International Conference on Machine Learning and Computational Intelligence-2017
International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 7 | ISSN : 2456-3307

Flower Pollination Algorithm for the Orienteering Problem

Madhushi Verma*[1], Tanvi Bothra[2], Surabhi Agarwal[2], K. K. Shukla[2]
[1]Department of Computer Science Engineering, Bennett University, Greater Noida, UP, India
[2]Department of Computer Science and Engineering, IIT(BHU), Varanasi, UP, India

ABSTRACT

The orienteering problem is an NP-Hard combinatorial optimization problem where the aim is to determine a Hamiltonian path that connects the stated source and target and includes a subset of the vertex set V such that the total collected score is maximized within the given time bound (T_max). Orienteering problem finds application in logistics, transportation, tourism industry etc. We have proposed an algorithm FPA_OP that can be implemented on complete graphs and its performance has been evaluated using standard benchmarks. Also, the results thus obtained have been compared against the latest heuristic for OP i.e. GRASP and it has been shown that for larger T_max, FPA_OP outperforms GRASP. Therefore, the decision maker can implement FPA_OP if he is willing to achieve a larger total collected score at the cost of time delay.

Keywords : Flower pollination algorithm, Metaheuristic, Orienteering problem, NP-Hard problems.

I. INTRODUCTION

Most of the optimization problems are NP-Hard and to solve these problems, we need efficient algorithms that can generate optimal solutions in polynomial time. However, it is difficult to obtain an algorithm that simultaneously possesses three properties: (1) computes optimal solutions, (2) for any instance and (3) in polynomial time [1]. To tackle the NP-Hard optimization problems, the decision maker needs to compromise with at least one of the above stated requirements. The problems that have large inputs and which cannot be solved in polynomial time can be dealt with using heuristic algorithms. Heuristic algorithms have the advantage of computing a solution for NP-Hard or NP-Complete problems with tolerable time and space complexity at the cost of optimality of the solution i.e., one needs to compromise with the quality of the solution to generate one with acceptable time and space complexity. However, the solution computed using a heuristic is most of the times a near to optimal solution and for real life applications it is sufficient to have an approximate or partial solution.

A metaheuristic is a combination of some local improvement methods and higher level techniques or strategies. It is basically an iterative generation process that helps in searching, generating or finding a heuristic (partial search algorithm) that explores and exploits the search space efficiently, avoids getting trapped in the local optima and performs a robust search to determine the near to optimal solution for the optimization problem at hand from the solution space [2, 3]. The advantage of metaheuristics is that it can also tackle the optimization problems with nonlinearity and multimodality. These days in industry and engineering applications, the problem under consideration is extremely complex and to generate an optimal solution for such problems is a challenging task. It has been found by several researchers that metaheuristic algorithms are quite efficient for solving such problems and the latest

trend is to apply nature-inspired metaheuristics for solving the critical optimization problems. Several nature-inspired metaheuristics have been developed by the researchers after studying the complex biological systems. These metaheuristics include the genetic algorithm, bat algorithm, firefly algorithm, ant colony optimization algorithm, particle swarm optimization algorithm etc. [4].

In this paper, the flower pollination metaheuristic has been implemented for the orienteering problem (OP). The OP finds a lot of practical applications especially in the fields like the tourism industry, logistics, transportation, networks etc. [5]. Few exact algorithms were suggested to solve OP [6, 7]. However, as OP is considered to be an NP-Hard problem, practically it is not possible to use exact algorithms for large instances. Therefore, the best way to tackle OP is to use some heuristic or approximation algorithms. Tsiligirides suggested the first heuristic for OP [9]. Since then several heuristics have been proposed [8, 10, 11, 12, 13, 14]. One of the latest heuristic for OP was introduced by Campos et al. [15]. Few approximation algorithms have also been proposed by Blum et al., Johnson et al., Fomin et al., etc. [16, 17, 18]. Some algorithms have also been proposed that can be applied on incomplete graphs as well [19. 20, 21]. Fuzzy and intuitionistic fuzzy versions of the orienteering problem have also been studied and a few models have been introduced to solve the problem [22, 23].

II. PREREQUISITES

2.1 Problem Definition

OP can be represented by an undirected weighted graph $G(V, E)$ where V and E signifies the set of vertices and set of edges respectively. The goal in OP is to compute a Hamiltonian path P which connects the stated source (v_1) and target (v_n), includes a subset (V') of the vertex set V such that the total collected score can be maximized with the given time budget (T_{max}). A score function $S: V \to \Re^+$ is associated with each vertex and a time function $t: E \to \Re^+$ is associated with each edge. Therefore, for a subset $E' of\ E$ and $V' of\ V$ we have $t(E') = \sum_{e \in E'} t(e)$ and $S(V') = \sum_{v \in V'} S_v$ [5]. OP can be depicted as an integer programming problem as shown below:

$$Max \sum_{i=1}^{N-1} \sum_{j=2}^{N} S_i x_{ij} \quad (1)$$

$$\sum_{j=2}^{N} x_{1j} = 1 \quad , \quad \sum_{i=1}^{N-1} x_{iN} = 1 \quad (2)$$

$$\sum_{i=1}^{N-1} x_{ik} \leq 1 \quad \forall\ k = 2, \ldots\ldots, N-1 \quad (3)$$

$$\sum_{j=2}^{N} x_{kj} \leq 1 \quad \forall\ k = 2, \ldots\ldots, N-1 \quad (4)$$

$$\sum_{i=1}^{N-1} \sum_{j=2}^{N} t_{ij} x_{ij} \ \leq\ T_{max} \quad (5)$$

$$2 \leq u_i \leq N \quad \forall\ i = 2, \ldots\ldots, N \quad (6)$$

$$u_i - u_j + 1 \leq (N-1)\left(1 - x_{ij}\right) \quad \forall\ i, j = 2, \ldots\ldots\ldots, N \quad (7)$$

$$x_{ij} \in \{0,1\} \quad \forall\ i, j = 1, \ldots\ldots, N \quad (8)$$

Variable u_i denotes the position of vertex v_i in the path. If vertex v_j is visited after vertex v_i then x_{ij} = 1 else x_{ij} = 0. The objective function of OP which is maximization of the total collected score is denoted by Eq. 1. The constraint that a path has v_1 as its source and v_N as its target is depicted by Eq. 2. Eq. 3 – 4 ensures that no vertex is visited more than once and the path remains connected. Eq. 5 takes care of the condition that the path satisfies the given time budget (T_{max}). Eq. 6 – 7 ensures the elimination of sub-tours [5].

2.2 Flower Pollination Algorithm (FPA)

The flower pollination algorithm (FPA) is a nature-inspired metaheuristic that was introduced by Xin-She Yang in 2012 [2] and here it has been used to solve the NP-Hard orienteering problem. FPA is inspired from the pollination process of the flowers. Pollination is the reproduction process of the flowers which is carried out through agents like bees, bats, birds, insects and other animals. These agents are called pollinators. Pollination can be either abiotic or biotic and can take place in two ways, namely self-

pollination and cross-pollination. When pollens are carried through some pollinators like insects then the pollination that takes place is called biotic and about 90% of the pollination activity is biotic. In 10% of the cases, pollens are carried through natural carriers like wind, water etc. and are termed as abiotic. If the pollination takes place with pollen from a flower of a different plant then it is called cross-pollination and if the fertilization happens due to the pollen coming from either the same flower or a different flower of the same plant then it is termed as self-pollination. Another term that can be associated with this process of pollination is the flower constancy. Honeybee is a pollinator that helps in implementing this phenomenon called flower constancy where the pollinators tend to visit only a specific species of flowers and ignore the other species that exist. Pollinators like birds, bees, insects etc. can fly to long distances. Therefore, biotic, cross-pollination over long distances can be termed as global pollination. Also, the behaviour of the biotic pollinators i.e., their jumps and flying distances etc. follows the Levy distribution as stated by Xin-She Yang [2].

III. ALGORITHM FPA_OP

Input: A graph $G(V,E)$ with t_{ij} (time taken to traverse) value of each edge (e_{ij}) connecting vertex v_i and $v_j \in V$, S_i (score) value of each vertex $v_i \in V$.
Output: A Hamiltonian path with the highest possible collected score such that total travel time is within the specified time budget T_{max}.

Initialize a population of n flowers /pollen gametes with random solutions
Find the best solution R^* in the initial population
Define a switch probability sp $\in$ [0,1]
while $(t < No\_of\_Iterations)$
for $i = 1$: n *(all n flowers in the population)*
if $rand < sp$
Do global pollination via $index2 = index1 + L(index(R^*) - index1)$
else
Randomly choose j and k among all the solutions
Do local pollination via $index2 = index1 + \epsilon(j - k)$
end if
Evaluate new solutions
If new solutions are better, update them in the populat
end for
Find the current best solution R^*
end while

In the above stated algorithm [2], initially n pollens are generated randomly where each pollen represents a possible solution i.e., a path satisfying the constraint that the total time taken by the path is less than the upper bound T_{max}. Then for the valid paths three values are evaluated: (1) the total time taken by the path, (2) the total score collected by the path and (3) the ratio of $score/time(S_i/\ t_{ij})$ for each of the valid paths. Then these paths are stored in a priority queue on the basis of the $(S_i/\ t_{ij})$ ratio. To determine the best solution R^* in the initial solution, any of the selection methods like the tournament selection, random selection, (μ, λ) selection, roulette wheel selection etc. can be implemented. The switch probability $sp \in [0,1]$ controls the type of pollination to be performed i.e., global pollination or local pollination. $rand$ is a randomly generated number and if it is less than sp then global pollination is performed else local pollination is performed. In case of global pollination, a function is used that randomly generates the value for $index1$. Then using $index1$ and Levy distribution L , the value for $index2$ is computed. The value for L is calculated using the following equation:

$$L \sim \frac{\lambda \Gamma(\lambda) sin(\Pi\lambda/2)}{\Pi} \frac{1}{s^{1+\lambda}} \quad (4.10)$$

Where, $\lambda = 1.5$ and s denotes the step size. To make the step size appropriate, so that neither it is too large nor too small, its value is calculated as

size of the priority queue/5 . Similarly, in local pollination again the value of $index1$ is considered and two randomly generated solutions j and k are used to calculate the value for $index2$. The value for ϵ is randomly chosen from the interval [0,1]. After the two parents are determined (i.e., the paths at $index1$ and $index2$), a crossover operation is performed to compute the child paths. Two points are found randomly in the parent2 (at $index2$) and the nodes that exists between the two points of parent2 are inserted in parent1, one by one at the best possible location (which leads to minimum increment in the total time taken). This way child1 is formulated. In a similar manner, two points are selected in parent1 and the nodes lying between the two points are added one by one to parent2 at its best location and this way another path i.e., child2 is created. Then it is checked whether these new paths (child1 and child2) are valid or not i.e., their total time taken is less than T_{max} or not. If the new path is valid then it is stored in the

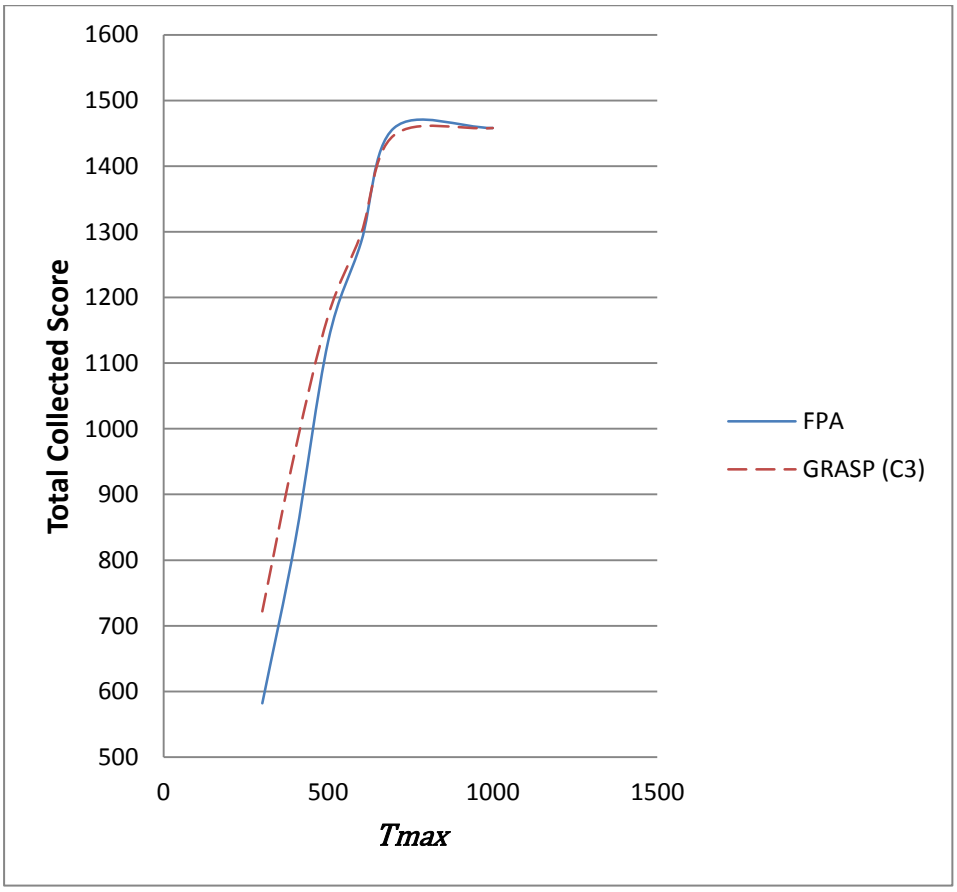

Figure 1. Comparison of the total collected score value achieved by GRASP and FPA algorithms for different T_{max} values when applied on a graph with 102 nodes, source=1, destination=102.

queue else it is discarded. If the new path that has been generated has a better score than the previous best solution, then it is updated in the queue. At the end, when the algorithm terminates, the path obtained with the highest value of total collected score forms the final solution.

IV. EXPERIMENTAL ANALYSIS

The code for *FPA_OP* was developed in C++ and compiled using CodeBlocks on an Intel Core i5 650 at 2.20 GHz. The code was implemented on instances with 32, 33, 64, 66, 102, 150 etc. nodes. Each instance

represents a complete graph. The results obtained for FPA_OP were compared against the best known algorithm in the literature for OP viz. the GRASP algorithm suggested in [15]. The results obtained (average of 10 runs) by FPA_OP were compared with the C3 method of GRASP and it was found that FPA_OP helps in obtaining higher total collected score value for larger T_{max}. However, at lower T_{max} values the results obtained through GRASP were better than FPA_OP. Figure 1 is a plot for a complete graph with 102 nodes, source=1 and destination=102. The results for the total collected score by GRASP and FPA_OP algorithms are compared for different T_{max} values and the observation stated above can be clearly seen in the plot. Therefore, the FPA_OP algorithm can be preferred in the cases were achieving a higher total collected score is the priority and the delay in time can be tolerated.

V. CONCLUSION

A meta-heuristic called the flower pollination algorithm suggested by Yang [2] has been implemented for OP (FPA_OP) and the results thus obtained for instances with different number of nodes has been compared with those obtained by running the GRASP algorithm [15]. It was found that in situations where achieving a better score is the priority at the cost of time delay, FPA_OP algorithm can be preferred as it helps in obtaining a higher total collected score than GRASP for larger values of T_{max}.

VI. REFERENCES

[1]. Williamson, D.P., and Shmoys, D.B. The Design of Approximation Algorithms, Cambridge University Press, (2010).

[2]. Yang, X.S. Flower Pollination Algorithm for Global Optimization, LNCS 7445, (2012): 240–249.

[3]. Yang, X.S., Karamanoglu, M. and He, X. "Multi-objective Flower Algorithm for Optimization." Proceedings of the International Conference on Computational Science, ICCS 2013, (2013): 861 – 868.

[4]. Fister, Jr. I., Yang, X. S., Fister, I., Brest, J., and Fister, D. "A Brief Review of Nature-Inspired Algorithms for Optimization." Elektrotehniski Vestnik 80, no. 3 (2013): 1–7.

[5]. Vansteenwegen, P., Souffriau, W., and Oudheusden, D. V. "The orienteering problem: A survey." European Journal of Operational Research 209, (2011): 1–10.

[6]. Laporte, G., and Martello, S. "The Selective Traveling Salesman Problem." Discrete Applied Mathematics 26, (1990): 193-207.

[7]. Hayes, M., and Norman, J. M. "Dynamic Programming in Orienteering: Route Choice and the Siting of Controls." Journal of the Operational Research Society 35, no. 9 (1984): 791-796.

[8]. Fischetti, M., Salazar, J., and Toth, P. "Solving the orienteering problem through branch-and-cut." INFORMS Journal on Computing 10, (1998): 133–148.

[9]. Tsiligirides, T. "Heuristic methods applied to orienteering." Journal of the Operational Research Society 35, (1984): 797–809.

[10]. Ramesh, R., and Brown, K. "An efficient four-phase heuristic for the generalized orienteering problem." Computers and Operations Research 18, (1991): 151–165.

[11]. Golden, B., Levy, L., and Vohra, R. "The orienteering problem." Naval Research Logistics 34, (1987): 307–318.

[12]. Wang, Q., Sun, X., Golden, B. L., and Jia, J. "Using artificial neural networks to solve the orienteering problem." Annals of Operations Research 61, (1995): 111–120.

[13]. Gendreau, M., Laporte, G., and Semet, F. "A tabu search heuristic for the undirected selective travelling salesman problem." European Journal of Operational Research 106, (1998): 539–545.

[14]. Schilde, M., Doerner, K. F., Hartl, R. F., and Kiechle, G. "Metaheuristics for the bi-objective orienteering problem." Swarm Intelligence 3, (2009): 179-201.

[15]. Campos, V., Marti, R., Sanchez-Oro, J., and Duarte, A. "GRASP with Path Relinking for the Orienteering Problem." Journal of the Operational Research Society 2013, (2013): 1–14.

[16]. Blum, A., Chawla, S., Karger, D. R., Lane, T., Meyerson, A., and Minkoff, M. "Approximation Algorithms for Orienteering and Discounted-Reward TSP." Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03), (2003): 1-10.

[17]. Johnson, D., Minkoff, M., and Phillips, S. "The prize collecting steiner tree problem: Theory and practice." Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms, (2000): 760–769.

[18]. Fomin, F. V., and Lingas, A. "Approximation algorithms for time-dependent orienteering." Information Processing Letters 83, (2002): 57–62.

[19]. Ostrowski, K., and Koszelew, J. "The Comparison of Genetic Algorithms which Solve Orienteering Problem using Complete and Incomplete Graph." Informatyka 8, (2011): 61-77.

[20]. Verma, M., Gupta, M., Pal, B., and Shukla, K. K. "Roulette Wheel Selection based Heuristic Algorithm for the Orienteering Problem." International Journal of Computers and Technology 13, no. 1 (2014): 4127-4145.

[21]. Verma, M., Gupta, M., Pal, B., and Shukla, K. K. "A Stochastic Greedy Heuristic Algorithm for the Orienteering Problem." Proceedings of the 5th International Conference on Computer and Communication Technology (ICCCT), (2014): 59-65.

[22]. Verma, M., and Shukla, K. K. "Application of Fuzzy Optimization to the Orienteering Problem." Advances in Fuzzy Systems 2015, (2015): 1-12.

[23]. Verma, M., and Shukla, K. K. "Fuzzy Metric Space Induced by Intuitionistic Fuzzy Points and Its Application to the Orienteering Problem." IEEE Transactions on Fuzzy Systems, 24, no. 2 (2016): 483-488