Solving Travelling Salesman Problem using Clustering Genetic Algorithm

R.SIVARAJ
Research Scholar & Assistant Professor (Sr.Gr.)
Department of Computer Science and Engineering
Velalar College of Engineering and Technology
Erode, Tamil Nadu, India.
rsivarajcse@gmail.com

Dr.T.RAVICHANDRAN
Principal
Hindusthan Institute of Technology
Coimbatore, Tamil Nadu, India.
dr.t.ravichandran@gmail.com

R.DEVI PRIYA,
Assistant Professor
Department of information Technology
Kongu Engineering College
Erode, Tamil Nadu, India.
scrpriya@gmail.com

***Abstract*— Solving NP hard problem like Travelling Salesman Problem (TSP) is a major challenge faced by analysts even though many techniques are available. Many versions of Genetic Algorithms are introduced by researchers to improve its performance in solving TSP. Clustering Genetic Algorithm (CGA) was recently introduced and this paper analyzes the results obtained by implementing it for TSP. It is observed that CGA effectively finds out more optimal solution sooner than the Standard Genetic Algorithm (SGA) in 3 different instances considered.**

Keywords- Genetic algorithms, Travelling Salesman Problem, Clustering genetic algorithms, Convergence Velocity.

I. Introduction

The "Traveling Salesman Problem" (TSP) is a common NP hard problem that can be used to test the effectiveness of Genetic Algorithm. The traveling salesman problem is defined in simple term as: “If there are n cities and the cost to travel between each pair of them is given, the objective is to find the cheapest and shortest path that anyone can follow to reach destination city from the starting city provided all the cities involved in the problem are traveled exactly once”.

GA for TSP cannot be designed very easily. Many problems use just numerical values and hence it is easy to define the structure of chromosome and perform genetic operations. But TSP uses permutation encoding which involves ordering of cities for which genetic operations cannot be directly applied. Many researchers are putting in more effort in producing good approaches for solving TSP kinds of problems.

In this paper a new algorithm called Clustering Genetic Algorithm is proposed which tends to perform better for TSP when compared with existing algorithms. The proposed approach not only works for TSP but also for other similar problems like Scheduling, Problems which involves ordering etc.

II. RELATED WORK

For more than a decade, many variations of Genetic Algorithms are being introduced to solve TSP kind of problems which are often faced in real life applications. In Moon et al (2002), the authors propose to use topological sorting which defines ordering of cities in a directed graph. In Zakir (2003), the sequential constructive crossover (SCX) operator constructs child chromosome using better edges based on their values present in the parents' structure. Hu and Di Paolo (2007) suggest mapping the represented solution of the pre-problem into the associated solution to the original problem. But designing pre-problem is extremely problem specific and requires much expertise of the user. Buthainah and Ali (2008) suggest using specialized crossovers and mutation to improve the solution. Gohar Vahdati et al (2009) proposed a new method using heuristic crossover and mutation operations to generate and to reinforce optimal chromosomes. Fozia et al (2009) proposed a new representation scheme of chromosomes using binary matrix where new fittest criteria is used for finding the optimal solution for TSP. Some literatures suggested hybrid Genetic Algorithms to improve the final optimality of TSP solution (Lee, 2004; White and Yen, 2004; Marinakis et al 2005).

III. CLUSTERING GENETIC ALGORITHM

Clustering Genetic Algorithm (CGA) is recently introduced in Sivaraj and Ravichandran et al.(2011) in which it is implemented for 0/1 Knapsack Problem. The methodology of CGA is simple which effectively groups chromosomes in the population into clusters using K- means clustering and then applies genetic operators as usual.

A. Design of CGA parameters for Travelling Salesman Problem

The encoding scheme chosen for the problem is Permutation encoding where genes in the chromosome represent the order in which one has to travel from the starting city. The idea here is that if there are n cities in the chosen problem, it has to be divided into x/n subgroups (clusters) where x represents the number of chromosomes in the entire population and n represents the size of clusters. Rather than searching from the starting city to the last city in each chromosome, the proposed approach effectively searches locally and combine the local best solutions to form a global final solution. All cities in the problem are clustered like shown in figure 1 using K-means clustering where each group will have collection of cities. Parent selection, genetic operators like crossover and mutation are applied and best ordering of the cities within each group are calculated independently as shown in figure 2 and finally the results from each group are combined to form a single solution as shown in figure 3. Thus it will reduce overhead for the algorithm to apply genetic operators as a whole which takes more time to execute.

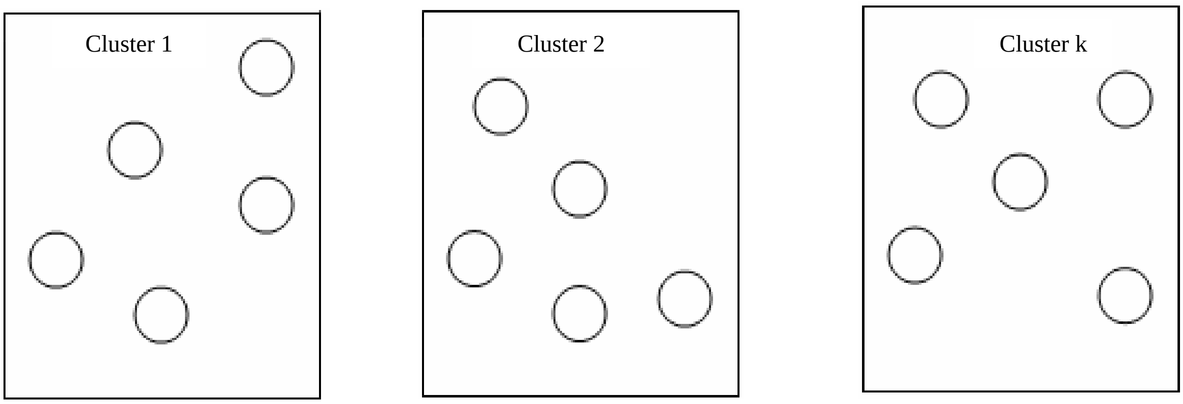

Figure 1. Grouping of chromosomes into K- Clusters

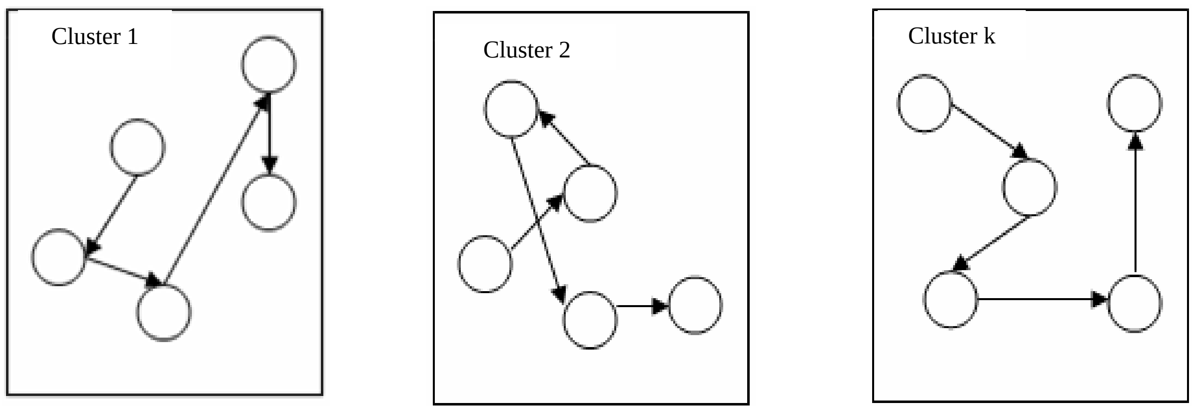

Figure 2. Local best solutions in each cluster

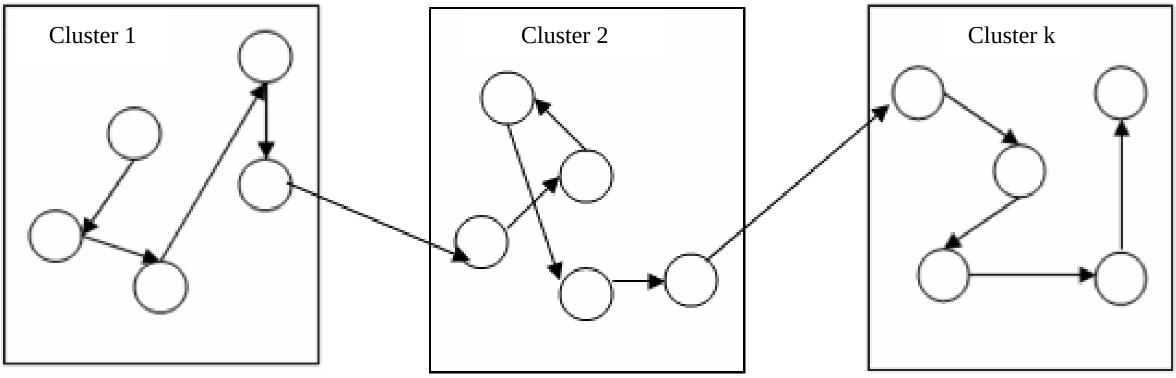

Figure 3. Final global solution for the given problem

IV. Implementation and Results

CGA is being implemented for different instances from TSPLIB. The instances chosen for our experiments are br17, ftv44 and Kro124p. The stopping condition for Genetic Algorithm may be either fixed number of generations or till convergence of chromosomes in the same values for five consecutive generations. The experiments are conducted with two setups.

(i) In the first setup, the algorithm is run until the solution converges (values remain unchanged for consecutive five generations).

The genetic operators and parameters chosen are:

- Initialization scheme : Random
- Population Size : 40
- Selection : Tournament selection
- Crossover : Two point
- Crossover probability : 0.90
- Mutation probability : 0.05
- Number of clusters : 3

Figures 4 and 5 show the distance obtained in the last generation and the total number of generations taken for convergence for br17 instance. Since this instance contains only 17 cities, finding the optimal route is not much difficult. In 15 different executions, CGA takes around only 20 generations to reach the optimal solution.

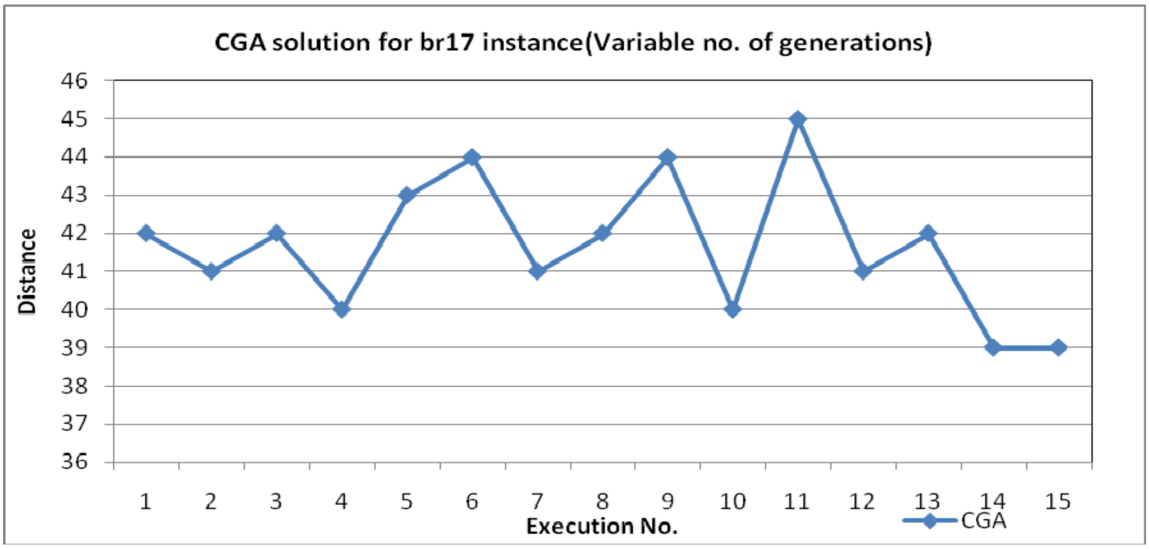

Figure 4 CGA solution for br17 instance (Variable no. of generations)

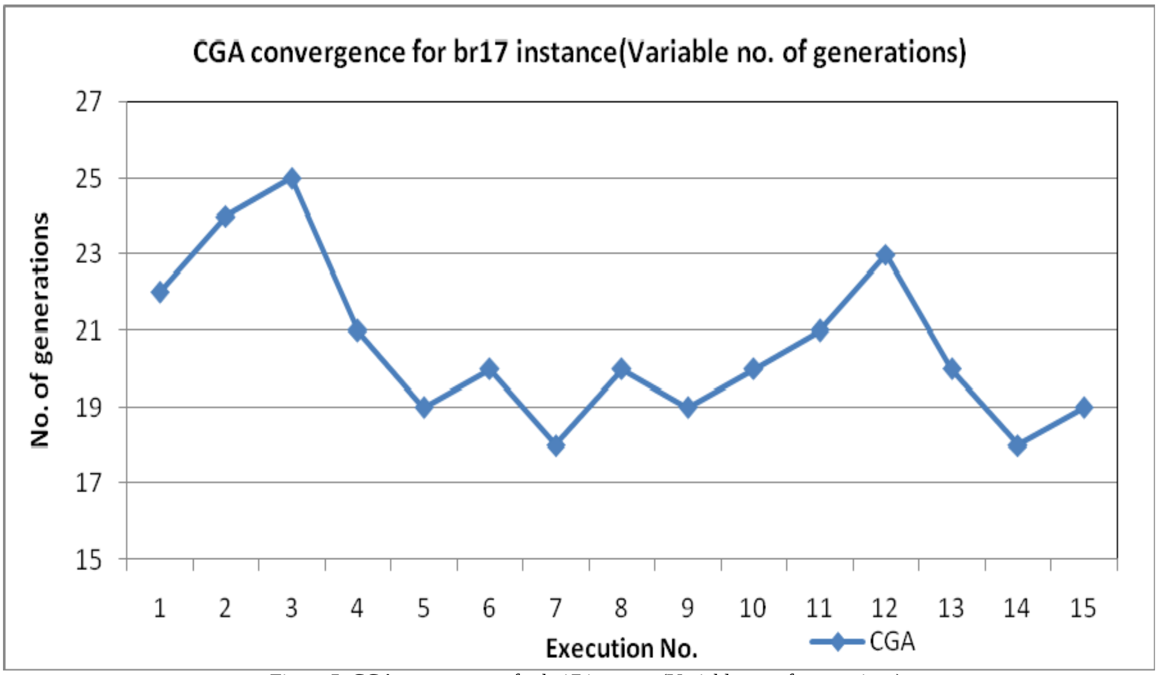

Figure 5. CGA convergence for br17 instance (Variable no. of generations)

The optimal distance for br17 instance is 39 and CGA is able to produce very near results. The final accuracy rate in finding the optimal solution is around 99% in CGA which is highly appreciable.

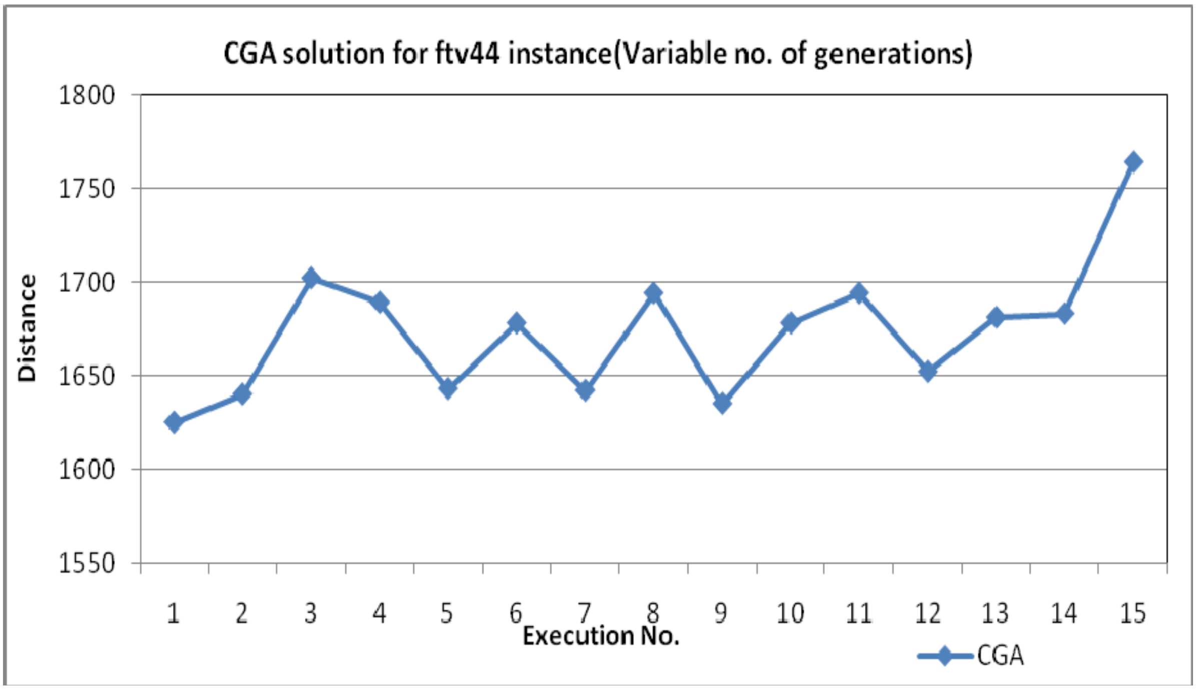

Figure 6. CGA solution for ftv44 instance (Variable no. of generations)

Figures 6 and 7 show the profits obtained and the corresponding number of generations taken for convergence for ftv44 instance. It shows that 27-32 generations are needed approximately to converge to the solution. The optimal distance already known for ftv44 instance is 1613. It could be inferred that the results produced by CGA are 94% accurate.

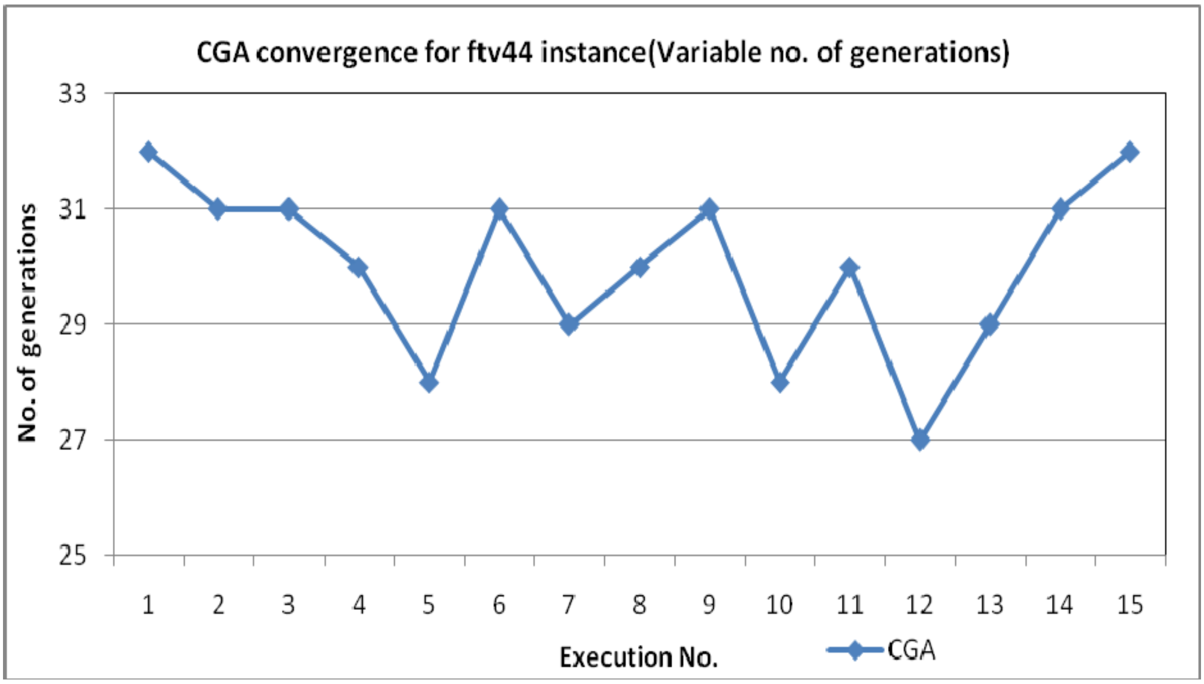

Figure 7. CGA convergence for ftv44 instance (Variable no. of generations)

CGA is also implemented for Kro124p which has quite large number of cities to deal with. Even in this complicated scenario, CGA showed 87% in terms of final accuracy. The number of generations needed is comparatively larger than other two instances. But if the population size is changed, the convergence and final solution is affected. Figures 8 and 9 show the profits obtained and the corresponding number of generations taken for convergence for Kro124p.

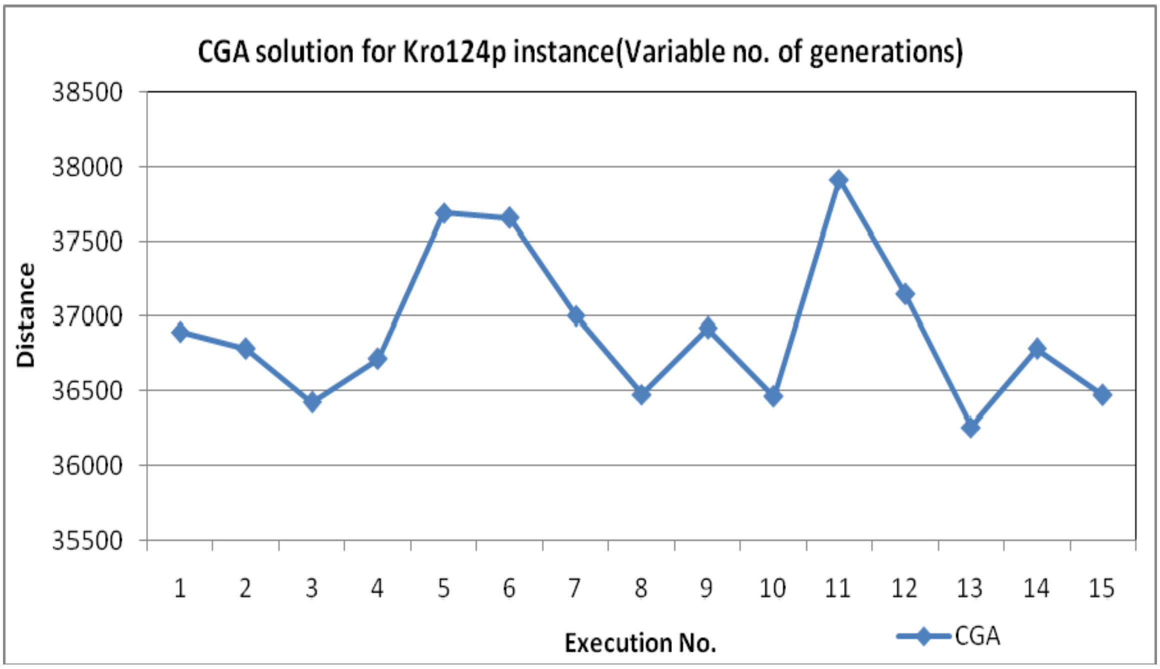

Figure 8. CGA solution for ftv44 instance (Variable no. of generations)

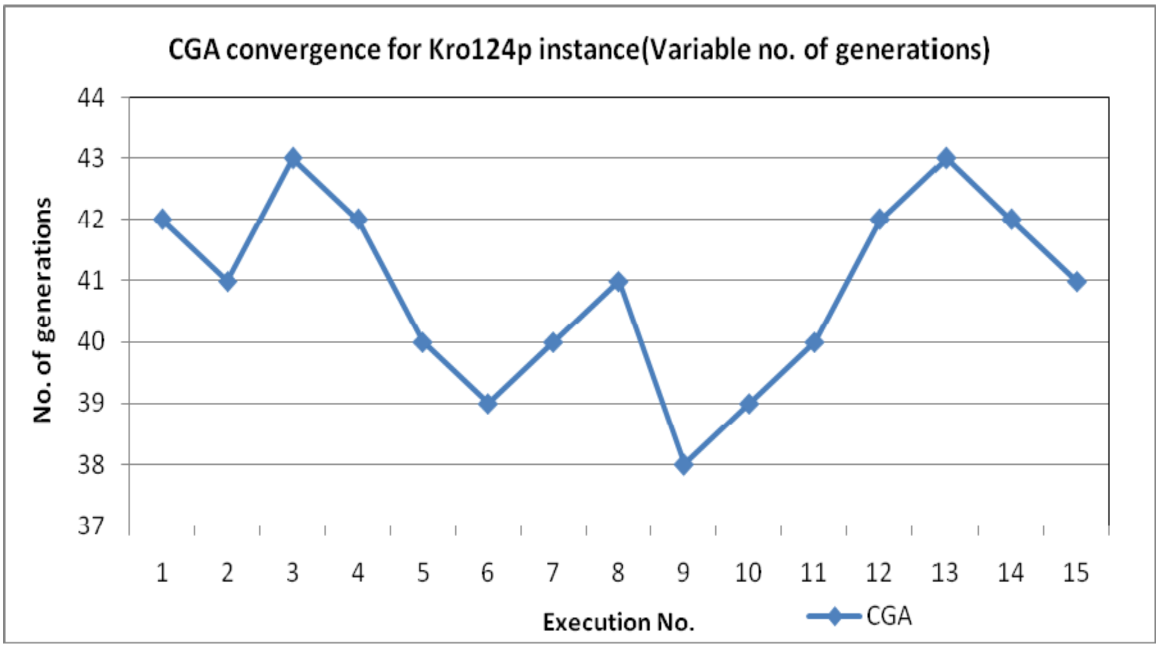

Figure 9 CGA convergence for ftv44 instance (Variable no. of generations)

(ii) In the second setup, the number of generations is fixed and solution that is obtained in the final generation is taken as the optimal solution found. Figures 10, 11 and 12 depict the final distances that are found by applying CGA on br17, ftv44 and Kro124p instances respectively. CGA produced results very near to the best optimal solution for the problems involving either small number or large number of cities.

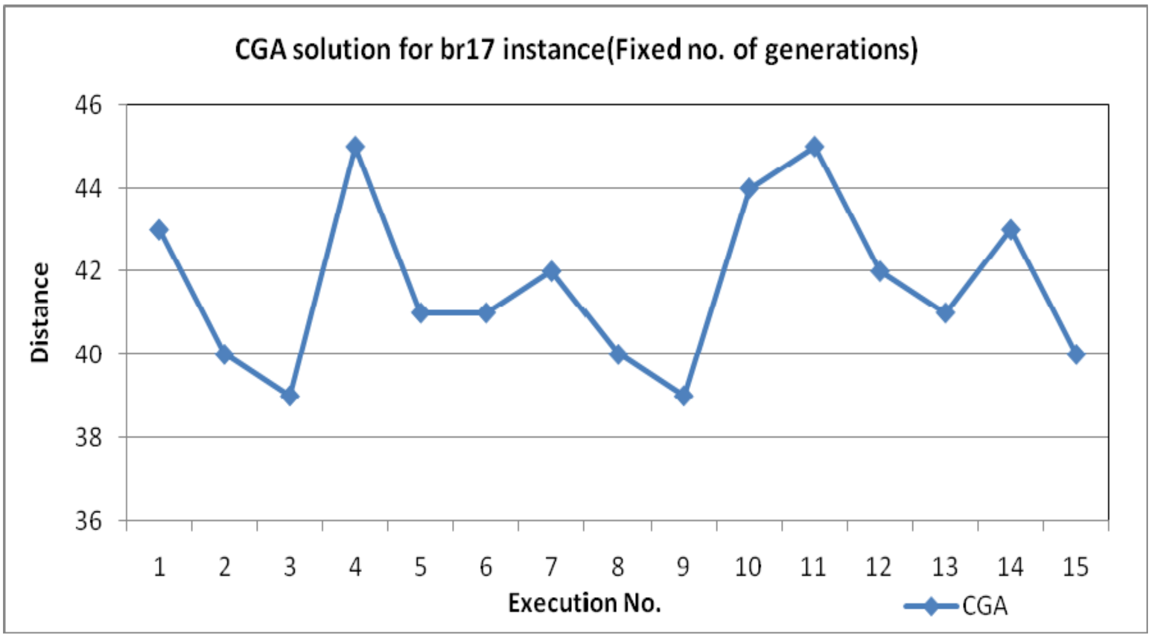

Figure 10. CGA solution for br17 instance (Fixed no. of generations)

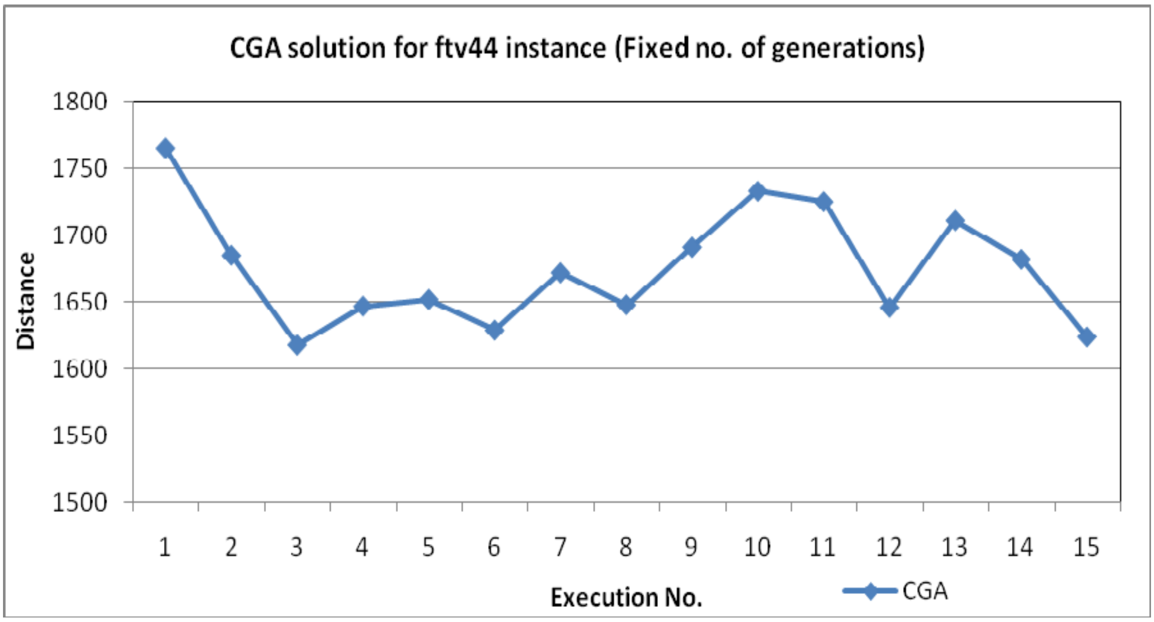

Figure 11. CGA solution for ftv44 instance (Fixed no. of generations)

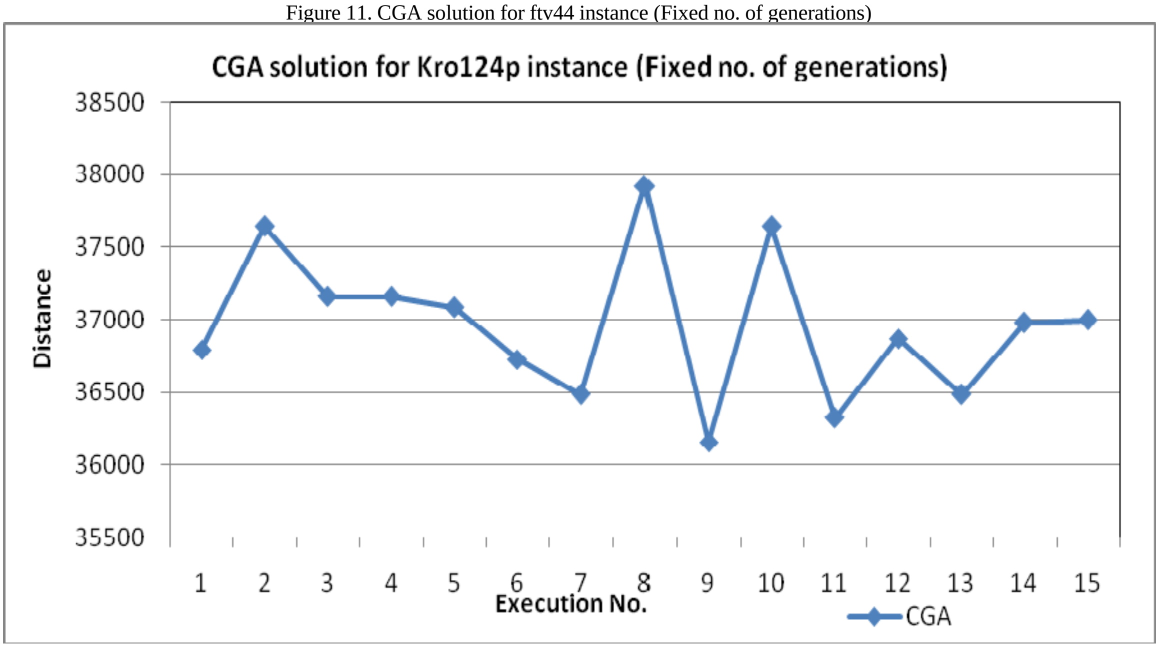

Figure 12 CGA solution for kro124p instance (Fixed no. of generations)

CGA produced more accurate results whether it is run for fixed number of generations or variable number of generations in all the instances considered thus making it suitable for applying to all kinds of scheduling problems.

V. CONCLUSION

The performance of CGA changes with different genetic parameters chosen. It introduces a checkpoint prior to selection mechanism in each generation by the usage of k-means clustering algorithm. It thus helps in selecting only the best chromosomes to be carried over to further generations and also helps in reducing the pressure for selection process. Even though clustering is an extra cost pulling factor than the standard Genetic Algorithm process, in the final solution it will effectively reduce the number of generations needed and improve the optimality of the solution. When final optimality of the solution is considered, this is negligible. Hence with improved performance of CGA, many complex problems in different domains and with different functionalities can be easily solved.

REFERENCES

[1] Buthainah Fahran Al-Dulaimi and Hamza A. Ali(2008), “Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique (TSPGA)”, World Academy of Science, Engineering and Technology Vol.38, pp.296-302.

[2] Hanif Khan Fozia, Nasiruddin Khan, Syed Inayatullah and Shaikh Tajuddin Nizami(2009), “Solving TSP problem by using genetic algorithm”, International Journal of Basic & Applied Sciences, Vol. 9, No.10, pp.79-88.
[3] Hu .X. B and Di Paolo.E (2007), “A hybrid genetic algorithm for the travelling salesman problem”, Proceedings of Nature Inspired Cooperative Strategies for Optimization: NICSO 2007,. Berlin: Springer, pp. 357–367.
[4] Lee .Z. J (2004), " Hybrid algorithm applied to travelling salesman problem”, IEEE International Conference on Networking, Sensing and Control, Mar 21-23, Taiwan, pp.237-242. Vol 1.
[5] Marinakis .Y, Migdalas .A and Pardalos .P. M(2005), “A Hybrid Genetic-GRASP Algorithm Using Lagrangean Relaxation for the Traveling Salesman Problem”, J.Comb. Optim. Vol.10, No.4, pp.311-326.
[6] Moon C, Kim J, Choi J, Seo Y (2002), “An efficient genetic algorithm for the traveling salesman problem with precedence constraints”, European Journal of Operational Research ,Vol.140, pp. 606–617.
[7] Sivaraj .R and Ravichandran .T (2011), “An Improved Clustering Based Genetic Algorithm for Solving Complex NP Problems”, Journal of Computer Science, Vol.7, No.7, pp. 1033-1037
[8] TSPLIB, http://www.iwr.uni - heidelberg.de/iwr/comopt/ Software/TSPLIB95.
[9] Vahdati G, Poostchi M, M. B. Naghibi S and Yaghoubi M. A New Approach to Solve Traveling Salesman Problem Using Genetic Algorithm Based on Heuristic Crossover and Mutation Operator, Proceedings of International Conference of Soft Computing and Pattern Recognition, SOCPAR 2009, 4- 7 Dec. 2009, pp. 112-116.
[10] White .C. M and Yen .G. G (2004), “A hybrid evolutionary algorithm for traveling salesman problem”, Congress on Evolutionary Computation (CEC2004) 2, pp.1473-1478.
[11] Zakir H. Ahmed (2003), “Genetic Algorithm for the Traveling Salesman Problem using Sequential Constructive Crossover Operator”, International Journal of Biometrics & Bioinformatics”, Vol.3 No.6, pp.25-30.