3})	7.6×10^{-3} (2.6×10^{-3})	3.78×10^{-3} (1.84×10^{-3})
f_7	N/A	N/A	−7917.1 (634.5)	−12554.5 (52.6)	−12596.5 (9.98×10^{-3})
f_8	47.1354 (1.8782)	46.4689 (2.4545)	89.0 (23.1)	0.046 (0.012)	0.0 (0.0)
f_9	N/A	N/A	9.2 (2.8)	0.018 (0.0021)	2.78×10^{-8} (5.15×10^{-8})
f_{10}	0.4498 (0.0566)	0.4033 (0.0436)	2.52×10^{-7}	0.016 (0.022)	0.0 (0.0)
f_{11}	N/A	N/A	1.76 (2.4)	9.2×10^{-6} (3.6×10^{-6})	7.85×10^{-7} (4.42×10^{-7})
f_{12}	N/A	N/A	1.4 (3.7)	1.6×10^{-4} (7.3×10^{-5})	1.11×10^{-5} (8.31×10^{-6})

a modified Ackley's Path Function 10 (f_9), the Griewank's function 8 (f_{10}), a generalized Penalized Function 1 (f_{11}), and a generalized Penalized Function 2 (f_{12}). Multimodal functions represent the most difficult class of problems, which possess many local optima and could trap an algorithm into one of its local optimal solutions.

4.2. Algorithm Implementation and Parameter Settings. In all cases, the population size is 150, in which the number of elite individuals is 15; the therapy crossover produces 120 individuals, and the mutation produces 15 ones. The therapy rates for good parent and bad parent are $p_{\text{good}} = 0.45$ and $p_{\text{bad}} = (1 - p_{\text{good}}) = 0.55$, respectively. For each test function, 50 independent trials with different seeds are performed using the MATLAB environment.

For complexity analysis, the mean number of function evaluations serves as a measure of required computational effort for an algorithm. Different from the crossover in traditional GAs, the exclusive feature of MEGA is the usage of the mutual-evaluation approach for genome therapy. Because

Table 4: Comparison of the computational effort (MNFE) with other EAs.

Test function	PSO	EO	CEP	FEP	MEGA
f_1	250,000	500,000	100,500	100,500	17,280
f_2	N/A	N/A	200,000	200,000	195,750
f_3	N/A	N/A	500,000	500,000	175,500
f_4	N/A	N/A	500,000	500,000	49,950
f_5	250,000	500,000	2000,000	2000,000	23,760
f_6	N/A	N/A	300,000	300,000	25,920
f_7	N/A	N/A	900,000	900,000	35,640
f_8	250,000	500,000	500,000	500,000	41,850
f_9	N/A	N/A	150,000	150,000	108,000
f_{10}	250,000	500,000	200,000	200,000	6,210
f_{11}	N/A	N/A	150,000	150,000	58,050
f_{12}	N/A	N/A	150,000	150,000	18,900

only an auxiliary chromosome should be extraevaluated for one crossover child, the number of required evaluations in each generation is (function evaluations per generation) = [(population size) × (1 + crossover fraction)] = [150 × (1 + 80%)] = 270. Therefore, the total number of function evaluations in each experimental run is equal to (functional evaluations per generation) × (no. of terminal generations).

4.3. Experimental Results. Table 2 summarizes the experimental results of the MEGA in 50 trials, which include (1) the global optimum ($f_{\min}$), (2) the mean best function value, (3) the standard deviation of the obtained function values, (4) the mean required generation, and (5) the mean number of function evaluations (MNFEs).

The first thing we can observe from Table 2 is that the obtained results are equal or really close to the "known" optimal solutions. Particulary, this paper uses a computational precision of 60 digits after point. Thus, the results "0" on f_8 and f_{10} in Table 2 mean that they are less than 10^{-60}. The standard deviation with respect to the functions f_8 and f_{10} is equal to zero; that is, the results of all 50 runs reach the optimum. All the obtained results approach the "known" optimal values with small differences. Secondly, the small standard deviations for all test functions also indicate that the MEGA consistently converges to the near-optimal solutions in all 50 trails. Finally, all of the mean numbers of function evaluations are relatively small. Therefore, the proposed MEGA can address a variety of numerical optimization functions effectively. To further analyze the solution capability of the proposed MEGA, the following sections describe the comparisons between the MEGA and two groups of optimization algorithms for the 12 benchmark functions.

4.4. Comparison with Other Evolutionary Algorithms. The performance of the MEGA is compared with four state-of-the-art EAs: particle swarm optimization (PSO) [17], evolutionary optimization (EO) [17], conventional evolutionary programming (CEP) [6], and FEP [6]. The comparison of the experimental results for 12 test functions is shown in Table 3.

For all the unimodal functions (f_1–f_6) in Table 3, we can observe that the MEGA can achieve dramatically the highest accuracy than others, while other four algorithms experience premature convergence on functions f_3, f_4, and f_5. For all the multimodal functions (f_7–f_{10}), the results shown in Table 3 clearly indicate that the MEGA can identify the actual optima of these functions with the highest accuracy. The MEGA can achieve better solution accuracy than other four EAs for all 12 benchmark functions.

The computational efforts required for the algorithms are measured by their mean numbers of function evaluations and depicted in Table 4. Obviously, the comparison demonstrates that the MEGA outperformed all the four algorithms for all the 12 functions with respect to the convergent ability. Therefore, the comparison indicates that the MEGA is both efficient and effective in solving the unimodal and multimodal benchmark functions.

4.5. Comparison with Other Genetic Algorithms. The performance of the MEGA is compared with those of four well-known GAs: cluster-based adaptive mutation genetic algorithm (CMGA) [18], orthogonal genetic algorithm with quantization (OGA/Q) [19], hybrid taguchi genetic algorithm (HTGA) [20], and StGA [7]. The OGA/Q is a quantized-version of the OGA, which incorporates with the Taguchi method to minimize the effect of chromosome variation without eliminating the population diversity [21]. The HTGA enhanced the Taguchi method as a new operation to adapt a dynamically extended precision method from a low-precision solution space to a high-precision one [20]. The above algorithms have been executed to solve the test functions and the results were reported in the literature. We will use these existing results for a direct comparison in Tables 5 and 6.

As the termination criteria used in these four algorithms are different, to make a fair comparison basis, we let the solution qualities obtained by our MEGA be slightly better than those of the four algorithms (in Table 5), and then, compared the mean computational effort at the given accuracy (in Table 6). For the unimodal functions (f_1–f_6), the convergence rate of an algorithm is a more important issue

Table 5: Comparison of the mean best value (and Std Dev) with other GAs.

Test function	CMGA	OGA/Q	HTGA	StGA (Variance)	MEGA
f_1	N/A	0 (0)	0 (0)	2.45×10^{-15} (5.25×10^{-16})	2.44×10^{-19} (1.33×10^{-18})
f_2	N/A	0 (0)	0 (0)	2.0×10^{-7} (2.95×10^{-8})	2.00×10^{-7} (4.27×10^{-7})
f_3	N/A	0 (0)	0 (0)	9.98×10^{-29} (6.90×10^{-29})	5.49×10^{-30} (2.74×10^{-29})
f_4	N/A	0 (0)	0 (0)	2.01×10^{-8} (3.42×10^{-9})	1.99×10^{-8} (2.42×10^{-8})
f_5	N/A	0.7520 (0.1140)	0.7 (0)	0.04435 (0.0)	0.03762 (0.02816)
f_6	N/A	6.301×10^{-3} (4.069×10^{-4})	1.000×10^{-3} (0)	8.4×10^{-4} (1.00×10^{-3})	3.78×10^{-3} (1.84×10^{-3})
f_7	−8722.16	−12569.4537 (6.447×10^{-4})	−12569.4600 (0)	−12569.5 (0.0)	−12596.5 (9.98×10^{-3})
f_8	157.76	0 (0)	0 (0)	4.42×10^{-13} (1.14×10^{-13})	0.0 (0.0)
f_9	N/A	4.440×10^{-16} (3.989×10^{-17})	0 (0)	3.52×10^{-8} (3.51×10^{-9})	2.78×10^{-8} (5.15×10^{-8})
f_{10}	0.3283	0 (0)	0 (0)	2.44×10^{-17} (4.54×10^{-17})	0.0 (0.0)
f_{11}	N/A	6.019×10^{-6} (1.159×10^{-6})	1.000×10^{-6} (0)	8.03×10^{-7} (1.96×10^{-14})	7.85×10^{-7} (4.42×10^{-7})
f_{12}	N/A	1.869×10^{-4} (2.615×10^{-5})	1.000×10^{-4} (0)	1.13×10^{-5} (4.62×10^{-13})	1.11×10^{-5} (8.31×10^{-6})

Table 6: Comparison of the computational effort (MNFE) with other GAs.

Test function	CMGA	OGA/Q	HTGA	StGA	MEGA
f_1	N/A	112,559	20,844	30,000	17,280
f_2	N/A	112,612	14,285	17,600	195,750
f_3	N/A	112,576	26,469	23,000	175,500
f_4	N/A	112,893	21,261	32,000	49,950
f_5	N/A	167,863	60,737	45,000	23,760
f_6	N/A	112,652	20,065	25,500	25,920
f_7	600,000	302,166	163,468	1,500	35,640
f_8	600,000	224,710	16,267	28,500	41,850
f_9	N/A	112,421	16,632	10,000	108,000
f_{10}	600,000	134,000	20,999	52,500	6,210
f_{11}	N/A	134,556	66,457	8,000	58,050
f_{12}	N/A	134,143	59,003	16,000	18,900

than the achieved satisfactory accuracy. Because of different problem dimensions used in the HTGA (i.e., $N = 100$), this study cannot compare its performance with other algorithms for these six unimodal functions. In Table 6, we can observe that the proposed MEGA requires fewer MNFEs than the OGA/Q for four of the six test functions. The convergence rate of the MEGA was similar to that of the StGA for a half of the six test functions. Because of the narrow valleys of f_2 and f_3, the MEGA was forced to change its searching direction continually; thus, it approached the high-accuracy optimum slowly. That is why the convergence rates of the MEGA were lower than those of other algorithms for f_2 and f_3.

For the multimodal functions (f_7–f_{12}), the quality of the solutions is more crucial than the required computational effort because the solution quality reveals the algorithm's ability to escape from local optima and achieves near-global solutions. From the obtained results of the numerical experiments in Tables 5 and 6, we can see that the MEGA was superior to the CMGA with respect to both solution accuracy and convergence rate. The solution accuracies achieved by the MEGA were similar to those of the OGA/Q, HTGA, and StGA for all the six multimodal functions. However, the MEGA converged slower than the HTGA and StGA for the multimodal function f_9. This finding implies that the proposed MEGA is robust and effective in solving the multimodal functions and can perform as good as the four state-of-the-art GAs.

5. Performance Evaluation for Multipath Routing Problems

5.1. Multipath Routing Problems. Multipath traffic engineering becomes more attractive for ubiquitous networks to satisfy the required QoS of mobile network applications [22]. For a given network topology with available link capacities, it is required to determine the optimal distribution of traffic requests on multipath routing subject to the constraints imposed by QoS specifications. The service network is modeled as a connected weighted, directed graph $G = (V, E)$, where the $V = \{v_1, v_2, \ldots, v_n\}$ is the vertex set of G, and the $E = \{e_1, e_2, \ldots, e_m\}$ is a finite set of edges. We consider a link-state routing environment, where each node has knowledge of its neighbor links and each source node knows (1) traffic load from node i to node j $(i \neq j)$, (2) QoS requirements/constraints (e.g., delay constraint), (3) residual bandwidth of each link, (4) link cost for the traffic, and (5) time delay for the traffic to pass through each link at a certain time period.

The link $e = (i, j) \in E$ connects a source node $i \in V$ to a destination node $j \in V$ with positive cost c_e, capacity b_e, and a delay function (i.e., $D_e : \mathfrak{R}^+ \rightarrow \mathfrak{R}^+$). The cost function represents the traffic-related delay of each edge and limit to positive cost. The delay requirement specifies the upper bound of delay tolerance, denoted as Δ_w.

In multipath routing problems, messages are routed for a set of OD pairs W, where t_w denotes the traffic demand volume (or bandwidth) for OD pair w. The set of all permissible paths for the wth OD pair is denoted as P_w. Let indicator function f_{wp} be a nonnegative continuous variable denoting the traffic flow allocated on path p of request w. Indicator function y^e_{wp} is 1 if link e belongs to path p for request w, and 0 otherwise; that is, $y_{pi} = \begin{cases} 1, & \text{if } e_i \in p \\ 0, & \text{otherwise,} \end{cases}$ where $i = 1, 2, \ldots, m$. The aggregate flow on link e is denoted as g_e, which must satisfy the capacity feasibility constraint; that is, $g_e = \sum_{w\in W} \sum_{p\in P_w} f_{wp} y^e_{wp} \le b_e$, for all $e \in E$.

The multipath routing problem is to find multiple paths to transmit the traffic demand for all OD pairs such that the total routing cost is minimal and the QoS constraints should be satisfied. The multipath routing can be modeled as a combinatorial linear programming problem in the following:

objective function,

$$\text{minimize} \quad \sum_{e\in E} c_e g_e \tag{7}$$

$$\text{subject to} \quad g_e = \sum_{w\in W} \sum_{p\in P_w} f_{wp} y^e_{wp} \quad \forall e \in E \tag{8}$$

$$g_e \le b_e \quad \forall e \in E \tag{9}$$

$$\sum_{e\in E} y^e_{wp} f_{wp} D_e (g_e) \le \Delta_w \quad \forall p \in P_w,\ w \in W \tag{10}$$

$$\sum_{p\in P_w} f_{wp} = t_w \quad \forall w \in W \tag{11}$$

$$f_{wp} \le t_w r_{wp} \quad \forall p \in P_w,\ w \in W \tag{12}$$

$$\sum_{p\in P_w} r_{wp} = x_w \quad \forall w \in W \tag{13}$$

$$r_{wp} \text{ binary} \quad \forall p \in P_w,\ w \in W. \tag{14}$$

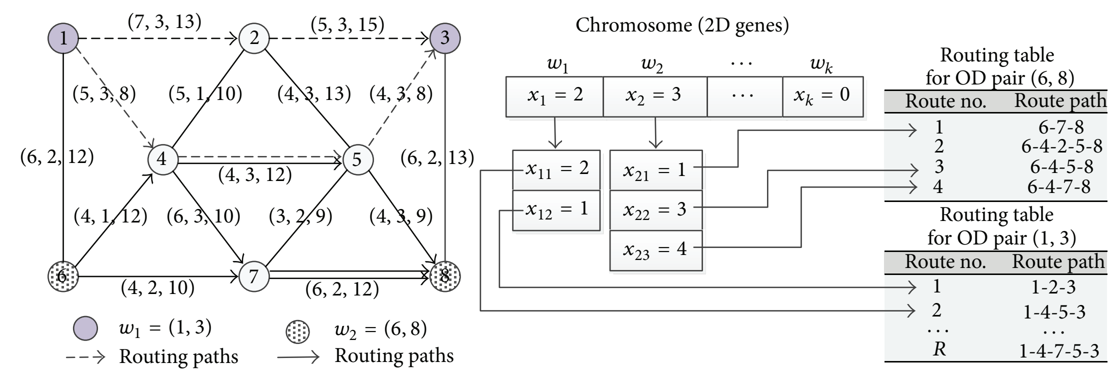
The objective function (in (7)) is to minimize the total routing cost in the multipath routing problem. The first constraint (in (8)) calculates the aggregate traffic on link e. The second constraint assures the capacity constraint on each link by (9). The third constraint (in (10)) ensures the total delay of each OD pair to satisfy a prespecified path-delay bound Δ_w. The constraint in (11) enforces that traffic demand of each OD pair should be satisfied. Equation (12) lets an auxiliary binary variable r_{wp} be 1 if the traffic f_{wp} along path p is larger than zero, and 0 otherwise. Equation (13) sums up the used routes and assigns the number to the wth gene x_w.

5.2. Algorithm Implementation for Multipath Routing Problems

5.2.1. Encoding. The encoding method is the most important steps towards solving real world problems using EAs. In this paper, the encoding method maps all multipath OD pairs into a chromosome based on route aspect. Since there are so many candidate paths between two nodes in the network graph, traditional GAs may consume considerable computational effort in searching infeasible solutions because genetic operations do not always preserve feasibility. Therefore, to reduce the search space, this work uses the K shortest path routing algorithm to precalculate the first R shortest paths and record in a routing table.

The MEGA maintains a population of chromosomes to optimize a given objective function for the multipath routing problem. Each chromosome can be represented by a two-dimensional array of integers. The first-dimension genes $\vec{x} = [x_1 \ x_2 \ \cdots \ x_N]^T$ represent the number of used routes for each OD pair. The second-dimension genes record the route number of each used path for realizing the demand of each OD pair. Thus, if the gene value of the wth gene is $x_w = k$, the vector $[x_{w1}, x_{w2}, \ldots, x_{wk}]^T$ represents k route numbers of subflows for OD pair w. Gene x_{wp} is an integer in interval $[1, R]$ to represent a route number for the OD pair w in its routing table. We use an example of network topology to illustrate the relationship between a chromosome, 2D genes, and routing tables in Figure 2.

Figure 2(a) depicts an example of a network graph, link parameters, and two multipath routings. Parameters along links are triple (cost, delay, and bandwidth). The first OD pair, that is, $w_1 = (1, 3)$, has two routing paths $p_{12} = \{(1, 4), (4, 5), (5, 3)\}$ and $p_{11} = \{(1, 2), (2, 3)\}$ to transmit traffic from node 1 to node 3. Thus, the first-dimension gene $x_1 = 2$.



Routing table for OD pair (6, 8)

Route no.	Route path
1	6-7-8
2	6-4-2-5-8
3	6-4-5-8
4	6-4-7-8

Routing table for OD pair (1, 3)

Route no.	Route path
1	1-2-3
2	1-4-5-3
...	...
R	1-4-7-5-3

(a) Network topology example with two OD pairs (b) Representation of a chromosome (c) Routing tables

Figure 2: Example of genotype coding: (a) network graph with link parameter (cost, delay, and bandwidth), (b) representation of chromosome by using 2D genes, and (c) two routing tables for paths for node 6 to node 8 and paths for node 1 to node 3.

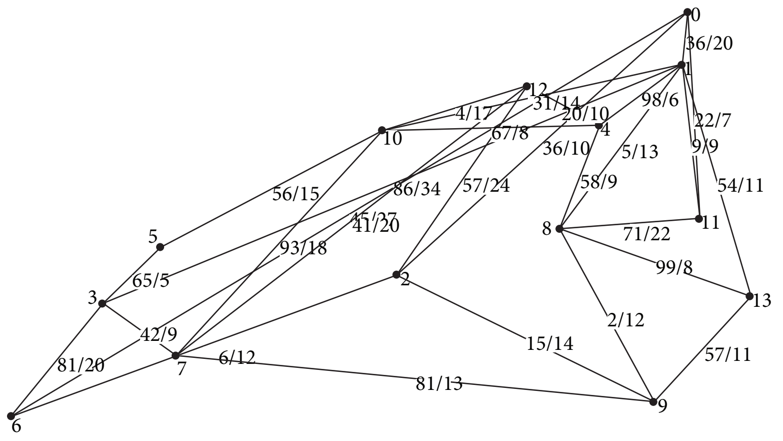


Figure 3: A randomly generated network with 14 nodes and average degree 4.

The route numbers in its routing table (in Figure 2(c)) corresponding to paths p_{12} and p_{11} are 2 and 1; thus, the second-dimension genes for w_1 (in Figure 2(b)) record $x_{11} = 2$ and $x_{12} = 1$, respectively. In Figure 2(a), the second OD pair, that is, $w_2 = (6, 8)$ routes its traffic along three subflows along paths p_{21}, p_{23}, and p_{24}. Therefore, the first-dimension gene is $x_2 = 3$ and the corresponding second-dimension genes are $x_{21} = 1$, $x_{22} = 3$, and $x_{23} = 4$, respectively.

5.2.2. Therapy Crossover for Multipath Routing. Each time the selection operation chooses two crossover parents from the population. The proposed mutual-evaluation approach calculates the merit of two selected genomes by comparing the changes in the chromosome fitness before and after interchanging the first- and second-dimension genomes with the other mating chromosome. The first-dimension genes can linearly combine with those in the other chromosome by using the proposed therapy crossover (in Section 3.2.1 (4) and (5)). The obtained results should be transformed into integer type, and therefore, the second-dimension genes should be modified. If the obtained result of the first-dimension gene is larger than before, we randomly select a suitable number of genes from the other parent to add into the second-dimension genes in this chromosome. If the result equals to the original value, we randomly select a small number of genes from the other parent to replace the original genes. Otherwise, a suitable number of genes should be randomly selected to remove from this chromosome.

5.2.3. Mutation for Multipath Routing. Mutation operation performs on an individual chromosome to flip one or more genes with a small probability (typically 0.001) and ensures that no point in the search space has a zero probability of being searched. According to a mutation probability, the mutation randomly selects a subset of genes and chooses new paths from its routing table. Thus, the route numbers of these new paths replace the original values of selected genes. The resulting chromosome is a new multipath routing plan that can increase population diversity.

5.3. Test Platform and Performance Metrics. In this paper, we use the well-known network generation tool [23] to create an asynchronous network based on the Waxman's techniques [24]. The network in Figure 3 illustrates a random generated

Table 7: The mean experimental results for the low-rate cases (0.25 Mbps) obtained by (a) MEGA and (b) TGA.

(a) MEGA

OD pairs	3	4	5	6	7	8	9	10
Case 1	116.16	160.66	193.84	234.28	267.5	298.12	334.64	368.47
Case 2	132.4	154.1	176.45	226.84	254.5	287.3	335.1	374.12
Case 3	100.84	141.1	171.5	210.56	254.67	291.6	321.64	341.6
Case 4	114.5	157.64	189.67	231.99	260.46	290.14	327.45	360.41
Case 5	125.1	170.14	203.46	248.82	176.45	308.2	343.71	386.46
Avg.	117.8	156.73	186.98	230.50	242.72	295.07	332.51	366.21

(b) TGA

OD pairs	3	4	5	6	7	8	9	10
Case 1	138.31	173.62	235.81	249.45	284.65	301.64	355.14	380.39
Case 2	154.96	184.60	193.59	236.07	282.32	321.73	363.22	392.32
Case 3	129.25	174.79	188.36	233.30	271.43	309.39	345.11	371.40
Case 4	137.98	192.39	203.39	250.22	266.99	313.95	354.95	384.30
Case 5	165.01	186.55	220.27	256.96	190.42	311.60	354.38	392.94
Avg.	145.10	182.39	208.28	245.20	259.16	311.66	354.56	384.27

Table 8: The standard deviation of results for the low-rate cases (0.25 Mbps) obtained by (a) MEGA and (b) TGA.

(a) MEGA

OD pairs	3	4	5	6	7	8	9	10
Case 1	4.28	5.12	7.01	12.80	11.79	14.31	15.26	16.82
Case 2	4.79	6.63	7.98	11.82	12.92	14.50	17.06	14.94
Case 3	3.49	6.89	6.76	10.58	13.01	14.30	15.13	16.92
Case 4	4.41	4.86	6.36	10.52	12.91	12.50	15.80	17.84
Case 5	5.27	5.60	6.23	9.62	13.47	14.18	15.71	15.24
Avg.	4.45	5.82	6.87	11.07	12.82	13.96	15.79	16.35

(b) TGA

OD pairs	3	4	5	6	7	8	9	10
Case 1	7.56	7.63	7.57	15.15	14.99	16.55	17.17	18.40
Case 2	7.04	9.34	8.35	15.36	14.23	17.49	18.59	19.31
Case 3	7.14	7.69	9.28	14.81	15.77	15.55	17.78	21.26
Case 4	8.24	9.47	9.05	14.66	15.37	16.50	18.72	18.06
Case 5	6.85	10.47	11.21	14.30	14.28	17.43	19.91	16.61
Avg.	7.37	8.92	9.09	14.86	14.93	16.70	18.43	18.73

graph with 14 nodes and the average degree of each node is four. Figure 3 only shows the cost/delay information along one direction link (from a smaller-ID node to a larger-ID one) to reduce the complexity of the graph representation. All links are assumed to have 1.5 Mbps of bandwidth capacity. For each test scenario, the OD pairs are randomly generated five times for each scenario to decrease the selection bias. Two kinds of transmission rates are assigned for the OD pairs: 0.25 Mbps (lowrate) and 0.5 Mbps (highrate).

The performance of the proposed MEGA is evaluated based on the following ways.

(1) *The total routing cost*: This cost reflects the algorithm's ability to construct multiple paths for all OD pairs by using low-cost and lightly utilized links.

(2) *The maximum end-to-end delay for OD pairs*: It indicates the algorithm's ability to satisfy the delay bound imposed by the service level agreement of applications.

An algorithm's effectiveness in allocating network resources can be judged by monitoring how frequently that algorithm fails to construct a set of acceptable OD pairs. There are two kinds of failure. One is that the created OD pair does not satisfy its delay bound. The other one is that the algorithm cannot find unsaturated links to create a path for OD pairs.

5.4. Computational Experiments for Multipath Routing Problems. The performance of the proposed MEGA is compared

Table 9: The mean experimental results for the high-rate cases (0.5 Mbps) obtained by MEGA and TGA.

Algorithm	MEGA				TGA			
OD pairs	3	4	5	6	3	4	5	6
Case 1	116.50	159.69	193.06	231.94	124.30	166.01	204.70	248.78
Case 2	110.84	142.02	180.71	209.44	120.51	149.94	200.26	223.59
Case 3	134.10	165.12	201.36	235.76	144.18	176.07	213.18	255.57
Case 4	100.94	136.34	170.46	192.10	124.78	150.85	179.52	209.35
Case 5	121.34	158.49	187.54	224.62	134.62	172.35	204.96	228.07
Avg.	116.74	152.33	186.63	218.77	129.68	163.04	200.52	233.07

Table 10: The standard deviation of results for the high-rate cases (0.5 Mbps) obtained by MEGA and TGA.

Algorithm	MEGA				TGA			
OD pairs	3	4	5	6	3	4	5	6
Case 1	4.58	5.50	7.59	10.04	5.79	6.55	8.55	11.13
Case 2	5.14	6.49	9.12	11.24	5.71	6.62	10.39	11.90
Case 3	3.72	4.98	7.12	9.56	5.21	6.47	9.12	11.14
Case 4	4.30	6.30	8.84	10.56	5.67	7.71	9.68	10.67
Case 5	5.04	6.81	9.40	10.54	6.93	8.43	10.66	12.20
Avg.	4.56	6.02	8.41	10.39	5.86	7.16	9.68	11.41

with the TGA for two kinds of multipath routing scenarios with respect to different transmission rates and different number of total OD pairs in the network. In the first test scenario, the experiments are performed for 100 independent runs with 10 generations in the MATLAB environment. The evolution curves of total routing costs with respect to lowrate and highrate are compared in Figures 4(a) and 4(b), respectively. The evolution curves in Figure 4(a) show that the efficiency of the MEGA is better than that of the TGA in both two transmission rates. Particulary, even though the transmit rate increases two times than the low-rate case, the growing ratio of the total routing cost in Figure 4(b) is still less than that obtained by the TGA.

The second scenario simulates the stress test to measure the robustness of these two algorithms. We increase the number of OD pairs from 3 to 10 and conduct 100 independent trials for each test case. The OD pairs are randomly generated five times for each scenario to simulate the great diversity of OD-pair selection for performance evaluation.

Tables 7 and 8 are the "mean" and "standard deviation" experimental results obtained by the proposed MEGA and TGA for five cases in the low-rate scenario (with 0.25 Mbps transmission rate). The mean experimental results obtained by the proposed MEGA are better than the results of the TGA on all the test cases. That is, the MEGA can find better routing paths to serve all multipath transmission requirements than TGA (in Table 7). For the standard deviations in 100 independent runs, the proposed algorithm achieved superior results compared with the TGA in all the test cases (in Table 8). This finding implies that the proposed algorithm is robust in solving multipath routing problems and can perform better than the TGA.

In the high-rate scenario, the test network cannot afford the data flows if the number of OD pairs is larger than 6. Thus, the following experiments only increase the number of OD pairs from 3 to 6 and the transmission rate is 0.5 Mbps for the high-rate cases. We simulate 100 independent trials for each test case and depict the "mean" and "standard deviation" of the experimental results obtained by the MEGA and TGA in Tables 9 and 10, respectively. Experimental results indicate that the proposed MEGA outperforms the TGA with respect to the "mean" and "standard deviation" of routing costs for all high-rate scenarios. Furthermore, the search ability of the MEGA is robust in obtaining consistent results and performs better than the TGA.

6. Conclusions and Future Works

To the best of our knowledge, the proposed MEGA is the first mutual-evaluation approach, which calculates each genome merit by interchange-compare-replace method. The genome evaluation facilitates the MEGA to perform an efficient search by dynamically shifting emphasis to significant genomes in the feasible space without abdicating any portion of the candidate schemata. The therapy crossover is also proposed to preserve better-performance schema patterns. Simpler than other modified approaches, the proposed MEGA can preserve high quality genomes during evolution period without using extra analyzing techniques.

The performance of the proposed algorithm was measured using 12 benchmark functions. The performance was compared with four existing EAs and four well-known GAs. The experiment results show that the MEGA is able to find near-optimal solutions, even though other algorithms experience difficulties in approaching the global optima on some functions. The behavior of the algorithm is also consistent as indicated by a small standard deviation among the 50 trials for each test function. Furthermore, the MEGA can increase the accuracy by several orders of magnitude than other algorithms in almost all test functions. That is, the MEGA can

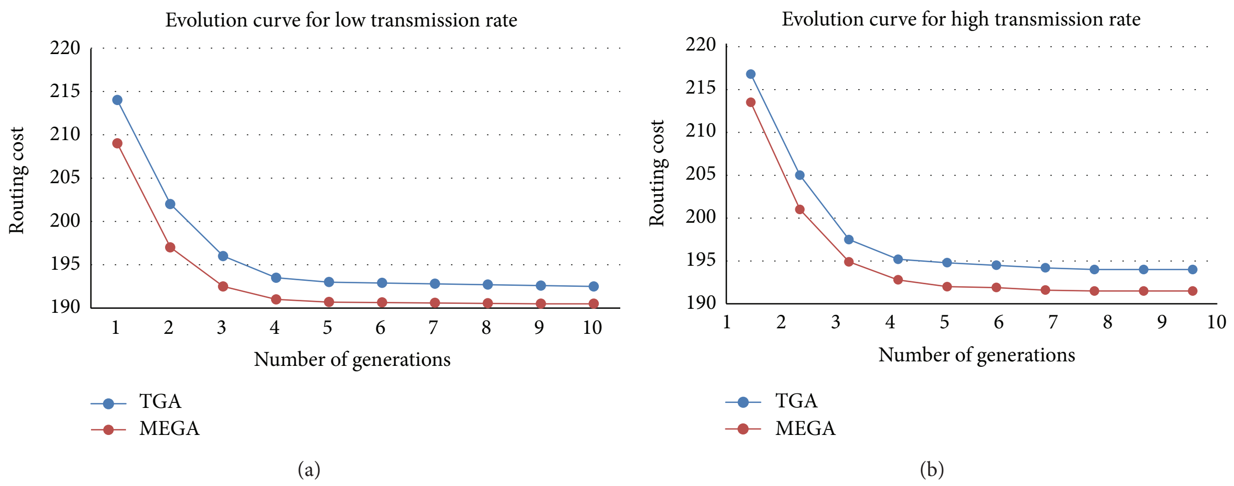


Figure 4: Evolution curves of two different transmission rates (a) 0.25 Mbps and (b) 0.5 Mbps.

outperform the existing global optimization algorithms with a dramatic improvement in terms of effectiveness.

Furthermore, because of the simplification property of the mutual-evaluation approach, this study slightly modified the encoding method of the MEGA to solve the multipath routing problems in multicommodity networks. The bandwidth-delay constraints are introduced to enforce the service quality of multimedia applications. The experimental results show that the MEGA not only can solve QoS constrained multipath routing problems, but also can outperform the TGA. That is, the proposed MEGA not only can reduce the effort of explicit function analysis but also can deal with a wide spectrum of real world problems.

The contributions of the paper are as follows. (1) We develop a novel mutual-evaluation approach which incorporates with a GA that has never been jointly considered in the literature. (2) We introduce a novel therapy crossover, which not only can evolve superior genomes but also can achieve global optima. (3) We introduce a novel chromosome representation for the MEGA to address multipath routing problems in a multicommodity network. (4) Experimental results show that the proposed MEGA can achieve significant performance gain over several well-known algorithms under the considered scenarios. The proposed MEGA has high exploration and exploitation abilities as a robust, statistically sound, and quickly convergent algorithm.

We have observed that there are many researches for routing problems. In the future, we will further compare with more state-of-the-art methods on QoS multipath routing. Particulary, the simplicity property of the MEGA can help to route dynamic services across heterogeneous environments. Therefore, developing a distributed MEGA to enhance its scalability for highly dynamic environments is also our future work.

References

[1] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin, Germany, 2nd edition, 1994.

[2] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, Mass, USA, 1996.

[3] L. Wei and M. Zhao, “A niche hybrid genetic algorithm for global optimization of continuous multimodal functions,” *Applied Mathematics and Computation*, vol. 160, no. 3, pp. 649–661, 2005.

[4] H. R. Mashhadi, H. M. Shanechi, and C. Lucas, “A new genetic algorithm with Lamarckian individual learning for generation scheduling,” *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1181–1186, 2003.

[5] J. A. Vasconcelos, J. A. Ramírez, R. H. C. Takahashi, and R. R. Saldanha, “Improvements in genetic algorithms,” *IEEE Transactions on Magnetics*, vol. 37, no. 5 I, pp. 3414–3417, 2001.

[6] X. Yao, Y. Liu, and G. Lin, “Evolutionary programming made faster,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[7] Z. Tu and Y. Lu, “A robust stochastic genetic algorithm (StGA) for global numerical optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, pp. 456–470, 2004.

[8] G. R. Harik, F. G. Lobo, and D. E. Goldberg, “The compact genetic algorithm,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 287–297, 1999.

[9] C. W. Ahn and R. S. Ramakrishna, “Elitism-based compact genetic algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 4, pp. 367–385, 2003.

[10] S. Rimeharoen, D. Sutivong, and P. Chongstitvatana, “Updating strategy in compact genetic algorithm using moving average approach,” in *Proceedings of IEEE Conference on Cybernetics and Intelligent Systems*, pp. 1–6, June 2006.

[11] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[12] J. McCall, “Genetic algorithms for modelling and optimisation,” *Journal of Computational and Applied Mathematics*, vol. 184, no. 1, pp. 205–222, 2005.

[13] E. C. Yeh and Y.-Y. Shyu, “New genetic algorithm with statistical gene evaluation,” in *Proceedings of the 1st International Joint Conference of NAFIPS/IFIS/NASA*, pp. 409–410, December 1994.

[14] N. Kubota, K. Shimojima, and T. Fukuda, “Role of virus infection in virus-evolutionary genetic algorithm,” in *Proceedings*

of IEEE International Conference on Evolutionary Computation (ICEC '96), pp. 182–187, May 1996.

[15] C. H. Lin, "A rough penalty genetic algorithm for constrained optimization," *Information Sciences*, vol. 241, pp. 119–1137, 2013.

[16] R. Hinterding, "Gaussian mutation and self-adaption for numeric genetic algorithms," in *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 384–388, December 1995.

[17] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: philosophy and performance differences," in *Proceedings of the Evolutionary Programming VII*, pp. 601–610, 1998.

[18] T.-Y. Sun, C.-C. Liu, S.-T. Hsieh, C.-G. Lin, and K.-Y. Lee, "Cluster-based adaptive mutation mechanism to improve the performance of genetic algorithm," in *Proceedings of the 6th International Conference on Intelligent Systems Design and Applications (ISDA '06)*, pp. 461–466, October 2006.

[19] Y.-W. Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 41–53, 2001.

[20] J.-T. Tsai, T.-K. Liu, and J.-H. Chou, "Hybrid Taguchi-genetic algorithm for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 4, pp. 365–377, 2004.

[21] Q. Zhang and Y.-W. Leung, "An orthogonal genetic algorithm for multimedia multicast routing," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 1, pp. 53–62, 1999.

[22] E. M. El-Alfy, S. N. Mujahid, and S. Z. Selim, "A Pareto-based hybrid multiobjective evolutionary approach for constrained multipath traffic engineering optimization in MPLS/GMPLS networks," *Journal of Network and Computer Applications*, vol. 36, pp. 1196–1207, 2013.

[23] H. Salama, "The multicast routing simulator," The Real-Time Communication Project, 1997, http://rtcomm.csc.ncsu.edu/index.htm.

[24] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.