International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-5 Issue-2, May 2015

Hybrid Local Search Based Genetic Algorithm and its Practical Application

Mustafa Tunay, Rahib H. Abiyev

Abstract:* *This paper presents an intense hybrid search method that uses Genetic Algorithms (GAs) and local search procedure for global optimization. The Genetic Algorithms (GAs) comprise a selection process, a crossover process and a mutation processes and local search procedure that uses Powell's method for updating the parameters of the objective functions. The performance of the designed algorithm is tested on specific benchmarking functions namely; Rastrigin function, Rosenbrock function, Schwefel's function 2.22, Schwefel's function 2.21 and Sphere's function. The computational results have demonstrated that the performance of Genetic Algorithms with Powell's Method is much improved specific benchmarking functions. The use of a hybrid search method approach allows it to speed up the learning of the system with faster convergence rates. The Genetic Algorithm with Local Search Procedure (GALSP) is applied for soling exam timetabling problem. The GALSP seems to be a promising approach and is comparable to specialized algorithm for solving a set of global optimization problems. The algorithms of these processes have been designed and presented in the paper.

Index Terms:* *Genetic algorithms, local search procedure, evolutionary theory, search methods.

I. INTRODUCTION

Optimization is the problem of making decisions to maximize or minimize an objective in the presence of complicating constraints. Nowadays, optimization techniques are widely used in areas of industrial operations, computer science, business and financial management, engineering design and control, and artificial intelligence to mention just a few. Optimization can bring efficiency throughout society wherever resources are constrained. There are many methods can be used in the process of taking a real world problem and transforming it into a formulation that can then be solved by the methods we have developed.

Therefore, there are many alternative methods to find the best solution for a real world problem. Many Multi-objective Evolutionary Algorithms [5,17] such as ant colony algorithms [23], particle swarm optimization (PSO) algorithms [9,12], hypercube optimization algorithm [2], artificial bee algorithm, hypercube optimization algorithm, [11,24] and genetic algorithms (GAs) [6,21] have developed many real evolution strategies for real world problems [8,19]. GAs are widely used to solve different real world problems, such as for evolutionary natural selection and genetic ideas, different engineering problems and for the solution of portfolio optimization problems using GA [1,13].

In [7,15,23] GAs and PSO are applied for optimization of power flow.

Evolutionary algorithms have been successfully applied in various fields of optimization to solve many global optimization problems.

There have been many studies on finding new methods for optimization problems. These methods are used to solve many difficult global problems [14]. Many research methods try to find the best to global problems. In this paper, the main steps of Genetic Algorithms (GAs) comprising selection, crossover and mutation form a new source of inspiration for the creation of an algorithm. This algorithm integrated into the principles of Genetic Algorithms with Local Search Procedure (LSP) is proposed for global optimization. The simplest form of Genetic Algorithm with Local Search involves four types of operators: selection with the roulette wheel, crossover, mutation and Powell's Method for a more optimal solution.

Local Search Procedure [16, 22] is optimization methods that maintain as the current best solution, and explore the search by steps within its ranges. Local Search Procedure usually checks the current solution against a more optimal solution which can be used in the next iteration as the new current best solution. The usage of GA favours finding a global optimal solution and to avoids the local optimum problem. However, sometimes the learning process with GA becomes time consuming. In order to speed up system learning the combination of GA with the local search procedure is considered in this paper.

The paper is structured as follows. Sec. 2 includes the description of proposed the GALSP. The processes and flowcharts of the algorithm are given. Sec. 3 and Sec. 4 include application of the algorithm on test functions. Comparative results with the GAs existing methods have been given in Sec. 5. Sec.6 presents the application of the GALSP to exam timetabling problems, finally in Sec.7 the conclusions are presented.

II. GENETIC ALGORITHM WITH LOCAL SEARCH PROCEDURE

The GAs involves three types of operators: the *selection process*, the *crossover process*, and the *mutation process*. This designed algorithm integrates the principles of the GAs with Powell's Method for global optimization. In the following subsections, the descriptions of each process are presented in detail. The GALSP begins with an initial population (Ps) which is randomly generated to create initial points. The initial value of

Manuscript Received on April 2015.

Mustafa Tunay, Department of Computer and Instructional Technologies Education, Eastern Mediterranean University, Mersin 10 Turkey, Famagusta, North Cyprus.

Prof. Dr. Rahib H. Abiyev, Department of Computer Engineering, Near East University, Mersin 10 Turkey, Nicosia, North Cyprus.

X^0 within GALSP is determined according to the change interval of the test (objective) functions. The population size, cross-over rate, mutation rate, elite size, initial points, the lower and upper boundaries of the points are input parameters of the algorithm. In this paper, we are presented real-valued single-objective unconstrained functions and also solution of the exam timetabling problem. At the first stage, we try to find the minimum (or equivalently the maximum) of a scalar objective function $f_{(x)}$. The parameters of $X = (x_1, x_2, \dots x_m)$ is represented as a vector (or set of points), where m represents the number of dimensions of the problem. A problem is finding the values of parameters of X that will minimize the objective function f, where $X \subseteq R^m$ is a bounded set in R^m, therefore f is a mapping of an ***m***-dimensional fitness function. The use of the proposed GA with Powell's method speeds up the learning of the system and finds a point $X_{min} \in X$ such that $f_{(x)min}$ will have a global minimum on X; that is $\forall x \in X: f_{(x)min} \leq f_{(x)}$. The process of finding the smallest value of (X_{min}) will be done through learning of the X parameters by means of GALSP. The details regarding the visualization of the flow- chart of the GALSP (with Powell's Method) are illustrated in Figure 1.

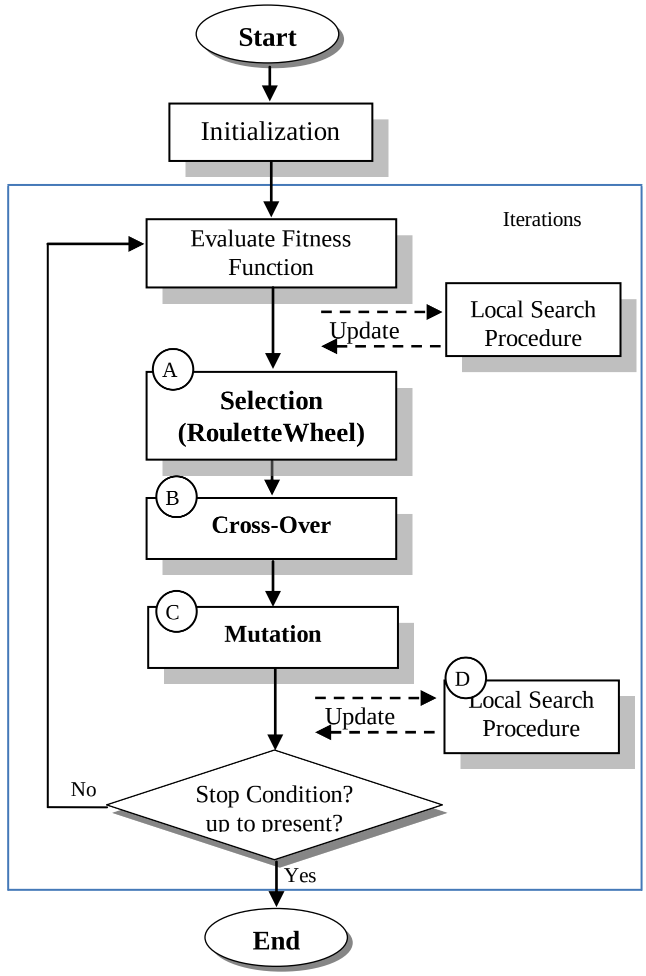

Figure 1. Flow chart of the GALSP. Here A is selection, B is cross-over, C is mutation processes and D is local search procedure.

As shown from the figure the GALSP includes four basic processes.

Step A (*selection process*)
Selection process is deployed to equate to survival of the fittest.

Step B (*cross-over process*)
Cross-over process is deployed to represents mating between populations.

Step C (*mutation process*)
Mutation process is deployed to introduce random modifications.

Step D (*local search procedure*)
Local search procedure is deployed explore the search by steps within the current range for a more optimal solution.

The GALSP starts by generating an initial population randomly. This population members are evaluated using a fitness function and applying Powell's Method for the best current solution and then determined best population member is saved. The selection is performed according to the fitness value. Then a roulette selection is applied to all population members. The better population member has more chances to be selected with a roulette wheel. The member that has more fitness is selected to the next generation. The values of fitness are then used to determine if the population is eliminated or preserved by applying the Genetic algorithm for a more optimal solution and compared with the current best population, far following crossover, mutation and local search procedure. With the principle of the strongest, the best adaptive population is maintained, and the less adaptive are replaced the generation of a new ones. This replaces the old ones. The whole process is repeated until the specific termination conditions are met. The convergence of the GALSP is shown below in Figure 2. The pseudo code regarding the GALSP (with Powell's Method) can be formed as:

`Algorithm`

```
Begin
  population:=0
    Initialize population Ps
     done:=  false
While not done Do
Calc. the fitness of each individual population
 Powell's Method and save the best so far
    Population: = population+1
      Selection (population) from (population -1)
        Crossover Cp
         Mutation  Mp
      Apply Powell's Method (x^0)
    done: = Optimization criteria met?
End While
Output `best solution`
End
```

Figure 2. Designed the GALSP

2.1 Selection Process

The *Selection process* determines which solutions are preserved and allowed to reproduce and those which are discarded. The primary selection operator aims to highlight the

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

good solutions and eliminate bad solutions in a population while keeping the size of the population constant.

Through the wheels, the population is selected according to fitness values. The best population is more likely to be selected. Thus, the populations have been selected based on their physical condition and so we excepted that selected population is among the strongest in the population and so we excepted that selected population will gradually increase in the average fitness, used in the next iteration, as the current solution. In this study, the average fitness of the population for i^{th} in roulette wheel selection is calculated as follows:

$$p_i(x) = \frac{f_i}{\sum_{j=1}^{N} f_i} \quad (1)$$

where N is the number of individuals in the population and f_j is the fitness of individual j in the population.

2.2 Cross-Over Process

The *crossover process* is a genetic operator that combines two parents from the population and produces a new population. In the paper, we select uniform crossover process, in which the individual bits of the string are compared between the two populations. The most popular crossover process selects two chains of solutions at random from the population, and some of the strings are exchanged between these chains. The selection point is chosen at random. The bits are exchanged with a fixed probability, crossover probability is usually 0.60.

A probability of crossover is also set up to give freedom to an individual chain solution to determine if the solution will cross or not. The details regarding the visualization of multipoint crossover process are given in Figure 3.

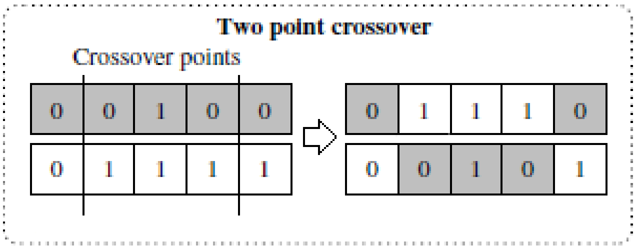

Figure 3. Visualization of two point crossover process

2.3 Mutation Process

This operator returns a random part of the bits in a population. The mutation may occur at each position in a bit string with a specific probability. While the crossing process has primary responsibility for finding the optimal solution, the mutation process is also used for this purpose. The mutation operator changes a 1 to 0 or 0 to 1, with a specific probability of mutation. The probability of mutation is generally kept low for fixed convergence. A high value of the mutation probability leads to a random search technique. The details regarding the visualization of mutation process are given in Figure 4.

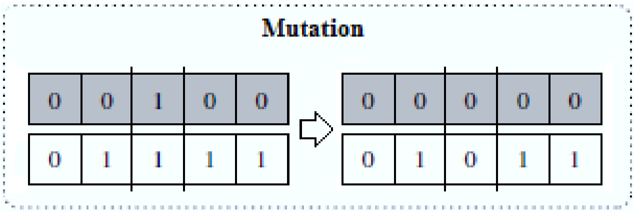

Figure 4. Visualization of mutation process

2.4 Local Search Procedure (Powell Method)

As mentioned, the GA is an effective research technique that can find the global optimum point of multimodal functions. But many times the GA-based search becomes time consuming. An effective approach to solve this problem is the use of the technique of local search with the combination of evolutionary optimization methods especially with GA. This speeds up the search process. In this paper, Powell's Method is combined with the GA search technique for finding the local optimum in the space of multimodal solutions.

The method minimizes the search function by using a bidirectional along each search vector p^i. The x^i point is determined as the point where the minimum of the function f occurs along the vector p^i

In Powell's method for the given input vectors $\boldsymbol{p}$ and $\boldsymbol{n}$, and the function f, it is necessary to find the scalar λ that minimizes the function $f(\boldsymbol{p}+\lambda^{\boldsymbol{n}})$. Then the replacement of $\boldsymbol{p}$ by $\boldsymbol{p}+\lambda^{\boldsymbol{n}}$ and $\boldsymbol{n}$ by $\lambda^{\boldsymbol{n}}$ takes place. The Powell's conjugate algorithm is given below:

Define the p^i set of independent vectors in x. x^0 is starting point;

```
While True
    For i = 1,…,n  do
        Replace x^i = x^(i-1) + λ^i p^i, where λ^i minimizes
          f (x^(i-1) + λ^i p^i)
      For i = 1,…,n - 1 do
          p^i = p^(i+1)
      end
        p^n = x^n - x^0
        Find λ^n that minimizes f (p^n + λ^n (x^n - x^0))
      x^0 = x^0 + λ^n (x^n - x^0)
   end
 end
```

The search results in the two vectors (x^1 , x^2) being generated by one-dimensional search in the same direction from different points, and so this is then used as the directions of the next search. Here is a sample session to find the optimum for the following function:

$$f(y) = 10 + (x_1 - 2)^2 + (x_2 + 5)^2 \quad (2)$$

The above function searches for the optimum two variables has the initial guess of [0 0] and step tolerance vector of [1e-5, 1e-5]. The search employs a maximum of 1000 iterations and a function tolerance of 1e-7.

III. TEST FUNCTIONS

This section presents the performance of GALSP tested on specific benchmark functions

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

which are widely used in the literature [7,20]. The details regarding the benchmark functions are as follows:

3.1 Rastrigin Function

The *Rastrigin function* is described as follows:

$$f_1(x) = 10D + \sum_{i=1}^{D} (x_i^{\,2} - 10\cos(2\pi x_i)) \quad (3)$$

where D is a number of dimension and $x_i = (x_1, x_2\ \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in interval of $-5.12 \le x_i \le 5.12$, $i = (1, 2\ \ldots., D)$. The global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0\ \ldots., 0)$.

3.2 Rosenbrock Function

The *Rosenbrock function* is described as follows:

$$f_2(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - (x_i)^2)^2 + (x_i - 1)^2] \quad (4)$$

where $D \ge 2$ is a number of dimension and $x_i = (x_1, x_2\ \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in interval of $-2.048 \le x_i \le 2.048$, $i = (1, 2\ \ldots., D)$. The global minimum $f(x) = 0$ is obtainable for $x_i = (1, 1\ \ldots., 1)$.

3.3 Schwefel's Function 2.21

The *Schwefel function* 2.21 is described as follows:

$$f_3(x) = \max_i (|x_i|, 1 \le i \le n) \quad (5)$$

where D is a number of dimension and $x_i = (x_1, x_2\ \ldots, x_D)$ is a dimensional row vector. This test area is usually evaluated to hypercube for $-10 \le x_i \le 10$ $i = (1, 2\ \ldots., D)$ and global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0\ \ldots., 0)$.

3.4 Schwefel Function 2.22

The *Schwefel function* 2.22 is described as follows:

$$f_4(x) = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} |x_i| \quad (6)$$

where D is a number of dimension and $x_i = (x_1, x_2\ \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in interval of $-10 \le x_i \le 10$ $i = (1, 2\ \ldots., D)$. The global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0\ \ldots., 0)$.

3.5 Sphere Function

The *Sphere function* is described as follows:

$$f_5(x) = \sum_{i=1}^{D} x_i^{\,2} \quad (7)$$

where D is a number of dimension and $x_i = (x_1, x_2\ \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in interval of $-5.12 \le x_i \le 5.12$ $i = (1, 2\ \ldots., D)$. The global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0\ \ldots., 0)$.

IV. Experimental Results and Discussion

The performance of GALSP is tested on the five benchmark functions that are described in subsection 3 and is implemented in Matlab/Simulink. The following the GALSP schemes were used: crossover with the rate of 0.60, mutation with the rate of 0.1, and local search procedure are employed. The GALSP searches the optimum of the function (2) having two variables that has the initial guess of [0 0] and step tolerance vector of [1e-5, 1e-5]. The search employs a maximum of 1000 iterations and a function tolerance of 1e-7 (tolF). The benchmark functions are evaluated by considering cases in which the problem population size is also set to 100. We have presented the success rate and the average number of the specific benchmark function evaluations equal to 30 runs. The details regarding the benchmark functions are given in Tables 1 - 4, respectively.

4.1 Rastrigin Function

The *Rastrigin function* is non-convex, multimodal and additively separable function. This test function produces several local minima. It is highly non-linear multimodal, but locations of the minima are regularly distributed. The finding of the minimum value of this test function is a fairly difficult problem for genetic algorithms due to the large search space and large number of local minima. However, the global minimum of this test function was obtained at (0, 0,…,0) points with 0.00e+00 accuracy error. The best and average fitness values of 30 runs of the GALSP for this test function are shown in Table 1.

Table 1: The performance of the GALSP for Rastrigin

Rastrigin Function				
	Iterations			
	1	**25**	**50**	**100**
Best	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Average	0.00e+00	0.00e+00	0.00e+00	0.00e+00

4.2 Rosenbrock Function

The *Rosenbrock function* is non-convex, unimodal and non-separable. The global minimum is inside narrow, parabolic valley, and through this valley it is difficult to find the minimum of the function. The global minimum of the test function was obtained using GALSP with a much better convergence. The minimum value of Rosenbrock function was obtained as 1.49e-10 at the point (1,1…,1). In general the GALSP yielded a better result. The best and average fitness values of 30 runs of the GALSP for this test function are shown in Table 2.

Table 2: The performance of GALSP for Rosenbrock

Rosenbrock Function				
	Iterations			
	1	**25**	**50**	**100**
Best	5.55e-11	4.50e-11	3.87e-11	4.52e-11
Average	6.66e-10	3.96e-10	2.97e-10	1.49e-10

4.3 Schwefel 2.21 Function

The *Schwefel function* 2.21 is continuous, non-differentiable, scalable, separable and unimodal function. The global minimum of the test function was obtained as 9.75e-19 at the point (0,0,…,0). This algorithm is

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

Benchmark Functions				
	Rastrigin	Rosenbrock	Schwefel 2.22	Sphere
GALSP				
Best	0.00e+00	2.55e-11	2.27e–18	1.44e–41
Avg	0.00e+00	1.25e-10	2.27e–18	2.75e–40
GA				
Best	1.073e+04	1.12e+03	1.490e+ 03	3.44e+03
Avg	1.075e+04	1.12e+03	1.492e+03	3.45e+03

sensitive to the number of iterations and parameters. The GALSP finds optimal or near-optimal solutions with much well convergence. The best and average fitness values of 30 runs of the GALSP for this test function are shown in Table 3.

Table 3: The performances of GALPS for Schwefel 2.21

4.4 Schwefel 2.22 Function

The *Schwefel function* 2.22 is continuous, scalable, non-differentiable, symmetric, and unimodal function. This test function produces the effect of convexity because its landscape is non-convex. The GALSP is applied for finding

Schwefel Function 2.22				
	Iterations			
	1	**25**	**50**	**100**
Best	1.40e-06	2.27e-18	2.27e-18	2.27e-18
Average	1.55e-05	2.27e-18	2.27e-18	2.27e-18

the optimum of this function. The minimum of this test function was obtained as 2.27e-18 at the point (0,0,…,0). The GALSP approach allows to speed up the learning of the system and respectively to decrease training time of this function with faster convergence. This algorithm is also insensitive to the number of iterations and parameters of the function. The best and average fitness values of 30 runs of the GALSP for this test function are shown in Table 4.

Table 4: The performances of GALPS for Schwefel 2.22

4.5 Sphere Function

The *Sphere function* is continuous, convex, a typical unimodal and additively separable test function that can be scaled up to any number of variables. Finding the minimum of this test function is a fairly easy problem. The global minimum of this test function was obtained with much better convergence with the last number of iterations using the GALSP. The Sphere function was obtained as 3.05e-37. The best and average fitness values of 30 runs of the GALSP algorithm for this test function are shown in Table 5.

Table 5: The performances of GALPS for Sphere

Sphere Function				
	Iterations			
	1	**25**	**50**	**100**
Best	3.70e-40	9.26e-39	2.52e-38	1.83e-40
Average	3.60e-37	3.67e-37	3.66e-37	3.05e-37

V. COMPARISIONS

The performance of the proposed GALSP is compared with the Genetic algorithm on benchmarking functions given above. The performance results obtained using the both techniques are presented in Table 6.

Table 6: Comparison of the performances of GALPS and GA

Schwefel Function 2.21				
	Iterations			
	1	**25**	**50**	**100**
Best	9.75e-19	9.75e-19	9.75e-19	9.75e-19
Average	9.75e-19	9.75e-19	9.75e-19	9.75e-19

For the comparison purpose, the GA algorithm is modelled using above test functions. Using the GA and the GALSP methods we have summarized the best and average number of the fitness function (specific benchmark function) evaluations over 30 runs.

The comparison has been performed at the same initial condition. Both the algorithms were tested using a set of common parameters. The simulations have been performed for 1000 iterations and D = 1000 dimensions of the test functions. As evident from the Table 6, the results obtained using GALSP algorithm has obtained the global minimum 0.00e+00 for the Rastrigin function, 1.25e-10 for the Rosenbrock function, 2.27e-18 for the Schwefel function 2.22 and 2.75e-40 for the Sphere function. The experimental results were obtained with the GALSP have better results than the GA. The convergences with the GALSP were very fast. The comparative results of the algorithms demonstrate that the performance of GALSP is much better than well-known the GA.

VI. APPLICATION

The exam timetabling problem [10,18] is one of important and constrained global optimization problem used for the scheduling of exams in institutions of higher education. The scheduling of exams timetabling problem consists of allocating a number of exams to a finite number of periods with specific constraints related to the avoiding the overlapping of exams having students in common, satisfying room and time constraints, etc. The following general conditions used to formulate the constraints:

- No students should have to take two exams in adjoining periods.
- No students should have to take two exams in one day.

The constraints form the basis for a feasible scheduling of exams timetabling problem for each period. For a more detailed description of timetabling, the reader is referred [3,4].

The problem is to find the near optimal solution of the exam timetabling. The GALSP algorithm is applied to solve the examination timetabling problem. In the exam timetabling problem the **E** exams should be scheduled in the **P** periods. Three periods in a day

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

are used for timetabling. Any of the two exams may have a conflict with each other. This means that there may be a number of students enrolled for both exams. The technique can be formally specified by defining the following:

1. All exam should be scheduled once, and only once in the timetable.

$$\sum_{p=1}^{P+1} T_{ip} = 1 \quad (8)$$

where, T_{ip} is an exam i scheduled in period p.

2. Exams should be scheduled within the same period (No conflicting).

$$\sum_{i=1}^{E-1} \sum_{j=i}^{E} \sum_{p=1}^{P} T_{ip} T_{jp} C_{ij} = 0 \quad (9)$$

where C_{ij} is a number of students taking both exams i and j.

3. Total number of seats required for any period is not greater than the number of seats available (Maximum sitting students per periods).

$$\sum_{p=1}^{P+1} T_{ip} S_i \leq S \quad (10)$$

where S_i is student taking exam i.

The actual process used by the GALSP technique is as follows;

- REPEAT
  FOR (each period)
    FOR (each event scheduled in period (*Exam*))
      Schedule event in the valid period causing (*same period*)
        Clashes (this includes the original period)
      Try and schedule any unscheduled events
- UNTIL check whether or not improvement is possible

The following experimental data is taken from ftp://ftp.cs.nott.ac.uk/ttp/Data/ for modeling of timetabling problem. The GALSP technique has been tested on a range of real data with the exception of varying to represent the real life information that are shown in Table 7. The data used is represented by the following codes:

carf92 Carleton University (1992), Ottawa
kfu King Fahd University, Dharam
nott Nottingham University, UK

Table 7. Dataset from the Universities

Data	Periods	Exams	Students	Enrollments
carf92	36	543	18,419	55,522
kfu	21	461	5,349	25,118
nott	23	800	7,896	34,265

The performance of the GALSP technique is tested on all above data sets with sizes subset of 100. These test functions are more applicable for the experimental evaluations of methods used in global optimization problems. Each of these successes has been tested with all the heuristics given 30 runs each and the average given results. In Table 8 has been shown the results obtained when using the heuristic relevant.

Table 8. Results of Applying Dataset

Data	2nd Order Same Day	2nd Order Overnight	Max Students per periods
carf92	365	718	2,000
Kfu	255	871	1,955
Nott	112	325	1,550

The proposed technique was tested on a decomposing large real-world timetabling problems encompassing a wide range of dimensionality. Experiments are carried out on an actual dataset taken from the University of Nottingham, University of Carleton and University of King Fahd. The focus of the experiments demonstrates that the GALSP algorithm effect on the quality when has been started on already better solutions for solving a set of optimal results. The process, however, is a little more complicated when room allocation (maximum sitting students per periods) is a part of the problem.

VII. Conclusion

This paper proposes a new intense hybrid search method that was inspired by evolutionary theory and based on GA with Local Search Procedure (Powell's method) for global optimization. The use of GALSP search approach speeds up the learning of the system and decreases training time of the system for a wide range of dimensionality. The simulations have been carried out using benchmarking functions; such as *Rastrigin function*, *Rosenbrock function*, *Schwefel function 2.22*, *Schwefel's function 2.21* and *Sphere's function*. The simulation results show that the GALSP based optimization has obtained good performance (reflected in the best and average fitness) for solving a set of global optimization problems. The comparative results of the GALSP and the GA demonstrate that the performance of GALSP is an improvement upon other one global optimization techniques. The algorithm is also applied to the solution of a timetabling problem.

REFERENCES

1. Abiyev Rahib H. (2009). Fuzzy Wavelet Neural Network for Prediction of Electricity Consumption. AIEDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing , 23(2) 109-118.
2. Abiyev Rahib H., Tunay M. (2015). Optimization of High Dimensional Functions through Hypercube Evaluation, Computational Intelligence and Neuroscience, Volume 2015, 2015.
3. Burke E. K, Elliman DG, Weare RF. ''A Hybrid Genetic Algorithm for Highly Constrained Timetabling Problems,'' 6th International Conference on Genetic Algorithms (July 1995,Pittsburgh, USA), 15-19.
4. Burke E. K, Newall JP.''A multistafe Evolutionary Algorithm for Timetable Problem'' Trans. On Evolutionary Computation 1999, 3(1).
5. Eiben A. E, Smith JE. Introduction to evolutionary computing. Springer 2003, New York.
6. Goldberg D. (1989). Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading MA.
7. Hansen N, Auger A, Ros R, Finck S, Posik P. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009, in Proceedings of the 12th International Conference on Genetic and Evolutionary Computation Conference (GECCO'10), Companion Material, Portland, ed. by M. Pelikan, J. Branke (ACM, New York, 2010), pp. 1689–1696.

8. Hansen N, Kern S. Evaluating the CMA evolution strategy on multimodal test functions, in Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN VIII), Birmingham. Volume 3242 of Lecture Notes in Computer Science, ed. by X. Yao et al. (Springer, 2004), pp. 282–291.
9. Holland J. H. Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.
10. Kalayci C. B., Gungor A. A Genetic Algorithm Based Examination Timetabling Model Focusing on Student Success for the Case of the College of Engineering at Pamukkale University, Turkey. Gazi University Journal of Science, 25(1) (2012) 137-153.
11. Karaboga D., Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing , 687–697.
12. Kennedy J. (1997). The particle swarm: social adaptation of knowledge, Proc. IEEE International Conference on Evolutionary Computation (Indianapolis, Indiana), IEEE Service Center 1997, Piscataway, NJ, 303–308.
13. Kramer O., Ciaurri D. E, Koziel S. (2011). Derivative-free optimization, in Computational Optimization and Applications in Engineering and Industry, Studies in Computational Intelligence (Springer, New York, 2011). pp. 61–83.
14. Lee K. Y., El-Sharkawi MA. Modern heuristic optimization techniques: theory and applications to power systems. Wiley 2008, New York.
15. Michalewicz Z. (1996). Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag 1996, 3rd edition, New York.
16. Moscato P. Memetic algorithms: a short introduction. In: Corne, D., Dorigo, M., Glover, F. (eds.) New Ideas in Optimization, pp. 219–234. McGraw-Hill 1999, New York.
17. Mukhopadhyay D., Maulik U., Bandyopadhyay S., and Coello Coello, CA. A Survey of Multiobjective Evolutionary Algorithms for Data Mining: Part I. IEEE Trans. On Evolutionary Computation 2014, 18(1).
18. Naji-Azimi Z. Comparison of metaheuristic algorithms for examination timetabling problem. Applied Mathematics and Computation 2004, 16(1-2) 337-354.
19. Ostermeier A., Gawelczyk A, Hansen N. A derandomized approach to self-adaptation of evolution strategies. Evol. Comput. 1994, 2(4), 369–380.
20. Pohlheim H. (December 2006). Genetic and Evolutionary Algorithm. Retrieved from website: http://www.Geatbx.com.
21. Rothlauf F. Representations for genetic and evolutionary algorithms. Springer 2006, Berlin.
22. Sudholt D. On the analysis of the (1+1) memetic algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2006), pp. 493–500. ACM Press 2006, New York
23. Vaisakh K, Srinivas LR, Meah K. Genetic evolving ant direction particle swarm optimization algorithm for optimal power flow with non-smooth cost functions and statistical analysis. Applied Softcomputing, 13, 4579-4593.
24. Zhu G., Kwong S. Gbest-Guided Artificial Bee Colony Algorithm for Numerical Function Optimization, 2010.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.