An Efficient Evolutionary Algorithm for Accurate Polygonal Approximatio

Shinn-Ying Ho (何信瑩)
Department of Information Engineering,
Feng Chia University,
Taichung, Taiwan, Republic of China
syho@fcu.edu.tw

Yeong-Chinq Chen (陳永慶)
Department of Information Engineering,
Feng Chia University,
Taichung, Taiwan, R epublic of China
d8527158@knight.fcu.edu.tw

Abstract

This paper proposes an efficient evolutionary algorithm (EEA) with a novel orthogonal array crossover (OAX) for solving the accurate polygonal approximation problem. Using the OAX, t he chrom osomes of the children are formed from the best combinations of the better genes from the parents rather th an the conventional random combinations of the parents' genes. The choice of the better genes is derived by way of a systematic reasoning approach for evaluating contribution of the individual gene based on the orthogonal arra . Genetic algorithm approach has been proposed and its performance is much better than that of several superior methods in obtaining accurate solutions of polygonal approximation problems. It is shown empirically that the proposed EEA outperforms the superior genetic algorithm proposed in literature under the same cost condition in the quality of the best solution, average solution, variance of solutions, and the convergence speed, especially in solving large polygonal approximation problems.

Key words : Genetic algorithm, Evolutionary algorithm, Polygonal approximation, Orthogonal arra crossover, Optimization.

1. INTRODUCTION

The approximation of shape representation is an important issue [1-18]. The statement of the polygonal approximation problem is simple as follows For a given number of approximating vertices, the objective is to minimize the error norm between the digital curve and the approximatin polygon. Many polygonal approximation methods can be found in literature such as dynamic programming approach [2,3], Newton's method [4] , iterative point elimination approach [5], sequential approach [6,7], split-and-merge approach [8-10], dominant point s detection approach [11-15], k-means based approach [16-17], and GA-based approach [18]. A new method for polygonal approximati using a genetic algorithm, proposed by Yin [18 , has been shown that its performance is much better than those of several superior methods in obtaining accurate solutions of polygonal approximation problems using the same benchmark [13].

In this paper we solve the polygonal approximation problem using an efficient evolutionary algorithm (EEA) with a novel orthogonal array crossover (OAX) that is an efficient general-purposed algorithm capable of solving large parameter optimization problems. The merit of EEA is the use of the orthogonal array (OA) to achieve an intelligent genetic crossove . The chromosomes of the children are formed from the best combinations of the better genes from the parents rather than the conventional random combinations of the parents' genes. The choice of the better genes is derived by way of a systematic reasoning approach for evaluating contribution of the individual gene based on OA. Theoretical analysis and experimental studies for the superiority of the intelligent crossover can be fo und in our recent work [19]. In this study, we try to achieve the following goals

(1) Show empirically that the proposed EEA outperforms the genetic algorithm approach proposed by Yin [18] in solving the accurate polygonal approximation problem using the same benchmark. The comparisons of performance evaluation take the following factors into consideration: the quality of the best solution, average solution, variance of solutions, and the convergence speed under the same cost condition.

(2) In real world applica tions, the image contours may be larger than the used laboratory contour benchmark [11]. To compare the performance for solving large polygonal approximation problems with the competitive Yin [18], we use a large real contour with 900 points, taken from di gitization of real objects in real applications, and apply it to test four superior polygonal approximation algorithms. Fr om the encouraging simulation results, we show that EEA i more superior to the other genetic algorithms i solving large polygonal approximation problems.

(3) A modified fitness function with a penalty term for dealing with the infeasible individuals is used t reveal the essentiality of maintaining feasibility for crossover operations and demonstrates the superiorit of the proposed OAX in solving accurate polygonal approximation problems.

2. PRELIMINARY OF CROSSOVER

Since genetic algorithm (GA) emplo s parallel search and has good performance in solving optimization problems [20,21], it can also be used to solve the optimal polygonal approximation problem. It has been theoretically proven that GA provides robust search even if the search space is not continuous. Crossover is the main operator of GAs. The crossovers for solving the binary string decision problem can be categorized into two classes, described in the following two subsections.

2.1 Crossover without maintaining feasibility

GA produces new solutions by recombining the encoded solutions from a population. Because GA works by recombining and altering solutions, maintaining feasibilit is difficult for many problems. Crossover may yield infeasible children from two feasible parents. This especially arises in combinatorial optimization where the encoding is the traditional bit string representation and the crossover is the general -purposed crossover. Generally, a penalty function approach is used to cope with the infeasible solutions as follows [22]. Given an optim izati problem,

$$Min.\ Z(x) \quad s.t. \quad x \in A, x \in B \qquad (1)$$

where x is a vector of decision variables, the constraints "$x \in A$" are relatively easy to satisfy, and the constraints "$x \in B$" are relatively hard to satisfy. The problem can be reformulated as

$$Min.\ Z(x) + p(d(x,B)) \quad s.t. \quad x \in A \qquad (2)$$

where d(x, B) is a metric function describing the distance of the vector x from the region B, and p(.) is a penalt function such that p(0) = 0. This is an exterior penalt function defined such that if the function p(.) grows quickl enough outside of B, the optimal solution of Eqn. (1) will also be optimal for Eqn. (2). Furthermore, any optimal solution of Eqn. (2) will provide an upper bound on the optimum for Eqn (1), and this bound will in general be tighter than that obtained by simply optimizing Z(x) over A. It can be recognized that when the strength of the penalt function is too large, the search algorithm is not allowed t enter the infeasible region, and consequently, might spend more time in finding feasible region than in finding the optimum. However, if the strength is to small, the search algorithm will spent too much time in evaluating invalid solutions. How to effectively determine the penalt function is in general problem-dependent.

Let α and β be parents pairs that have the same numbers of "0" and "1" in a bit strings chromosome before performing crossover operations. Assume that feasible solution has the same number of "0" and "1" in a chromosome. An integer number s between [1, N -1] is randomly, where N is the bit string length. The children α' and β' are obtained by swapping all the bits of α and β after position s using a traditional one -cut-point crossover. For example, feasible parents α and β with N = 10 and s = 5 can generate infeasible children α' and β' as follows

α = 0 1 0 0 1 | 1 1 0 0 0 $\quad$ α' = 0 1 0 0 1 | 1 0 1 0 1

β = 0 0 1 0 0 | 0 1 0 1 0 1 $\quad$ β' = 0 0 1 0 0 | 1 1 0 0 0

To cope with the infeasible children problem, the penalt function must be added into the fitting function. A general-purpose penalty function for binary string decision problem is described as follows. Let the number of "1" in a chromosome be K and the number of "1" after a crossover operation be W. The penalty function can be defined as p(x) = γ |W – K|, where γ is a weighting constant. However, how to efficiently determine the γ value is not an easy task.

2.2 Crossover with maintaining feasibility

One solution for preserving feasibility of children is t increase the complexity of the crossove . Some application-dependent crossovers have been proposed [23]. A GA repair operator, used by Yin [18], adopts a similar two-cut-point crossover for polygonal approximation. At first, an integer number s between [1, N -2] is randomly generated as the first cut point and, then, check the numbers of corresponding (0,1)-pair and (1,0) -pair after positi s in the parents' bit strings. If one of them is zero, regenerate another number until the nu mbers of (0,1)-pair and (1,0)-pair are both non-zero. The second cut point can be determined in the position t such that the numbers of (0,1)-pair and (1,0)-pair between s and t are identical. It is guaranteed by swapping (0,1)-pairs and (1,0)-pairs simultaneously between parents to contain the same number of 1's for the string offspring using the two-cut-point crossover. Take the same example using α and β. If s = 7, the numbers of (0,1)-pair and (1,0)-pair are 2 and 0. Therefore, this crossover will fail since the second cut point ca n not be determined. If s = 6, it can obtain the second cut point in the position t = 8 and, then, the feasible children α' and β' can be obtained as follows

α = 0 1 0 0 1 1 | 1 0 0 $\quad$ α' = 0 1 0 0 1 1 | 0 1 0 0

β = 0 0 1 0 0 1 | 0 1 0 1 $\quad$ β' = 0 0 1 0 0 1 | 1 0 0 1

Since the offspring always are feasible solutions, no penalt function is needed.

Although, the above mentione d two-cut-point crossover can always produce feasible offspring, the children can be formed from only the exchange of the middle substring. We will propose an orthogonal array crossover OAX, which utilizes the decomposition of the chromosome into a variable number of substrings, can produce the children using the best combinations of the better substrings from the parents. The choice of the better substrings is derived by way of a systematic reasoning approach for evaluatin contribution of the individual substring based on OA.

3. PROPOSED EFFICIENT EVOLUTIONAR ALGORITHM

EEA uses a novel OAX based on the ability of OA, which is descried in Section 3.1. Section 3.2 presents OAX procedure. The illustration of OAX using a concise example is given in Section 3.3. EEA is provided in Section 3.4.

3.1 Orthogonal arrays and factor analysis

Orthogonal array (OA) and factor analysis, which are representative methods of quality control [24], also work to improve the crossover more efficiently. The superiority of OA on obtaining better results for large parameter optimization problems has been demonstrated in our recent

work [19,25]. The definition of the O is as follows. Let there is N factors of two levels. The number of total combination is 2^N. Columns of two factors are orthogonal when 4 pairs, (1,1), (1,2), (2,1), and (2,2), occur equally in all experiments. When any two factors in an experimental set are orthogonal, the set is called an OA. To establish an OA of N factors of two levels, we obtain an integer $n = 2^{\lceil \log(N+1) \rceil}$, build an orthogonal arra $L_n(2^{n-1})$ with n rows and (n-1) columns, and select N columns. Factor analysis can evaluate the effects of factors on the evaluation function, rank the most effective factors, and determine the best level for e ach factor such that the evaluation is optimized. Orthogonal experiment design can reduce the number of experiments for the factor analysis. The number of OA experiments for single factor analysis is only n. For instance, Table 1 shows an orthogonal arra $L_8(2^7)$. Let y_t be the positive function evaluation value of experiment no. t. Define the main effect of factor j with level k S_{jk},

$$S_{jk} = \sum_{t=1}^{n} Y_t^2 \times \left[\begin{array}{l} the\ level\ of\ Exp.no.t \\ of\ factor\ j\ is\ k \end{array} \right] \quad (3)$$

where

$$[condition] = \begin{cases} 1 & if\ the\ condition\ is\ true \\ 0 & otherwise\ , \end{cases}$$

and

$$Y_t = \begin{cases} y_t & if\ the\ function\ is\ to\ be\ \max . \\ 1/y_t & if\ the\ function\ is\ tobe\ mix. \end{cases}$$

Note that the main effect reveals the individual effect of a factor. The most effective factor j has the largest main effect difference (MED) $| S_{j1} - S_{j2} |$. If $S_{j1} > S_{j2}$, the level 1 of factor j is better than the level 2 on the contribution for the optimization function. Otherwise, level 2 is better.

3.2 Orthogonal array crossover OAX

Let the length of the bit string chromosome be N and the number of 1's be K. The newly produced strings (or offspring) after OAX are forced to contain the same number of 1's as their parents. The OAX is as follows

Step 1: Determine the maximal number M of segments using M-1 cut points such that the numbers of 1 's in any two corresponding segments of the parents are identical and any two corresponding segments are not identical . If M = 0, OAX will not b applied before performing the mutation operation. For example, let N=10, K=5, and two parents strings P_1 and P_2 be as follows. It can be determined that M = 3.

P_1 = 1 0 1 0 1 1 1 0 0 0

P_2 = 0 1 0 1 0 1 0 1 0 1

Step 2: Select the first M columns of OA $L_n(2^{n-1})$ where $n = 2^{\lceil \log(M+1) \rceil}$. Note that one segment in a chromosome is regarded as a factor in OA. Let level 1 and level 2 o f factor j represent the j^{th} segments coming from P_1 and P_2, respectively.

Step 3: Evaluate the function value $_t$ for experiment no. t where t = 1, 2, ..., n.

Step 4: Compute the main effect S_{jk} where j = 1, 2, ..., M and k = 1, 2.

Step 5: Determine the best level for each segment. Select level 1 for the j^{th} segment if $S_{j1} > S_{j2}$. Otherwise, select level 2.

Step 6: The chromosome of the first child is formed from the best combinations of the better segment derived from the corresponding parents.

Step 7: Rank the most effective factors from rank 1 to rank M. The factor with large MED has higher rank.

Step 8: The chromosome of the second child is formed similarly as the first child except that the segment with the lowest rank adopts the other level.

3.3 Illustration by a concise example

In this section, OAX is illustrated by a concise example and the simulation results are show in Tables 1 and 2. Let P_1 and P_2 be parents, and C_1 and C_2 be the children after performing OAX procedure. The polygon figure-8 with 45 points (N=45) and 9 vertices (K=9) in Figure 1(d) is adopted. The feasible solution must have 9 1's and 36 0's in a chromosome. The maximal number M of segments for P_1 and P_2 is 6. Therefore, the first 6 columns of OA $L_8(2^7)$ are used. Let

P_1 = (100000 0010000 100010000 110100000000 1000000 0100)

and

P_2 = (001000 1000000 000010100 000010100001 0010000 1000).

The values of all variables for factor analysis are shown in Table 1. For instance, in the 3rd experiment, the function evaluation value y_3 (177.88) can be obtained using Eqn.(4) from the 1st, 4th and 5th segments of P_1, and the 2nd, 3rd, and 6th segments of P_2. The main effect S_{31} (8.28) can be obtained using $S_{31} = y_1^{-2}+y_2^{-2}+y_7^{-2}+y_8^{-2}$ from Eqn. (3). The child C_1 can be derived from combining the 2nd, 3rd, 5th and 6th segments of P_1, and the 1st and 4th segments of P_2. Based on MED, the child C_2 is only different in segment 6 with the lowest rank from C_1. The OAX results are shown in Table 2.

Table 1. Orthogonal array $L_8(2^7)$ and factor analysis.

	Factors								
Exp. no.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Function Evaluation value y_t (E_2)**	
1	1	1	1	1	1	1	1	y_1	87.11
2	1	1	1	2	2	2	2	y_2	63.10
3	1	2	2	1	1	2	2	y_3	177.88
4	1	2	2	2	2	1	1	y_4	136.59
5	2	1	2	1	2	1	2	y_5	161.08
6	2	1	2	2	1	2	1	y_6	61.70
7	2	2	1	1	2	2	1	y_7	138.00
8	2	2	1	2	1	1	2	y_8	50.48
$S_{j1}(10^{-4})$	4.68	6.84	8.28	2.54	8.18	6.16	-		
$S_{j2}(10^{-4})$	7.46	5.30	3.86	9.60	3.96	5.98	-		
MED(10^{-4})	2.78	1.54	4.41	7.05	4.23	0.18	-		
Rank	4	5	2	1	3	6	-		

Table 2. Results of OAX.

	1	2	3	4	5	6	E_2
P_1	100000	0010000	100010000	110100000000	1000000	0100	87. 1
P_2	001000	1000000	000010100	000010100000	0010000	1000	129.14
C_1	001000	0010000	100010000	000010100001	1000000	0100	46.82
C_2	001000	0010000	100010000	000010100001	1000000	1000	49.44

3.4 Efficient evolutionary algorithm

In this section, solving polygonal approximation problems using EE is described in detail. Assume the given digital curve S has N points and the specified degree of the approximating polygon is K, then this problem has search space $C(N, K)$. Let $\alpha = a_1a_2...a_N$ be a binary string of length N and the number of 1's in α be K where $a_i = 1$ means the i^{th} point of S is chosen as a vertex of the pol ygon and $a_i = 0$ means the i^{th} point is eliminated. Consequently, α represents a possible solution of the polygonal approximation problem. Let the perpendicular distance between the i^{th} point of S and the nearest line segment in the approximating polygon be $e_i(\alpha)$. The error norm is define as $[e_i(\alpha)]^2$, since integral square error can evaluate the whole accurate effect of fitness function. The objective of the problem is to minimize the approximation error between S and its approximating polygon. The fitness function value of α can be define as

$$f(\alpha) = E_2(\alpha) = \sum_{i=1}^{N} [e_i(\alpha)]^2 \qquad (4)$$

EEA can be written as follows.

Step 1: Initiation Randomly generates an initial population with size N_{pop}. Each individual I_i contains K 1's and $(N-K)$ 0's where $i = 1, 2,..., N_{pop}$.

Step 2: Elitist strategy : Repeat the following steps for $i=2$ to N_{pop} :

2a: Select I_1 and I_i as the parents and produce two children I_{c1} and I_{c2} using OAX.

2b: Replace I_1 and I_i using the best and the second best individuals according to fitness performance among I_1, I_i, I_{c1} and I_{c2}, respectively.

Step 3: Evaluation: Evaluate the function values for all individuals.

Step 4: Selection Use the rank selection that replace the worst P_s * N_{pop} individuals using the best individual to form the new population where P_s is the selection probability.

Step 5: Crossover Select P_c * N_{pop} parents for OAX where P_c is the crossover probability. Apply OAX to the selected pairs of parents. Two individuals with the better fitness function valves among t he parents and children replace the two children for the elitist strategy.

Step 6: Mutation: Apply the swap mutation operator t the randomly selected P_m * N_{pop} individuals in the new population where P_m is the mutation probability. To prevent the fitness value fr om deteriorating, mutation is not a pplied to the best individual.

Step 7: Termination test: If a prespecified stopping condition is satisfied, end the algorithm Otherwise, go to Step 3.

4. EXPERIMENTAL RESULTS AND PERFORMANCE COMPARISON

4.1 Performance evaluation using the small benchmark

In order to demonstrate the superiority of our algorithm, we compare its performance with that of Yin's geneti algorithm (YinGA) [18], which has shown the superiorit as the best approach in the accurate polygonal approximation problem domain. The three benchmarks are chromosome with 60 points Figure 1(a), figure-8 with 45 points Figure 1(d), and circles with 102 points Figure 1(g). The parameters of EEA are as follows: N_{pop} = 100, p_s = 0.04, p_c = 0.5, and p_m = 0.2. The simulation of the small benchmark conducts ten independent runs for EEA. All the reported results are obtained under the same cost (number of function evaluation, FITs). The simulation results of YinGA and EEA are shown in Figure 1 (b), (e) and (h) and Figure 1 (c), (f) and (i), respectively. The average error values are summarized in Table 3. It can be seen that EE outperforms YinGA in the quality of the best solution, average solution, variance of solutions, and the convergence speed. The convergence spee and average accuracy for YinGA and EE are illustrated in Figure 2.

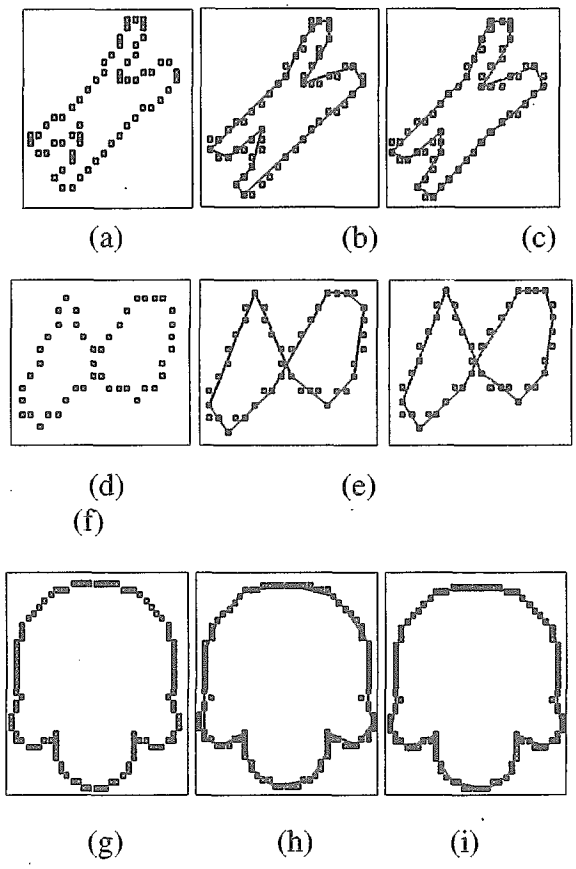


Figure 1. Three small benchmarks and the comparative results of YinGA and EEA. (a) Chromosome; (b) Apply YinGA using 14 vertices; (c) Apply EEA using 14 vertices, (d) figure-8; (e) Apply YinGA using 10 vertices; (f) Appl EEA using 10 vertices, (g) circles; (h) Apply YinGA usin 19 vertices; and (i) Apply EEA using 19 vertices.

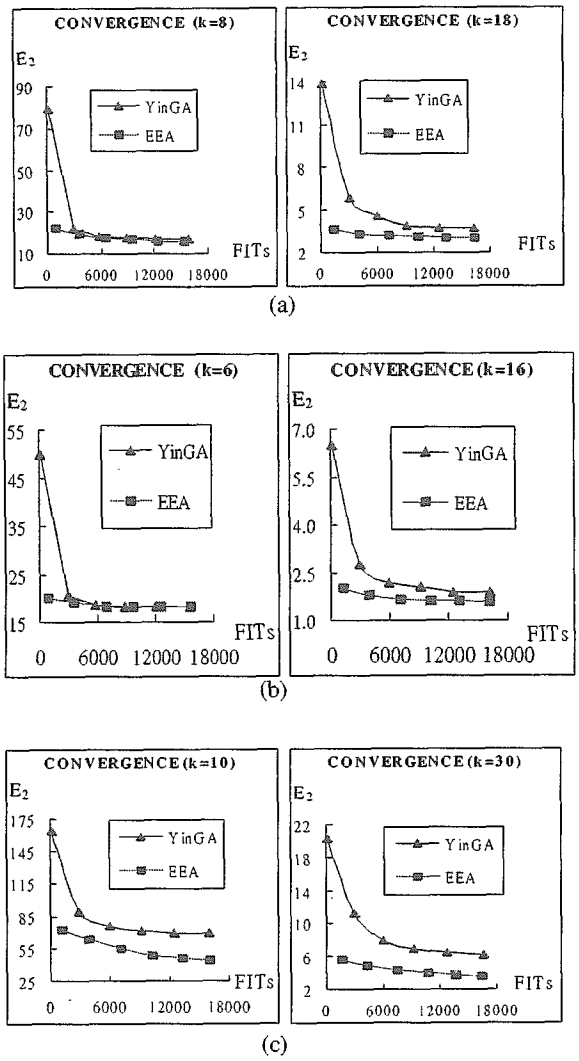


Figure 2. The convergence speed and average accuracy of YinGA and EEA for various benchmarks (a) chromosome, (b) figure-8, and (c) circles with various K's value.

Table 3. Experimental results of small bechmarks for YinGA and EEA.

Results of Figure 1(a)						
K	BEST E_2		AVERAGE E_2		VARIANCE	
	YinGA	EEA	YinGA	EEA	YinGA	EEA
8	17.41	13.43	16.87	15.51	11.96	2.33
9	13.82	12.08	14.47	13.49	7.21	1.63
12	7.99	5.82	7.97	6.79	2.46	0.87
14	5.47	4.17	6.02	5.11	1.44	0.56
15	5.22	3.80	5.15	4.32	0.75	0.28
17	4.58	3.13	4.56	3.55	0.35	0.16
18	4.17	2.83	3.68	3.04	0.18	0.05
total	58.66	45.26	58.72	51.81	24.35	5.88
EEA/ YinGA	77.2%		88.2%		24.1%	

Results of Figure 1(d)						
K	BEST E_2		AVERAGE E_2		VARIANCE	
	YinGA	EEA	YinGA	EEA	YinGA	EEA
6	17.49	17.49	18.25	18.22	2.15	0.41
9	6.84	4.54	5.41	4.78	0.81	0.17
10	3.91	3.69	4.42	3.94	0.37	0.05
11	3.83	2.90	3.52	3.20	0.18	0.05
13	2.24	2.04	2.54	2.38	0.14	0.06
15	2.01	1.61	1.98	1.83	0.10	0.05
16	1.94	1.41	1.87	1.58	0.08	0.03
total	38.26	33.68	37.99	35.93	3.83	0.82
EEA/ YinGA	88.0%		94.6%		21.4%	

Results of Figure 1(g)						
K	BEST E_2		AVERAGE E_2		VARIANCE	
	YinGA	EEA	YinGA	EEA	YinGA	EEA
10	52.95	38.92	69.27	44.07	113.3	76.50
12	42.85	26.00	40.99	29.53	86.52	5.08
14	29.93	17.39	27.22	20.14	31.47	4.69
17	17.41	12.22	20.03	14.58	18.24	2.23
18	14.80	11.34	16.35	12.86	6.81	1.56
19	14.94	10.04	15.03	11.52	5.06	0.92
22	12.91	7.19	11.40	8.52	2.29	0.59
27	7.04	3.74	7.51	5.03	1.22	0.52
30	6.61	2.84	6.12	3.57	0.93	0.34
total	199.4	129.7	213.9	149.8	265.8	92.43
EEA/ YinGA	65.0%		70.0%		34.8%	

4.2 Performance evaluation using the large benchmark

Most of the existent works deal only with laboratory contours, that is to say, curves specifically design ed or selected by the authors to show the goodness of their approaches; or curves with a small number of points commonly utilized in the literature in order to make their results comparable [11 p.685]. In this section, a large contour with 900 points taken from digitization of a real popular USA map in practical applications is used t analyze the effects of various crossovers, as shown in Figure 3(a). To demonstrate the superiority of the orthogonal array crossover and reveal the essentiality of maintaining feasible solutions in solving accurate polygonal approximation problems, additional two crossovers that may produce infeasible individuals are applied. The fitness function considering the infeasible solution is modified as follows

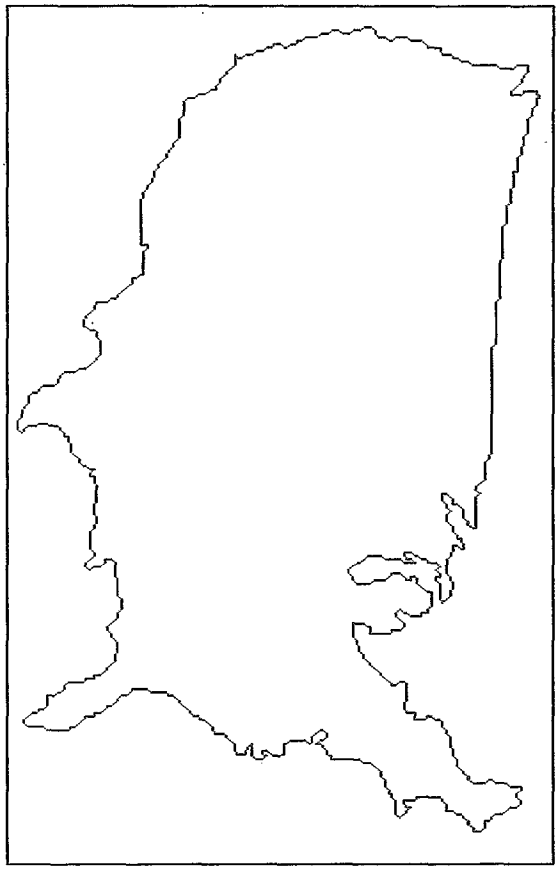
$$f(\alpha)=E_2(\alpha)=\sum_{i=1}^{N}[e_i(\alpha)]^2+\gamma|W-K| \qquad (5)$$

The first crossover is the traditional simple one-cut-point crossover that may produce infeasible offspring. Let YinGAP be the same genetic algorithm as YinGA except that the one -cut-point crossover is incorporated and the fitness function with the penalty term in Eqn. (5) is applied simultaneously. In order to generate infeasible individuals for testing the capability of OAX in EEA, a hybrid orthogonal array crossover is used. The hybrid crossover is the same as OAX described in Section 3.2 except Step 1. In the Step 1, a simple one-cut-point crossover is first used. Then, determine the maximal number M of segments usin M-1 cut-points such that the numbers of 1's in any two corresponding segments of the parents are identical except the last segment which may not be identi cal. Let EEAP be the same algorithm as EEA except that the hybrid crossover and Eqn. (5) are used. The parameters of YinGA and YinGAP are as follows $\gamma = 20$, $N_{pop} = 100$, p_s, p_c, and p_m are the same as those of YinGA for small benchmarks, and the generation number of stopping condition is 200. The parameters of EEA and EEAP are as follows $\gamma = 20$, $N_{pop} = 50$, $p_s = 0.04$, $p_c = 0.5$, and $p_m = 0.18$. All the results are obtained under the same cost condition as YinGA and YinGAP. The simulation of the large benchmark conducts ten independent runs for the four algorithms, YinGAP, YinGA, EEA, and EEAP, and the average results are summarized in Table 4. Two approximating polygons derived from YinGA and EEA are illustrated in Figure 3(b) and 3(c), respectively. The convergence speed and average accuracy of the four algorithms are shown in Figure 4. The simulation results reveal the following

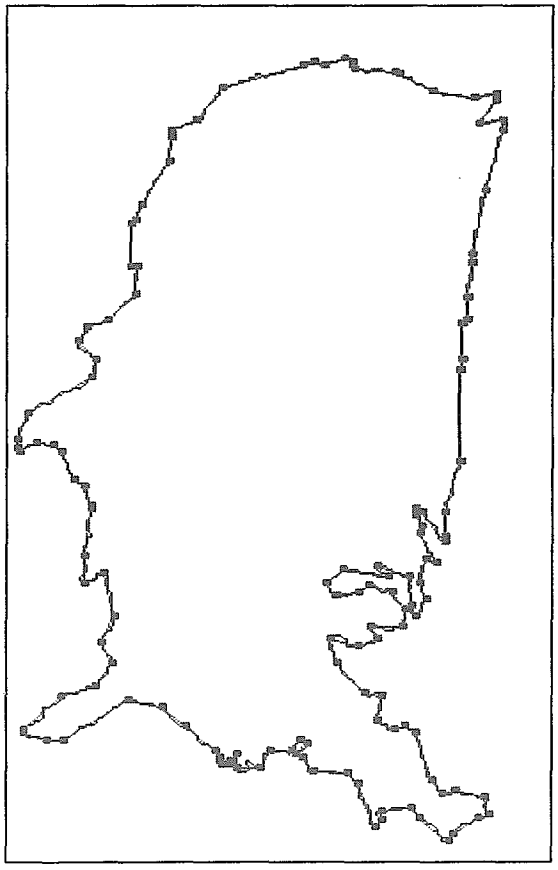
(1) The crossover, which can maintain feasibility, is superior to the crossover which may produce infeasible individuals.

(2) EEA with the O AX is more superior to the investigated GA with other crossovers, even applied to the infeasible individuals. Note that EEAP is superior to YinGA.

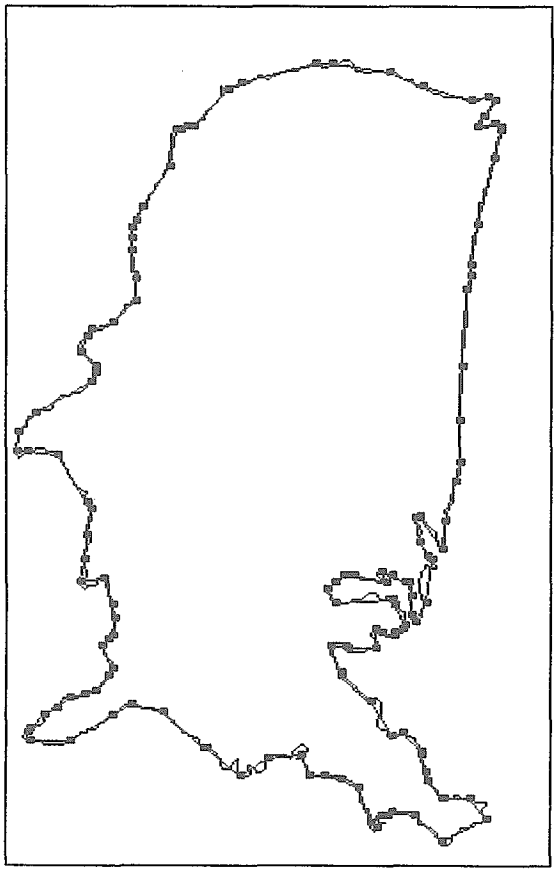
(3) The relationship pairs for the number of points and the average E_2 error ratio 'EEA/YinGA' are (45, 94.6%), (60, 88.2%), (102, 70.0%), and (900, 36.97%) for polygons in Figure 1(d), 1(a), 1(g) and Figure 3(a), illustrated in Tables 3 and 4. The large number of points results in small error ratio. Compare the ratios of the large benchmark with those of the small benchmark, it can be recognized that EEA with OAX is more efficient to solve the large polygonal approximation problems.



(a)

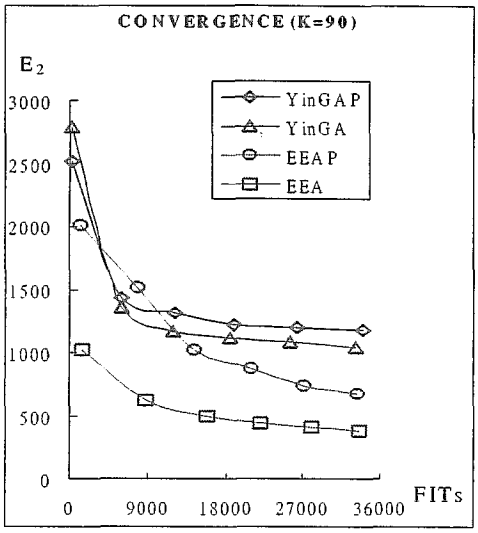


(b)



(c)

Figure 3. The large contour of USA ap and the comparative results of YinGA and EEA. (a) A large contour of USA map with 900 points, (b) Apply YinGA using 135 line segments, and (c) Apply EEA using 135 line segments.



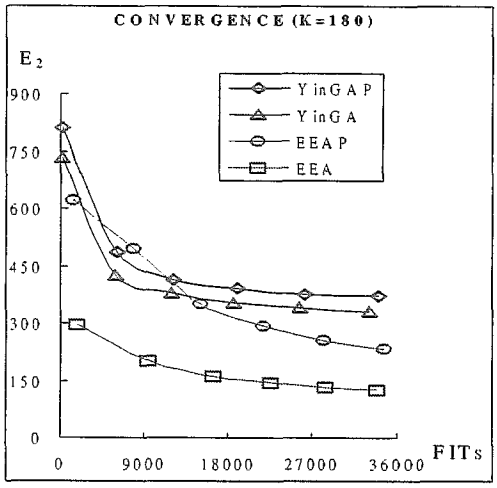


Figure. 4. The convergence speed and average accuracy of YinGA, YinGAP, EEA and EE P for the contour of USA map with various K's value.

Table 4. Experimental results of large benchmark for various algorithms.

BEST E_2				
K	YinGAP	YinGA	EEAP	EEA
90	853.33	832.33	462.60	323.51
135	472.32	399.34	269.05	161.67
180	304.26	252.85	180.59	107.54
total	1629.91	1484.52	912.24	592.72
EEA/ others	36.37%	39.93%	64.97%	—

AVERAGE E_2				
K	YinGAP	YinGA	EEAP	EEA
90	1182.82	1040.47	674.96	380.49
135	571.90	542.89	370.68	201.28
180	371.36	332.15	234.71	126.44
total	2126.08	1915.51	1280.35	708.21
EEA/ others	33.31%	36.97%	55.31%	—

VARIANCE				
K	YinGAP	YinGA	EEAP	EEA
90	16366.92	11010.24	8944.00	1389.53
135	7271.09	4743.53	2720.49	382.71
180	1657.45	1314.79	714.99	136.80
total	25295.46	17068.56	12379.48	1909.04
EEA/ others	7.55%	11.18%	15.42%	—

5. CONCLUSION

It has been demonstrated that the proposed EEA outperforms the existent genetic algorithm approach, especially in solving large polygonal approximation problems in the quality of the best solution, average solution, variance of solution, and convergence spee . By observing the general statistical analyses in EEA and YinGA, it is easy to see that EEA outperforms YinGA, especially in large real contour problems. Even if usin general penalty design, EEAP using OAs is better then YinGAP and YinGA. The results demonstrate that EE makes use of the systematic reasoning abilit of OAs can efficiently obtain accurate solutions. As a result, EEA is suitable for solving C(N, K) pro·blems, since polygonal approximation problem is a canonical type of the C(N, K) problems.

REFERENCES

[1] J. C. Perez, E. Vidal, "Optimal polygonal approximation of digital curves," Pattern Recognition Letters, 15, pp. 743-750, 1994.

[2] R. Bellman, "On the approximation of curves by line segments using dynamic programming," Commune. ACM 4, 6, 1961.

[3] J. G. Dunham, "Optimum uniform piecewise linear approximation of planner curves," IEEE Trans. Pattern Analysis Mach, Intell. PAMI-8, pp. 67-75, 1986.

[4] T. Pavlidis, "Polygonal approximation by Newton's method," IEEE Trans. Comput. 26, pp. 800-807, 1977.

[5] Arie Pikaz, I. Dinstein, "An algorithm for polygonal approximation of digital curves based on iterative points elimination," Pattern Recognition Letters, 16, pp. 557-563, 1995.

[6] Y. Kurozumi, W. A. Davis, "Polygonal approximation by the minimax method," Co mputer Graphics Image Processing, 19, pp. 248-264, 1982.

[7] K. Wall, P. E. Danielsson, "A fast sequential method for polygonal approximation of digitized curves, Computer Vision," Graphics, Image Processing, 28, pp. 220-227, 1984.

[8] T. Pavilids, S. Horowitz, "Segmentation of plane curves," IEEE Trans. Comput. 23, pp. 860-870, 1974.

[9] J. G. Leu, L. Chen, "Polygonal approximation of 2-D shapes through boundary merging," Pattern Recognition 28, pp. 571-579, 1988.

[10] B. K. Ray, K. S. Ray, "A new split-and-merge technique for polygonal approximation of chain coded curves," Pattern Recognition Letters, 16, pp. 161-169, 1995.

[11] Jose. Manuel. Inesta, Mateo. Buendia, Maraia. Angeles. Sarti, "Reliable polygonal approximations of imaged real objects through dominant point detection," Pattern Recognition, 31, pp. 685-697, 1998.

[12] B. K. Ray, K. S., Ray, "An algorithm for detection of dominant points and polygonal approximation of digitized curves," Pattern Recognition Letters, 13, pp. 849-856, 1992.

[13] C. H. The, R. T. Chin, "On the dominant points on digital curves," IEEE Trans. Pattern Anal. March Intell, 11(8), pp. 859-872, 1989.

[14] W. Y. Wu, M. J. Wang, "Detecting the dominant points by the curvature-based polygonal approximation," CVGIP: Graphical Models and Image Processing, 55, pp. 79-88, 1993.

[15] Philippe Cornic, "Another look at the dominant point detection of digital curves," Pattern Recognition Letters, 18, pp. 13-25, 1997.

[16] T. Y. Phillips, A. Rosenfeld, "An ISODAT algorithm for straight line f itting," Pattern Recognition Letters, 7, pp. 291-297, 1988.

[17] P. Y. Yin, "Algorithms for straight line fitting using K-means," Pattern Recognition Letters, 19, pp. 31-41, 1998.

[18] P. Y. Yin, "A new method of polygonal approximation using genetic algorithm," Pattern Recognition Letters, 19, pp. 1017-1026, 1998.

[19] Shinn-Ying Ho, Li-Sun. Shu, Hung-Ming. Chen, "Intelligent genetic algorithm with a new intelligent crossover using orthogonal arrays," GECCO-99: Proc. of the Genetic and Evolutionary Computation Conf., pp. 289-296, 1999.

[20] D. E. Goldberg, "Genatic Algorithms in Search Optimization and Machine Learning, " Addison-Wesley, Reading MA, 1989.

[21] J. H. Holland, "Adaptation in Natural and Artificial Systems," Univ. of Michigan Press, Ann Arbor ; MI, 1975.

[22] David. W. Coit, Alice. E. Smith, "Penalty guided genetic search for reliability design optimization," Pattern Recognition, 30, pp. 895-904, 1996.

[23] N. R. Pal, S. Nandi, M. K. Kundu, "Self crossover: a new genetic operator and its application to feature selection," Internal. J. Systems Sci., 29, pp. 207 -212, 1998.

[24] S. Taguchi, S. Konishi, "Orthogonal arrays and linear graphs," Dearbon MI: America Supplier Institute, 1987.

[25] Shinn-Ying Ho, Hung-Ming Chen, Li-Sun Shu, "Solving large knowledge base partitioning problems using an intelligent genetic algorithm," GECCO-99: Proc. of the Genetic and Evolutionary Computation Conf., pp.1567-1572, 1999.