ISSN 1749-3889 (print), 1749-3897 (online)
International Journal of Nonlinear Science
Vol.24(2017) No.3, pp.169-174

Using Genetic Algorithm for the Discrete Domination Over Time Problem

Nazanin Abbasnezhad *, Javad Mehri-Tekmeh
Faculty of Mathematical Sciences, University of Tabriz, Tabriz, Iran
(Received 5 March 2017, accepted 7 September 2017)

Abstract: Domination in graphs has been studied by many researchers in the graph theory and has applications in various fields that many of these problems in real world involve the notion of time. In this paper, we generalize the standard domination problem by introducing an element of time and call it a discrete domination over time problem. We formulate this problem and show that it is NP-complete. We find the best value of crossover probability and run the genetic algorithm for some graphs.

Keywords: Domination in graphs; Domination over time; Discrete time; Genetic algorithm

1 Introduction

Domination in graphs has been studied by many researchers in the graph theory and has applications in various fields such as, computer communication networks, land surveying, monitoring, communication and facility location like the location of hospital and fire station [13]. A domination set S is a subset of the nodes in a graph such that every node in the graph either belongs to S or has a neighbor in S. The standard domination problem searches a domination set of minimum cardinality. The first results on domination was published in 1977 by Cockayne and Hedetniemi [5]. Following the publication of this paper, domination in graphs has become an area of focus and a vast body of research has been developed [1, 13]. Several polynomial time algorithm for the domination problem have been constructed on trees, internal graphs and permutation graphs. Cockayne et al. [4] gave the first linear-time algorithms for the domination problem in trees.

Many problems in real world involve the notion of time that isn't captured by standard domination problem. For example, communication and facility location like the location of hospital and fire station. Considering an element of time, we generalize the standard domination problem. Also, there are many network problems in which the time parameter has been studied by some researchers, including minimum cost flow over time and Maximum flow over time problems [7–9, 21].

The genetic algorithm (GA) is the most popular technique in evolutionary computation research. GAs are based on the principle of genetics and evolution [20]. The GA was introduced by Holland in the 1960s and the 1970s [16]. Srinivas and Patnaik [22] examined the crossover and mutation probability to optimize the genetic algorithm's performance. Goldberg [12] and Chambers [2] have discussed several applications of GAs in optimization problems. It has been applied to many fields in engineering, computing, biology and has been used to solve the problems such as knapsack problem [3, 15], traveling salesman problem [17, 19] and others [6, 18]. Also, the dominating set problem is NP-complete [13] and in general, it can not be solved exactly in polynomial time. So, the various GAs has been applied for solving this problem. Hedar and Rashad [14] gave the hybrid genetic algorithm for minimum dominating set problem and Giap and Ha [11] proposed the parallel genetic algorithm for this problem.

To start, we present some notations and the programming model that are needed in the paper. Suppose that a graph $G = (V, E)$ is a simple graph with a vertex set V of order $|V| = n$ and an edge set E of size $|E| = m$. Each arc (i, j) has a transit time τ_{ij}. The adjacency matrix of the graph G is denoted by A.

Our goal in this paper is to introduce and formulate a discrete domination over time (DDOT) problem and we show that this problem is NP-complete. Finally, we empirically extract a suitable crossover probability and we also run a genetic algorithm on some graphs.

*Nazanin Abbasnezhad. *E-mail address*:n_ abbasnezhad@tabrizu.ac.ir.

IJNS.2017.12.15/986

The present paper organized as follows. In the next section, we introduce a discrete domination over time (DDOT) problem. The section 3 describes a genetic algorithm (GA) to solve a DDOT problem. In section 4, we report computational results to solve various instances.

2 Discrete domination over time

We introduce the generalized model for domination problem in discrete time. In order to illustrate it with an example, consider a network of the possible locations of fire stations by a graph G. The station at vertex u is active and can serve to the directly connected vertices, if $x_u(t) = 1$ (depends on the time parameter). We need to do monitoring in the specific times up to the time horizon T. Every direct path has a known delay in reaching destination, but there is no delay in using from the storage of each station. We need to identify which station is active in a time point t such that the number of active stations in the network will be minimum.

Definition 1 *A discrete dominating over time is a function $x_j : \mathbb{Z} \to \{0, 1\}$, $j \in V$ such that the node j is not active ($x_j(t) = 0$), for $t \notin \{0, 1, \ldots, T\}$ and for every $\theta \in \{0, 1, \ldots, T\}$, it satisfies the following constraints*

$$\sum_{t=0}^{\theta} x_j(t) + \sum_{i \in V} A_{ij} \sum_{t=0}^{\theta} x_i(t - \tau_{ij}) \geq \theta, \qquad \forall j \in V, \tag{1}$$

where A is the adjacency matrix of the graph G, $\tau : E \to \mathbb{R}_{\geq 0}$ is a (symmetric) delay function (transit times).

The set of active nodes (Dom_t) at a time point t is a minimum dominating over time set, if $\sum_{j \in V} \sum_{t=0}^{T} x_j(t)$ has the minimum value among all discrete dominating over time functions. We consider $\tau \in \{0, 1, \ldots, T\}$.

We formulate a Discrete Domination Over Time problem (DDOT) in which the time variable $t \in \{0, 1, \ldots, T\}$.

$$\begin{aligned} \text{DDOT:} \qquad & \min \quad \sum_{j \in V} \sum_{t=0}^{T} x_j(t) \\ & \text{s.t.} \\ & \sum_{s=0}^{t} x_j(s) + \sum_{i \in V} A_{ij} \sum_{s=0}^{t} x_i(s - \tau_{ij}) \geq t, \\ & \qquad\qquad\qquad\qquad j \in V, \;\; t \in \{0, 1, \ldots, T\}, \\ & x_j(t) \in \{0, 1\}, \qquad j \in V, \;\; t \in \{0, 1, \ldots, T\}, \end{aligned} \tag{2}$$

where A is the adjacency matrix of the graph G, $\tau : E \to \mathbb{R}_{\geq 0}$ is a known (symmetric) delay function.

3 Genetic algorithm

David Johnson [10] showed that the dominating set problem is NP-complete and we have the following theorem.

Theorem 1 *The DDOT problem is NP-complete.*

Proof. If all the transit times on the arcs are zero and the time horizon $T = 1$, then the DDOT problem reduces to a standard domination problem. NP-completeness of the standard domination problem implies that the DDOT problem is NP-complete. ■

So, we apply GA to solve the DDOT problem. In GA, we represent a candidate solution to a problem as a chromosome in a generation. Each solution of problem is represented by 0-1 vector x with dimension equal to $n(T + 1)$ where T is the time horizon. If the i^{th} node at a time point t is contained in the solution, then a component $x_{i(t+1)}$ will be set to 1, (this node will be active) otherwise $x_{i(t+1)}$ has the value 0.

Fitness Function

The goal of the algorithm is to solve the DDOT problem. So, the fitness function for problem needs to minimize the number of active nodes until T contained in this chromosome and the solution has to be the discrete dominating over time function. The fitness function is shown as follows [11]:

$$fit = \begin{cases} -\infty & x \text{ is not a discrete dominating over time function} \\ n(T+1) - \sum_{t=0}^{T} |Dom_t| & x \text{ is a discrete dominating over time function} \end{cases} \tag{3}$$

In practice, instead of $-\infty$, we set this value equal to 0. The value of fitness function increases as the number of nodes in $Dom_t, (t = 0, 1, \ldots, T)$ decrease. Furthermore, the total number of active nodes is smaller or equal to $n(T+1)$ (the upper bound for the number of active nodes), therefore the fitness value is always non-negative.

Selection

We apply the roulette wheel selection method. In this method, n chromosomes are selected for the next generation by turning the wheel n times.

Crossover

We apply the one-point crossover. In this method, we select parents set of chromosomes using selection probability and choose a random pair of chromosomes from parents. Then, we choose a random cutting point and swap the tails of two parents to make a new offspring. Finally, we replace the parents with the offsprings in the next generation.

Mutation

We use the standard mutation. In this process, we randomly select a chromosome and choose two genes at random and swap them.

4 GA performance

In this section, we present the results of the genetic algorithm performed on some simple graphs. GAs starts with an initial population generated at random. Each individual represents a solution of the DDOT problem.

Example 2 *Consider the example shown in Figure 1 with six nodes and nine arcs. The time delays are as follows:*

$$\begin{array}{lllll} \tau_{12} = 2, & \tau_{13} = 5, & \tau_{23} = 1, & \tau_{24} = 3, & \tau_{25} = 4, \\ \tau_{35} = 7, & \tau_{36} = 2, & \tau_{45} = 2, & \tau_{56} = 1. & \end{array}$$

The optimal value of the objective function is 16 and the value obtained by the genetic algorithm is 18. The optimal solution of DDOT is obtained as follows:

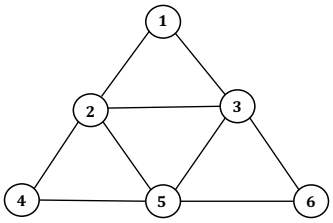

Figure 1: Hajos graph for example 2.

$$x_1^*(t) = \begin{cases} 1, & t = 0, 2 \\ 0, & otherwise \end{cases} \qquad x_2^*(t) = \begin{cases} 1, & t = 0, 1, 2, 7 \\ 0, & otherwise \end{cases}$$

$$x_3^*(t) = \begin{cases} 1, & t = 0, 1, 2, 5 \\ 0, & otherwise. \end{cases} \qquad x_4^*(t) = \begin{cases} 1, & t = 0, 6 \\ 0, & otherwise. \end{cases}$$

$$x_5^*(t) = \begin{cases} 1, & t = 0, 1, 5, 7 \\ 0, & otherwise. \end{cases} \qquad x_6^*(t) = \begin{cases} 1, & t = 3, 8 \\ 0, & otherwise. \end{cases}$$

Table 4 shows the result of the algorithm for some experiments with crossover probability from 0.7 to 0.8 with the step value 0.01. The best value is 0.74. The graphs used for experiments in the Table 4 are randomly built graphs with 10 - 200 nodes and T=10. Transit times are also random nonnegative numbers. The one-point crossover probability and mutation probability are 0.74 and $0.26(= 1 - 0.74)$, respectively. The population size is equal to 80 and the maximum number of generations is 2000. For each graph, we ran the algorithm 10 times and recorded the best, average and worst solution.

The experiments show that the average result is good for the graphs with the number of nodes smaller than or equal to 100.

The Figure 2 represents the trend of the fitness function value with respect to the number of generations for a random graph with 210 nodes.

5 Conclusions

We generalized the domination problem in discrete time and presented the mathematical model of the discrete domination over time problem. We showed that the standard domination problem is the special case of the discrete domination over time problem. Since the standard domination problem is NP-complete, the DDOT problem is NP-complete, too. We used GA to solve it and found the best value of crossover probability, empirically. The experimental results show that the best answer is good enough for the problems.

The standard domination problem and 2-packing problem have duality properties. As a future work and to make the results better, it is a good point to start with the definition of 2-packing problem over time and then use it to improve the GA results.

Table 1: GA performance

Index	Nodes	Optimal Value	Algorithm's result		
			Best	Average	Worst
1	10	25	26	27.20	29
2	20	39	40	42	44
3	30	52	54	55.60	57
4	40	68	73	74.30	77
5	50	91	94	97.60	102
6	60	81	87	89.80	92
7	70	84	89	91.90	96
8	80	92	94	100.60	106
9	90	112	118	124.30	130
10	100	128	135	141.10	146
11	110	142	151	155	160
12	120	173	193	195.90	204
13	130	192	210	215.80	221
14	140	190	205	210.80	220
15	150	199	218	225.30	231
16	160	188	207	209.90	213
17	170	187	199	202.80	207
18	180	201	212	220.80	228
19	190	253	277	288.90	294
20	200	269	300	307.90	318

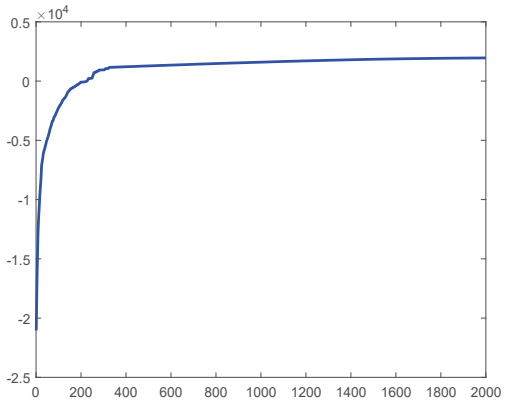

Figure 2: GA performance for a random graph with 210 nodes.

References

[1] C. Berge. *The Theory of Graphs*. Dover books on mathematics. Dover, 2001.

[2] L.D. Chambers. *The Practical Handbook of Genetic Algorithms: Applications, Second Edition*. CRC Press, Inc., 2000.

[3] P.C. Chu and J.E. Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(1):63–86, 1998.

[4] E. Cockayne, S. Goodman, and S. Hedetniemi. A linear algorithm for the domination number of a tree. *Information Processing Letters*, 4(2):41 – 44, 1975.

[5] E.J. Cockayne and S.T. Hedetniemi. Towards a theory of domination in graphs. *Networks*, 7(3):247–261, 1977.

[6] K. Deep, Sh. Barak, V.K. Katiyar, and A.K. Nagar. Minimization of molecular potential energy function using newly developed real coded genetic algorithms. *IJOCTA*, 2(1):51–58, 2012.

[7] L. Fleischer and M. Skutella. Minimum cost flows over time without intermediate storage. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-03)*, pages 66–75, 2003.

[8] L.R. Ford and D.R. Fulkerson. Constructing maximal dynamic flows from static flows. *Operations Research*, 6(3):419–433, 1958.

[9] L.R. Ford and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[10] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co. New York, NY, USA. 1979.

[11] C.N. Giap and D.T. Ha. Parallel genetic algorithm for minimum dominating set problem. In *2014 International Conference on Computing, Management and Telecommunications (ComManTel)*, pages 165–169, 2014.

[12] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[13] T.W. Haynes, S. Hedetniemi, and P. Slater. *Fundamentals of Domination in Graphs*. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis, 1998.

[14] A.R. Hedar and I. Rashad. Hybrid genetic algorithm for minimum dominating set problem. In *ICCSA (4)*, Lecture Notes in Computer Science, pages 457–467, 2010.

[15] A. Hoff, A. L?kketangen, and I. Mittet. Genetic algorithms for 0/1 multidimensional knapsack problems. In *Proceedings Norsk Informatikk Konferanse, NIK '96*, 1996.

[16] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[17] P. Jog, J.Y. Suh, and D.V. Gucht. Parallel genetic algorithms applied to the traveling salesman problem. *SIAM Journal on Optimization*, 1(4):515–529, 1991.

[18] A. Johar, S.S. Jain, and P.K. Garg. Transit network design and scheduling using genetic algorithm ? a review. *IJOCTA*, 6(1):9–22, 2016.

[19] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga, I. Inza, and S. Dizdarevic. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial Intelligence Review*, 13(2):129–170, 1999.

[20] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998.

[21] M. Skutella. An introduction to network flows over time. In J. Vygen W. Cook, L. Lov?sz, editor, *Research Trends in Combinatorial Optimization*, pages 451–482. Springer Berlin Heidelberg, 2009.

[22] M. Srinivas and L.M. Patnaik. Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(4):656–667, 1994.