J Inf Process Syst, Vol.13, No.3, pp.546~558, June 2017
https://doi.org/10.3745/JIPS.01.0017

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Multicast Tree Construction with User-Experienced Quality for Multimedia Mobile Networks

Hoejung Jung* and Namgi Kim*

Abstract
The amount of multimedia traffic over the Internet has been increasing because of the development of networks and mobile devices. Accordingly, studies on multicast, which is used to provide efficient multimedia and video services, have been conducted. In particular, studies on centralized multicast tree construction have attracted attention with the advent of software-defined networking. Among the centralized multicast tree construction algorithms, the group Takahashi and Matsuyama (GTM) algorithm is the most commonly used in multiple multicast tree construction. However, the GTM algorithm considers only the network-cost overhead when constructing multicast trees; it does not consider the temporary service disruption that arises from a link change for users receiving an existing service. Therefore, in this study, we propose a multiple multicast tree construction algorithm that can reduce network cost while avoiding considerable degradation of service quality to users. This is accomplished by considering both network-cost and link-change overhead of users. Experimental results reveal that, compared to the GTM algorithm, the proposed algorithm significantly improves the user-experienced quality of service by substantially reducing the number of link-changed users while only slightly adding to the network-cost overhead.

Keywords
GTM, Mobile Network, Multicast Tree Construction, SDN

1. Introduction

With the recent widespread use of high-speed networks and mobile devices, multimedia and video traffic on the Internet is rapidly increasing. In particular, multimedia and video-related traffic is forecast to triple from 2015 to 2020, growing at a rate of 21% annually [1]. With this trend of continuously increasing traffic, considerable research on multicast has been conducted to provide multimedia data to many users efficiently. Multicast transmission is widely used in multimedia services because it can efficiently transmit the same data to many users while reducing network cost [2]. However, existing multicast algorithms cannot avoid forming locally suboptimal multicast trees. This is because routers construct multicast trees with their respective distributed information. Recently, studies on multicast transmission using centralized information in a software-defined network (SDN) have been conducted to overcome the problems inherent in these existing multicast algorithms [3-7]. In SDN, the controller can construct globally optimal multicast trees using global topology information.

Manuscript received March 29, 2017; accepted April 12, 2017.
Corresponding Author: Namgi Kim (ngkim@kyonggi.ac.kr)
* Dept. of Computer Science, Kyonggi University, Suwon, Korea ({junghj, ngkim}@kyonggi.ac.kr)

This is because a centralized controller, rather than a switch, manages the path information for the flow.

An SDN-based multicast tree construction algorithm can also solve the classic multicast tree construction optimization problem represented by the Steiner tree problem in networks (SPN) [8,9]. In the SPN, a Steiner tree forms a multicast tree for network cost minimization. The SPN is an NP-complete problem. Thus, from the perspective of the SPN, algorithms exist that heuristically form a single multicast tree (SMT) [10,11]. The SMT is used only for a single service. Therefore, a method must exist that constructs a multiple multicast tree (MMT) efficiently for use with multiple services. The group Takahashi and Matsuyama (GTM) algorithm [12], which is a typical heuristic algorithm that efficiently constructs the MMT, singles out SMTs that can be problematic because of bandwidth, replaces them with an alternative tree while having low-cost overhead, and constructs the MMT that approximates the Steiner tree.

Most of the multicast tree construction algorithms proposed thus far have only focused on network cost minimization. Multicast tree construction algorithms, which limit their focus to network overhead, update trees without considering the user's perspective. Thus, numerous link changes may occur on the path between the client currently receiving a service and the server. These link changes produce a temporary change in the quality of service from the user's point of view. Therefore, if many link changes occur, the overall user-experienced quality of service may degrade [13]. Therefore, in this study, we propose a multicast tree construction algorithm that generates low network overhead while generally maintaining the user-experienced quality of service. It accomplishes this by considering both network and link-change costs when constructing the MMT. The proposed algorithm can construct the MMT that considers the quality of user service by determining the overhead incurred from tree changes based on both the network cost and link-changed user.

The remainder of this paper is organized as follows. Section 2 investigates the SDN multicast that constructs a multicast tree based on both the multicast concept, which efficiently provides multimedia services, and global information. We also review existing multicast tree construction algorithms, SMT and MMT. In Section 3, we describe our proposed multicast tree construction algorithm. Section 4 describes the experimental environment, parameters, and metrics used in verifying the performance of the proposed algorithm. We test both the GTM and the proposed algorithm according to the metrics and then compare and analyze their results. Section 5 presents the conclusion and suggests future research.

2. Background Knowledge and Related Studies

A multicast transmission mechanism enables one or more senders to deliver the same data to one or more receivers. Multicast transmission copies a data packet in the branch node on the multicast tree construction tree and delivers it to the lower nodes. Compared to unicast transmission, in which a sender must transmit a data packet as many times as the number of receivers, multicast transmission can reduce network overhead. Thus, multicast transmission is widely used in multimedia services such as IPTV, video conferencing, and real-time streaming.

Thus far, the most important property to consider when constructing a multicast tree has been network cost minimization. An SPN is a classical problem encountered when constructing a multicast tree with respect to network cost minimization. A Steiner tree on the SPN is a minimum cost multicast

tree and represents an NP-complete problem that cannot be solved in polynomial time, except in some cases [8,9]. The TM [10] and KMB [11] algorithms solve the problem heuristically by constructing a multicast tree that is approximates the minimum cost. However, even algorithms such as the TM and KBM cannot be easily applied because routers have distributed information in existing router-based networks.

2.1 SDN-Based Multicast Tree Construction

The SDN was proposed in order to improve the limitations of existing distributed router-based networks. An SDN is composed of application, control, and data planes, as shown in Fig. 1 [14]. The SDN manages the flow path for all switches in the controller using a protocol such as OpenFlow. An SDN switch using the OpenFlow protocol has more than one flow table, and the controller can centrally control the flow path through this protocol [15]. If multicast transmission is implemented using the SDN, constructing an efficient multicast tree with global network information beyond a locally optimal limit of existing multicast transmission [3-7] is possible.

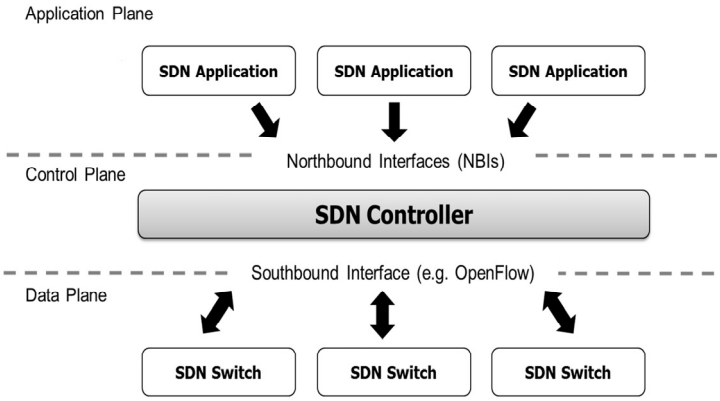


Fig. 1. Structure of software-defined network (SDN).

A typical study of SDN multicast is CastFlow [3]. CastFlow is an open source for a multicast clean-slate approach on the SDN, which is a programmable network. CastFlow manages numerous users who can dynamically join and leave multicast groups. In the multicast group configuration of CastFlow, path calculation uses the SMT as a construction algorithm to reduce processing delay. CastFlow uses the BRITE tool [16] to create a topology and selects a subset of users to form a multicast group. Once the network topology is created, the minimum spanning tree (MST) of the multicast group is computed using Prim's algorithm [17], and the path list for each user corresponding to the multicast group is found on the MST. In addition, the process of eliminating duplication links between paths for each user is started. Through this process, the SDN controller can easily add an entry to the flow path table of the SDN switch. However, existing CastFlow studies use the MST algorithm during multicast tree creation and, as a result, all node switches are visited in the network topology. Therefore, with respect to the SPN, a considerable overhead is incurred when constructing the SMT.

2.2 Single Multicast Tree Construction Algorithms

To implement multicast transmission efficiently, constructing an efficient multicast tree is crucial. Accordingly, studies on SMT construction algorithms based on the multicast priority property have been conducted and of these, those algorithms whose priority property is the minimum cost of the tree have been researched the most. The most typical SMT construction algorithms for this single service are the TM [10] and KMB [11] algorithms. The TM algorithm is computationally intensive because it not only must locate the minimum path between nodes that do not belong to the service member group, but it also must calculate the distance between member nodes and nodes that do not belong to the member group. However, the TM algorithm constructs more efficient SMT in terms of cost because it considers a greater number of cases than does the KMB algorithm.

In addition to the TM and KMB, the layer-based serving with layer switching (LBS-LS) algorithm provides multicast graph construction for multiple senders [5]. The LBS-LS algorithm adds delay and jitter constraints to the shortest paths between the sender and receiver when constructing a multicast graph for a single content service. In addition, in the event it cannot find an appropriate path, the LBS-LS algorithm constructs a multicast graph by locating a path having the largest available bandwidth. Global cost minimization may not be achieved because the LBS-LS algorithm constructs a multicast graph with the shortest distance first between the sender and receiver. However, it can prioritize delay minimization between each sender and receiver. Thus, an advantage exists in constructing multicast for multiple senders per single service.

2.3 Multiple Multicast Trees Construction Algorithms

When constructing a multicast tree for a single service, constructing only one SMT is simple. However, the MMT construction for multiple services is more complicated because the effects among multiple trees must be considered. In particular, in the event a service is added in real time or the members of a multicast group join and later leave, the MMT construction becomes more complicated.

Studies on efficiently implementing an MMT construction that supports multiple services have been conducted [6,7,12,18-20]. The GKMB [18] and GTM [12] algorithms are typically employed in MMT construction. The GKMB algorithm uses the KMB algorithm when constructing the SMT, whereas the GTM algorithm uses the TM algorithm. The TM algorithm has advantages over the KMB algorithm with respect to network cost and considers a greater number of cases in the process of SMT construction. Therefore, the GTM algorithm has advantages over the GKMB algorithm regarding network cost because it considers a greater number of cases.

When the SMT, which is constructed to provide an added multicast service, must be added to an existing network topology, the GTM algorithm compares the additional and existing trees in order to choose the one having lower overhead and to provide other existing services. It then changes the trees based on this comparison. Here, the network cost overhead NO_{T_i} of tree T_i is the difference between the network cost $C(T_i)$ of the currently constructed original tree and the network cost $C(T'_i)$ of an alternative tree, as shown in the following equation.

$$NO_{T_i} = C(T'_i) - C(T_i) \tag{1}$$

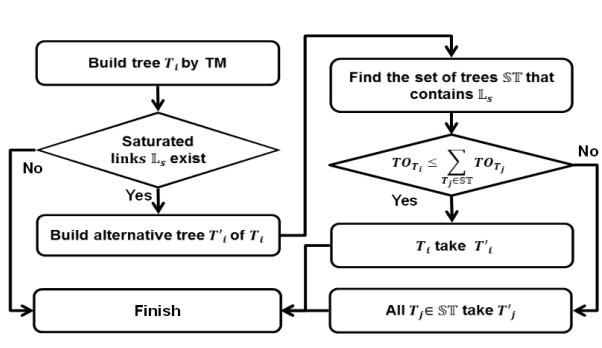
The implementation of the GTM algorithm using (1) is as follows. First, when a new service is added, construct a newly inserted tree T_i belonging to the MMT using the TM algorithm. Second, confirm that the residual bandwidth in the network topology is sufficient for a new tree to be inserted. If it is sufficient, terminate the GTM algorithm; if it is insufficient, construct an alternative tree T'_i on the network topology using the TM algorithm in the third step, excluding the bandwidth insufficient link $\mathbb{L}_s$. In the fourth step, calculate NO_{T_i} that occurs when T_i is changed to T'_i according to (1). In the fifth step, find the most recent trees set $\mathbb{ST}$ that contains $\mathbb{L}_s$. In the sixth step, find T'_j for every tree T_j in $\mathbb{ST}$ and find NO_{T_j}. Finally, compare $\sum_{T_j \in \mathbb{ST}} NO_{T_j}$ with NO_{T_i}. If $\sum_{T_j \in \mathbb{ST}} NO_{T_j}$ is less than NO_{T_i}, update every tree T_j in $\mathbb{ST}$ with T'_j. Otherwise, update T_i with T'_i.

The GTM algorithm has an advantage when used to construct the MMT heuristically in order to minimize the network cost. However, it does not consider user-experienced quality of service as a result of a link change. Therefore, in this study, we propose a multicast tree construction algorithm that considers not only the network cost but also the user-experienced quality of service.

3. Proposed Algorithm

In this study, the proposed GTM with user (GTMU) algorithm effectively constructs the MMT by reducing the network overhead in a centralized network such as SDN while considering the user-experienced quality of service. The GTM algorithm considers only the network cost overhead NO_{T_i} when calculating the overhead between the original tree T_i and an alternative tree T'_i. By contrast, the GTMU algorithm includes not only the network cost overhead but also the link-changed user overhead LO_{T_i}, which is the number of users whose links on the service path are changed because of a tree change. Eq. (2) calculates the total overhead TO_{T_i} between T_i and T'_i in the GTMU algorithm. Here, α is an averaging factor between $NO_{T_i}$and LO_{T_i} and values between 0 and 1 are used. If α = 1, the MMT is constructed while considering only NO_{T_i}, exactly as in the GTM algorithm. If α = 0, the MMT is constructed while considering only LO_{T_i}.

$$TO_{T_i} = \alpha \times NO_{T_i} + (1 - \alpha) \times LO_{T_i} \tag{2}$$



Symbol	Definition
T'_i	Alternative tree of the tree T_i
$\mathbb{L}_s$	Saturated link set
NO_{T_i}	Network cost overhead of the tree T_i
LO_{T_i}	Link-changed user overhead of the tree T_i
TO_{T_i}	Total overhead of the tree T_i
$C(T_i)$	Cost of the tree T_i
$\mathbb{ST}$	Tree Set that contains the $\mathbb{L}_s$

$$TO_{T_i} = \alpha \times NO_{T_i} + (1 - \alpha) \times LO_{T_i}$$

Fig. 2. Flow of the proposed GTM with user (GTMU) algorithm.

The process flow of the GTMU algorithm using (2) is shown in Fig. 2. Each step in the GTMU algorithm is described as follows.

Step 1 – construct a new SMT T_i: This is performed when a new multicast service member set is given. The T_i construction uses the TM algorithm [10], which constructs a tree according to the network cost of previous SMT construction algorithms. In the TM algorithm, the Dijkstra algorithm [21] is used to find the distance between the visiting and receiver nodes of the tree.

Step 2 – check bandwidth insufficient (saturation) link $\mathbb{L}_s$: This determines whether the newly constructed tree in Step 1 in the network topology can accommodate the newly formed SMT. Each time a service is added, the bandwidth on the network topology is updated to reflect the used amount. To accommodate T_i added to the current network topology, available bandwidth must exist on the link included in T_i. If a link in T_i does not have available bandwidth, it is labeled a saturation link and is included in the set $\mathbb{L}_s$. If $\mathbb{L}_s$ is an empty set, insert T_i as is into the existing network topology and terminate the algorithm.

Step 3 – construct an alternative tree of T_i : In the network topology excluding the saturation link $\mathbb{L}_s$, calculate an alternative tree T'_i of T_i using the TM algorithm. Then, calculate TO_{T_i} using (2).

Step 4 – find the set of most recent trees $\mathbb{ST}$ that contains saturation link set $\mathbb{L}_s$: This step concerns the SMT of services prior to adding T_i. Here, find the trees passing through the link included in $\mathbb{L}_s$. Among the trees, find the tree T_j that most often passes through the saturation link, and include it in the saturation tree set $\mathbb{ST}$. In the network topology excluding the saturation link for T_j, construct an alternative tree T'_j using the TM algorithm, and calculate TO_{T_j}. Next, remove links that are no longer saturated in $\mathbb{L}_s$ by changing T_j to T'_j. This process is repeated until $\mathbb{L}_s$ becomes an empty set.

Step 5 – compare the overhead of the newly inserted tree and the overhead of previous trees that may be changed instead of the new: In this step, compare the overhead of the newly inserted tree TO_{T_i} with that of previous trees $\sum_{T_j \in \mathbb{ST}} TO_{T_j}$ that may be changed instead of the new. If $TO_{T_i} \leq \sum_{T_j \in \mathbb{ST}} TO_{T_j}$, change T_i to T'_i. Otherwise, we change all $T_j \in \mathbb{ST}$ to T'_j.

To explain the proposed algorithm, we assume a service environment such as that shown in Fig. 3. In this figure, Services 1 and 2 are already running and Service 3 has been newly inserted. When the SMT for Service 3 is constructed using the TM algorithm, the link existing between Nodes 3 and 5 is saturated and the bandwidth becomes insufficient. In this situation, the proposed GTMU algorithm finds an alternative tree for Services 1–3, as shown in Fig. 4. In addition, the algorithm determines both the network cost overhead *NO* and link-changed user overhead *LO* in order to calculate the total overhead *TO*. In this example, the *TO* is calculated by setting α to 0.5. The results are presented in Table 1. As shown in the table, Service 2 is the lowest of the three services. Therefore, the original tree of Service 2 is changed to an alternative tree, and the service is continued. The final tree construction is shown in Fig. 5.

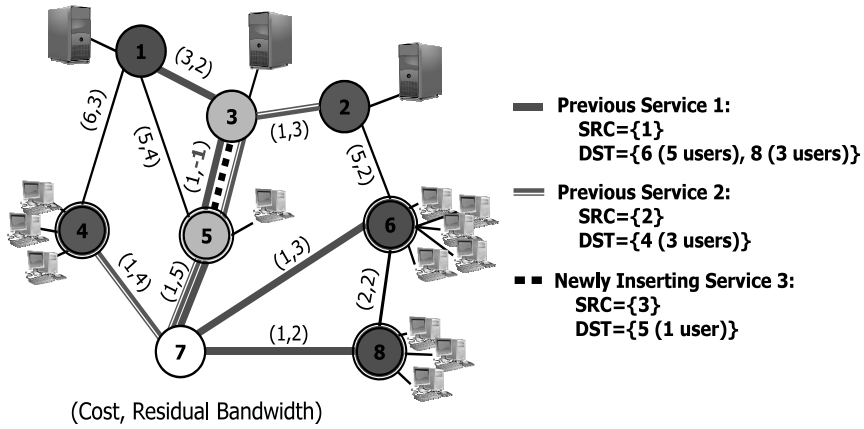


Fig. 3. Link saturation to be solved for Services 1–3.

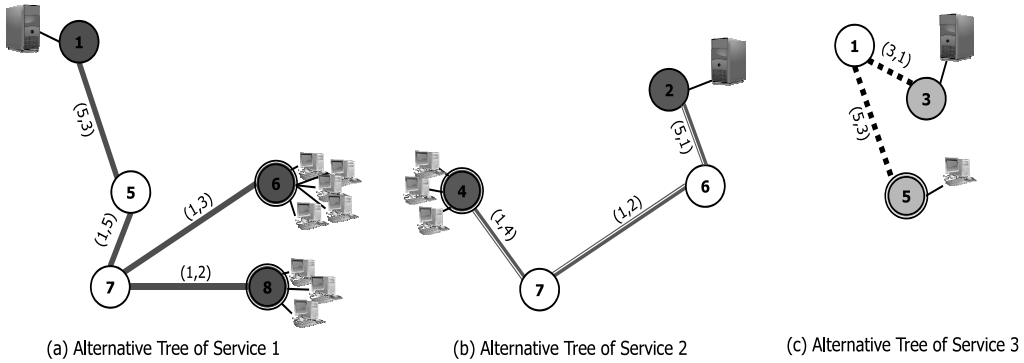


Fig. 4. Alternative trees for Services 1–3.

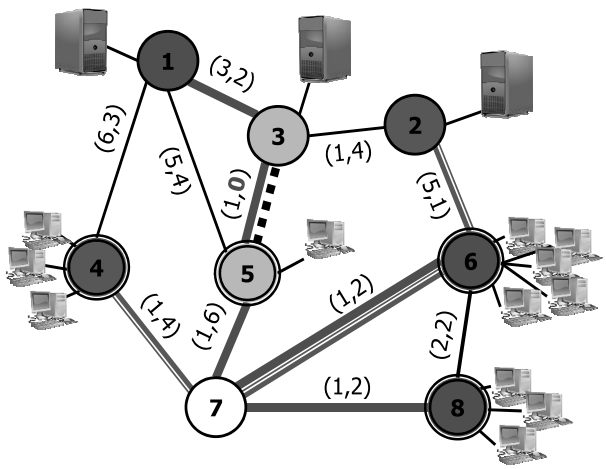


Fig. 5. Tree construction after solving link saturation for Services 1–3.

Table 1. Overhead calculation for Services 1–3

	Service 1	Service 2	Service 3
Network cost of original tree T_i	7	4	1
Network cost of alternative tree T'_i	8	7	8
Network cost overhead NO_{T_i}	1	3	7
Link-changed user overhead LO_{T_i}	8	3	0
Total overhead TO_{T_i} (α = 0.5)	4.5	3	3.5

4. Performance Evaluation

4.1 Simulation Environments

For our experiment, we constructed a network topology by inputting the simulation parameters listed in Table 2 into the BRITE tool [16]. The location of the nodes was randomly set using the Waxman model [22]. The numbers of nodes and links were set to 200 and 400, respectively. The bandwidth of the link was set randomly from 1 to BW_{MAX}, and the value of BW_{MAX} was set to 5 (low capacity), 10 (middle capacity), and 20 (high capacity) depending on network capacity.

Table 2. Simulation parameters for the network topology

Parameter	Value
Network topology construction tool	BRITE [16]
Random graph model	Waxman [22]
Total number of nodes	200
Total number of links	400
Link bandwidth (min, max)	
Low capacity	(1, 5)
Middle capacity	(1, 10)
High capacity	(1, 20)

The services required by users on the network topology were set using the simulation parameters as listed in Table 3. The required bandwidth per service was set to 1. The number of source nodes per service was 1, and the number of destination nodes for each service was randomly set to a value between 1 and 20. The number of users for each service per destination node also was randomly set to a value between 1 and 20.

Table 3. Simulation parameters for services

Parameter	Value
Required bandwidth per service	1
Number of source node per service	1
Number of destinations per service	1–20
Number of users per destination node	1–20

4.2 Simulation Results

Fig. 6 shows the simulation results in the low-capacity network (BW_{MAX} = 5). The greater the value of α, the more weight given to *NO* when calculating the *TO* value. In addition, if α = 1, the MMT was constructed while considering only the network cost overhead, as in the GTM algorithm. Fig. 6(a) shows the total network cost based on the number of multicast input trees. Results show that the total network cost was lowest when α was 1. In other words, the simulation confirmed that the GTM algorithm offers the greatest advantage in terms of network cost. In addition, as the value of α increased, the total network cost decreased. Here, the network cost overhead occurring with MMT construction was regarded with greater importance. However, we should note that even if the value changed, the network cost overhead did not change considerably. This shows that the total network cost did not change considerably even if both the link-changed user overhead and network cost overhead were considered.

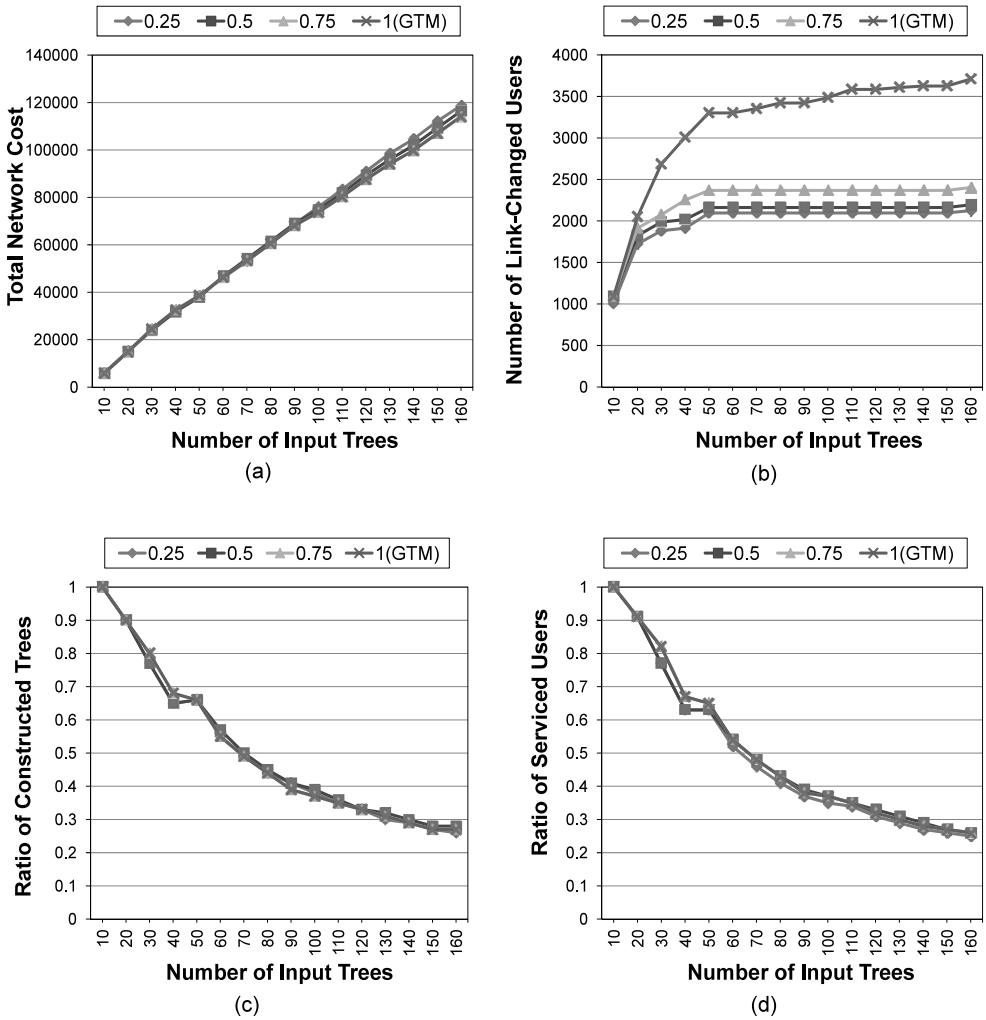


Fig. 6. Simulation results in the low-capacity network.

Fig. 6(b) shows the number of link-changed users based on the number of input trees. The figure shows that the GTM algorithm with an α value of 1 had the most link-changed users. This is because the GTM algorithm used only the network cost for overhead computation without considering the number of link-changed users when selecting an alternative tree. By contrast, the proposed algorithm considered the number of link-changed users to a greater extent when calculating the total overhead while selecting an alternative tree. Thus, the lower the value of α, the fewer the link-changed users. Therefore, this study showed that by using the proposed GTMU algorithm, an advantage was gained in maintaining the number of link-changed users below a certain level without a substantial increase in total network cost.

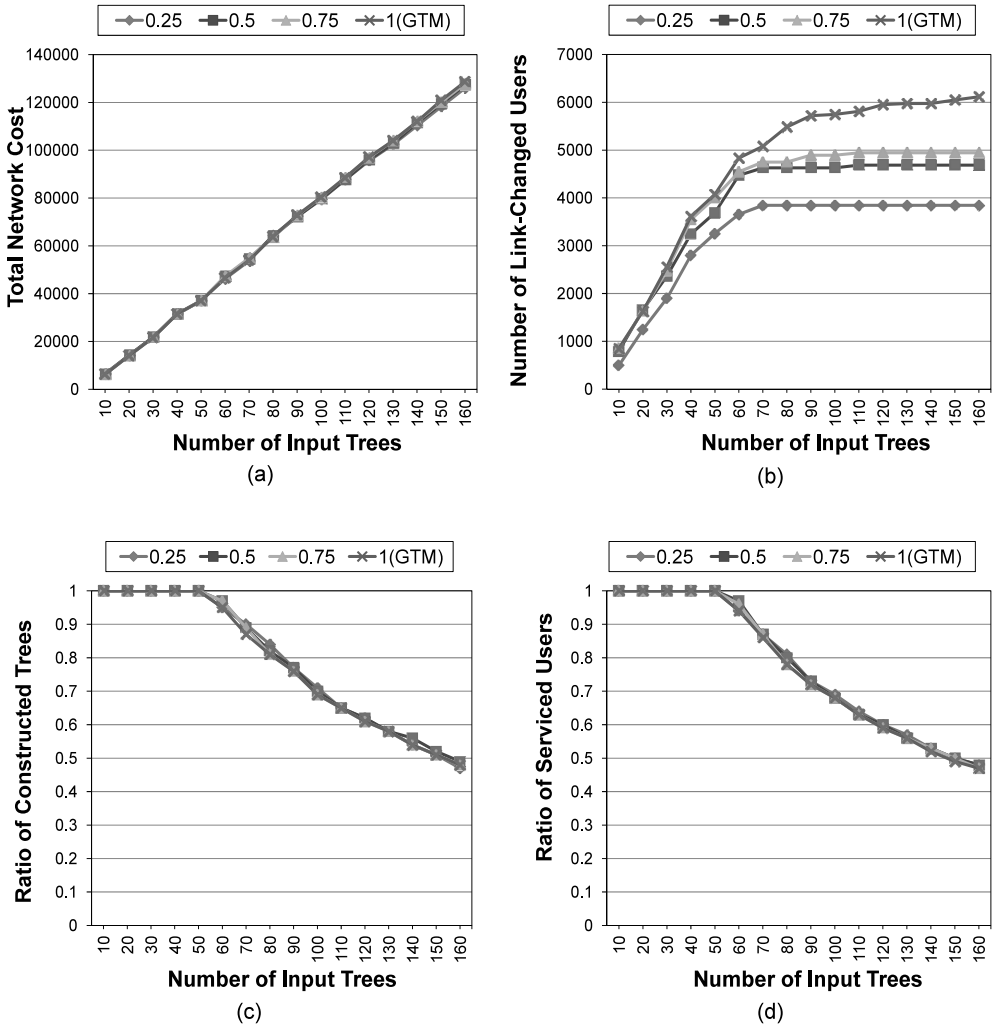


Fig. 7. Simulation results in the medium-capacity network.

Figs. 7 and 8 show the experimental results in the medium- and high-capacity networks. As the results show, even though the number of input trees increased, the total network cost steadily decreased. This is because the number of service trees that can be accommodated increased as the link capacity

increased. The number of link-changed users also increased steadily as the number of input trees increased. However, the medium- and high-capacity networks had consistently similar experimental results to those of the low-capacity network.

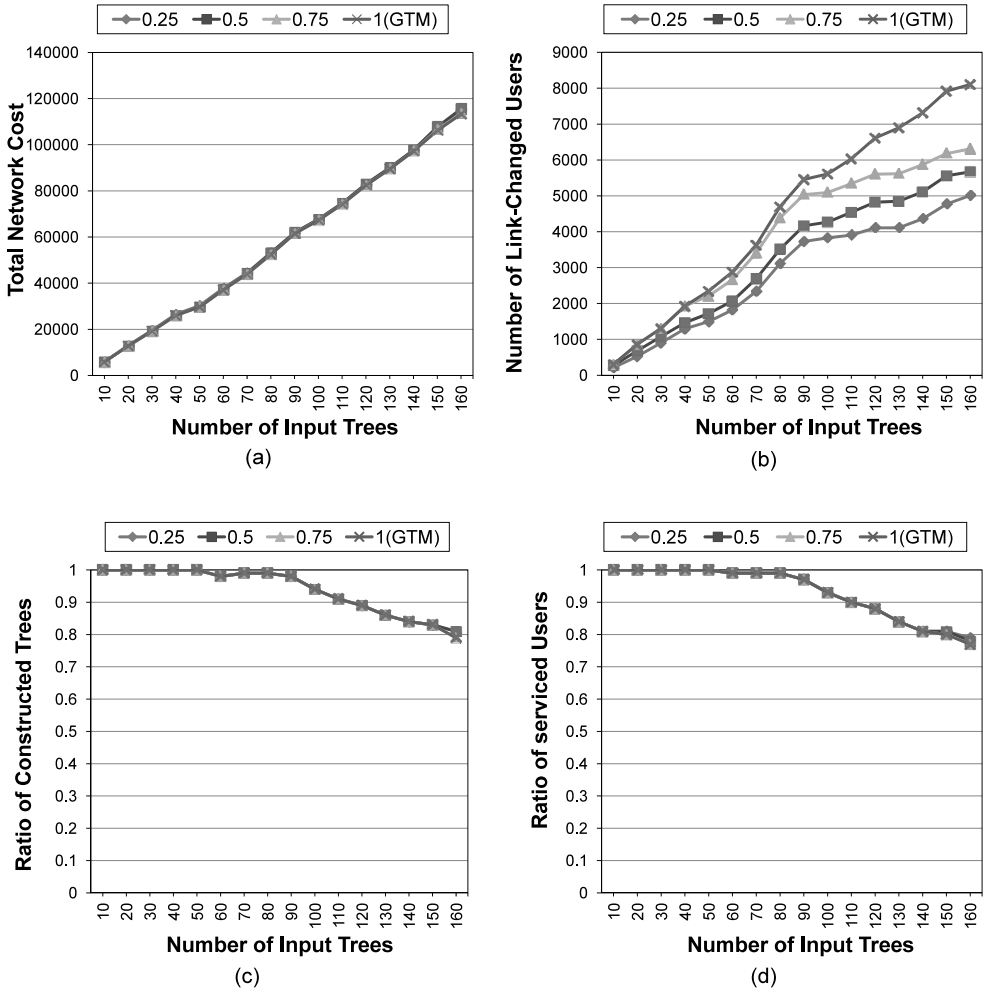


Fig. 8. Simulation results in the high-capacity network.

5. Conclusion

In this study, we proposed a multicast tree construction algorithm that considers not only network cost but also the number of link-changed users to provide better quality of service to users. Our study showed that the proposed GTMU algorithm works efficiently on an SDN architecture and can integrate and manage the flow paths and monitoring information for all switches to obtain global information about a network topology. We showed that our GTMU algorithm can be used to construct multiple multicast trees to support several multimedia services for mobile users on the network topology. When a new multicast tree is added, the GTMU algorithm adds a new service by locating the tree with the

small link change overhead. The GTMU algorithm consider the service quality for users already receiving services by considering both the link-changed user overhead and the network cost overhead when calculating the link change overhead. Comparison results of the existing GTM and proposed GTMU algorithms showed that the GTMU increased the network-cost overhead slightly. However, the difference in the network cost between them was relatively small, and our study showed that GTMU could significantly improve the quality of service without much network-cost overhead by substantially reducing the number of link-changed users. For future research, we plan to study algorithms that are suitable for use in more realistic environments by subdividing the trees into path links based on destination nodes.

Acknowledgement

This work was supported by Kyonggi University Research Grant 2016.

References

[1] Cisco VNI Forecast Widget [Online]. Available: http://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-widget/forecast-widget/index.html.

[2] L. H. Sahasrabuddhe and B. Mukherjee, "Multicast routing algorithms and protocols: a tutorial," *IEEE Network*, vol. 14, no. 1, pp. 90-102, 2000.

[3] C. A. C. Marcondes, T. P. C. Santos, A. P. Godoy, C. C. Viel, and C. A. C. Teixeira, "CastFlow: clean-slate multicast approach using in-advance path processing in programmable networks," in *Proceedings of IEEE Symposium on Computers & Communications (ISCC)*, Cappadocia, Turkey, 2012, pp. 94-101.

[4] S. Xu, C. Wu, and Z. Li, "Software defined mobile multicast," in *Proceedings of 2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Dallas, TX, 2015, pp. 208-216.

[5] N. Xue, X. Chen, L. Gong, S. Li, D. Hu and Z. Zhu, "Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1617-1629, 2015.

[6] J. R. Jiang and S. Y. Chen, "Constructing multiple Steiner trees for software-defined networking multicast," in *Proceedings of the 11th International Conference on Future Internet Technologies*, Nanjing, China, 2016 pp. 1-6.

[7] M. Sun, X. Zhang, L. Wang, H. Shi, and W. Zhang, "A multiple multicast tree optimization solution based on software defined network," in *Proceedings of 2016 7th International Conference on Information and Communication Systems*, Irbid, Jordan, 2016, pp. 168-173.

[8] P. Winter, "Steiner problem in networks: a survey," *Networks*, vol. 17, no. 2, pp. 129-167, 1987.

[9] F. K. Hwang and D. S. Richards, "Steiner tree problems," *Networks*, vol. 22, no. 1, pp. 55-89, 1992.

[10] H. Takahashi and A. Matsuyama, "An approximate solution for the Steiner problem in graphs," *Mathematica Japonica*, vol. 24, no. 6, pp. 573-577, 1980.

[11] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for Steiner trees," *Acta Informatica*, vol. 15, no. 2, pp. 141-145, 1981.

[12] C. P Low and N. Wang, "An efficient algorithm for group multicast routing with bandwidth reservation," *Computer Communications*, vol. 23, no. 18, pp. 1740-1746, 2000.

[13] P. Brooks and B. Hestnes, "User measures of quality of experience: why being objective and quantitative is important," *IEEE Network*, vol. 24, no. 2, pp. 8-13, 2010.

[14] C. Banse and S. Rangarajan, "A secure northbound interface for SDN applications," in *Proceedings of 2015 IEEE Trustcom/BigDataSE/ISPA*, Helsinki, Finland, 2015, pp. 834-839.

[15] Open Networking Foundation, "OpenFlow Switch Specification," 2015 [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf.

[16] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: an approach to universal topology generation," in *Proceedings of 9th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Cincinnati, OH, 2001, pp. 346-353.

[17] R. C. Prim, "Shortest connection networks and some generalizations," *Bell Labs Technical Journal*, vol. 36, no. 6, pp. 1389-1401, 1957.

[18] X. Jia and L. Wang, "A group multicast routing algorithm by using multiple minimum Steiner trees," *Computer Communications*, vol. 20, no. 9, pp. 750-758, 1997.

[19] C. P. Low and X. Song, "On finding feasible solutions for the delay constrained group multicast routing problem," *IEEE Transactions on Computers*, vol. 51, no. 5, pp. 581-588, 2002.

[20] Y. L. Wang, "Based QoS constrained group multicast routing for multimedia communication," in *Proceedings of 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering*, Changchun, China, 2010, pp. 296-299.

[21] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269-271, 1959.

[22] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617-1622, 2002.

Hoejung Jung

She received the B.S. degree in Computer Science from the Kyonggi University, Korea, in 2015. She is currently M.S. candidate in Computer Science from Kyonggi University. Her research interests include image processing, sensor networks, and wireless systems.

Namgi Kim

He received the B.S. degree in Computer Science from Sogang University, Korea, in 1997, and the M.S. and the Ph.D. degrees in Computer Science from KAIST in 2000 and 2005, respectively. From 2005 to 2007, he was a research member of the Samsung Electronics. Since 2007, he has been a faculty of the Kyonggi University. His research interests include sensor system, wireless system, and mobile communication.