

A Minimum Spanning Tree Algorithm for Efficient P2P Video Streaming System

Khaled Ragab, Asrar Ul Haque

King Faisal University, College of Computer Sciences and Information Technology, Saudi Arabia

{kabdultawab,ahaque}@kfu.edu.sa

Abstract— Peer-to-Peer (P2P) file downloading and streaming applications have recently attracted a large number of users on the Internet. Currently, several P2P video streaming systems have been deployed to reduce server cost. They can be classified into two categories live and on-demand streaming systems. The live streaming systems disseminate live video contents to all the peers in real time. On the other hand, the on-demand video (VoD) streaming system enables peers to enjoy the flexibility of watching a video.

An implicit, self-organized and unstructured Multicast Overlay Network (MON) of peers has been proposed [1]. It is organized into MON-Clusters. Each MON-Cluster is composed of independent edge-disjoint Hamilton cycles. The existence of cycles leads to duplication of packets. MON was designed to multicast/flood small-data size (e.g. warning messages) to all peers. Duplication caused due to flooding of small data-packets does not have significant impacts on network traffic. However, duplication of video-packets will have significant impacts on network traffic. Thus the proposed MON is not appropriate to multicast video packets. This paper proposes a novel (Minimum Spanning Tree)-Cluster (MST-Cluster) algorithm over each cluster to avoid duplicated packets. As a result, the MST-Cluster algorithm reduces the network traffic for efficient video streaming.

Index Terms— Peer-to-Peer, Overlay Networks, Video Streaming, Minimum Spanning Tree algorithms.

I. INTRODUCTION

THE video streaming applications have recently fascinated a large number of users over the Internet. According to *AccuStream iMedia Research*, the number of video streams served increased 52% in 2008, and reach 34 billion views. In addition, over the last ten years audiences have accessed 142.7 billions of pieces of video [9]. According to *comScore*, in July 91 million viewers watched five billion online video on *YouTube*. Americans watched 558 million hours of online during the month [10].

The most familiar solution for streaming video over the Internet is the client-server service model such as *YouTube*. This model uses variants of technologies such as *Content Delivery Network (CDN)* to push video content from server to its clients through delivery servers. These servers are placed strategically at the network edges. Scalability is the major challenge for server based video streaming solutions. Despite the fact that *Google* has its impressive *CDN*, on 23 November, 2008 it relied on a third party *Akamai* to stream a *YouTube* live concert to 700,000 concurrent viewers. Actually, *Google* and *Akamai* *CDN* infrastructures are not capable to handle

live video stream effectively. Instead, Peer-to-peer (P2P) video streaming systems can handle it efficiently which will be far cheaper because peers are serving most of the video to others. P2P file downloading and streaming applications have recently attracted a large number of users on the Internet. Currently, several P2P video streaming systems have been deployed to reduce server cost. They can be classified into two categories *live* and *on-demand* streaming systems. The live streaming systems disseminate live video contents to all peers in real time. On the other hand, the on-demand video (VoD) streaming system enables peers to enjoy the flexibility of watching a video. It realizes the goal of watch whatever you want whenever you want.

The construction of a P2P overlay network of peers is the core of the P2P multimedia streaming systems. It enables media dissemination without overloading any server. The P2P overlay networks are classified into three different approaches: *tree-first* (e.g. *ALMI* [6], *Yoid* [7]), *mesh-first* (e.g. *Narada* [8], *Scattercast* [9]) and *implicit* (e.g. *CAN* [2], *Community Overlay Network CON* [3]) approaches [4]. We proposed an implicit self-organized $\langle \delta, \mu \rangle$ -Multicast Overlay Network ($\langle \delta, \mu \rangle$ -MON) that enables scalable and efficient data dissemination [1]. The $\langle \delta, \mu \rangle$ -MON is organized into 2-tiers architecture of *MON-Clusters*, where δ is the maximum physical number of hops between any two end-nodes in each *MON-Cluster*. Thus, end-users accessing from different areas are able to broadcast data into each *MON-Cluster* within a constant δ physical number of hops. However in a dense-mode, increasing the size of *MON-Clusters* induces a long logical diameter that manifests high stress per physical links and high delay to disseminate data. Thus, in [1], we limited the expansion of each *MON-Cluster* within a specific logical diameter μ . Each *MON-Cluster* is constructed with at most μ logical hops over δ physical hops. Thus, end-users are able to efficiently disseminate data into each *MON-Cluster* within μ logical hops over δ physical hops with reasonable stress per physical links and bandwidth consumption. Each *MON-Cluster* is constructed as a 2χ -regular graph composed of χ independent edge-disjoint *Hamilton Cycles (HC)* [5]. Within each *MON-Cluster* a peer can disseminate data to its neighbors, who in turn broadcast same data to all their neighbors, except the neighbor from which it was received. This dissemination process goes on till the data reaches all nodes in *MON*. Because of the existence of cycles in each *MON-Cluster*, the dissemination process floods *MON* and results in many duplicated packets that

increases the network traffic. This dissemination process shows insignificant impacts on network traffic if the disseminated data size is small. On the other hand, it shows significant impacts on network traffic when disseminating multimedia on *MON* because each *MON-Cluster* includes cycles. Thus, it is required to build a *Minimum Spanning Tree (MST)* over each *MON-Cluster* to avoid the duplicated packets and reduce the network traffic. This paper proposes a novel algorithm for constructing *MST*. It runs simultaneously over each *MON-Cluster*.

Given $G(E, V)$ a connected, undirected graph with $|E|=m$, $|V|=n$ and diameter d , Minimum Spanning Tree (MST) of G is an acyclic subset of edges that connects all the vertices of the graph and whose total weight is the minimum. MST is a well-known combinatorial problem with a rich history [11] and is one of the most widely studied distributed problems for its theoretical and practical implications. *Gallagher, Humblet, and Spira* [22] presented the pioneering distributed MST algorithm known as *GHS*. Since then there have been considerable amount of work done to improve the time complexity and to prove correctness of the *GHS* algorithm [13,24,25,26,27]. *Faloutsos and Molle* [23] gives brief description of *GHS* and *AWE* [27] and suggests modifications for correct execution of *AWE*.

Lotker, Patt-Shamir, and Peleg [14] has proposed a distributed algorithm of $O(lgn)$ communication round for overlay network where n processes are connected to each other. *Young, Chen, Ma, Krishnamurthy, Peterson and Wang* [21] has used the *GHS* algorithm to create k -MST which can give k -redundant trees for multimedia applications, data collection, multi-path routing, and file mirroring. The MST is widely used for approximation algorithms, e.g. the degree bounded MST [25]. The search for a bounded-diameter MST finds its applications in designing networks such as: telecommunication, linear light-wave network, and distributed system considering mutual exclusion and bit-compression [26]. *Raidl and Julstrom* [26] has given an evolutionary algorithm for bounded-diameter MST. *Han, McMahon, and Sugden* [23] has proposed a branch and bound algorithm for Capacitated MST (*CMST*). The *CMST* is one of fundamental and most significant sub-problem in many telecommunication network designs as well as in the optimal design of local computer networks [13]. The MST is used to design heuristics for the strong minimum energy topology problem [17]. The first efficient localized method has been presented to construct a bounded degree planar connected structure called Incident MST whose total edge length is within a constant factor of MST [18]. In sensor networks, various spanning trees including MST have been used for data aggregation and energy saving [29]. More recent arguments favoring the use of MST have been made in [19] to minimally avoid network partitions. Recently an approximation distributed algorithm has been suggested with $\Omega(V/lgn)$ [31].

Any distributed *MST* solution requires local optimization which leads to global optimization based on network

wide knowledge. The edges that belong to the MST are called the *tree edges*; the rest of the edges, which are not part of MST of G , are called the *non-tree edges (NTE)*. Essentially inclusion of any *NTE* in a MST results in creation of cycles in the MST. Most of the earlier works[22] have focused on finding the tree edges. The careful selection of the tree-edges contribute to the $O(nlgn)$ factor in the message complexity. The novel idea exploited in this paper is distinct in the way that only the *NTEs* are tracked. The message complexity of this novel algorithm is $O(m)$ while the time complexity is $O(d)$. Moreover the messages are of bounded length i.e. $O(lgn)$. The algorithm is straightforward. It works in two phases and requires only three types of messages. There are six simple modules to generate and to aggregate these three types of messages.

The remainder of this paper is organized as follows. Section 2 briefly presents the architecture of the proposed multicast overlay network $\langle \delta, \mu \rangle$ -*MON*. Section 3 gives an overview of the proposed *MST-Cluster* algorithm. Section 4 draws a conclusion of this paper.

II. $\langle \delta, \mu \rangle$ -MULTICAST OVERLAY NETWORK

This section describes the proposed multicast overlay network architecture.

Definition 1 ($\langle \delta, \mu \rangle$ -Clustering): Given a graph $G = (V, E, \langle \delta, \mu \rangle)$, where vertex $v \in V$, edge $e \in E$. The pair $\langle \delta, \mu \rangle$ is integer. It clusters the graph G into connected sub-graphs (*MON-Cluster*) C . Each cluster has a diameter less than or equal to the upper bound diameter μ over δ physical links. In other words, the distance between any two vertices in the *MON-Cluster* is less than $\langle \delta, \mu \rangle$, where δ, μ are the clustering parameters.

A. $\langle \delta, \mu \rangle$ -MON Architecture

The 2-tiers architecture of *MON* is shown in figure 1 and explained as follows. It is composed of multiple *MON-Clusters* C_k ; $k=1, \dots, \omega$ in level 1. Each C_k has a *tutor* node (*leader*) called t_k . The physical distance between any $i \in C_k$ and t_k is $d(i, t_k) \leq r$; $\delta=2r$ as shown in figure 1. The distance between two vertices i and j , $h(i, j)$, is the number of logical hops of a shortest path between i and j . Its maximum value over all pair of vertices, $D_\tau^{(k)} = \max \{h(i, j); \forall i, j \in C_k\}$, is the diameter of the *MON-Cluster*

C_k at instance of time τ . Where $D_\tau^{(k)}$ is less than or equal the upper bound diameter μ . In level 2 a cluster that contains all tutor nodes is constructed and called *Tutors-Cluster* as shown in figure 1. The size of the *Tutors-Cluster* is denoted by ω and equaled to the number of *MON-Clusters* in the level one.

B. MON-Cluster

Definition 2 (MON-Cluster): The $\langle \delta, \mu \rangle$ -*MON cluster* is defined by a subset of nodes which are mutually reachable by a path of length at most δ physical hops and μ logical hops for some fixed δ and μ .

Definition 3 The size, $|C_j|$ of a MON-Cluster C is the number of nodes in C .

Each MON-Cluster (C) and Tutors-Cluster are constructed as a 2χ -regular graph composed of χ independent edge-disjoint Hamilton Cycles (HC) [5]. Each node has 2χ neighbors (node connectivity). Those neighbors are labeled as $\mathbf{n}_p^{(1)}$, $\mathbf{n}_s^{(1)}$, $\mathbf{n}_p^{(2)}$, $\mathbf{n}_s^{(2)}$, ..., $\mathbf{n}_p^{(\chi)}$, $\mathbf{n}_s^{(\chi)}$.

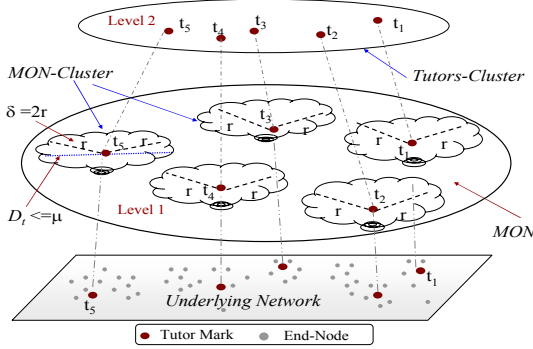


Figure 1: The 2-tiers $\langle \delta, \mu \rangle$ -MON architecture.

For each i , $\mathbf{n}_p^{(i)}$ denotes the neighbor node's predecessor and $\mathbf{n}_s^{(i)}$ denotes the neighbor node's successor on the i -th HC. For example, figure 2 shows a MON-Cluster consists of two Hamilton cycles with diameter $D_t = 3$ (logical hops) over the underlying network. The physical distance between the tutor node t_1 and any node is less than or equal two (i.e. radius $r = 2$ and diameter $\delta = 2r$) backbone physical hops between their routers. The cluster C is composed of HC that enjoys the advantage of joining or leaving processes that require only local changes.

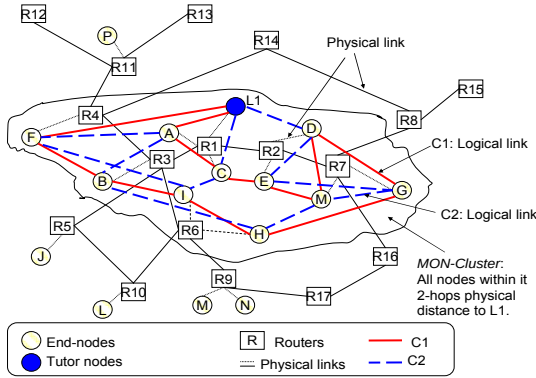


Figure 2: An example of MON-Cluster.

Definition 4 (MON-Cluster's Diameter): The diameter of the 2χ -regular graph (C_j) at instance of time τ is bounded by

$$1 + 2\chi + 2\chi(2\chi - 1) + \dots + 2\chi(2\chi - 1)^{D_t^{(j)} - 1} = \frac{2\chi(2\chi - 1)^{D_t^{(j)}} - 2}{2\chi - 2} = N_\tau(2\chi, D_t^{(j)}) = |C_j|_\tau \quad (1)$$

This value is called the Moore bound [6], and it is known that, for $D_t^{(j)} \geq 2$ and $2\chi \geq 3$, where $|C_j|_\tau$ is the estimated size of MON-Cluster C_j at instance of time τ .

From equation 1 if $|C_j|_\tau - 1 \leq (2\chi)^{D_t^{(j)}} [7]$ then

$$D_t^{(j)} \geq \frac{\log(|C_j|_\tau - 1)}{\log 2\chi} \quad (2).$$

Thus, by estimating the size of the j -th MON-Cluster the diameter $D_t^{(j)}$ can be determined.

Further details about the step-step construction scheme of the $\langle \delta, \mu \rangle$ -MON was described in [1].

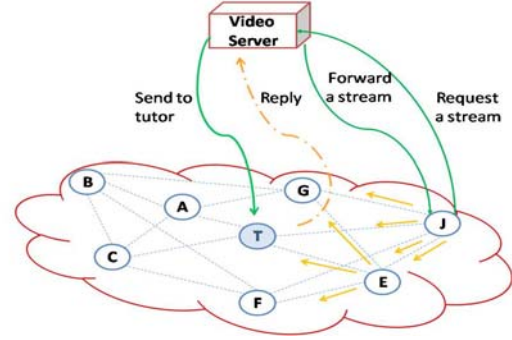


Figure 3: Video Streaming Process in MON-Cluster.

C. Multicast Process

As soon as a node $j \in C_k$ sends a request to a video server for receiving certain video stream as shown in figure 3, the server sends to the tutor of the C_k to check whether no node belongs to C_k has already requested the same stream. The tutor has a list of nodes who registered to disseminate different streams into C_k . If no node have already disseminated the requested stream, the tutor sends a positive reply to the server to allow the node j to receive this stream. Then, the video server pushes the required stream to the node j . Then, node j disseminates the received packets p_i over the $\langle \delta, \mu \rangle$ -Multicast Overlay Network processes the following steps:

- Node j broadcasts the packet p_i to its neighbors, who in turn broadcast p_i to all their neighbors, except the neighbor from which it was received.
- When a tutor receives the packet p_i , then it broadcasts this packet to its neighbors in the MON-Cluster.
- In addition, each tutor node in turn broadcasts p_i to its neighbors in the Tutors-Cluster.
- This application level-multicast process goes on till the message packet p_i reaches all nodes in MON.

In the multicast process, each node/tutor autonomously monitors the received packets to forward only one copy of the duplicated received ones.

III. MST-CLUSTER ALGORITHM

A. Overview

In each MON-Cluster the tutor initiates forming MST for its cluster. The MST algorithm runs in parallel in each cluster. It is designed to operate in two phases: the orientation phase and the NTE detection phase. It uses three types of messages: *request*, *acknowledge* and *notify*. The first phase is the orientation phase during which

request and acknowledge messages are generated. This phase is used to convert an undirected graph to a directed graph and to locate candidate edges. During the *NTE* detection phase notify messages are generated.

The tutor of a *MON-Cluster* initiates the orientation phase. The *request* messages are sent to adjacent nodes for further downstream propagation. A node upon receiving a request, propagates the request by generating a new request message for the downstream nodes. However, at the same time it also generates an immediate acknowledge for the requester. The notify initiator, a node without having any children, launches *NTE* detection phase by generating notify messages. In the return trip of information during the *NTE* detection phase, the nodes aggregate and generate notify messages. Once a node receives all the expected notify messages from its children, it generates new notify messages for its parents. The algorithm uses a set of six modules which are the *request generator*, *acknowledge generator*, *notify generator*, *request aggregator*, *acknowledge aggregator*, and *notify aggregator*.

B. Graph Orientation

Given an undirected graph, the request and notify messages are propagated using an acyclic directed graph to avoid deadlock during the *NTE* detection phase. An orientation of an undirected graph G is an assignment of exactly one direction to each of the edges of G so that there is no cycle. The graph orientation also ensures that there is at least one sink which we call notify initiator which initiates the *NTE* detection phase. There are many graph orientation algorithms [12,30]. However many of these are not suitable for the MST algorithm because of their complexity of message size. The graph orientation algorithm used in the paper uses messages of size $O(\lg n)$. Let $G=(V,E)$ be an undirected graph and $\bar{G}=(V,\bar{E})$ be an orientation of G . An edge $(v_i, v_j) \in \bar{E}$ denotes that vertex v_i is the parent of v_j .

Let $\{v_i, v_j, v_k\} \in V$ and $\{(v_i, v_j), (v_j, v_k), (v_k, v_i)\} \in E$. Let id_i represent the *id* of i^{th} vertex. Let rr_i denotes set of nodes from which vertex i has received a request. The level function f maps each node to an integer depending on the arrival of the first request message at that node. Therefore the level assignment is a function:

$f: V \rightarrow \{0..n\}$ where $f(v_i)=l_i$ denotes level of i^{th} vertex. It is shown in the appendix that a directed graph is acyclic if and only if its nodes can be numbered in such a way that for all arcs (v_i, v_j) , $l_i \leq l_j$.

C. Cycle Detection and Elimination

A major challenge in reducing the messaging cost of any distributed MST algorithm lies in detecting edges, called *non-tree edges*, which create cycles in the MST. GHS totally eliminates inclusion of such edges that might create a cycle. It has been argued [26] that GHS operates by growing fragments which are part of the final MST and any merging decision has to be taken after communicating messages over the fragment itself,

contributing to higher time complexity of $O(n \lg n)$. The MST algorithm [26], referred here as GKP, during initial phases create fragments of MST to reduce the cost of detection and elimination of non-tree edges. The cycles are created due to inclusion of non-tree edges as the fragments of MST are merged. The cycles are removed in subsequent phases of GKP.

The approach proposed in this paper only considers detection of those edges which are *NTE*. Given an undirected graph, an acyclic directed graph is built through graph orientation (as discussed in the previous subsection). The candidate edges are identified along a path originating from the tutors. The edge having the maximum cost among all the edges along a path from tutor to one of the notify initiator is called the candidate edge. When more than two paths merge at a single node, the candidate edges are compared to find the non-tree edges.

During the *NTE* detection phase each node identifies a *virtual* path to the notify initiator and forwards the candidate edge of that path to its parents. The node identifies the path to be used further upstream by aggregating two virtual paths received from two different children. Out of those two paths, it finds out the path which has the link with the larger cost which is essentially a non-tree edge. The path with lower maximum link cost is then used to find non-tree edges of the induced sub-graph produced by inducing its parents. The correctness of the algorithm lies in the following path optimality theorem and the cycle theorem.

D. Path Optimality Theorem

A spanning tree T^* is a minimum spanning tree if and only if it satisfies the following condition: For every edge (k,l) , called non-tree edge (NTE), that is not in the MST, $\omega(i,j) \leq \omega(k,l)$ for every edge (i,j) contained in the path in T^* connecting nodes k and l .

Proof. The proof is straightforward [28]. Let T^* be an MST and (i,j) be contained in the path connecting nodes k and l . If $\omega(i,j) > \omega(k,l)$ then replacing (i,j) with (k,l) would generate a spanning tree of lesser cost than T^* which contradicts the assumption.

Conversely, let (i,j) be any edge in T^* and let $[H, \bar{H}]$ be a cut and $(i,j) \in [H, \bar{H}]$ and be any non-tree edge $(k,l) \in [H, \bar{H}]$ and $\omega(i,j) \leq \omega(k,l)$. Since the condition, $\omega(i,j) \leq \omega(k,l)$ is true for every edge in T^* , $(i,j) \in [H, \bar{H}]$ and for any corresponding non-tree edge $(k,l) \in [H, \bar{H}]$, T^* must be a minimum spanning tree.

An example of MST algorithm run within a cluster is shown in figure 4.

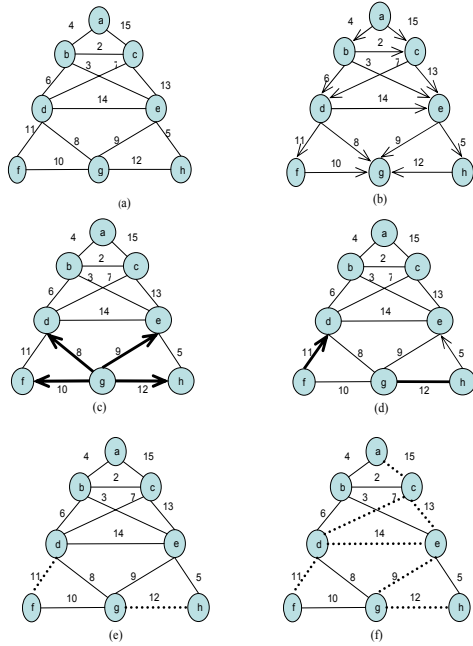


Figure 4. An Example of MST algorithm within a cluster. (a) example graph with cost of edges (b) parent-child relationship between different adjacent nodes at the end of the orientation phase (c) notify messages send by g with bold lines representing the candidate-edges (d) notify messages send by f and h with bold lines representing the candidate-edges (e) Detection of NTEs by d and e with dotted lines representing the NTEs (f) at the end of execution - all the non-tree-edges with dotted lines representing the NTEs and solid lines representing edges forming the MST

IV. CONCLUSION

The proposed $\langle \delta, \mu \rangle$ -Multicast Overlay Network enabled the end-users accessing from different areas to efficiently broadcast data within a constant μ logical hops over λ physical hops. Multicasting process over $\langle \delta, \mu \rangle$ -MON induced appropriate performance for disseminating data with small size over $\langle \delta, \mu \rangle$ -MON. This paper proposes a novel MST algorithm that enables efficient multicasting video streams to a set of peers over the proposed $\langle \delta, \mu \rangle$ -Multicast Overlay Network is feasible. It reduces the traffic induced by the duplicated packets.

REFERENCES

- [1] K. Ragab, A. Yonezawa, "A Self-organized Clustering-based Overlay Network for Application Level Multicast", Journal of Networks, Academy publisher, Vol. 4, No. 1, April 2009
- [2] S. Ratnasamy, Et al., "A Scalable Content- Addressable Network," Proc. Of SIGCOMM'01, California, USA.
- [3] K. Ragab, K. Mori, "ACIS-Hierarchy: Enhancing Community Communication Delay for Large-Scale Information Systems," IEICE Trans. on Communication, Vol. E87-B, No. 7, pp. , July 2004
- [4] K. Ragab "Autonomous Decentralized Community Communication Technology for Assuring Information Dissemination," Ph.D. Thesis Tokyo Institute of Technology, 2004.
- [5] B. Jackson and H. Li, "Hamilton Cycles in 2-Connected k -regular Bipartite Graphs," Comb. Theory Series A, 44 (2): 177-186, 1998.

- [6] N. Biggs, "Algebraic Graph Theory", Cambridge Math. Library ISBN 0-521-45897-8 pbk, (1974, 1993 (2nd edition))
- [7] A. H. Dekker, B. D. Colbert, "Network robustness and graph topology," ACM Proc. 27th Int. Conf. on Australian Computer Science, pp 359-368, 2004.
- [8] Granbaum B. Et al., "Pairs of Edge-disjoint Hamiltonian Circuit" Journal of Aequationes Mathematicae, vol. 4, No. 1-2, 1976.
- [9] Accustream iMedia Research Homepage. <http://www.accustreamsearch.com>
- [10] ComScore measuring the digital world. <http://www.comscore.com>
- [11] R.L. Graham, Pavol Hell, On the History of the Minimum Spanning Tree Problem, Annals of the History of Computing, Vol. 7, Number 1, January 1985
- [12] Valmir C. Barbosa and Jayme L. Szwarcfiter, Generating All The Acyclic Orientations Of An Undirected Graph, Information Processing Letters 72, 1999.
- [13] Michalis Faloutsos, Mart Molle, A Linear-Time Optimal-Message Distributed Algorithm For Minimum Spanning Trees, Distributed Computing, Vol. 17, Number 2, August 2004.
- [14] Zvi Lotker, Elan Pavlov, Boaz Patt-Shamir, David Peleg, MST Construction In $O(\log \log N)$ Communication Rounds, In Proc. Of The Fifteenth Annual ACM Symposium On Parallel Algorithms And Architectures, June 2003.
- [15] Jochen Könemann, R. Ravi, Primal-Dual Meets Local Search: Approximating MST's With Nonuniform Degree Bounds, In Proc. Of The Thirty-Fifth ACM Symposium On Theory Of Computing, June 2003.
- [16] Günther R. Raidl, Bryant A. Julstrom, Evolutionary Computing And Optimization: Greedy Heuristics And An Evolutionary Algorithm For The Bounded-Diameter Minimum Spanning Tree Problem, In Proc. of the 2003 ACM symposium on Applied computing, March 2003.
- [17] Xiuzhen Cheng, Bhagirath Narahari, Rahul Simha, Maggie Xiaoyan Cheng, and Dan Liu, Strong Minimum Energy Topology in Wireless Sensor Networks: NP-Completeness and Heuristics, IEEE Transactions On Mobile Computing, Vol. 2, No. 3, July-September 2003.
- [18] Xiang-Yang Li, Yu Wang, Peng-Jun Wan, Wen-Zhan Song and Ophir Frieder, Localized Low-Weight Graph and Its Applications in Wireless Ad Hoc Networks, IEEE INFOCOM 2004.
- [19] Javier Gomez, Andrew T. Campbell, A Case for Variable-Range Transmission Power Control in Wireless Multihop Networks, IEEE INFOCOM 2004.
- [20] Pankaj K. Agarwal, David Eppstein, Leonidas J. Guibas, Monika R. Henzinger, Parametric and Kinetic Minimum Spanning Trees, In Proc. 39th Annual Symp. on Foundations of Computer Science, pp. 596-605.
- [21] Anthony Young, Jiang Chen, Zheng Ma, Arvind Krishnamurthy, Larry Peterson, and Randolph Y. Wang, Overlay Mesh Construction Using Interleaved Spanning Trees, In Proc. INFOCOM 2004. March 2004.
- [22] Robert G. Gallager, Pierre A. Humblet, and P. M. Spira, A Distributed Algorithm For Minimum Weight Spanning Trees, ACM TOPLAS, Vol. 5, No. 1, pp. 66-77, January 1983.
- [23] Nancy Lynch, Distributed Algorithms, pp 509-524, Morgan Kaufmann Publishers, Inc., 1997.
- [24] Michalis Faloutsos and Mart Molle, Optimal Distributed Algorithm for Minimum Spanning Trees Revisited, Symposium on Principles of Distributed Computing, 1995

- [25] Juan A. Garay and Shay Kutten and David Peleg, A Sub-Linear Time Distributed Algorithm for Minimum-Weight Spanning Trees, In Proc. Foundations of Computer Science, 34, 1993.
- [26] B. Awerbuch, Optimal Distributed Algorithms For Minimum Weight Spanning Tree, Counting, Leader Election, And Related Problems, pp. 230—240, In Proc. of STOC, 1987.
- [27] Baruch Awerbuch, Oded Goldreich, David Peleg, and Ronen Vainish, A Tradeoff Between Information And Communication In Broadcast Protocols, pp 238--256, Journal. of the ACM, 37(2), April 1990.
- [28] Ravindra K. Ahuja, Thomas L. Magnanti, James B. Orlin, Network Flows: Theory, Algorithms, and Applications, 1993, Prentice Hall pp 516-520.
- [29] J.I. Khan and A.U. Haque, Maximizing Lifetime of Power-limited Network with Active Minimum Spanning Tree Aggregation, Parallel and Distributed Computing and Systems, PDCS 2004, MIT Cambridge, USA.
- [30] Eugene Lawler, Combinatorial Optimization Network and Matroids, pp 29-30, Dover Publications Inc, 2001.
- [31] Maleq Khan and Gopal Pandurangan. "A Fast Distributed Approximation Algorithm for Minimum Spanning Trees", *Distributed Computing*. Pages 391-402, Vol. 20, No. 6, Apr. 2008.

APPENDIX

Let $G=(V,E)$ be an undirected graph and $\vec{G}=(V,\vec{E})$ be an orientation of G . An edge $(v_i, v_j) \in \vec{E}$ denotes that vertex v_i is the parent of v_j . Let $\{v_i, v_j, v_k\} \in V$ and $\{(v_i, v_j), (v_j, v_k), (v_k, v_i)\} \in E$. Let id_i denote the id of i^{th} vertex. Also it is assumed that $id_i \neq id_j$ i.e. id is unique for every vertex. Let rr_i denote nodes from which vertex i has received a request. The level function f maps each node to an integer depending on the arrival of the first request message at that node. Therefore the level assignment is a function: $f: V \rightarrow \{0..n\}$ where $f(v_i) = l_i$ denotes level of i^{th} vertex.

$$f(v_i) = \begin{cases} 0 & \text{if } |rr_i| = 0 \text{ and } v_i \notin \text{session initiator} \\ 1 & \text{if } v_i \in \text{session initiator} \\ l_j + 1, & \text{if } f(v_j) = l_j \text{ and } j \in rr_i \text{ and } l_i = 0 \\ l_i & \text{if } l_i \neq 0 \end{cases} \quad (7)$$

Then level is used to get $\vec{G}$.

$$i \in rs_j \Rightarrow \begin{cases} (v_i, v_j) \in \vec{E} & \text{if } v_j \in \text{update initiator} \\ (v_i, v_j) \in \vec{E} & \text{if } l_i < l_j \text{ or } (l_i = l_j \text{ and } id_i < id_j) \\ (v_j, v_i) \in \vec{E} & \text{otherwise} \end{cases} \quad (8)$$

A. Graph Orientation Theorem

A directed graph is acyclic if and only if its nodes can be numbered in such a way that for all arcs (v_i, v_j) , $l_i \leq l_j$.

Proof. Let in the directed graph C be a cycle consisting of at least three vertices. Let v_i, v_j, v_k be the three vertices and $\{(v_i, v_j), (v_j, v_k), (v_k, v_i)\} \in \vec{E}$.

Case I: If $l_i \neq l_j \neq l_k$ then since (v_i, v_j) and $(v_j, v_k) \in \vec{E}$ then $l_i < l_j < l_k$. Since $l_j < l_k$ so $(v_k, v_i) \notin \vec{E}$. Hence cycle C cannot exist.

Case II: If $l_i = l_j = l_k$ then since (v_i, v_j) and $(v_j, v_k) \in \vec{E}$ then $id_i < id_j < id_k$. Since $id_j < id_k$ so $(v_k, v_i) \notin \vec{E}$. Hence the cycle C cannot exist.

Case III: If $l_i = l_j \neq l_k$ then since (v_i, v_j) and $(v_j, v_k) \in \vec{E}$ then $l_i < l_k$. Since $l_i < l_k$ so $(v_k, v_i) \notin \vec{E}$. Hence the cycle C cannot exist.

Conversely, let v_i, v_j, v_k be the three vertices and $\{(v_i, v_j), (v_j, v_k), (v_k, v_i)\} \in E$.

Case I: If $l_i < l_j < l_k$ then $\{(v_i, v_j), (v_j, v_k), (v_i, v_k)\} \in \vec{E}$. So there is no cycle.

Case II: If $l_i = l_j = l_k$ and without loss of generality, if $id_i < id_j < id_k$, then $\{(v_i, v_j), (v_j, v_k), (v_i, v_k)\} \in \vec{E}$. So there is no cycle.

Case III: If $l_i = l_j \neq l_k$ and without loss of generality, if $id_i < id_j$ then $(v_i, v_j) \in \vec{E}$. Now there can be two sub cases. In the first sub case, if $l_j < l_k$ then $l_i < l_k$, implying $(v_i, v_k) \in \vec{E}$. So there cannot be a cycle. In the second sub case, if $l_j > l_k$ then $(v_k, v_j) \in \vec{E}$. Hence there is no cycle.