Improved environmental adaption method
and its application in test case generation

K.K. Mishra*, Shailesh Tiwari and A.K. Misra
Computer Science and Engineering Department, MNNIT Allahabad, Allahabad, India

Abstract. An Environmental Adaption Method (EAM) has been recently proposed in which, each solution updates its structure on the basis of current environmental fitness and its own fitness. EAM has a new operator known as adaption operator. In this paper, an improved EAM, henceforth called as IEAM has been proposed and proper tuning of parameters of algorithm has been done to find the optimal solution in minimum time. IEAM has been tested on a set of benchmark functions. Results show the superiority of IEAM over PSO-TVAC, SADE and other optimization algorithms. Even for complex functions, the performance of IEAM is very good. In addition to this, IEAM has also been applied to software testing to generate test cases for white box testing.

Keywords: Evolutionary computing, improved EAM, PSO

1. Introduction

The term optimization refers to the minimization or maximization of a real function by systematically choosing the values of real or integer variables within an allowed set. Although a number of gradient based approaches [19–21] are available in mathematics to identify optimal solutions, but writing computer algorithm to implement accurate solution of differential equation is very difficult. There is only one way to solve optimization problem and that is to write a randomized algorithm which can search for optimal solution in a large problem search space. As in many cases, users do not have any idea about the nature of function so direction of search cannot be predicted. Keeping this fact in mind, most of these randomized algorithms are derived from the nature because the nature provides a good framework to search good solutions when search direction is not clear. Many nature inspired algorithms such as Evolutionary Programming, Evolutionary Strategy, Differential Evaluation, Genetic Programming [22–25], Genetic Algorithm [10] and Particle Swarm Optimization (PSO) have already gained wide acceptance in solving optimization problems.

Although a number of randomized algorithms are available in literature for solving optimization problems yet their design objectives are same. Each algorithm has been designed with certain goals such as minimizing total number of fitness evaluations to reach nearly optimal solution and to capture diverse optimal solutions in multi-modal solutions. Moreover all these algorithms should be able to escape from local optimal solutions which are possible only if random parameters used in these algorithms are properly tuned [6, 7]. Each randomized algorithm use some operators that are used to search the optimal solutions either by exploiting or by exploring the problem search space.
Among all these operators, only some operators govern major portion of search and these operators are known as dominating operators. These operators are very important because if we know their nature it can be easily determined that for which type of optimization problems this algorithm will be more suitable. For example in a random algorithm, if major portion of search is done by those operators, which perform the search by exploiting the search space then this algorithm will be suitable for uni-modal problems which have only one optimal solution and as such once algorithms find good solutions, it can exploit that region to retrieve optimal solution.

However in multi-modal problems, there may be many optimal solutions and the whole search space would be explored until a solution near to global optimal is captured. So requirement of uni-modal and multi-modal problems are different. This indicates that even if an algorithm is very effective in solving uni-modal problems, it can be trapped in local optimum solution in multi-modal problems. While reverse is not true i.e. any multi-modal optimization algorithm can be applied to solve uni-modal problems. For example, main operators of Genetic algorithm and Particle Swarm Optimization focus on exploitation of good regions, that is, why these algorithms works well with uni-modal optimization problems. However these algorithms are not good in capturing multiple solutions in multi-modal problems [8]. These algorithms should be modified when they are applied to multi-modal problems. Recently, an Environmental Adaption Method [EAM] [9] has been proposed for finding solutions to optimization problems. The algorithm uses three operators: adaption, alteration and selection operator. In EAM, the major contribution in searching is provided by Adaption operator, which explores the search space to search good solutions. Minor contribution is due to alteration operator, this operator exploit the search space, EAM works well with both uni-modal as well as multi-modal problems. However, the performance of EAM can be enhanced further; to search whole search space, the operator must exploit all finite regions having a very high probability of containing the optimal solution. In this way, improved version can achieve a balance between exploitation and exploration.

In the present paper, an improved version of Environmental Adaption Method [26] has been proposed in which some ideas taken from PSO is incorporated in EAM, so that, its operators can exploit all good regions which were discovered by adaption operator of EAM. Proposed algorithm selects these regions by utilizing the information received from the known phenotypic structures of best solutions. This newly designed algorithm will perform well with both types of problems i.e. uni-modal and multi-modal problems. Different state of art algorithms are compared on benchmark functions to check its performance. Moreover, this algorithm has been applied to generate test cases for white box testing. Before discussing the proposed algorithm, let us take a brief overview of the Environmental Adaption Method and the Particle Swarm Optimization Algorithm.

2. Background detail

2.1. Environment adaption method [EAM]

EAM is a computerized implementation of the process used by the species while they adapt themselves to survive in changing environment. According to this theory all existing species have a tendency to update their phenotypic structure to attain better phenotypic structure and to contribute in creation of better environment as compared to previous one. An environmental fitness equal to the average fitness of all species residing in current environment and has been used to measure the fitness of current and previous environment. Using its current and environmental fitness, each species refine its structure. Sometimes, during refinement some alteration may occur due to environmental noise. Finally, best species selected from newly created species and the existing species form the new environment. Those species which do not contribute to the new environment will die. Since genomic changes require current environment, the probability of getting a good species in the next generation is very high.

Alike GA and PSO, this algorithm is also a population based algorithm. This algorithm processes a randomly generated initial population using three operators. The first operator is named as the adaption operator and this causes every solution of current generation to update its phenotypic structure taking its current fitness and environmental fitness in to account. Second operator simply performs alteration, which may result due to environmental noise. After the applications of both of these operators on the current generation, next generation is generated. Selection operator is used to select best solutions from the combination of current and previous generation forming current generation with better environmental fitness. Those solutions which are not selected will be discarded from the current generation. This process is repeated until either we get
the desired solution or maximum number of iterations has reached.

**Adaption Operator:** Adaption operator updates a solution \( Pi \) as follows:

\[
P_{i+1} = (\alpha \ast (\text{Decoded value in decimal of binary coding of } P_i) + \beta) \% 2^L
\]

Where \( F(P_i) \) is the fitness value of \( P_i \), \( \alpha \) and \( \beta \) are the random numbers that are to be decided as per the requirement of problem. \( L \) represents the total number of bits in an individual. \( F_{avg} \) is the average fitness value of current population, representing the current environmental fitness.

**Alteration:** Alteration operator generates new solution \( P_{i+1} \) by flipping one or more bits of \( P_i \). Alteration is carried out according to the following pseudo-code:

```
Begin
for each member in the population
  for each bit in the member’s genome
    Invert the bit with a probability of \( P_{alt} \)
  end
Each new member so generated is considered for selection for the next generation.
```

**Selection:** Best solutions equal to size of initial population are selected from the combination of current and previous generation to form current generation.

2.2. Particle swarm optimization (PSO)

PSO is a robust optimization technique and was first proposed by J. Kennedy, R. Eberhart [1]. In this version, a set of potential solutions (called as particles), are dispersed at various points in the solution space. Each point has some objective function value, and thus has fitness, for its current location. Each particle’s movement is influenced by the following two factors:

- The best fitness achieved by the particle so far. (BFS)
- The best fitness achieved by any particle in the neighborhood. (BPN)

Many new variants of PSO have been proposed and are available in the literature. In canonical PSO size of its neighbors is equal to the size of population. In Local PSO modal each particle updates its position by considering some solutions as its neighbors. Interested reader may refer [1–5] for detailed information.

M. Senthil Arumugam et al. [6] proposed a new version of PSO (called as GLBestPSO) which approach global best solution in different manner. Further modification of same algorithm is done in 2009 by the same authors [7].

Although PSO has very good convergence rate but it may face problem of stagnation. Tuning of random parameters is very important issue and if properly done it can avoid premature convergence of PSO on local optimal solutions. Many papers [3–5] suggested solution of this issue by giving various methods to tune random parameters (Like \( w, c_1, c_2, r_1 \) and \( r_2 \)).

3. Proposed approach

Since each optimization algorithm is designed is to minimize searching time to reach global optimal solution and to escape from local optimal solution. This improved version of EAM is also designed to fulfill these requirements. To improve the convergence rate of existing EAM and to prevent it to converge on local optimal solution, a slight change has been done in the adaption operator of basic EAM. In EAM each solution was updating its structure only on the basis of current environmental fitness and its own fitness; there was no role of best particles. However as initially no information is available about the structure of optimal solution, all solutions will update their structure in the direction of best solution in the hope that optimal structure exist somewhere nearby the structure of best solution. This idea was implemented in PSO. In proposed version, same idea is used to modify the adaption operator of EAM. However to avoid the problem of stagnation, best solution will use adaption operator of EAM and will explore full search space. Moreover proper tuning of parameters \( \alpha \) and \( \beta \) have been done so that global optimal can be searched as soon as possible. In improved EAM all solutions other than best will use following adaption operator

\[
P_{i+1} = (\alpha \ast (\text{Decoded value in decimal of binary version of } P_i) F(X_i)/F_{avg} \\
+ \beta(G_i - P_i)) \% 2^L \ldots (1)
\]

Where \( G_i \) is the position vector of best particle, \( P_i \) is the position value of a particle that is updating its structure. For best solution following adaption operator will be used
\[ P_I + 1 = (\alpha \times (\text{Decoded value in decimal of binary version of } P_{in})) F(X_i)/F_{avg} + \beta/(2^{l}). \]  

This adaption operator can be used both for exploitation and exploration of the search space. In updated version there will be no change in alteration and selection operator. This is represented in the Fig. 1.

Also in proposed version, setting of parameters \( \alpha \) and \( \beta \) has done cleverly so that global optimal solution can be retrieved in early generations. For all solutions other than best, the value of \( \alpha \) will be taken as 1 and the value of \( \beta \) taken as high somewhere around .5. However for best solution the value of \( \alpha \) and \( \beta \) will be taken as very high close to \( 2^l \) and .5. This is done because in all optimization problems first requirement is to search optimal value as soon as possible, in initial generations proposed values will force algorithm to search the optimal value by exploiting the regions around best solutions. However at the same time, best solution will explore whole search space to find out other good regions. This process is repeated again and again until all solutions will tend to converge in to one optimal structure. To check whether this condition is reached, in each generation the value of \( F_{avg}/F(Gi) \) will be checked. With the help of this value, one can identify whether solutions are forming a cluster or not. For example in initial generations this value will be very less, say around 3 or 4. This will happen because in initial generations solution will be distributed over full search space so the initial population will contain a mixture of good and bad solutions. In late generations, when all population is converging toward one structure (this is possible only if all solutions are targeting either local optimum or global optimum), this value will become 1. However to prevent algorithm in being trapped in local optima, a check would be applied when the value of \( F_{avg}/F(Gi) \) will be equal to or greater than .8. This procedure will decide whether the algorithm is converging on global optimal solution or not. When the value of \( F_{avg}/F(Gi) \) will be equal to or greater than .8 the value of \( \alpha \) will be taken as very high (a value which is near to highest value). This is done to perturb solutions of current generations so that premature convergence can be avoided. The reason behind it can be explained as follows, if all solutions of previous generation were approaching toward global optimum then no new solution will replace the existing solutions in the population. As selection operator will choose only best solutions. If the search was targeted toward local best then this stretching of \( \alpha \) will force new generation to capture new good solution and will again create an unstable population that will contain a mixture of good and
bad solutions. This stretching of α is repeated again and again until we get stable population. These algorithms will definitely able to cover many optimal solutions in minimum number of iterations. Proposed IEAM can be explained as follows:

3.1. Proposed algorithm

Input: MAXG (maximum number of generations)  
PnS (Population size)  
Output: Q* (Final set)  
Other variables: n (number of generation)  
O' (intermediate offspring)  
O (final offspring)  
P (temporary pool)  
in (ith individual of nth generation)  
Gi (Best individual of ith generation)

Step
1. Set n = 0.
2. Generate initial population POP0 by PnS random individuals.
3. Apply Improved Adaptation operator (calculation given below) to each member of population POPn and form On'. For best solution

\[ \text{Gi} \rightarrow \text{Gi} + 1 = (\alpha \cdot \text{Decoded value in decimal of binary version of Gi}) + F(\text{Xi}) / F_{avg} + \beta \{ (\text{Gi} - \text{Pi}) \} \mod (2^l) \]

For all other solutions

\[ \text{Pi} \rightarrow \text{Pi} + 1 = (\alpha \cdot \text{Decoded value in decimal of binary version of Pi}) + F(\text{Pi}) / F_{avg} + \beta \{ (\text{Gi} - \text{Pi}) \} \mod (2^l) \]

4. Apply alteration operator as per the probability of alteration On' and form On.
5. \( P_n = \{ \text{POP}_n \cup \text{On} \} \)
6. \( n = n + 1 \)
7. If \( n < \text{MAX}_G \)
   Then go to 10
   Else go to 8
8. Set \( Q' \) = on the basis of fitness function choose n best solutions of \( P_n \)
9. Return \( Q' \)
10. \( \text{POP}_n \) = on the basis of fitness function select \( P_n \) fittest individuals from \( P_{n-1} \); go to 4

3.2. Details of algorithm

Types of Encoding: Alike EAM [9] any type of coding can be used with this optimization algorithm. Binary coding solutions are represented as binary number and in real parameter coding this algorithm will be directly applied on the actual values of each solution. IEAM works well with both type of encoding. Following steps are used in binary version of IEAM

1) Population initialization: A fixed number of random initial solutions serve as the first generation population.
2) Creating next generation: Application of Improved adaptation, alteration and selection operator on current generation will generate next generation.

The function of each operator can be explained as follows:

2a) Improved Adaption operator: Adaption operator can both explore and exploit the problem search space. This operator will work as follows

\[ \text{Gi} \rightarrow \text{Gi} + 1 = (\alpha \cdot \text{Decoded value in decimal of binary version of Gi}) + F(\text{Xi}) / F_{avg} + \beta \{ (\text{Gi} - \text{Pi}) \} \mod (2^l) \]

Where \( \text{Gi} \) is the position vector of best particle, \( \text{Pi} \) is the position value of a particle that is updating its structure. For best solution following adaption operator will be used

\[ \text{Xi} \rightarrow \text{Xi} + 1 = (\alpha \cdot \text{Decoded value in decimal of binary version of Xi}) + F(\text{Xi}) / F_{avg} + \beta \{ (\text{Gi} - \text{Pi}) \} \mod (2^l) \]

Where \( F(\text{Pi}) \) is the fitness value of \( \text{Pi} \), α and β are the random numbers that are to be decided as per the requirement of problem. \( 2^l \) represents the total number of bits in an individual. \( F_{avg} \) is the average fitness value of current population.

2b) Alteration Operator: In early generation, this operator is responsible for exploit a particular region. In late generation, this operator will work as a explorer. Alteration is carried out according to the following pseudo code:
for each member in the population
for each bit in the member’s genome
invert the bit with a probability of $P_{alt}$.
Each new member so generated is considered for
selection for the next generation.

2) **Selection:** Best solutions equal to size of initial
population are to be selected from the combination
of next generation and previous generation to form
current generation.

3) **Generation step and evolution**
   Each generation step consists of generating new
   offspring by the formula described above, followed
   by alteration of existing population by $P_{alt}$ prob-
   ability and applying selection of best fit members
   generated. Figure 2 describes the working of this
   step.

4) This evolution process is repeated until either the
   solution with desired fitness is obtained or num-
   ber of generation exceeds the maximum number
   of generation. The detailed flowchart for the above
   stated Procedure is described below in the Fig. 3.

4. Result analysis

The performance of IEAM is compared with SAD,
PSO, IPOP-SEP-CMA-ES, AMALGAM IDEA and
BIPOP-CMA-ES algorithm. To check the efficiency of
improved version of Environmental Adaption Method
(IEM), this algorithm has been applied to standard
benchmark functions like Axis parallel hyper-ellipsoid,
Sphere, Rastrigrin, Griewank, Schwefel and
Zakharov. Out of these functions Axis parallel hyper-
ellipsoid, Sphere and Quatric are uni-modal functions
and Rastrigrin, Griewank, Schwefel and Zakharov are
Multi-modal functions. All benchmark functions were
analyzed for different dimensions $D = 10, 20, 30$, and
in the first experiment size of initial population for each
algorithm was taken as 50 which was extended to 100
in later experiments, each experiment were repeated for
five times and average values are shown in tables and
number of generation is $G = 1000$. The value of SAD
need not be initialized as they will be updated auto-
matically. Each algorithm is executed five times; the
proposed results are the average of the best minimum
values obtained and the average of the standard devia-
tion after all the five runs of the algorithm. A statistical
data containing minimum and standard deviation has
been calculated for each experiment. The detailed result
analysis for each function is given below: Table 1 shows
the different benchmark functions used in this paper.

A) Uni-modal functions.

1) **Axis parallel hyper-ellipsoid function:**
From Table 2 and Fig. 4, it is clear that the per-
formance of IEAM is far better than SAD and
PSO-TVAC on Axis parallel hyper-ellipsoid function.
Its performance is little better than IPOP-SEP-CMA-ES
and AMALGAM IDEA and almost equal to BIPOP-
CMA-ES. When the population size is increased to
100, it performs better than BIPOP-CMA-ES on higher
dimensions.
Table 1

<table>
<thead>
<tr>
<th>Benchmark Function</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axiz parallel hyper-ellipsoid</td>
<td>( f(x) = \sum_{i=1}^{n-1} x_i^2 )</td>
</tr>
<tr>
<td>Sphere</td>
<td>( f(x) = \sum_{i=1}^{n} x_i^2 )</td>
</tr>
<tr>
<td>Rastrigrin</td>
<td>( f(x) = \sum_{i=1}^{n} [x_i^2 - 10 \cos(2\pi x_i) + 10] )</td>
</tr>
<tr>
<td>Griewank</td>
<td>( f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \frac{1}{\sqrt{n}} \cos \left( \frac{x_i}{\sqrt{n}} \right) + 1 )</td>
</tr>
<tr>
<td>Schwefel</td>
<td>( f(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{</td>
</tr>
<tr>
<td>Zakharov</td>
<td>( f(x) = \sum_{i=1}^{n} x_i^2 + \left( \sum_{i=1}^{n} x_i \right)^2 + \left( \sum_{i=1}^{n} x_i \right)^4 )</td>
</tr>
</tbody>
</table>

Table 2

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE (MEAN, STDDEV)</th>
<th>PSO-TV AC (MEAN, STDDEV)</th>
<th>IPOP-SEP-CMA-ES (MEAN, STDDEV)</th>
<th>AMALGAM IDEA (MEAN, STDDEV)</th>
<th>BIPOP-CMA-ES (MEAN, STDDEV)</th>
<th>IEAM (MEAN, STDDEV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>1000</td>
<td>10</td>
<td>22.934678, 2.2987444</td>
<td>22.748911, 1.87299765</td>
<td>20.993261, 1.559346</td>
<td>20.97896, 1.530143</td>
<td>20.834189, 1.535624</td>
<td>20.957634, 1.536241</td>
</tr>
<tr>
<td>20</td>
<td>27.87665</td>
<td>25.744665</td>
<td>22.155302</td>
<td>21.984342, 2.436158</td>
<td>21.358923, 2.570836</td>
<td>2.994761, 2.570836</td>
<td>2.9063417, 2.5478403</td>
<td>2.5973457, 2.4937545</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>10</td>
<td>15.80040, 1.987444</td>
<td>12.422762, 1.7251827</td>
<td>11.653212, 1.5677412</td>
<td>11.041523, 1.5014588</td>
<td>10.984423, 1.4562412</td>
<td>10.958637, 1.3962412</td>
</tr>
</tbody>
</table>

Fig. 4. Comparison among SADE, PSO-TV AC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES and IEAM on Axis Parallel Hyper-Ellipsoid for Min value (population size 100).

2) Sphere: Sphere function is a convex uni-modal function. Although for sphere function a big difference was not obtained for smaller and higher dimensions, BIPOP-CMA-ES performs slightly better than IEAM on higher dimension of 30. Surprisingly for an intermediate dimension size of 20, for population size of 50, SADE and IEAM perform better than all other algorithms. Table 3 and Fig. 5 confirm our analysis.

3) Quartic function: Quartic function is also a uni-modal function of degree four. BIPOP-CMA-ES outperforms IEAM on all dimensions for both 50 and 100.
Table 3
Statistical analysis of SADE, PSO-TV AC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES, IEAM for Sphere function

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE</th>
<th>PSO-TV AC</th>
<th>IPOP-SEP-CMA-ES</th>
<th>AMALGAM IDEA</th>
<th>BIPOP-CMA-ES</th>
<th>IEAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>100</td>
<td>10</td>
<td>0.013378</td>
<td>0.013072</td>
<td>0.0156458</td>
<td>0.0144684</td>
<td>0.012664</td>
<td>0.0111231</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.000905</td>
<td>0.001240</td>
<td>0.003215</td>
<td>0.003164</td>
<td>0.001876</td>
<td>0.0010022</td>
</tr>
<tr>
<td>20</td>
<td>0.012201</td>
<td>0.030501</td>
<td>0.0044231</td>
<td>0.013246</td>
<td>0.003185</td>
<td>0.0018764</td>
<td>0.0011864</td>
<td>0.0011864</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.001438</td>
<td>0.007708</td>
<td>0.0023634</td>
<td>0.0031459</td>
<td>0.0014687</td>
<td>0.0014687</td>
</tr>
<tr>
<td>30</td>
<td>0.011783</td>
<td>0.011608</td>
<td>0.0111284</td>
<td>0.011865</td>
<td>0.0014566</td>
<td>0.0011867</td>
<td>0.0011867</td>
<td>0.0011867</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>10</td>
<td>0.012879</td>
<td>0.012870</td>
<td>0.013607</td>
<td>0.015309</td>
<td>0.0113655</td>
<td>0.010022</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.000936</td>
<td>0.000871</td>
<td>0.0007561</td>
<td>0.0009934</td>
<td>0.0009214</td>
<td>0.000902</td>
</tr>
</tbody>
</table>

Bold value represents standard deviation (std).

Table 4
Statistical analysis of SADE, PSO-TV AC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES, IEAM for Quartic function

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE</th>
<th>PSO-TV AC</th>
<th>IPOP-SEP-CMA-ES</th>
<th>AMALGAM IDEA</th>
<th>BIPOP-CMA-ES</th>
<th>IEAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>100</td>
<td>10</td>
<td>0.001661722</td>
<td>0.0097766</td>
<td>0.0007862</td>
<td>0.007910</td>
<td>0.002609</td>
<td>0.006654</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.0045642</td>
<td>0.0032453</td>
<td>0.0003975</td>
<td>0.0004065</td>
<td>0.0002536</td>
<td>0.0002002</td>
</tr>
<tr>
<td>20</td>
<td>0.0172334</td>
<td>0.0134444</td>
<td>0.0126969</td>
<td>0.0175163</td>
<td>0.003319</td>
<td>0.00010725</td>
<td>0.00010725</td>
<td>0.00010725</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.0047742</td>
<td>0.0003328</td>
<td>0.0003618</td>
<td>0.0003543</td>
<td>0.0002095</td>
<td>0.0002153</td>
</tr>
<tr>
<td>30</td>
<td>0.0228175</td>
<td>0.0210898</td>
<td>0.012812</td>
<td>0.026817</td>
<td>0.0204586</td>
<td>0.00005051</td>
<td>0.00005051</td>
<td>0.00005051</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.0049918</td>
<td>0.0039933</td>
<td>0.00036211</td>
<td>0.00031302</td>
<td>0.00015984</td>
<td>0.0002429</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>10</td>
<td>0.0057780</td>
<td>0.0067968</td>
<td>0.0003083</td>
<td>0.0079111</td>
<td>0.0006324</td>
<td>0.0006383</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.0006425</td>
<td>0.0002538</td>
<td>0.00020361</td>
<td>0.0001906</td>
<td>0.0000941</td>
<td>0.0001001</td>
</tr>
<tr>
<td>20</td>
<td>0.0102312</td>
<td>0.0097723</td>
<td>0.0008122</td>
<td>0.006975</td>
<td>0.007263</td>
<td>0.0072543</td>
<td>0.0072543</td>
<td>0.0072543</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.001042</td>
<td>0.0003328</td>
<td>0.00002107</td>
<td>0.0001269</td>
<td>0.00013524</td>
<td>0.0001253</td>
</tr>
<tr>
<td>30</td>
<td>0.0122465</td>
<td>0.0090863</td>
<td>0.014094</td>
<td>0.0173103</td>
<td>0.0043325</td>
<td>0.00097562</td>
<td>0.00097562</td>
<td>0.0009426</td>
</tr>
</tbody>
</table>

Bold value represents standard deviation (std).

100 population sizes. As shown in Table 4 and Fig. 6 IEAM still gives much better results than SADE, PSO-TVC and IPOP-SEP-CMA-ES.

B) Multi-modal functions.

1) Rastrigrin function: Rastrigrin function is one of the simple multi-modal functions with cosine modulation. It may have many local minima with a unique global minimum. For this Multi-modal function we get a series of good results in which IEAM is better than its other counter parts. The minimum values are always better than SADE, PSO-TV AC, IPOP-SEP-CMA-ES, AMALGAM IDEA, and BIPOP-CMA-ES. Table 5 and Fig. 7 shows result analysis.

2) Griewank Function: Griewank is also a multi-modal function with cosine modulation. It is clear from the Table 6 and Fig. 8 that the gap in performance of IEAM when compared with all other algorithm is huge. It clearly outmatches all the algorithms by a large margin.
Table 5

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE</th>
<th>PSO-TV AC</th>
<th>IPOP-SEP-CMA-ES</th>
<th>AMALGAM IDEA</th>
<th>BIPOP-CMA-ES</th>
<th>IEAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>100</td>
<td>10</td>
<td>2.813423</td>
<td>2.819838</td>
<td>2.460263</td>
<td>2.766625</td>
<td>1.846521</td>
<td>1.725161</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>20</td>
<td>15.773627</td>
<td>11.486118</td>
<td>11.939921</td>
<td>11.290994</td>
<td>11.492316</td>
<td>10.478952</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>30</td>
<td>37.473776</td>
<td>30.483194</td>
<td>32.976456</td>
<td>30.946132</td>
<td>29.946264</td>
<td>28.351672</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>10</td>
<td>1.597393</td>
<td>1.526443</td>
<td>1.123864</td>
<td>1.031264</td>
<td>1.063413</td>
<td>1.000209</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>20</td>
<td>1.897956</td>
<td>1.433130</td>
<td>1.509843</td>
<td>1.497561</td>
<td>1.497470</td>
<td>1.423344</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>30</td>
<td>2.921780</td>
<td>1.897269</td>
<td>1.826974</td>
<td>1.774632</td>
<td>1.785420</td>
<td>1.631562</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>10</td>
<td>0.329307</td>
<td>0.209212</td>
<td>0.09020361</td>
<td>0.081996</td>
<td>0.080941</td>
<td>0.020157</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>20</td>
<td>15.965878</td>
<td>11.909010</td>
<td>11.693962</td>
<td>11.297462</td>
<td>10.946290</td>
<td>10.478952</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>30</td>
<td>36.973841</td>
<td>30.725416</td>
<td>30.870319</td>
<td>29.849531</td>
<td>28.369461</td>
<td>28.351672</td>
</tr>
</tbody>
</table>

Bold value represents standard deviation (std).

Table 6

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE</th>
<th>PSO-TV AC</th>
<th>IPOP-SEP-CMA-ES</th>
<th>AMALGAM IDEA</th>
<th>BIPOP-CMA-ES</th>
<th>IEAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>100</td>
<td>10</td>
<td>0.057967</td>
<td>0.058913</td>
<td>0.0369842</td>
<td>0.03197896</td>
<td>0.03974126</td>
<td>0.021387</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>20</td>
<td>0.031974</td>
<td>0.023736</td>
<td>0.029462</td>
<td>0.0219792</td>
<td>0.0279676</td>
<td>0.022100</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>30</td>
<td>0.044682</td>
<td>0.025964</td>
<td>0.0312966</td>
<td>0.0297967</td>
<td>0.0284913</td>
<td>0.021322</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>10</td>
<td>0.013739</td>
<td>0.011147</td>
<td>0.001976</td>
<td>0.0014632</td>
<td>0.00161324</td>
<td>0.001035</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>20</td>
<td>0.012475</td>
<td>0.009796</td>
<td>0.046384</td>
<td>0.0390996</td>
<td>0.021943</td>
<td>0.010331</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>30</td>
<td>0.004266</td>
<td>0.023390</td>
<td>0.0026453</td>
<td>0.0028976</td>
<td>0.0025973</td>
<td>0.003247</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>10</td>
<td>0.021481</td>
<td>0.015396</td>
<td>0.016432</td>
<td>0.0104632</td>
<td>0.003792</td>
<td>0.001458</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>20</td>
<td>0.002026</td>
<td>0.001124</td>
<td>0.001169</td>
<td>0.0010199</td>
<td>0.002627</td>
<td>0.001354</td>
</tr>
</tbody>
</table>

Bold value represents standard deviation (std).

Fig. 7. Comparison among SADE, PSO-TVAC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES and IEAM on Rastrigin for Min value (population size 100).

Fig. 8. Comparison among SADE, PSO-TVAC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES and IEAM on Griewank for Min value (population size 100).

3) Schwefel: Schwefel is one of the most tough benchmark functions. It is a multi-modal function with sine modulation. To be noted; we execute a two test cases for schwefel one for two-dimensional 50 population size and other for two-dimensional 100 population size. From Table 7 and Fig. 9, an idea about the performance of IEAM can be taken. For lower population we get very good result for IEAM and for higher population,
Table 7: Statistical analysis of SADE, PSO-TVAC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES, IEAM for Schwefel function

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE</th>
<th>PSO-TVAC</th>
<th>IPPOP-SEP-CMA-ES</th>
<th>AMALGAM IDEA</th>
<th>BIPOP-CMA-ES</th>
<th>IEAM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>1000</td>
<td>0.004412</td>
<td>0.003010</td>
<td>0.004361</td>
<td>0.003713</td>
<td>0.003165</td>
<td>0.003327</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>1000</td>
<td>0.004474</td>
<td>0.003010</td>
<td>0.003461</td>
<td>0.003713</td>
<td>0.003165</td>
<td>0.003227</td>
</tr>
</tbody>
</table>

Bold value represents standard deviation (std).

Table 7b: Statistical analysis of SADE, PSO-TVAC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES, IEAM for Zakharov function

<table>
<thead>
<tr>
<th>Population</th>
<th>Iterations</th>
<th>Dimension</th>
<th>SADE</th>
<th>PSO</th>
<th>IPPOP-SEP-CMA-ES</th>
<th>AMALGAM IDEA</th>
<th>BIPOP-CMA-ES</th>
<th>IEAM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
<td>(MEAN STD. DEV)</td>
</tr>
<tr>
<td>50</td>
<td>1000</td>
<td>10</td>
<td>2.784612</td>
<td>2.456022</td>
<td>2.803408</td>
<td>2.783421</td>
<td>2.973458</td>
<td>2.695367</td>
</tr>
<tr>
<td>20</td>
<td>7</td>
<td>10</td>
<td>5.098371</td>
<td>5.688962</td>
<td>5.01762</td>
<td>5.096421</td>
<td>4.62384</td>
<td>4.33416</td>
</tr>
</tbody>
</table>

However for Zakharov also IEAM is better than its counterparts on lower dimensions.

Fig. 9: Comparison among SADE, PSO-TVAC, IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES and IEAM on Schwefel function for Min value (Population size 100).

its performance remains unchanged and is far better than others.

4) Zakharov: Zakharov is also complex multi-modal function. We evaluate zakharov on three dimensions (10, 20 and 30) and population size 50, evaluation is shown in Table 7. The function minimum values have a very little difference say, for 30 dimension population size t the minimum value were 2.043, 2.006, 2.0058, 2.0035, 2.009 and 1.999 for SADE,PSO-TVAC,IPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES and IEAM respectively.

However for zakharov also IEAM is better than its counterparts on lower dimensions.

The mean and stddev is calculated for each algorithm for 25 runs. Wilcoxon’s rank sum test is performed between IEAM and SADE, PSO, IPPOP-SEP-CMA-ES, AMALGAM IDEA, BIPOP-CMA-ES at 0.05 significance level. “-“ denotes the performance of an algorithm worse than, “+”, better than, and “≈”, similar to the performance of the proposed IEAM.

5. Real life application of IEAM

Although IEAM has been established as an optimization algorithm yet its performances are not checked on a real life optimization problem. In this section, a real life application of IEAM is discussed. This problem is related to software testing and its main objective is to generate test cases for white box testing. Software testing is the most commonly used technique for improving the quality of Software. It is a labor intensive process, and typically accounts for about half of the total cost of software development and maintenance [11]. To gain confidence that software will work as per user expectations, this newly developed software must be carefully exercised by the fault revealing test cases. Searching appropriate test data is not an easy task. It is very difficult. This is an optimization problem where a set of test cases are to be searched from...
Table 8

<table>
<thead>
<tr>
<th>PERFORMANCE</th>
<th>SADIE</th>
<th>PSO</th>
<th>IPOP-SEP</th>
<th>CMA-ES</th>
<th>AMALGAM</th>
<th>IDEA</th>
<th>BIPOP-CMA-ES</th>
</tr>
</thead>
<tbody>
<tr>
<td>+3</td>
<td>20</td>
<td>19</td>
<td>14</td>
<td>13</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+5</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+8</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 10. Test case generation framework.

5.1. Test case generation using IEAM for white box testing

In this section, IEAM has been used to generate new test cases whose branch coverage is better as compared to existing test cases. First of all, an initial set of test cases are generated randomly from the domain of input variables. To generate new test cases, IEAM needs an input population. A set of randomly generated test cases will work as an initial population for the IEAM. For test case generation, framework shown in Fig. 10 has been used. This framework has three modules as follows:

1) Starting module
2) Optimization Algorithm
3) Test case coverage tool

Starting module: This module contains all information that is required to start other modules. Following parameters are stored in this module.

1. Information required for starting IEAM: maximum number of generations, number of solutions in the population, type of coding.
2. Information required to generate new population: number of test data in each individual, type of input variable, value of population related parameters, for example in case of IEAM this will store the value of Favg (average fitness of the population).
3. Fitness function used to evaluate the fitness of individual. Information about test tool.

Optimization function used to evaluate the fitness of individual. Information about test tool.

Test case coverage module: This module implements the functionality of optimization algorithm. Its purpose is to generate initial population and new populations with the help of necessary operators.

Test case coverage module: This module implements the functionality of coverage tool. IEAM has been used as an optimization algorithm to implement this framework. The optimization algorithm IEAM is used to search appropriate test cases which have better branch coverage as compared to existing test cases. Here branch coverage of test case is taken as fitness function. Gcov coverage tool [a coverage tool available as free open source software at [http://gcc.gnu.org/onlinedocs/gcc/Gcov.html]] is used to identify the percentage of branch covered by each test case.

The process of test case generation with IEAM can be explained by following pseudo code.
The proposed algorithms have been applied to generate test cases for white box testing. The goal of the proposed algorithm is to generate test cases that have higher branch coverage capability as compared to initial test cases. To gauge the performance of our proposed algorithms EAM on four programs
1) Triangle Program 2) Max- Find Program 3) F-closure 4) First fit algorithm,
(a) Apply Gcov and Calculate the percentage of branches covered by each test case, this will work as fitness of each solution.
(b) Calculate the value of average fitness, by taking the average of fitness of solutions.
(c) Apply improved adaption operator to each test case and create intermediate population.
(d) Apply alteration operator to some solutions of intermediate population.
(e) Merge initial population with intermediate population and sort this population according to branch coverage.
(f) Select N best solution and generate new population.
(g) Number of generations = number of generations + 1
3 end while.
4 print: Final test suite.
We have compared the performance of EAM with Genetic Algorithms and EAM. Total twenty iterations have been used and it is seen that EAM and IEAM have a very fast convergence rate. The fitness values for Triangle Program are as shown in the Table 9.

6. Conclusion
To improve the performance of EAM algorithm, an improved version of EAM has been suggested. Result analysis shows that Improved Environmental Adaptation Method (IEAM) is better than all other optimization methods. Moreover this paper establishes the application of IEAM in generating the test suites for white box testing that achieve maximum branch coverage. Further following has been established

• A new framework has been designed to automate the process of test case generation.
• With the help of EAM and IEAM the average branch coverage of initial test suite has been increased.
• The performance of EAM, IEAM and GA has been tested on standard benchmark programs and

<table>
<thead>
<tr>
<th>Generation</th>
<th>Triangle problem</th>
<th>Max-Find program</th>
<th>F-closure program</th>
<th>First-fit algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GA</td>
<td>EAM</td>
<td>GA</td>
<td>EAM</td>
</tr>
<tr>
<td>1</td>
<td>68.5</td>
<td>55</td>
<td>67.5</td>
<td>68.75</td>
</tr>
<tr>
<td>2</td>
<td>68.93</td>
<td>72.5</td>
<td>69.68</td>
<td>68.75</td>
</tr>
<tr>
<td>3</td>
<td>71.93</td>
<td>72.5</td>
<td>64.93</td>
<td>75</td>
</tr>
<tr>
<td>4</td>
<td>53.93</td>
<td>72.5</td>
<td>64.93</td>
<td>75</td>
</tr>
<tr>
<td>5</td>
<td>53.43</td>
<td>72.5</td>
<td>68.68</td>
<td>75</td>
</tr>
<tr>
<td>6</td>
<td>64.43</td>
<td>72.5</td>
<td>67.43</td>
<td>75</td>
</tr>
<tr>
<td>7</td>
<td>60.93</td>
<td>72.5</td>
<td>69.93</td>
<td>75</td>
</tr>
<tr>
<td>8</td>
<td>63.43</td>
<td>72.5</td>
<td>71.18</td>
<td>75</td>
</tr>
<tr>
<td>9</td>
<td>59.43</td>
<td>72.5</td>
<td>64.93</td>
<td>75</td>
</tr>
<tr>
<td>10</td>
<td>60.93</td>
<td>72.5</td>
<td>68.68</td>
<td>75</td>
</tr>
<tr>
<td>11</td>
<td>57.93</td>
<td>72.5</td>
<td>67.43</td>
<td>75</td>
</tr>
<tr>
<td>12</td>
<td>55.93</td>
<td>72.5</td>
<td>62.43</td>
<td>75</td>
</tr>
<tr>
<td>13</td>
<td>59.43</td>
<td>72.5</td>
<td>69.93</td>
<td>75</td>
</tr>
<tr>
<td>14</td>
<td>54.93</td>
<td>72.5</td>
<td>68.68</td>
<td>75</td>
</tr>
<tr>
<td>15</td>
<td>60.43</td>
<td>72.5</td>
<td>66.18</td>
<td>75</td>
</tr>
<tr>
<td>16</td>
<td>59.93</td>
<td>72.5</td>
<td>66.18</td>
<td>75</td>
</tr>
<tr>
<td>17</td>
<td>58.43</td>
<td>72.5</td>
<td>66.18</td>
<td>75</td>
</tr>
<tr>
<td>18</td>
<td>61.93</td>
<td>72.5</td>
<td>67.43</td>
<td>75</td>
</tr>
<tr>
<td>19</td>
<td>59.93</td>
<td>72.5</td>
<td>68.68</td>
<td>75</td>
</tr>
<tr>
<td>20</td>
<td>65.5</td>
<td>72.5</td>
<td>69.93</td>
<td>75</td>
</tr>
</tbody>
</table>
IAAM algorithm is found to be the most suitable algorithm for this application.

- Even for complex programs, where conditions or predicates in the software restrict the input domain, performance of IEM and EAM was very good as compared to GA algorithm.

References


