# AN EFFICIENT ALGORITHM FOR SHORTEST PATH MULTICAST ROUTING UNDER DELAY AND DELAY VARIATION CONSTRAINTS

Mohamed F. Mokbel
Department of Computer Science
Purdue University
West Lafayette, Indiana 47907
e-mail: mokbel@cs.purdue.edu

Wafaa A. El-Haweet
Department of Computer Science
and Automatic Control,
Alexandria University, Egypt.
e-mail: w_elhaweet@yahoo.com

M. Nazih El-Derini
Department of Computer Science
and Automatic Control,
Alexandria University, Egypt.
e-mail: nazih-s@safwa.com

**KEYWORDS**
Communications, Network, Real Time

**Abstract**

A new heuristic algorithm is proposed for constructing multicast tree for multimedia and interactive applications that require certain quality of services, QoS. We consider two main QoS parameters that are the delay and delay variation. The problem is formulated as constructing shortest path tree under delay and delay variation constraints. The tree is used to concurrently transmit packets from source to multiple destinations such that exactly one copy of any packet traverses the links of the multicast tree. This problem is known to be NP-Complete. The proposed heuristic algorithm solves this problem in polynomial time and gives near optimal tree. We first mention some related work in this area then we formalize the problem and introduce the new algorithm with the proof of its complexity. Other heuristic algorithms are examined and compared with the proposed algorithm via simulation.

## 1. INTRODUCTION

Handling group communication is a key requirement for numerous applications that have one source which sends the same information concurrently to multiple destinations. Multicast routing refers to the construction of a tree rooted at the source and spanning all destinations. Generally, there are two types of such a tree, the Steiner tree and the shortest path tree. Steiner tree or group-shared tree tends to minimize the total cost of the resulting tree, this is an NP-Complete problem. Number of heuristics to this problem can be found in (Winter 1987; Hwang and Richards 1992). Shortest path tree or source-based tree tends to minimize the cost of each path from source to any destination, this can be achieved in polynomial time by using one of the two famous algorithms of Bellman (Bellman 1957) and Dijkstra (Dossey *et al* 1993) and pruning the undesired links. Recently, with the rapid evolution of multimedia and real-time applications like audio/video conferencing, interactive distributed games and real-time remote control system, certain QoS need to be guaranteed in the resulted tree. One such QoS is the end-to-end delay between source and each destination, where the information must be sent within a certain delay constraint $\Delta$. By adding this constraint to the original problem of multicast routing, the problem is reformulated and the multicast tree should be either delay-constrained Steiner tree, or delay-constrained shortest path tree. Delay constrained Steiner tree is an NP-Complete problem (Kompella *et al* 1992), several heuristics are introduced for this problem (Kompella *et al* 1992; Kompella *et al* 1993; Widyono 1994; Wi and Choi 1995; Zhu *et al* 1995; Jia 1998) each trying to get near optimal tree cost, without regarding to the cost of each individual path for each destination. Delay-constrained shortest path tree is also an NP-Complete problem (Mokbel *et al* 1999). An optimal algorithm for this problem is presented at (Widyono 1994), but its execution time is exponential and used only for comparison with other algorithms. Heuristics for this problem is presented in (Sun and Langendoerfre 1995; Mokbel *et al* 1999), which tries to get a near optimal tree from the point of view of each destination without regarding the total cost of the tree.

In this paper, we add a delay variation constraint to the problem of delay constrained shortest path. The delay variation constraint is a bound on the variation among the delays along the individual paths from source to each destination. This constraint is first mentioned in (Rouskas and Baldine 1997) where the problem of finding a

multicast tree that satisfies both delay and delay variation constraint is proposed with the proof that it is an NP-Complete problem. In (Haberman and Rouskas 1996) the cost factor is added, which results in a new problem for finding a multicast tree with least cost and satisfying both delay and delay variation constraint. This problem is also an NP-Complete as proved in (Haberman and Rouskas 1996).

The importance of delay variation arises from several applications. For example, in a teleconference, it is important that the speaker be heard by all participants at the same time, or the communication may lake the feeling of an interactive discussion. When multicast messages are used to update multiple copies of replicated data item in a distributed database system, minimizing the delay variation would minimize the length of time that the database is in an inconsistent state. In a distributed game where each player is connected to a game server and competes against other participants, it will not be fair to delay one of the players more than the others by a certain value.

As proposed in (Rouskas and Baldine 1997), buffering can be used to control the delay variation but it may not be practical. Buffering can be placed at the source node, switching nodes or at the receivers. Buffering at the source would require the source to maintain additional information about all destinations. This will make each message buffered different amount of time for each destination and hence transmitted by the source multiple times which violates the meaning of multicast routing. Buffering at the switch nodes suffers from the same problems as buffering at source node. Buffering at receiver nodes is more reasonable and it may have a good effect but it requires that all destinations cooperate together. However, this may be accepted if all destinations have the same task like the case of replicated database, but if the destinations are competitors, it will be difficult to cooperate. Furthermore, the amount of needed buffering is proportional to the maximum variation of end to end delays. Providing bounds for delay variation will result in more efficient usage of buffering resources.

The rest of this paper is organized as follows. In section 2, the problem of shortest path under delay and delay variation constraints is formulated. A new heuristic algorithm for this problem is proposed in section 3. The complexity analysis of the proposed algorithm is introduced in section 4. In section 5, large number of simulation experiments have been done to analyse the performance of the proposed algorithm in comparison to other algorithms Finally, the paper is concluded in section 6.

## 2. PROBLEM FORMULATION

The communication network is modeled as a directed, simple, connected weighted graph $G=(V,E)$, where $V$ is the set of nodes and $E$ is the set of directed links. Each link $e$ in $E$ connects two nodes $u$, $v$ in $V$ and is represented as $e(u,v)$. Two non-negative real value functions are associated with each link, the cost function $Cost(u,v)$ represents the utilization of the link and the delay function $Delay(u,v)$ represents the delay that the packet experiences through passing that link including switching, queuing, transmission and propagation delays. Links are asymmetrical, i.e. $Cost(u,v)$ and $Delay(u,v)$ do not necessarily equal $Cost(v,u)$ and $Delay(v,u)$. A sequence of links that connects two nodes $u$, $v$ are represented by a $Path(u,v)$ with $Cost(Path(u,v))$ which is equal to the sum of the costs of all its links, and $Delay(Path(u,v))$ which is equal to the sum of the delays of all its links. Multicast group $M \subseteq V$ is a set of nodes that receives packets from source $S \in V$. The least cost tree (LC) is a tree originating at the source $S$ and spanning all members of $M$ with minimum cost for each of them individually. The least delay tree (LD) is a tree originating at the source $S$ and spanning all members of $M$ with minimum delay for each of them individually.

Using the previous notations and definitions, the problem of shortest path multicast routing under delay and delay variation constraints can be formulated as follows: Given a directed, simple, connected weighted graph $G(V,E)$, multicast group $M \subseteq V$, a multicast source node $s \in V$ , a delay constraint $\Delta$, and a delay variation constraint $\delta$ where $\delta < \Delta$. Find a subgraph $G`(V`,E`)$ where $(\{s\} \cup M) \subseteq V` \subseteq V$ and $E` \subseteq E$ that has no cycles and no incoming edge in the source node $s$ and all the leaves should be members of $M$. $G`$ should satisfy the following three conditions:

1- Minimum $Cost(Path(s,v))$ $\forall v \in M$

2- $Delay(Path(s,v)) < \Delta$ $\forall v \in M$

3- $|Delay(Path(s,u)) - Delay(Path(s,v))| < \delta \ \forall \ u, v \in M$

This problem is known to be NP-Complete Haberman and Rouskas 1996). So, any polynomial solution tends to find a near optimal solution. If $\delta >$

Δ, the problem is reduced to the delay constrained shortest path problem.

## 3. THE DELAY AND DELAY VARIATION CONSTRAINED SHORTEST PATH ALGORITHM (DVCSP)

DVCSP is a heuristic algorithm to find a near optimal subgraph for the shortest path problem under delay and delay variation constraints. It is a centralized algorithm where all the information about the network topology is supposed to be kept at the source node. The algorithm is divided into two separate phases. Phase I, the delay constraint phase, is responsible for yielding a set of paths for each destination. These paths should satisfy the delay constraint. If for some destinations, phase I is failed to find any path to that destination, then the algorithm reports failure to satisfy the delay constraint. Phase II, the delay variation constraint phase, collects all the paths from phase I and chooses one path from each destination so that the delay variation constraint is satisfied. If phase II is failed, then the algorithm reports failure to satisfy the delay variation constraint and returns a subgraph with the minimum possible delay variation that can be resulted from DVCSP algorithm. The two phases are described in the next two subsections, more details and explanations can be found in (Mokbel 1999).

### 3.1 Phase I: The Delay Constraint Phase

This phase is similar to the algorithm proposed in (Mokbel *et al* 1999) where it starts by an empty token at the source node *S*. The token is continuously duplicated from node to node yielding a set of tokens where each token keeps track with the cost and delay it experiences so far, also it keeps track with the path from source node *S* to the node it currently belongs to. To control the excessive number of tokens that results from duplication, we limit the number of tokens that can be kept in any node at the same time by the number *K*. However, phase I differs from DCSP algorithm proposed in (Mokbel *et al* 1999) in the way of choosing the best *K* tokens to be kept in any node. While in DCSP we choose the least delay tokens, here, in DVCSP algorithm we have a special handling of the best *K* tokens to achieve delay variation constraint. For delay variation constraint, we need to get one token from each destination. Choosing a token from any destination is independent of its delay since all tokens satisfy the delay constraint. The chosen token may be the highest delay one, the least delay one or any one, this depends on the other tokens chosen in other nodes. Using this idea as a key point of the algorithm, we can say that there is no need to keep two tokens in one node with relatively small difference in delay. So, the main task of phase I in DVCSP algorithm is to keep *K* tokens in each node with reasonable delay difference between each two tokens. Recalling that every token *T* satisfies the relation $0 \le Delay(T) \le \Delta$, phase I divides the interval [0, Δ] into *K* equal segments, each with size $\Delta / K$ as shown in Fig. 1.

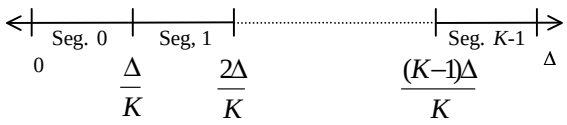


**Fig. 1.** Dividing the interval [0,Δ] into K equal segments

Each node is allowed to keep only one token in each segment and hence the maximum number of tokens that can be kept in each node is *K*. This guarantees an equal distribution of the tokens along the interval [0, Δ], where each segment is handled independently of the other segments. For any token *T* that needs to be duplicated from node *u* to node *v*, its segment is calculated from the equation:

$$Segment\ \ No. = \frac{(Delay(T) + Delay(u,v)) \times K}{\Delta}$$

So, we consider three constraints that can prevent token duplication from node *u* to node *v*.

1. The sum of token delay and *Delay(u,v)* will exceed the delay constraint Δ.
2. The token *T* visited the node *v* before.
3. There is another token $T_1$ in the same segment that *T* wants to duplicate in and $Delay(T_1) < Delay(T)$.

The first constraint means that the token is going to violate the delay constraint while the second means that the token will do a loop. The third constraint, a heuristic one, means that we choose to duplicate the token with lowest delay in each segment.

Each destination node $u \in M$ keeps track with at most 3*K* tokens representing three tokens in each segment. For each segment the destination keeps track with the least delay, least cost and the highest delay tokens passed through this segment. The least delay token is used to guarantee finding a solution for the delay constraint. The highest delay token is used to give an upper bound for the delay in this node. The least cost token is used as a

moderation between the least delay and the highest delay tokens, it is also used to minimize the cost of the resulted graph. These tokens are used by phase II to select only one of them. Phase I is completed when all the tokens in the system are either killed or kept in the destination nodes.

### 3.2 Phase II: The Delay Variation Constraint Phase

Phase II is responsible of satisfying the delay variation constraint. This phase is started by collecting all the tokens from the destinations that left from phase I in one array sorted in ascending order according to the delay field of each token. Since, at most each destination can have 3*K* tokens, so, the total number of tokens in this array will be at most 3*mK* tokens. Two pointers $T_s$, $T_e$ are moved through the sorted array where always $T_s<T_e$ and $|Delay(T_e)–Delay(T_s)| < \delta$. The pointers constitute a window that contains a number of tokens, we call it a valid window if it contains at least one token from each destination. Phase II examines all the valid windows and chooses the one with the minimum cost. If there is no valid window in the array, then the phase II reports failure to satisfy the delay variation constraint and runs another procedure to get the minimum possible delay variation. The other procedures use the same sorted array but moving only one pointer examining all possible changes in delay for each destination and returning a token from each destination where these tokens give the minimum possible delay variation among them.

## 4. CORRECTNESS AND COMPLEXITY OF DVCSP

***Lemma 1****: DVCSP algorithm always finds a solution for the delay constrained problem if one exists and if DVCSP fails to find a feasible solution for the delay constraint, then there is no other algorithm can find it.*

***Proof:*** In case of $K$=1, we have only one segment at each node. Any collision at this node will result in selecting the node with the least delay, so we guarantee that the least delay token continues its trip and results in the least delay path. If $K$>1, each node will have $K$ segments and since we deal with each segment independently and keep the least delay token in each one, so the least delay token in the first segment that lies in the interval $[0,\Delta/K]$ is the least delay token that can reach this node. If for any destination $u \in M$, the least delay token does not satisfy the delay constraint, then this means that there is no feasible solution and no other algorithm can find a solution since the shortest path delay is optimal with respect to the delay constraint.

***Lemma 2:*** *Phase II in DVCSP algorithm finds an optimal solution for delay variation constraint from the data returned from Phase I.*

***Proof:*** In phase II, we collect all the tokens from the destinations in one array. The maximum number of tokens is 3*mk*. Since, we initialize the pointers $T_s$ and $T_e$ at the first element of the array, then we move them step by step to get the window $[T_s,T_e]$. After examining this window, we move to the next available window. This means that we check all the possible windows in the array then we choose the best one in terms of cost. This is an exhaustive search, which yields the optimal solution. When the search fails to satisfy the delay variation constraint, we invoke a procedure to get the minimum delay variation. This procedurs sets $T_s$ again to the first element of the array. By moving $T_s$ step by step, and for every step we calculate the variation of delay we have, we do an exhaustive search for the best delay variation.

***Lemma 3:*** *The worst case complexity of DVCSP is* $O(K^2M^2)$ *if* $KM^2 > N^2$ *and* $O(KN^2)$ *if* $KM^2 < N^2$, *where K is an integer value ranged from 1 to N, M = |M| the number of nodes in the multicast group and N = |V| the number of nodes in the network.*

***Proof :*** In phase I, The algorithm is continuously looping till all the tokens in the network finish their trips, and since the maximum length of any trip is bounded by the number of nodes in the network *N*, then the algorithm will loop at most *O(N)* times in duplicating tokens. For each time of duplication the algorithm checks all the *N* nodes for existence of tokens, so checking nodes will be $O(N^2)$ . For each node, we will process each token in it and since there will be at most *K* tokens in each node then tokens will be processed $O(KN^2)$. For each token we will test whether it can be duplicated or not for all neighbor nodes, so, if we assume that the network has an average degree *d*, then token duplication will be tested $O(dKN^2)$. But since we use real networks which always has a small node degree, we can consider that *d* is a small constant number, so the complexity of phase I will be $O(KN^2)$. In phase II, the maximum number of elements in the sorted array is 3*mK* elements. Sorting this array can be done in *O(MK log MK)*. Moving $T_s$ from the first element of the array to the last element is *O(MK)*. At each time $T_s$ is moved, $T_e$ will be updated. The worst case can happen when the delay variation constraint is so large that $T_e$ every time will be moved from $T_s$ to the end of the array. This will make the movement of $T_e$ also *O(MK)*. Moving $T_e$ inside $T_s$ results in making

phase II $O(M^2K^2)$. In case of failure to satisfy delay variation constraint, phase II calls a procedure to get the minimum delay variation. This procedure moves $T_s$ from the first element in the array to the last element, so, the worst case complexity of moving $T_s$ is $O(MK)$. At every move, we calculate the maximum and minimum value of delay to get the maximum variation of this array. Since we have $M$ destinations, so, the worst case complexity of this procedure is $O(M^2K)$. So, phase II contains three sequential complexities which are $O(MK \log MK)$, $O(M^2K^2)$ and $O(M^2K)$. Since phase I and phase II are executed sequentially, then DVCSP, in its worst case complexity, will be executed in $O(KN^2 + Mk \log MK + M^2K^2 + M^2K)$ steps. For the four terms that constitute the complexity order, the term $M^2K^2$ will be dominant if $KM^2 > N^2$ which makes DVCSP of $O(M^2K^2)$. If $KM^2 < N^2$, the term $KN^2$ will dominate the other three terms and the total complexity of DVCSP will be $O(KN^2)$.

## 5. PERFORMANCE ANALYSIS

### 5.1 Random Graph Generator

To guarantee fair simulation results, we use the same graph generator (Waxman 1988) that is used in all problems related to multicasting. $N$ nodes are randomly distributed over a rectangular area with size $2000 \times 2000$ where each node placed at a location with integer coordinates. The probability of edge existence between any two nodes $u$, $v$ can be calculated from the function:

$$P(u, v) = \beta \exp\left(\frac{-d(u, v)}{L\,\alpha}\right)$$

Where $d(u, v)$ is the distance between nodes $u$, $v$. $L$ is the maximum distance between any two nodes. $\alpha$ and $\beta$ are two parameters used to adjust the degree of the graph and the density of short and long edges. After calculating the above function for each pair, the resulted graph does not necessary to be connected, so, we add random edges to obtain a connected graph. The cost of any edge $e(u, v)$ equals to the distance $d(u, v)$ and the delay is a random value between 1 and 10. Finally, for each algorithm to be correctly evaluated, we run it on 3000 different graphs and taking the average.

### 5.2 Simulation Results

We compare our algorithm DVCSP with three different values of $K$ against the least delay tree LD that comes from applying Dijkstra algorithm on the link delay, CDKS algorithm (Sun and Langendoerfre 1995) and DCSP algorithm **(**Mokbel *et al* 1999**)**. For all algorithms we have two performance factors, the failure rate which is the rate that the algorithm fails to satisfy the delay variation constraint and the average cost per path of the resulted tree of each algorithm whether it satisfies the delay variation constraint or not.

In Fig. 2, it can be observed that DVCSP always gives low failure rate even with tight delay variation constraint. LD, CDKS and DCSP give a very high and non practical failure rate, this indicates that these algorithms can not be used in delay variation constrained problem. All algorithms have a zero failure rate when delay variation constraint equals the delay constraint. In Fig.3, we can see that the low failure rate we get comes on expense of the cost. LD, CDKS and DCSP give horizontal lines since they are not affected be δ. When relaxing δ, DVCSP gives results better than CDKS, this means that our algorithm gives better cost although it has an additional constraint. However the cost of DVCSP can not be better than DCSP which is the basis of our algorithm.

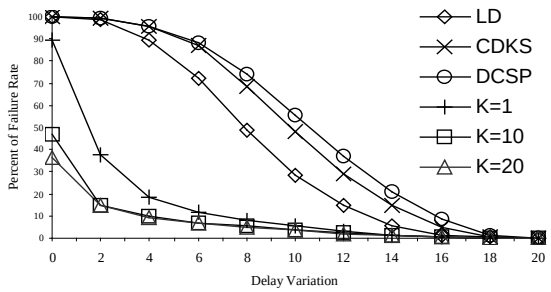


**Fig. 2**. Effect of delay variation constraint on failure rate with network size=50, Δ=20, multicast group=5, average degree=4.

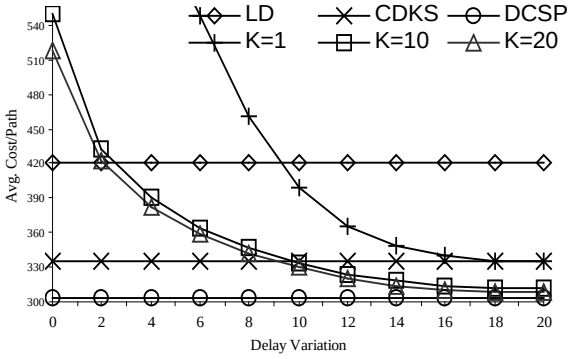


**Fig. 3**. Effect of delay variation constraint on average cost per path with network size=50, Δ=20, multicast group=5, average degree=4.

In Fig. 4, it can be observed that DVCSP dominates other algorithms as the multicast group size increases. LD, CDKS and DCSP give 100% failure rate when multicast group $M$>15, while DVCSP give near zero failure rate for all $M$ at $K$=10 and $K$=20. Also $K$=1 gives a comparable results. Fig. 5 shows that DVCSP with $K$=10 and $K$=20 gives better cost than CDKS, and DVCSP with $K$=1 gives very high cost which makes it non practical at all. This is because at $K$=1 we try to find a solution with very low resources which results in a very high cost.

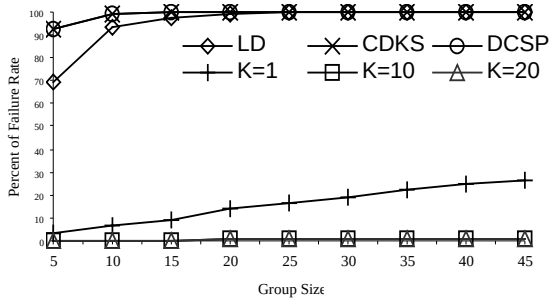


**Fig. 4**. Effect of multicast group size on failure rate with network size = 50, Δ = 20, δ = 5, average degree = 10.

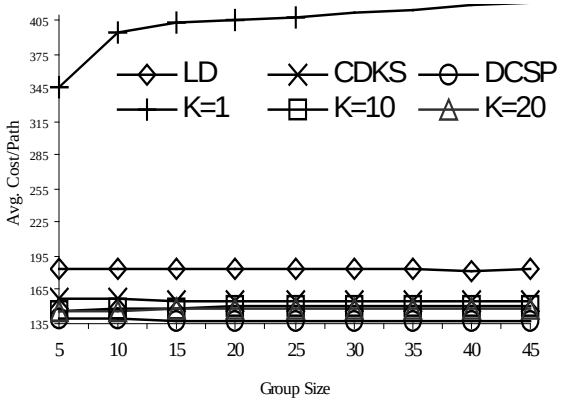


**Fig. 5**. Effect of multicast group size on average cost per path with network size=50, Δ=20, δ=5, average degree= 10.

Fig. 6 shows the effect of changing network size on the failure rate. It can be observed that the results from DVCSP dominates other algorithms with large difference. Also the failure rate of DVCSP with $K$=10 and $K$=20 almost the same. Fig. 7 shows the effect of good failure rate on the cost. It can be seen that DVCSP with $K$=1 is not practical in terms of cost. For small networks $N$<30, CDKS outperforms DVCSP. However, as the size of the network increases the performance of DVCSP increases. Also, DCSP always gives better results than all other algorithms.

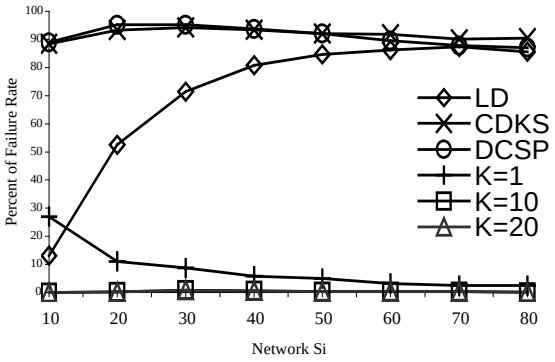


**Fig. 6.** Effect of network size on failure rate with Δ=20, δ = 5, multicast group = 5, average degree = 8.

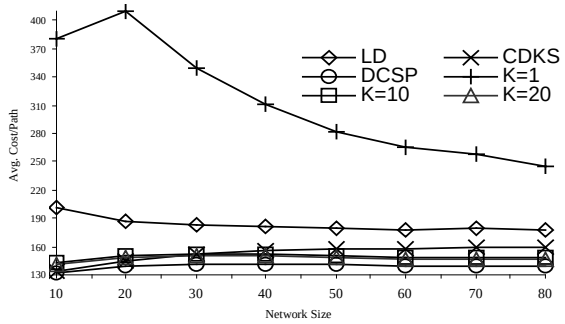


**Fig. 7**. Effect of network size on average cost per path with Δ = 20, δ = 5, multicast group = 5, average degree = 8.

## 6. CONCLUSIONS

In this paper, we considered the problem of multicast routing with two QoS parameters, which are delay and delay variation. The problem is formulated as shortest path problem under delay and delay variation constraints. This problem is known to be NP-Complete. A polynomial time heuristic algorithm, called DVCSP, for this problem is proposed. The algorithm has running time complexity of $O(K^2M^2)$ if $KM^2 > N^2$ and $O(KN^2)$ if $KM^2 < N^2$ , where $K$ is an integer value ranged from 1 to $N$, $M = |M|$ the number of nodes in the multicast group and $N = |V|$ the number of nodes in the network. Large number of simulation experiments has been done to compare the proposed DVCSP algorithm with other algorithms that is mainly for delay constrained shortest path problem. We consider two performance factors, the failure rate and the average cost per path. For the failure rate, DVCSP always dominates all other algorithms with a very great difference. For the average cost per path, DVCSP sometimes gives results that dominates LD and CDKS algorithms in

case of large degree and large network size. On the other side, DCSP algorithm always dominates DVCSP in terms of cost. DVCSP suffers additional cost which is paid for satisfying the additional constraint of delay variation. For all simulated experiments. Simulation results recommend that using $K = 10$ almost has the same results as in case of $K = 20$ while using $K = 1$ always results in a very high cost which makes it not practical. So, $K = 10$ may be considered as a good value in DVCSP algorithm.

## REFERENCES


[Bellman 1957]
Bellman, R. 1952, *Dynamic Programming*. Princeton University Press, 1957.

[Dossey *et al* 1993]
Dossey, J.; A. Otto; L. Spence; and C. Eynden 1993 *Discrete Mathematics*, Second Edition, Harper Collins College Publishers.

[Haberman and Rouskas 1996]
Haberman, B. and G. Rouskas 1996. "Cost, Delay, And Delay Variation Conscious Multicast Routing", Technical Report. TR-97-03, Department of Computer Science, North Carolina State University.

[Hwang and Richards1992]
Hwang, F. and D. Richards 1992. "Steiner Tree Problems", *Networks*, vol. 22, no. 1, pp 55-89, January 1992.

[Jia 1998]
Jia, X. 1998 "A Distributed Algorithm of Delay-Bounded Multicast Routing for Multimedia Applications in Wide Area Networks" *IEEE/ACM Transactions on Networking*, Vol. 6, No.6, pp. 828-837.

[Kompella *et al* 1992]
Kompella, V.; J. Pasquale; and G. Polyzos, 1992 "Multicasting for Multimedia Applications", *in Proceedings of IEEE INFOCOMM'92*, pp. 2078-2085.

[Kompella *et al* 1993]
Kompella, V.; J. Pasquale; and G. Polyzos, 1993 "Multicast Routing for Multimedia Communication", *IEEE/ACM Transactions on Networking*, vol.1, no. 3, pp 286-292.

[Mokbel 1999]
Mokbel, M. 1999 "New Algorithms for Multicast Routing in Real Time Networks", Master Thesis, Department of Computer Science and Automatic Control, Faculty of Engineering, Alexandria University.

[Mokbel *et al* 1999]
Mokbel, M.; W. El-Haweet; and M. El-Derini 1999 "A Delay Constrained Shortest Path Algorithm for Multicast Routing in Multimedia Application", *in Proceedings of IEEE Middle East Workshop on Networking*.

[Rouskas and Baldine 1997]
Rouskas, G. and I. Baldine, 1997 "Multicast Routing with End-to-End Delay and Delay Variation Constraints", *IEEE Journal of Selected Areas in Communications*, vol.15, no.3, pp 346-356.

[Sun and Langendoerfre 1995]
Sun, Q. and H. Langendoerfre, 1995 "Efficient Multicast Routing for Delay-Sensitive Applications", *in Proceedings of the second Workshop on Protocols for Multimedia Systems (PROMS '95)*, pp. 452-458.

[Waxman 1988]
Waxman, B. 1988, "Routing of Multipoint Connections", *IEEE Journal on Selected Areas of Communication*, Vol. 6, No. 9, pp. 1617-1622.

[Wi and Choi 1995]
Wi, S. and Y.Choi. 1995 "A Delay-Constrained Distributed Multicast Routing Algorithms", *in Proceeding of the twelfth International Conference on Computer Communication (ICCC '95)*, pp. 883-838.

[Widyono 1994]
Widyono, R. 1994 "The Design and Evaluation of Routing Algorithms for Real-Time Channels", Technical Report ICSI TR-94-024, University of California at Berkeley, International Computer Science Institute,.

[Winter 1987]
Winter, P. 1987 "Steiner Problem in Networks: A Survey", *Networks*, vol. 17, no. 2, pp. 129-167.

[Zhu *et al* 1995]
Zhu, Q.; M. Parsa; and J. Garcia-Luna-Aceves. 1995 "A Source-Based Algorithm for Delay-Constrained minimum-Cost Multicasting", *in Proceeding of IEEE INFOCOM'95*, pp. 337-385.