International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-2 Number-4 Issue-6 December-2012

Hybrid Genetic Algorithm with PSO Effect for Combinatorial Optimisation Problems

M. H. Mehta
Assistant Professor, Information Technology Department,
Sardar Vallabhbhai Patel Institute of Technology (SVIT),
Vasad-388306, India.

Abstract

In engineering field, many problems are hard to solve in some definite interval of time. These problems known as "combinatorial optimisation problems" are of the category NP. These problems are easy to solve in some polynomial time when input size is small but as input size grows problems become toughest to solve in some definite interval of time. Long known conventional methods are not able to solve the problems and thus proper heuristics is necessary. Evolutionary algorithms based on behaviours of different animals and species have been invented and studied for this purpose. Genetic Algorithm is considered a powerful algorithm for solving combinatorial optimisation problems. Genetic algorithms work on these problems mimicking the human genetics. It follows principle of "survival of the fittest" kind of strategy. Particle swarm optimisation is a new evolutionary approach that copies behaviour of swarm in nature. However, neither traditional genetic algorithms nor particle swarm optimisation alone has been completely successful for solving combinatorial optimisation problems. Here a hybrid algorithm is proposed in which strengths of both algorithms are merged and performance of proposed algorithm is compared with simple genetic algorithm. Results show that proposed algorithm works definitely better than the simple genetic algorithm.

Keywords

Genetic algorithm (GA), Particle Swarm Optimisation (PSO), Crossover, Mutation

1. Introduction

In engineering field, many problems are having inherent characteristic of not being solved in some polynomial time if input size is large. Complexity of such problems, depend on the input size given to them. Traditional methods [1] are available for solving them effectively but these conventional methods take large amount of time for computation. Thus heuristic methods have been proposed for solving these NP-Hard problems. Ant-Colony optimisation [2], Artificial-Bee Colony optimisation [3], Swarm Intelligence [4] and Genetic Algorithms [5] are evolutionary algorithms developed for solving optimisation problems. Genetic algorithms basically copy by human genetics process. Charles Darwin's famous "Survival of the fittest" strategy is followed by GAs to reach towards the optimal solution. Although, to achieve perfect optimal answer is hard for any evolutionary algorithm, but near optimal answer can be obtained with less time in comparison of the traditional methods. Genetic algorithms operate using four steps: 1) Initial population generation, 2) Selection Operator to choose the best parent chromosome and pushing it to go ahead in the process using fitness function, 3) Crossover operator to generate child chromosomes, And 4) Mutation Operators to give variation and diversity in the population so that minimums can be avoided. Particle swarm optimisation algorithm assumes particles flying in search space with some velocity and position. Their velocity adjustment is dynamic in nature. The strength of PSO lies in the fact that it uses historical information to reach to the optimal answer. For this, in PSO, local best of particle and global best of any particle is maintained. Where as both GA and PSO are powerful algorithms but if applied individually to solve any problem, neither GA nor PSO has been totally successful. They both have failed individually in terms of either running time and obtained results. GA's strengths are various operators, diversity provided by mutation and its simplicity while PSO's power lies in the fact that it utilizes historical data to achieve better results.

In this paper, proposed is an algorithm that is hybrid as it combines both genetic algorithm and PSO advantages. Simple traditional genetic algorithms are not having benefit of using historical data to improve the result and to reach to the solution. I have taken

Symmetric Travelling Salesman Problem [6] as my combinatorial optimisation problem. In symmetric TSP, the cost connecting one city to other remains same in both directions - forward and backward journey. Asymmetric TSP and dynamic TSP are other types of TSP problems. TSP is a well-known example of NP-Hard Combinatorial optimisation problem. Many optimisation problems can be reduced to a simple definition of TSP in which a salesman has a list of number of cities to visit. The salesman has to cover every city exactly once. He has to start from a city and has to make a round trip such that minimum expenditure tour is done and he is finished with start city. Here, every road/edge connecting the two cities are given some cost. The aim is to find minimum cost tour by making a round trip covering each city exactly once. In TSP, every city is connected with every other city. It can be observed by experiments that as number of cities increase in TSP, obtaining the optimal cost trip becomes toughest gradually. The real test of TSP lies in finding optimal solution as n (= number of cities) increases, possible tours to explore becomes (n-1)! /2. So for 5 cities, we have only 12 trips to analyse, but if number of cities increase to just 10 then we have now 181440 trips to study. Here (n-1)! /2 are taken as only one start city is considered and duplicate trips are removed in symmetric case. My experiment shows that proposed algorithm performs better than simple genetic algorithm for two classical TSPLIB [7] problems.

Remainder of this paper is organized as follows: Section II describes the Genetic Algorithm. Section III explains the concept of particle swarm optimisation algorithm. Section IV illustrates simple GA, which is designed to use in this paper. Simple GA is designed at first place so that it can be considered for comparison purpose for other sections. Section V shows complete proposed algorithm structure. The section also explains mechanism being considered in the proposed algorithm. Experiments and Results are narrated in Section VI. Section VII concludes the paper.

2. Genetic Algorithm

In 1975, J. Holland first proposed genetic algorithm. Genetic algorithm is an iterative search and optimisation method, which, works on the principle of human evolution process. Genetic algorithm has taken idea of working from human genetics. Any genetic algorithm has basically four steps to execute. The first step is initial population generation by choosing appropriate encoding method. After creating the initial population, by selection operator the optimal value is chosen based upon the fitness function. In TSP, the round trip that is travelled by the salesman is fitness function and it should be minimum. Selection operator's task is to select the most optimal population(s) that can be forwarded in the evolution and generate best children for the future generations. Many different types of selection operators are available in the literature [8]. After selecting the best population, crossover operator operates on the selected population, to generate new off springs. Crossover operators actually make permutations on the chromosomes that result in new chromosomes. Various crossover operators' efficiencies and effects are different [1]. To provide population diversity, Mutation occurs in genetic algorithm. Mutation operator necessarily gets whole process of GA out of any local minima if realized by the algorithm. Mutation operator is an important requirement for genetic algorithm so that whole process of getting optimal value does not fall into some valley. In traditional genetic algorithms, more emphasis is given on crossover operators. Mutation operators are given few chances to occur in GA. The best fitness value is considered in each iteration and the process terminates when it reaches to its stopping criteria such as number of total iterations.

3. Particle Swarm Optimisation

Swarm Intelligence is the whole new branch of algorithms, which take motivation for their operation from the nature. Particle swarm optimisation [4] works by mimicking the social behaviour of birds and fishes. In 1995, Eberhart and Kennedy first proposed particle swarm optimisation. Standard PSO 's framework consists of individuals who fly in the search space with some velocity. The velocity of each individual is adjusted according to its own flying experience and its companions' flying experience. This velocity adjustment is dynamic in nature. The i^{th} particle is given by $X_i = (x_{i1,}\ xi2_{,\dots,}x_{in})$. The best previous position of the ith particle is calculated and represented by $P_i = (p_{i1,}p_{i2,\dots,}p_{in})$. The best particle among all the particles referred as global best is represented by symbol g. Rate of change that is velocity of particle's position is given by $V_i = (v_{i1,}v_{i2,\dots,}v_{in})$. The particles are measured based on the following equations:

$v_{id} = w * v_{id} + c_1 * r1* (p_{id} - x_{id}) + c_2 * r2* (p_{gd} - x_{id})$ (1)
$x_{id} = x_{id} + v_{id}$ (2)
In above equations, c_1 and c_2 are two positive constants and r1 and r2 are two random functions in range 0 to 1 including 0 and 1. Symbol of weight is w and first part of equation 1 is dealing with previous velocity of the particle, where as second part is "cognitive" part and third part is "social" part. Two constants c_1 and c_2 are also known as individual factor and societal factor respectively. Both genetic algorithm and particle swarm optimisation are considered as main algorithms for forming a hybrid algorithm in this paper.

4. Simple Genetic Algorithm (SGA)

Simple genetic algorithm used in this paper follows four steps as described above. Initial population is generated in first step. After generating the initial population, selection operator chooses the best trip, which is having minimum value. Here, elitism selection operator is chosen which selects the best individual parent having minimum fitness value that is minimum tour cost. Order crossover [9] is chosen for generating off springs in the paper. Order crossover works as follows: from the two parents, to have two new off springs, two random points are chosen form every parent. The string between random points is directly copied into respective child. Then after, the first child starts copying the string from parent 2 after the second random point but checks that digit already used from parent 1 should not occur in it. When child 1 reaches end of the string of parent 2, it starts from the starting point of parent 2 string and continues the process in a way that duplicate numbers are removed from the final offspring. When it reaches the first random point it stops. Off spring 1 is arranged. In similar way, off spring 2 is prepared. Example can be like this: two parent strings 12564387 and 14236578 are considered. Two random points are chosen as 2 and 5. Two off springs generated are shown in Table I.

After generation of off springs, Mutation operator is executed. Mutation operator does not work on two chromosomes. It works only on single chromosome and changes current population so that process of getting minimum tour is not hindered into local minima.

Thus, mutation necessarily aids the whole process from coming out of the valley. Mutation operators occur at fixed intervals in the simple GA algorithm designed in this paper. Mutation Operators used in this paper are as follows: 1) Inversion Mutation, 2) Reciprocal/Exchange Mutation, and 3) Slide Mutation [10]. Here also, in each operator, two random points are chosen as two and five. Table II, III and IV illustrate all three operators.

Table I: Order Cross Over

Parent-1)	12-564-387	→	Offspring-1)	23-564-781
Parent-2)	14-236-578	→	Offspring-2)	54-236-871

Table II: Inversion Mutation

Chromosome	12-564-387	→	New Chromosome	12-465-387

Table III: Reciprocal/Exchange Mutation

Chromosome	12-387-564	→	New Chromosome	12-783-564

Table IV: Slide Mutation

Chromosome	12-387-564	→	New Chromosome	12-873-564

Inversion mutation operator can be narrated as follows: for any chromosome string, any two random points are selected. After selection of random points, the string between these two random points is reversed to generate new chromosome string. Exchange mutation operator's mechanism of working is in following way: for any chromosome string, two random points are chosen and the string between two points is partially exchanged. Where as in slide mutation, after selection of two random points from the chromosome, the string between two points is rotated similar to left shift operator of digital logic register operator.

So in the simple GA used in this paper, selected population undergoes crossover operation generally to generate new off springs but is updated by mutation operators at fixed intervals only. Simple genetic algorithm's structure used in this paper is shown in Fig. 1 [11].

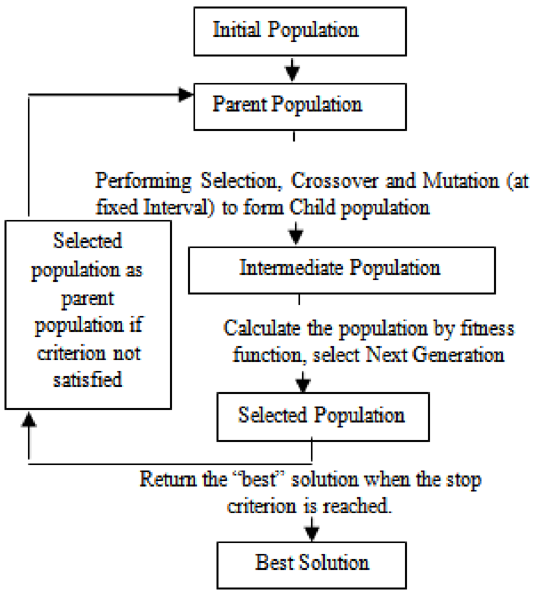
Initial Population
Parent Population
Performing Selection, Crossover and Mutation (at fixed Interval) to form Child population
Selected population as parent population if criterion not satisfied
Intermediate Population
Calculate the population by fitness function, select Next Generation
Selected Population
Return the "best" solution when the stop criterion is reached.
Best Solution

Figure 1: Simple Genetic Algorithm (SGA) [11]

Table V: Two-Point Turn over Mutation Operator

Chromosome	123-456-789	→	New Chromosome	321-456-987

Table VI: Mutual Mutation Operator

Chromosome	12345678	→	New Chromosome	13572468

Or

Chromosome	12345678	→	New Chromosome	24681357

5. Proposed Algorithm

Traditionally, genetic algorithms do not use any historical information in its operation at all. Each iteration carried out in GA yields best minimum chromosome of population under consideration only but it does not compare the current best value to the previous iterations' calculated minimum optimums. This deficiency of GA gives poor results. My proposed algorithm is mainly dependent on the idea of using historical information taking inspiration from PSO and merging this idea into GA making it a hybrid algorithm.

Algorithm starts with initial population generation. Then it generates new population by elitism selection operator. This population is stored for comparison purpose. Except first iteration, in all iterations previously stored population's calculated distances are compared with current generated population's distances. If historical data gives optimum result than current considered population, previous population is forwarded in the process every time and current population is ignored. Every time fresh population is generated after the comparison only.

At fixed number of iterations, mutation occurs on the generated population with historical data where as for other iterations, crossover operator is executed. Algorithm stops when number of decided iterations is performed and final best cost is achieved. Proposed algorithm is depicted in Figure 2.

6. Experiments and Results

Implementation is done in Mat lab 7.9 on Intel Core 2 Duo processor with 3 GB Memory. In both algorithms initial population is set to 100. Initial population contains 100 parents to maintain similarity in both algorithms. Iterations were kept to 10000.

Two TSPLIB problems were considered for comparison. As in TSPLIB [7], the best answers are given different algorithms' results can be compared with them. For every problem, four runs are taken and the best cost is finally considered.

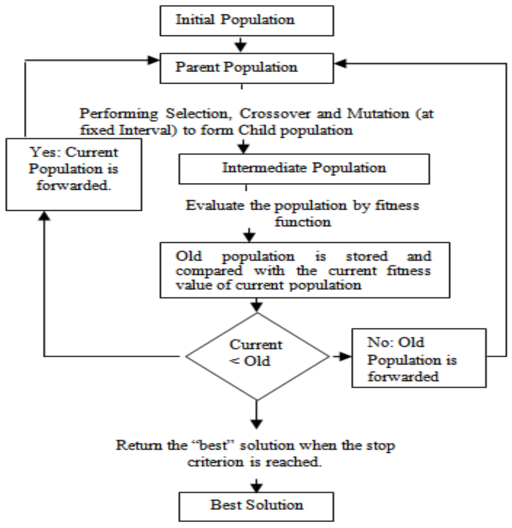
Initial Population
Parent Population
Performing Selection, Crossover and Mutation (at fixed Interval) to form Child population
Yes: Current Population is forwarded.
Intermediate Population
Evaluate the population by fitness function
Old population is stored and compared with the current fitness value of current population
Current < Old
No: Old Population is forwarded
Return the "best" solution when the stop criterion is reached.
Best Solution

Figure 2: Proposed Algorithm

Table VII: Results for SGA

TSP Problem	Optimal Answer	Run1	Run2	Run3	Run4	Best Cost
Gr24	1272	2865	2688	2626	2839	2626
Swiss42	1273	3956	4040	3756	3939	3756

Table VIII: Results for Hybrid-GA with PSO Effect

TSP Problem	Optimal Answer	Run1	Run2	Run3	Run4	Best Cost
Gr24	1272	1795	1802	1777	1817	1777
Swiss42	1273	3006	2767	3048	2919	2767

Results in Table VII and Table VIII show that SGA with mutation at fixed interval performs poor in the comparison of Hybrid-GA. Proposed algorithm gets 1777 for Gr24 city problem, which is far better than 2626 achieved by SGA. Similarly for Swiss42 city problem, Hybrid-GA proposed in paper gets 2767 as best cost answer, while SGA gets 3756 as best route. The results clearly show that Hybrid-GA performs far better than SGA and is nearer to optimal answers. In

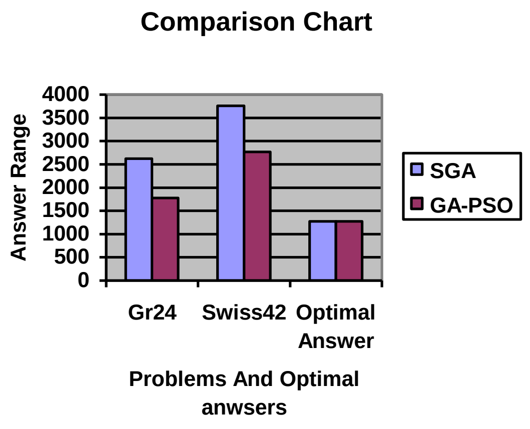

Figure 3: Graphical Comparison of Proposed algorithm with SGA

Figure 3, Graphical comparison is presented in chart form, which again shows that Hybrid GA gives more optimal answers than SGA.

7. Conclusion

In this paper, proposed is a Hybrid Genetic Algorithm for solving Symmetric TSP as combinatorial optimisation problem. Traditional GA helps in dealing with NP-Hard problems, but simple GA cannot be utilized to get very near optimal answer. Here SGA and Proposed Hybrid GA is compared. Comparison shows that Hybrid-GA performs far better than SGA as it achieves more nearer answers to optimal answers.

The paper proves the fact that historical information should be used in optimisation algorithms to reach to optimum answers. Emphasis on combination of two known algorithms can aid in getting good results. Although, Hybrid-GA gets good results than SGA, more thorough analysis of different strategies related to combinatorial optimisation can really help in improving the proposed algorithm.

In this paper, algorithm-running time is not considered. Time can be considered as one of the major comparison factor as well in future.

Acknowledgment

I thank to almighty and my dearest parents and elder sister for giving me inspiration and support. I heartily thank all the persons who have helped directly or indirectly in any way to complete this work.

References

[1] ABDOUN Otman and ABOUCHABAKA Jaafar “A comparative study of adaptive crossover operators for genetic algorithms to resolve the travelling salesman problem” IJCA (0975-8887) October 2011.
[2] Carlos M. Fernandes, Agostinho C. Rosa and Vitorino Ramos “Binary Ant Algorithm” GECCO’07 London, ACM, July 7-11, 2007.
[3] Anan Banharnsakun, Tiranee Achalakul, Booncharoen Sirinaovakul “ABC-GSX: A hybrid method for solving the Traveling Salesman Problem” IEEE 2010.
[4] Mei-Ping song, Guo-Chang gu “Research on particle swarm optimisation: a review” IEEE 2004.
[5] Golberg, David E. "Genetic algorithms in search, optimization, and machine learning." Addion wesley 1989 (1989).
[6] Jacek M. Zurada “Introduction to artificial neural systems”.
[7] TSPLIB http://www.tsp.gatech.edu/index.html.
[8] Noraini, Mohd Razali, and John Geraghty. "Genetic algorithm performance with different selection strategies in solving TSP." (2011).

[9] Monica Sehrawat and Mr. Sukhvir Singh "Modified Order Crossover (OX) Operator" IJCSE – ISSN:0975-3397 Vol.3 May 2011.

[10] Basima Hani Hasan and Moutaz Saleh Mustafa "Comparative Study of Mutation Operators on the Behavior of Genetic Algorithms Applied to Non- deterministic Polynomial (NP) Problems" IEEE 2011.

[11] M.H. Mehta and V.V. Kapadia "Intelligent Adaptive Mutation Based Genetic Algorithm For Combinatorial Optimisation Problems" Accepted and Registered, IEEE ICECT Kanyakumari, April 6-8, 2012.

Ms. Mala H. Mehta is an Assistant Professor at Sardar Vallabhbhai Patel Institute of Technology, Vasad. She has completed her Bachelors of Engineering in Information Technology with Distinction from Gujarat University in 2005. She has completed her Masters of Engineering in Computer Engineering from Gujarat Technological University in 2012 with 9.08 CPI. She has published five research papers in various national and international conferences. She was given "Young Investigator Award" at an International Conference as well.