Computational Aspects of Some Algorithms for The Multiperiod Degree Constrained Minimum Spanning Tree Problem

View the article online for updates and enhancements.

Computational Aspects of Some Algorithms for The Multiperiod Degree Constrained Minimum Spanning Tree Problem

Wamiliana[1,*], M Usman[1], Warsono[1] and Warsito[2]

[1]Dept. of Mathematics, Universitas Lampung, Indonesia
[2]Dept. of Physics, Universitas Lampung, Indonesia

*Email: Wamiliana.1963@fmipa.unila.ac.id

Abstract. Given a graph G(V,E), where V is the set of vertices and E is the set of edges connecting vertices in V, and for every edge e_{ij} there is an associated weight $c_{ij} \geq 0$, The Multi Period Degree Constrained Minimum Spanning Tree (MPDCMST) is a problem of finding an MST while also considering the degree constrained on every vertex, and satisfying vertices installation on every period. The restriction on the vertex installation is needed due to some conditions such as fund limitation, harsh weather, and so on. In this research some algorithms developed to solve the MPDCMST Problem will be discussed and compared. **Keywords:** multi periods, degree constrained, minimum spanning tree, computational aspect, comparative analysis

1. Introduction

Combinatorial or discrete optimization problems occur in many applications in daily-life problem, for example: scheduling problems (such as airline crew scheduling, database query design, and network design problem (transportation, electricity, communication, and so on). Moreover, combinatorial optimization problems occur in many diverse areas such as graph theory, linear and integer programming, number theory and artificial intelligence. Network design as part of combinatorial optimization mainly used the concept of graph in representing the problem. The nodes or vertices in graphs can be used to represent the component in network such as cities/computers/stations and so on, while the edges can be used to represent roads/cables/ train tracks and so on. The edges in graph can be assigned a number which can be used to represent distance/time/cost and so on. There is no a right way to draw a graph, an edge can be drawn as a straight line, a curve or other forms of line. Therefore, because of its flexibility, graph theoretical concepts are used in many networks design problems.

One problem that arises in Network design problem is the Multiperiod Degree Constrained Minimum Spanning Tree in which will be discussed later. This paper is organized as follow: Introduction is given in Section 1. In Section 2 the backbone of the problem and the problem itself will be discussed. In Section 3, Results and Discussion will be given, and in this section the comparative analysis will be discussed, followed by Conclusion.

2. The Problem: Multiperiod Degree Constrained Minimum Spanning Tree

One of the famous concepts in graph theory is the concept of tree. The tree is defined as a connected graph without any cycle. In chemistry, the tree structure was used by Cayley in 1857 to represent and counting the number of isomers of hydrocarbon C_nH_{2n+2}, and Kirchhoff developed the tree concept in

1847 to solved an electrical network using the system of simultaneous linear equations which give the current in each branch and around each circuit [1]. However, in network design problem, the concept of Minimum Spanning Tree (MST) is more applicable to be used.

Given an undirected, connected graph G(V,E), where associated for every edge $e_{ij} \in E$, there exists a cost/weight $c_{ij} \geq 0$, the Minimum Spanning Tree (MST) of G is the spanning tree T that minimizes the total cost. The spanning tree of a graph G is a tree which contains all the vertices of G. The first algorithm for finding a minimum spanning tree was suggested by Böruvka in 1926, who developed an algorithm for the most economical layout for a power-line network [2]. In many literatures Boruvka's algorithm also was called as Sollin's algorithm. Minimum spanning tree algorithms have been studied extensively and a variety of fast algorithms have been developed, including Prim's [3] and Kruskal's [4] algorithms. In many applications, Prims' and Kruskal's algorithms are preferable to be used.

The Degree Constrained Minimum Spanning Tree (DCMST) is concerned with finding a minimum-weight spanning tree whilst satisfying degree requirements on the vertices. The DCMST problem uses MST as the backbone and added restriction on every vertex. This problem arises in many network design problems such as: the design of telecommunication, transportation, energy networks, computer communication, sewage and plumbing. This problem had been investigated by many researchers and either exact or heuristics algorithms had been developed, but due to its NP completeness, heuristics are more likeable. Some heuristics for DCMST to be noted here are: greedy algorithm based on Prim's and Kruskal's algorithm [5], Genetic Algorithm [6], Iterative Refinement [7], Simulated Annealing [8], and Tabu Search [9-11].

The Multiperiod Degree Constrained Minimum Spanning Tree Problem is a problem derived from the DCMST by adding period restriction on every vertex. In real situation, it is possible that the process of installation or connection of some components to the network is done in some periods due to harsh weather, fund limitation or other factors. Moreover, there are also possible that some components are more important than others and need to be connected/installed right away. This problem was introduced in 2002 by Kawatra [12] and proposed hybrid method of branch exchange and Lagrangean relaxation which implemented with problems up to 100 vertices. The problem suggested by Kawatra used directed graph (used arcs instead of edges). Modifying the problem suggested by Kawatra by using undirected graph, some algorithms based on Kruskal's algorithms were developed using 3 periods and implemented using problems up to 100 vertices [13 -15], and in [16] the comparative analysis of some algorithms developed based on Kruskal's algorithm was given.

Because Kruskal's algorithm doesn't guarantee that the components installed are connected with the network during the process of installation of the components (although at the end of the process all components are connected), then developing algorithm based on Prim's algorithm is one alternative. In [17-18] some algorithms for solving the MPDCMST based on Prims' algorithms were developed.

In the undirected MPDCMST problem, HVT_i is defined as the set of vertices that must be installed/connected on the ith period of before, and $MaxVT_i$ as the maximum number of vertices can be installed/connected on i^{th} period. To illustrate the modified MPDCMST problem by using undirected graph, a simple example is given below:

Suppose there are 10 sites to be connected to build a water pipe network. One of the sites is a reservoir (denote as vertex 1 or central vertex). This site must already set in the network. The aim is to find the cheapest way to connect all sites in the network, but satisfied the requirements (some vertices must be connected in certain period or before, and also do not violated the degree restriction on every vertex). The cost for connecting those building with the reservoir is given in the following table:

Table 1. The cost for connecting edge e_{ij}

Edge	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	e_{17}	e_{18}	e_{19}	$e_{1,10}$	e_{23}	e_{24}	e_{25}	e_{26}	e_{27}	e_{28}
Cost	740	572	447	835	427	807	362	832	120	221	109	276	741	978	352
Edge	e_{29}	$e_{2,10}$	e_{34}	e_{35}	e_{36}	e_{37}	e_{38}	e_{39}	$e_{3,10}$	e_{45}	e_{46}	e_{47}	e_{48}	e_{49}	$e_{4,10}$
Cost	368	403	505	921	757	884	369	886	545	639	253	750	251	187	857
Edge	e_{56}	e_{57}	e_{58}	e_{59}	$e_{5,10}$	e_{67}	e_{68}	e_{69}	$e_{6,10}$	e_{78}	e_{79}	$e_{7,10}$	e_{89}	$e_{8,10}$	$e_{9,10}$
Cost	807	926	781	605	112	559	411	473	743	882	693	851	509	434	828

Assume that we use degree bound 3 for every vertex, and the number of periods also 3. Moreover, we set $HVT_1 = \{2\}$, $HVT_2 = \{3\}$, and $HVT_1 = \{4\}$, and $MaxVT_i = \left\lfloor \frac{(n-1)}{3} \right\rfloor$. Since vertex 1 (reservoir) must be already in the network, then for the first period, $MaxVT_1 = 3$. Thus, there are two more vertices with the smallest cost can be connected in the first period except vertex 2. Therefore, for the first period, we get the following:
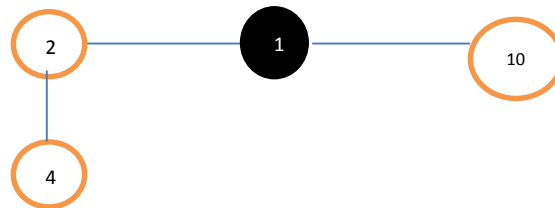

Figure 1. Network after the first period

The total cost for the first period is 969.

For the second period, vertex 3 is priority vertex, and since $MaxVT_2$ also 3, then we can add two more vertices with the smallest cost, and we get the following network:
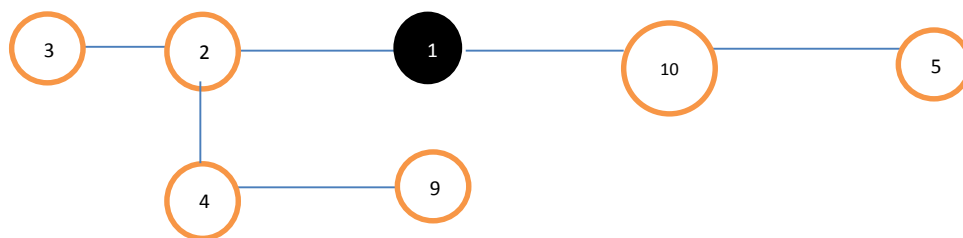

Figure 2. Network after the second period

The total cost after the second period is finished: 1489

After the second period is done, only left 3 vertices are still not connected, which are vertices 6,7 and 8. Since the priority vertex on this period already connected, then the algorithm search for the smallest edges that connect the network with those three vertices. Note that on this period, there are degree violation detected for vertex 4 and vertex 2 during the process installation (for example: the smallest edge is e_{46} with cost 253, but the degree of vertex 4 already 3 (connected with vertex 1, 2 and 8)). Therefore, after the third period we get the following network with the total cost of 2710.
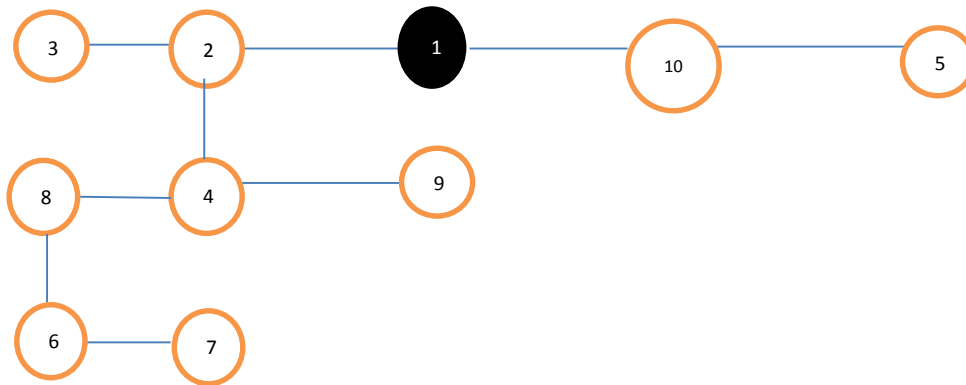
Figure 3. Network after the third period

3. Results and discussion

Eight algorithms are compared, four algorithms (WADR1, WADR2, WADR3, and WADR4) are based on Kruskal's algorithm, and the other four (WADR5, WAC1, WAC2, and WAC3) are based on Prim's algorithm. The following table gives some characteristics of the algorithms:

Table 2. Comparison of some algorithms for MPDCMST

No	Algorithm's name	Developed based on	Process of installation vertices in HVT_i
1	WADR1	Kruskal's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm will add the other priority vertices to be connected on that period.
2	WADR2	Kruskal's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm search the next vertex with the smallest incidence edge to be connected (maybe not priority vertex).
3	WADR3	Kruskal's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm will add the other priority vertices to be connected on that period, and used DFS technique.
4	WADR4	Kruskal's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm search the next vertex with the smallest incidence edge to be connected (maybe not priority vertex), and use DFS technique.
5	WADR5	Prim's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm search the next vertex with the smallest incidence edge to be connected (maybe not priority vertex).
6	WAC1	Prim's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm will add the other priority vertices to be connected on that period.
7	WAC2	Prim's algorithm	No priority installation for vertices in HVT_i, but the all vertices on HVT_i must be connected at the end of ith period.
8	WAC3	Prim's algorithm	Priority based on vertex index, if the vertices in HVT_i already connected on the previous period, the algorithm will add the other priority vertices to be connected on that period, and use DFS technique.

All eight algorithms used degree bound 3, and three periods. Moreover, all algorithms also used the same HVT_i and $MAXVT_i$, and implemented on the same data set. The data implemented consists of 300 random table problem represented complete graph with vertex order of 10 to 100, with increment of 10. For every vertex order, 30 problem are implemented.

From Table 2, we can see that there is one factor will also influence the quality of the solutions besides the main differences of the algorithms (developed by modified Kruskal's or Prim's and using Depth First Search technique): the priority vertices on HVT_i. The time for doing the installation/connection of priority vertices will affect the quality of the solution. Figure 4 below shows the comparative of the solutions.
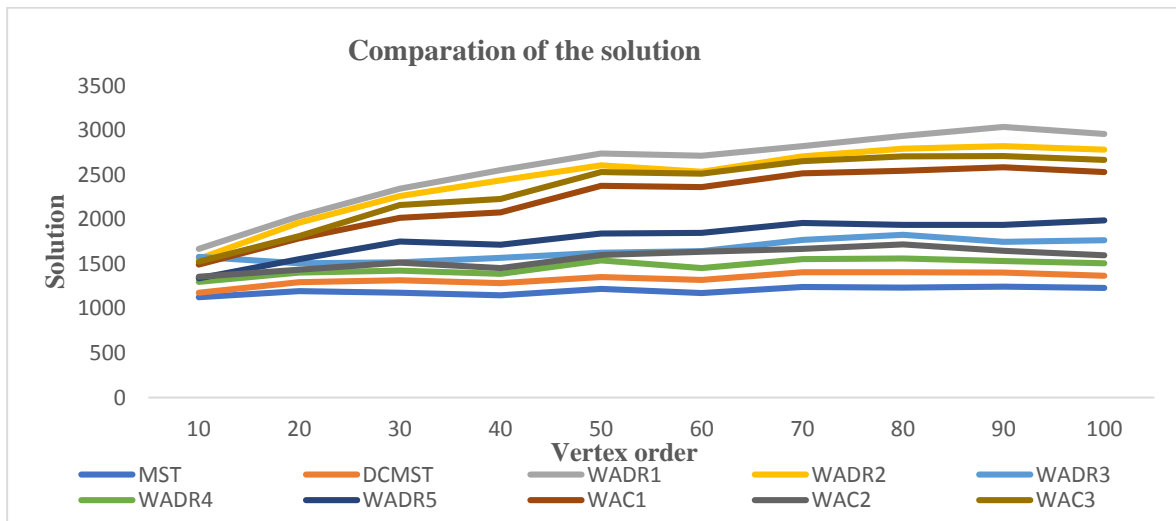
Figure 4. Comparison of the solutions of some algorithms for solving the MPDCMST problem

4. Conclusions

We have compared some algorithms for solving the MPDCMST problem. From the results we can conclude that for the algorithms developed based on Kruskal's algorithm, WADR4 performs the best, and for the algorithms developed based on Prim's algorithm, WAC2 perform the best. Among all algorithm compared, WADR4 is the best. However, in WADR4 the connection property is not guaranteed because that is possible during the connection process constitutes a forest, although at the end all vertices/components are connected. Therefore, if the connectivity is a must, WAC2 is the best.

References

[1] Vasudev C 2006 *Graph Theory with Application* (New Delhi: New Age International Limited)
[2] Graham R L and P Hell 1982 *On the history of the Minimum Spanning Tree Problem* (New Jersey: Bell Laboratories)
[3] Prim R C 1957 Shortest connection networks and some generalization *Bell Syst. Tech. Journal*, **36** pp 1389-1401
[4] Kruskal J B 1956 On the shortest spanning tree of a graph and the travelling salesman problem *Proc. Amer. Math. Soc.* **7** pp 48-50
[5] Narula S C and A H Cesar 1980 Degree-constrained minimum spanning tree *Computer and Operation Research* **7** pp 239-249
[6] Zhou G and M Gen 1997 A note on genetics algorithms for degree-constrained spanning tree problems *Networks* **30** pp 91-95
[7] Deo N and N Kumar 1997 *Computation of Constrained Spanning Trees: A Unified Approach Network Optimization* (Berlin: Springer-Verlag) pp 194-220
[8] Krishnamoorthy M, A T Ernst and Y M Sharaila 2001 Comparison of algorithms for the degree constrained minimum spanning tree *Journal of Heuristics* **7** 6 pp 587-611
[9] Caccetta L and Wamiliana 2001 Heuristics algorithms for the degree constrained minimum spanning tree problems *Proceeding of the International Congress on Modelling and Simulation (MODSIM)* pp 2161-2166

[10] Wamiliana and L Caccetta 2003 Tabu search based heuristics for the degree constrained minimum   spanning tree problem *Proceeding of South East Asia Mathematical Society* pp 133-140

[11] Wamiliana and L Caccetta 2012 The modified CW1 algorithm for the degree restricted minimum   spanning tree problem *Proceeding of International Conference on Engineering And   Technology Development* pp 36-39

[12] Kawatra R 2002 A multi period degree constrained minimum spanning tree problem *European Journal of Operational Research* **143** pp 53p-63

[13] Wamiliana, D Sakethi and R Yuniarti 2010 Computional aspect of WADR1 and WADR2 alghoritms for multi period degree constrained minimum spanning tree problem *Proceeding SNMAP* pp 208-214

[14] Wamiliana, Amanto and M Usman 2013 Comparative analysis for the multi period degree constrained minimum spanning tree problem *Proceeding The International Conference on Engineering and Technology Development (ICETD)* pp 39-43

[15] Wamiliana, M Usman, D Sakethi, R Yuniarti and A Cucus 2015 The hybrid of depth first search technique and kruskal's algorithm for solving the multiperiod degree constrained minimum spanning tree problem *The 4th International Conference on Interactive Digital Media (ICIDM)*

[16] Wamiliana, F A M Elfaki, M Usman and M Azram 2015 Some greedy based algorithms for multi periods degree constrained minimum spanning tree problem *ARPN Journal of Engineering and Applied Sciences* **10** 21 pp 10147-10152

[17] Wamiliana, Asmiati, M Usman, A Hijriani and W C Hastono 2018 Comparative analysis of some modified prim's algorithms to solve the multiperiod degree constrained minimum spanning tree problem *Indian Journal of Science and Technology* **11** 11 pp.1- 6.

[18] Wamiliana, Warsono, Asmiati, A Hijriani and W C Hastono 2018 different time installationeffect on the quality of the solution for the multiperiod installation problem using modified prim's algorithm *Far East Journal of Electronics and Communications* 18(2) pp. 291-300