Critical Path Problem for Scheduling Using Genetic Algorithm

Harsh Bhasin and Nandeesh Gupta

Abstract The critical path problem, in Software Project Management, finds the longest path in a Directed Acyclic Graph. The problem is immensely important for scheduling the critical activities. The problem reduces to the longest path problem, which is NP as against the shortest path problem. The longest path is an important NP-hard problem, which finds its applications in many other areas like graph drawing, sequence alignment algorithms, etc. The problem has been dealt with using Computational Intelligence. The paper presents the state of the art. The applicability of Genetic Algorithms in longest path problem has also been discussed. This paper proposes a novel Genetic Algorithm-based solution to the problem. This algorithm has been implemented and verified using benchmarks. The results are encouraging.

Keywords Longest path problem · Genetic algorithms · NP-Hard problems Heuristics

1 Introduction

Handling non-deterministic polynomial (NP) time problems is one of the most precarious tasks. The problems which can be solved in polynomial time by deterministic algorithms are referred to as polynomial time problems or P problems [1].

H. Bhasin (✉)
School of Computer and Systems Sciences,
Jawaharlal Nehru University, New Delhi, India
e-mail: i_harsh_bhasin@yahoo.com

N. Gupta
Department of Computer Science,
YMCA University of Science and Technology,
Faridabad, India
e-mail: guptanandeesh@gmail.com

M. Pant et al. (eds.), *Soft Computing: Theories and Applications*,
Advances in Intelligent Systems and Computing 583,
https://doi.org/10.1007/978-981-10-5687-1_2

Finding shortest path in a given graph is an example of P-type problem. NP problems can be solved by non-deterministic algorithms in polynomial time [2]. These problems are further divided into two types: NP-complete and NP-hard.

In NP-complete problems, solution can be verified in polynomial time. Hamiltonian cycle problem is an example of NP-complete problem. Problems which can neither be solved in polynomial time by deterministic algorithms, nor their solutions can be verified in polynomial time are called NP-hard problems. The longest path problem is an example of NP-hard problem.

Generally, NP-hard problems are optimization problems. However, global optimization is an illusion. Therefore, soft computing approaches like Genetic Algorithms (GAs) are used to handle such problems. GAs are heuristic search processes that are based on survival of the fittest [3].

The paper presents a GA-based solution for the critical path problem in software planning and management. The approach has been implemented and the results are encouraging.

The organization of the paper is as follows: Section 2 presents the review. Section 3 explains the basics of GA. Section 4 presents the background. Section 5 presents the proposed work. The next section presents the results and the last section concludes.

2 Review

As stated earlier, longest path problem is an NP-hard problem, which finds its application in various fields. In order to access the state of the art and place the proposed technique in the right perspective, a comprehensive literature review has been carried out. The review has been carried out in accordance with the guidelines proposed by Kitchenham [4]. This section has been divided into various subsections, which are as follows:

1. Research questions
2. Review methodology
3. Review

Research questions: In order to carry out a meaningful review, appropriate research question should be crafted. The review, in turn, should be able to answer the research questions. The aim of this review is to answer the following research questions.

RQ. 1. What is the trend in research in NP problems notably the longest path problem?

RQ. 2. What are the existing techniques used to tackle the problem?

Review methodology: the present study intends to summarize the work concerning NP problems in general and longest path problem in particular. The study also explores the applicability of Diploid Genetic Algorithms (DGAs) in longest path problem.

The databases that have searched while carrying out the review are as follows:

1. ACM Digital Library
2. Science Direct
3. IEEE
4. Wiley
5. Springer

From amongst these, the journals having high impact factor were selected. The papers which have been considered important in the topic have also been selected even if they were not from above journals.

Initially, all the papers related to longest path problem were included in the search, from those papers, the papers which were related to computer science discipline were selected, a further search was carried out by reading the abstract.

The selected papers were filtered according to the proposed technique. The results of the review have been presented in the next section.

The data was collected in a scientific way and the prime criterion was the quality of a paper. The summary of the review has been presented in Table 1.

Answer to RQ1. The search was carried out as follows. The keyword that was used to search the relevant articles was <longest path problem>. This was followed by applying various criteria to filter out the irrelevant papers. English was selected as the language for the papers. The topics selected were <Computer Science>, <AI> and <Theoretical Computer Science>. The year was selected as follows: the beginning and the end years were set same, For example, to select the papers of 2015, the beginning and the end year was 2015. Thus resulting in a total of 1209 articles and 12,496 chapters, in Springer link. Similar searches were carried out in

Table 1 Review of techniques to solve longest path problem

Papers	Proposed work
[5, 6]	These papers relate travelling salesman problem with the longest path problem and considers it to be a special case of longest path problem. It also discusses some of the algorithms from longest path problem
[7]	The paper applies some algorithms on super classes of interval and permutation graph and discusses the issue concerning the comparability of the graphs
[8]	The paper presents an approximation algorithm from the problem. The complexity of the proposed algorithm is $2\hat{}O(\log^{1-\varepsilon} n)$
[9]	The paper uses the concept of Hamiltonian Path problem to handle the longest path. It explores weighted trees, block graphs, ptolemaic graphs, and cacti and then solves the problem
[10]	The paper proposes $O(n)$ algorithm for finding the longest path in a bipartite graph
[11]	The paper proposes an $O(n^6)$ algorithm for finding the longest path in a biconvex graph and an $O(n^4)$ approximation algorithm for the same problem
[12]	The paper shows that interval graphs can be used to solve the longest path problem in polynomial time

other databases. However, it was found that most of them were not related to the core problem itself.

The total number of papers finally selected was 8.

Answer to RQ2. The critical path problem reduces to the longest path problem, using problem reduction approach. The problem reduces to that of finding a simple path of longest length which does not have any repeated vertex. Though the shortest path problem can be solved in polynomial time, this version of the longest path problem is an NP-hard problem.

The researches have been able to find a solution for Directed Acyclic Graph (DAG) which is used in many applications such as compiler, but the solution for undirected graphs still eludes the fraternity.

It was earlier suggested that if a graph having all the weight edges is given, then the procedure used to find the shortest path can also be used to find the longest path by simply converting all the weights to negative. However, the selection would not work as the susceptibility of creation of negative length cycles is high.

The longest path terminating at a given vertex can also be found by its incoming neighbors and incrementing the maximum length recorded by those neighbors. In the case, where the given vertex does not have any neighbor, the length of the longest path is Zero.

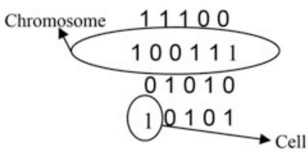
The problem has also been handled using approximation algorithms. The approximation ratio of the proposed solution is approximately O (N/ log N) using color coding technique.

Except for the above, the technique has also been applied on other NP-complete problems. The review considered all those problems in which GA were applied. [13–15].

3 Genetic Algorithm

Genetic Algorithm (GA) is a heuristic search process which is based on the theory of survival of the fittest [3, 16]. It is generally used in optimization problems. The process starts with the creation of a population of biological units called chromosomes. Chromosomes further consist of a number of smaller units called cells. These cells may be binary, decimal, or even hexadecimal. Fig. 1 shows a population having cells which are binary.

Fig. 1 A binary Population



This work uses binary chromosomes which then together form a binary population. There are many genetic operators, some of them are: crossover, mutation, and roulette wheel selection. These genetic operators help us modify the parent chromosomes, thus helping us to reach the requisite solution.

1. **Crossover.** It helps to produce new chromosomes (child chromosomes) with traits of both the parent chromosomes. The crossover population merges the two parents for producing the child. There are many types of crossover: One-point, Two-point, Multipoint, and Uniform crossover.
   The One-point crossover is the simplest crossover in which a random point (R) is selected in the parent chromosomes. Now, the child is produced by merging the part to the left of R in first chromosome and that to the right in second chromosome. Similarly, the other child will consist of the cells of first parent after R and cells of second parent before R.
   The Two-point crossover is done by amalgamating the parent chromosomes wherein two random points are selected on basis of which the formation of new chromosome takes place.
2. **Mutation.** It is an operation which is carried out so as to break the local maxima. One of the ways of implementing mutation is by simply complementing a randomly chosen bit of a random chromosome.
3. **Selection.** It is a process in which some chromosomes having higher fitness values are selected from the population and then these chromosomes are replicated so as to form a new population with a majority of chromosomes having high fitness values, thus increasing the final solution.

Genetic algorithms have been successfully used for optimization problems, particularly in NP problems. This work explores the use of genetic algorithms in one such NP-hard problem called longest path problem.

4 Background

In computation theory problems can be broadly divided into two parts namely P and NP. The first class of problems is those for which there exists a deterministic Turing machine (Enigma machine) which can accomplish the task in polynomial time. Examples of such problem are linear search and most of the sorting algorithms. For the other class of problems, there exists a non-deterministic Turing machine which can accomplish the given task in polynomial time [1].

These problems are further of two types namely NP-complete and NP-hard. In NP-complete problems; though they are non-deterministic algorithms for solving problem, however, if the solution is given it is not difficult to verify whether it is correct in polynomial time. These are generally decision-type problems. The other kind of problems neither has an algorithm which takes place in polynomial time.

These problems are generally optimization problems. The critical path problem is one such problem.

The problem reduces to the longest path problem, which calls for finding out the longest path in the graph. There are two versions of the problem one is the NP-complete and the other is NP-hard.

In the first version, the problem is to find out there exists a path having K vertices? The problem is NP complete as if a path is given it can be easily said whether the solution is correct or not. The other version of the problem calls for finding out the longest path in the given graph, this version is NP-hard. During the literature review, it was found that the NP completeness of the first problem can be easily proved by converting it into Hamiltonian cycle problem [17].

As per the literature review, it was found that the approximation algorithms using color coding, though has minimum complexity it has, as yet not been verified on all the possible benchmarks.

Applications:

1. **Critical Path Method for scheduling on activity.** Critical Path Method is a technique that helps us to plan various projects consisting of numerous activities. Usually, some of these activities are dependent on others, i.e., in order to begin with some of our activities, we need to finish some others, thus making it a complex project. It also helps us to find the total time or the maximum time taken to complete our project and also points out more prior activities. Thus, helping us to determine various factors such as cost and speed [18].
2. **Layered Graph Drawing.** Longest path problem helps in layered graph drawing. It is a technique of drawing graphs where in, the edges are generally directed downwards and the vertices are generally in some particular horizontal row, such graphs reduce the possibility of getting the edges which cross and also reduce the number of inconsistently oriented edges. Both of which are NP-hard problems [19].
3. **Sequence Alignment Algorithms.** The longest path problem has also been used by some researchers to handle one of the most important problems in Bioinformatics that is Sequence alignment [1, 20].

5 Proposed Work

The longest path problem, as stated earlier is an NP-Hard problem. The work proposes a novel solution to solve the problem. The procedure has been explained in this section. The technique has been implemented and the results are encouraging. The results have been presented in the next section.

The application of GA to a problem requires an effective problem reduction approach. The first step of GA is population generation. The population consists of chromosomes. The number of cells in a chromosome is determined as follows.

For the graph with N vertices, the nearest power of two is found. Let this power be K then,

$$N \leq 2^K \tag{1}$$

Now each chromosome consists of N multiplied by K ($N \times K$) number of cells. This is done so that a chromosome can be divided into N parts each containing K cells. These K cells (a binary number) is converted into a decimal number and its modulus with n is then taken. The number obtained, if same as that obtained earlier (in any set of the same chromosome), then the bits are re-randomized. This procedure essentially generates a permutation of the n numbers. However, the number of chromosomes would determine the number of permutations. These limited permutations would lead us to the best (or almost best) solution. This logic also justifies the need of using GA.

This is followed by assigning fitness to each chromosome. The fitness is assigned as follows. The sequence generated (essentially a path, in the given graph), has some cost. This cost can be obtained by the given graph. In the implementation, the cost matrix of the graph is given as the input. The fitness of a chromosome is directly related to the cost. A chromosome having more cost should have more fitness. The formula employed to find the fitness is as follows.

$$\text{Fitness} = K \times (\text{The distance of the path determined by chromosome, where } K \text{ is a parameter}) \tag{2}$$

This is followed by Roulette Wheel Selection. The selection procedure has been explained in the preceding sections. The work uses one-point crossover and two-point crossover but the results of the two-point crossover have been presented in the following section. This is because on an average one-point crossover does not show any better result as compared Fig. 2.

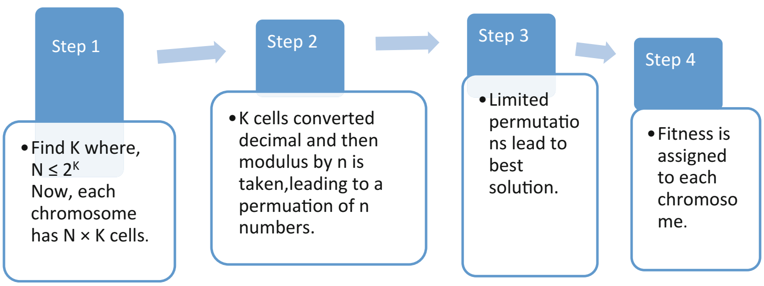


Fig. 2 Proposed algorithm for the longest path problem

The crossovered population is then made to undergo mutation. The stopping criterion is the number of generations (as stated in the next section) or the stagnation in the average fitness value.

6 Result and Conclusion

The problem has been implemented in MATLAB. The code has been tested on a few graphs of moderate size. The size of the graphs for which the algorithm was tested was 5, 7, 8, 9, and 10. The number of generations was restrained owing to the limited time. However, the results were encouraging. Table 2 gives the summary of the results and the actual answer.

Initial Population: 20
Number of generations (Maximum): 50
Crossover: Mentioned in the previous section.
Mutation: Mentioned in the previous section.
Selection: Roulette wheel.

The graphs were selected keeping in consideration the completeness of the path and feasibility of maximum number of paths Figs. 3 and 4.

Table 2 Results

n	Path found by the algorithm	Actual
5	34	34
5	32	34
5	34	34
5	30	34
5	29	34
7	43	47
7	42	47
7	47	47
7	45	47
7	47	47
9	62	65
9	62	65
9	62	65
9	62	65
9	62	65
10	90	90
10	82	90

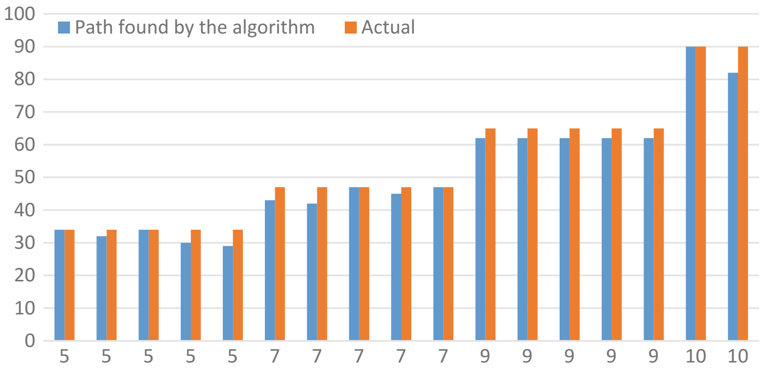


Fig. 3 Graphical representation of the results

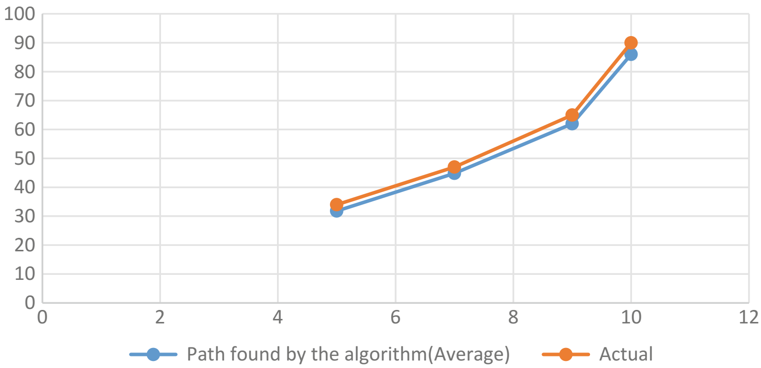


Fig. 4 Comparison of average cost in various trials

The work implements the proposed algorithm for the critical path problem. The problem is now being tested for larger population. Also, DGA is being applied to solve the problem. An extensive literature review of DGA has already been carried out [21]. The technique has also been applied to dynamic travelling salesman problem [22].

References

1. Bhasin, H.: Algorithm Analysis and Design. Oxford University Press (2015)
2. Cormen, L., Rivest, S.: Introduction to algorithms. Second Edition, Prentice Hall of India (1990)
3. Goldberg, D.E.: Genetic algorithms in search, Optimization and Machine Learning. Addison-Wesley, Reading, MA (1989)

4. Kitchenham, B.A., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007–01, School of Computer Science and Mathematics, Keele University (2007)
5. Hardgrave, W.W., Nemhauser, G.L.: On the relation between the traveling-salesman and the longest-path problems. Oper. Res. **10**, 647 (1962)
6. Bhasin, H., Singla, N.: Harnessing cellular automata and genetic algorithms to solve travelling salesman problem. ICICT, 72–77 (2012)
7. Ioannidou, K., Mertzios, G.B., Nikolopoulos, G.B.: The longest path problem is polynomial on interval graphs. In: MFCS'09, Springer-Verlag, Berlin, Heidelberg (2009)
8. Karger, D., Motwani, R., Ramkumar, G.: On approximating the longest path in a graph. Algorithmica **18**(1), 82–98 (1997)
9. Uehara, R., Uno, Y.: Efficient algorithms for the longest path problem. In: ISAAC (2004)
10. Uehara, R., Valiente, G.: Linear structure of bipartite permutation graphs and the longest path problem. Inf. Process. Lett. **103**(2), 71–77 (2007)
11. Ghosh, E.: Hamiltonicity and longest path problem on special classes of graphs. Research Thesis, Indian Institute of Technology, Madras (2011)
12. Ioannidou, K., Nikolopoulos, S.D.: The longest path problem is polynomial on cocomparability graphs. Lect. Notes Comput. Sci. Springer **6410**, 27–38
13. Bhasin, H., Ahuja, G.: Harnessing genetic algorithm for vertex cover problem. IJCSE (2012)
14. Bhasin, H., Singla, N.: Genetic based algorithm for N-puzzle problem. Int. J. Comput. Appl. **51**(22) (2012)
15. Bhasin, H., Mahajan, R.: Genetic algorithms based solution to maximum clique problem, Int. J. Comput. Sci. Eng. **4**(8) (2012)
16. Bhasin, H., Behal, G., Aggarwal, N., Saini, R.K., Choudhary, S.: On the applicability of diploid genetic algorithms in dynamic environments. Soft Computing, 1–8. Springer (2015)
17. Karp, R.: Reducibility among combinatorial problems. Complexity of computer computations. In: Miller, R., Thatcher, J. (eds.) Plenum Press pp. 85–103 (1970)
18. Senior, B.A.: Critical Path Method Implementation Drawbacks: A Discussion Using Action Theory
19. Suderman, M.: Layered Graph Drawing. ACM, McGill University, Montreal, Quebue, Canada (2005)
20. Attwood, T.K., et al.: Introduction to Bioinformatics. Pearson Publications (2009)
21. Bhasin et al., H.: On the applicability of Diploid Genetic Algorithm. AI and society. Springer **31**(2), 265–274 (2016)
22. Bhasin, H., et al.: On the applicability of Diploid Genetic Algorithm in Dynamic Environments. Soft Computing, Springer, First Online, 1–10 (2015)

Springer

http://www.springer.com/978-981-10-5686-4

Soft Computing: Theories and Applications
Proceedings of SoCTA 2016, Volume 1
Pant, M.; Ray, K.; Sharma, T.K.; Rawat, S.;
Bandyopadhyay, A. (Eds.)
2018, XXII, 865 p. 302 illus., Softcover
ISBN: 978-981-10-5686-4