PAPER

Load-balanced and interference-aware spanning tree construction algorithm for TDMA-based wireless mesh networks

Hiroshi TOKITO[†a)], *Nonmember*, Masahiro SASABE[††b)], Go HASEGAWA[†††c)], *and* Hirotaka NAKANO[†††d)], *Members*

SUMMARY Wireless mesh networks have been attracting many users in recent years. By connecting base stations (mesh nodes) with each other via wireless connections, these network can achieve a wide-area wireless environment with flexible configuration and low cost at the risk of radio interference between wireless links. When we utilize wireless mesh networks as infrastructures for Internet access, all network traffic from mobile nodes goes through a gateway node that is directly connected to the wired network. Therefore, it is necessary to distribute the traffic load by deploying multiple gateway nodes. In this paper, we propose a spanning tree construction algorithm for TDMA-based wireless mesh networks with multiple gateway nodes that works to maximize the traffic volume transferred between the mesh network and the Internet (system throughput) by taking account of the traffic load on the gateway nodes, the access link capacity and radio interference. Through a performance evaluation, we show that the proposed algorithm increases the system throughput regardless of the bottleneck position and achieves up to 3.1 times higher system throughput than a conventional algorithm.

key words: *wireless mesh network, spanning tree, load balance, radio interference*

1. Introduction

Wireless mesh networks (hereafter, called mesh networks) have been attracting many users in recent years [1, 2]. As shown in Fig. 1, base stations (mesh nodes) connect with each other via wireless connections in mesh networks. An end node (station) connects to one of the mesh nodes located within its transmission range. The station can communicate with a gateway node, which is a mesh node directly connected to the wired network, by means of multi-hop communication with the help of mesh nodes on the path to the gateway node. Here, the path is determined by the spanning tree construction algorithms [3, 4]. Mesh networks can achieve a wide-area wireless environment with flexible configuration and low cost, however they are also susceptible to radio interference because mesh nodes depend on wireless connections. Because of these favorable properties, there have already been several experimental trials [5–9].

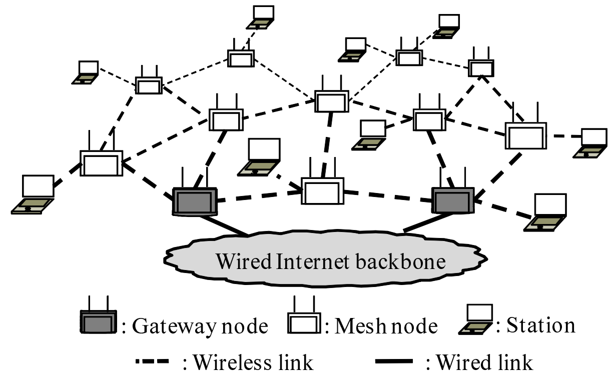


Fig. 1 Mesh network.

In the future, mesh networks are expected to become a basic infrastructure for Internet access in areas with poor wired network infrastructures, such as rural areas and isolated islands. However, all network traffic from stations must go through the gateway node [10]. Since the gateway node is the single entry point for all traffic between the mesh network and the wired Internet, the capacity of the gateway node's access link typically limits the capacity of the mesh network.

Therefore, the deployment of multiple gateway nodes is needed to efficiently distribute the entire traffic load. In conventional spanning tree construction algorithms [3, 4], a mesh node selects the closest gateway node in terms of the path length between the mesh node and gateway node. As a result, large amounts of traffic may concentrate in some gateway nodes and congestion occurs depending on the geographic distribution of gateway nodes, mesh nodes, and stations. This indicates that simply deploying multiple gateway nodes will not result in effective load balancing. For better results, each mesh node must select a gateway node to access the Internet based on the traffic load and access link capacity of the various gateway nodes. However, when a mesh node selects a path that is longer than the

[†]The author is with the Graduate School of Information Science and Technology, Osaka University.
[††]The author is with the Graduate School of Engineering, Osaka University.
[†††]The author is with the Cybermedia Center, Osaka University.
a) E-mail: h-tokito@ist.osaka-u.ac.jp
b) E-mail: sasabe@comm.eng.osaka-u.ac.jp
c) E-mail: hasegawa@cmc.osaka-u.ac.jp
d) E-mail: nakano@cmc.osaka-u.ac.jp

shortest path, the potential for radio interference in the mesh network increases. Consequently, both load balancing among gateway nodes and suppressing the radio interference are significant to maximize the traffic volume transferred between the mesh network and the Internet.

Here, the geographic distribution of traffic load in the mesh network may change with time due to addition/removal of gateway/mesh nodes, arrivals/movements/departures of stations, and changes of traffic demand for stations. The path for each station should be reconstructed as soon as these environmental changes occur. This means that the time complexity for constructing the spanning tree should be small as possible.

In this paper, to satisfy these requirements, we propose a spanning tree construction algorithm, called load-balanced and interference-aware tree construction (LITC), for mesh networks that have multiple gateway nodes and are based on time division multiple access (TDMA). The purpose of LITC is to maximize the traffic volume transferred between the mesh network and the Internet while suppressing the time complexity for constructing the spanning tree as possible. LITC first constructs a spanning tree that minimizes the radio interference in the mesh network necessary for maximizing the traffic volume. Then, LITC reconstructs the spanning tree in order to disperse the traffic load on the gateway nodes, while taking account of the impact of the increase in radio interference, in order to maximize the traffic volume transferred between the mesh network and the Internet. Through a performance evaluation, we show that LITC can increase the amount of the traffic transferred between the mesh network and the Internet regardless of the bottleneck position on the access link of the gateway node or on the wireless link. We further show that LITC achieves up to 3.1 times higher performance than a conventional spanning tree construction algorithm.

The rest of this paper is organized as follows. In Section 2, we describe the current standardization of mesh networks and related work on spanning tree construction algorithms that consider load balancing or radio interference in mesh networks. In Section 3, we discuss the network model and the performance metrics. We present the conventional and proposed spanning tree construction algorithms in Section 4. In Section 5, we show the effectiveness of our proposed algorithm through several simulations. Finally, Section 6 gives the conclusions of this paper.

2. Related work

IEEE 802.11s and 802.16 working groups define an architecture and a protocol for mesh networks, respectively [11, 12]. IEEE 802.11s assumes that a mesh network is composed of approximately 30 wireless LAN access points. The default routing method is the hybrid wireless mesh protocol (HWMP) [13], which is based on a modified ad hoc on-demand distance vector (AODV) protocol [14] called radio metric AODV (RM-AODV) [15]. If all the access points are located on fixed points and the topology of the mesh network does not change, HWMP applies proactive routing by building a spanning tree. Each mesh node selects its parent node by a metric based on the condition of wireless resource, that is, airtime, and does not consider the access link capacity on the gateway node.

IEEE 802.16 standardizes the wireless metropolitan area network (WMAN) mesh network. IEEE 802.16 supports two modes: point to multipoint (PMP) mode and mesh mode. In the PMP mode, each mesh node directly communicates with the gateway node. In the mesh mode, each mesh node communicates with the gateway node through multi-hop communication by relaying its traffic to a mesh node that is randomly selected among the available mesh nodes within its transmission range. The proposed algorithm is applicable to both modes and can maximize the traffic volume transferred between the mesh network and the Internet.

IEEE 802.11s and 802.16 mesh networks are built by adding necessary information to route request (RREQ)/route relay (RREP) messages and mesh network configuration (MSG-NCFG)/mesh network entry (MSG-NENT) messages, respectively. The proposed algorithm requires information of the traffic load on the gateway node and the radio interference in the mesh network. These information can also be exchanged among mesh nodes by extending the messages.

Various routing methods used to balance the load for communications between mesh nodes in a mesh network have been proposed in Refs. [16–18]. These studies proposed routing metrics to increase the throughput. Draves *et al.* [16] proposed a routing metric as a function of average transmission time per packet, that is, the weighted cumulative expected transmission time (WCETT). Liu and Liao [17] proposed normalized bottleneck link capacity (NBLC) as the routing metric. NBLC is the function of the available time that the channel assigned to the bottleneck link along a path can use. Note that the available time is normalized by the path length. By using these metrics, a mesh node can select a path according to its traffic load. In addition, Koksak and Balakrishnan [18] proposed a routing metric based on the expected transmission count (ETX) [19], which is the function of the average number of transmissions including retransmission per packet. The authors extend ETX to modified ETX (mETX) in order to handle the transient nature of successful packet transmission. Even under mesh networks with unstable wireless conditions, mETX achieves high throughput. Although these methods aim to maximize the throughput of mesh nodes, not all consider the possibility of traffic congestion on the gateway node.

For load balancing in a mesh network with gateway nodes by means of spanning tree construction, Chen *et al.* [20] and Nguyen *et al.* [21] proposed spanning tree construction algorithms that focus on the transmission time of a mesh node and the mesh nodes interfered by its transmission, and the contention window size, respectively. These algorithms distribute the traffic load on a mesh node and the traffic load on a gateway node by reducing the use of wireless links on which traffic is concentrated. Kuran *et al.* [22] proposed the spanning tree construction algorithm for avoiding congestion by using the queue length of the busy wireless link as an indicator for detecting the start of congestion. In this algorithm, each mesh node changes its parent node when congestion occurs. As a spanning tree construction algorithm for distributing the traffic load on a gateway node, Lakshmanan *et al.* proposed multi-gateway association (MGA) [23]. In MGA, a mesh node determines the path to the gateway node based on the amount of network resources on each link along the path. Although these algorithms are effective when the wireless link connected to a gateway node is a bottleneck, they do not assume that the access link capacity on a gateway node is a bottleneck. In this paper, we propose a spanning tree construction algorithm that maximizes the traffic volume transferred between the mesh network and the Internet, regardless of the bottleneck position, by taking account of not only the conditions of the wireless links but also the conditions of the access links.

Alternatively, many researchers have been trying to improve the throughput by considering the radio interference in IEEE 802.16 mesh networks [24–26]. Jin *et al.* [24] proposed a spanning tree construction algorithm that considers the relationship between the interference of the wireless links and their traffic demand. Jiao *et al.* [25] focused on the energy consumed by transmitting one-byte data to a parent node. Each mesh node selects a parent node to minimize the total energy consumed by all mesh nodes on the path. As a result, the area in which a mesh node interferes with the transmission from other mesh nodes decreases. Wei *et al.* [26] also proposed a spanning tree construction algorithm that minimizes radio interference in the mesh network by using blocking metric which represents goodness of a path in terms of radio interference caused by the transmission along with the path (See the details in Section 4.1.2). Since the blocking metric is easily observed, the proposed algorithm uses the blocking metric when it constructs the spanning tree to maximize the internal traffic volume of the mesh network.

3. System model

In this section, we explain the model of a mesh network and the performance metric.

3.1 Network model

We assume a communication graph $G = (V, E)$, where $V = \{v_1, \ldots, v_m, \ldots, v_n\}$ is the set of mesh nodes ($n \geq m \geq 1$, n is the number of mesh nodes, m is the number of gateway nodes, and mesh nodes from v_1 to v_m are gateway nodes), and E is the set of wireless links $l_{i,j} = (v_i, v_j)$. A mesh node v_i has the wireless link $l_{i,j}$ to a mesh node v_j when the following condition is satisfied:

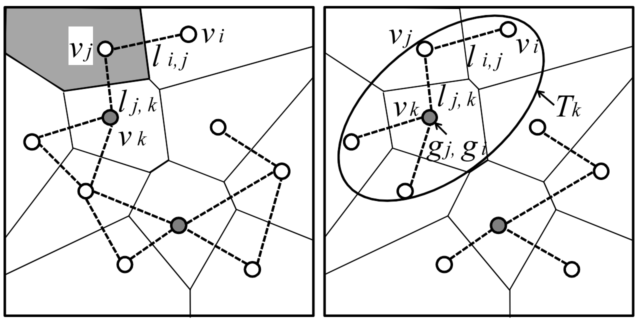
$$\|v_i - v_j\| \leq t_i \tag{1}$$

where t_i is the transmission range of v_i. Each mesh node has a path to a gateway node, which is calculated by one of the spanning tree construction algorithms (we describe the details in Section 4). Let us denote $T_k = (V_k, E_k)$ as the spanning tree whose root is the gateway node v_k, where V_k and E_k are the set of mesh nodes and the set of wireless links consisting of the spanning tree, respectively. In addition, let us denote c_k as the capacity of the access link on a gateway node v_k and g_i as the ID of the gateway node through which the traffic from mesh node v_i passes.

Here, we describe the traffic demand for stations. First, we assume the outbound traffic that goes from stations to gateway nodes. The traffic demand d_i for mesh node v_i is defined as the sum of the traffic demand $d_i^{(s)}$ for connected stations and the traffic demand $d_{*\to i}$ for the mesh nodes from which v_i receives traffic. When we assume that stations are uniformly located in the mesh network, each station connects to the nearest mesh node, and the traffic volume from each station is identical, we can derive $d_i^{(s)}$ as the function of the size of the Voronoi area [27] of v_i. Likewise, we define $d_{j\to i}$ as the amount of traffic which v_j sends to v_i on the wireless link $l_{j,i}$. We further define N_i as the set of neighboring mesh nodes of v_i on the spanning tree. As a result, $d_{*\to i}$ is obtained as $\sum_{v_j \in N_i} d_{j\to i}$, the traffic demand d_i for v_i becomes $d_i^{(s)} + \sum_{v_j \in V_i} d_{j\to i}$.

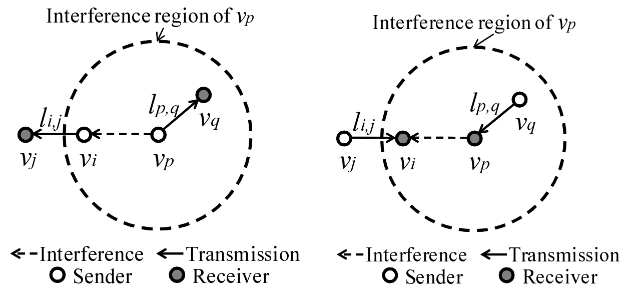
Next, we consider the inbound traffic that goes from the gateway nodes to the stations. When we assume that the amount of traffic sent to a station is identical among all stations, the traffic volume which v_i sends to the connected stations becomes $\delta d_i^{(s)}$, where δ is the ratio of the outbound traffic volume to the inbound traffic volume. The derivation of traffic demand for each mesh node is the same as that in the case of outbound traffic. Hereafter, we assume only the case of the outbound traffic for simplicity. Although we can adopt other definition of traffic demand, the above assumption seems to be valid in terms of fairness among stations.

Fig. 2 shows an example of the mesh network discussed in this paper. In Fig. 2 (a), v_k is a gateway node, and v_i and v_j are mesh nodes. The dotted line is the wireless link. The area separated by the solid



(a) Topology. (b) Spanning tree.

Fig. 2 An example of a mesh network discussed in this paper.



(a) RTS message. (b) CTS message.

Fig. 3 Interference based on the RTS/CTS model.

line equals the Voronoi area of each mesh node. $l_{i,j}$ is the wireless link between v_i and v_j, and $l_{j,k}$ is the wireless link between v_j and v_k. We can obtain $d_j^{(s)}$ by calculating the size of the shaded area. Fig. 2 (b) shows an example of the spanning trees that are constructed on the topology in Fig. 2 (a). On the spanning tree T_k rooted by v_k, d_j and $d_{j\to k}$ become $d_j^{(s)} + d_i^{(s)}$ since v_j relays the traffic from v_i. Furthermore, g_i and g_j become k.

We define the radio interference between wireless links based on the request to send/clear to send (RTS/CTS) model [28]. Fig. 3 (a) shows the case in which wireless link $l_{i,j}$ experiences interference by wireless link $l_{p,q}$. Suppose the situation in which v_p sends an RTS message to v_q and another mesh node v_i receives the RTS message. The RTS/CTS model considers the communication from v_p to v_q (link $l_{p,q}$) and that from v_i to v_j (link $l_{i,j}$) to interfere with each other. The interference caused by the exchange of a CTS message is similar in Fig. 3 (b). Therefore, the condition whereby $l_{p,q}$ interferes with $l_{i,j}$ is denoted as follows:

$$\begin{aligned} &\|v_i - v_p\| \le \gamma_p, \quad \|v_j - v_p\| \le \gamma_p, \\ &\|v_i - v_q\| \le \gamma_q, \quad \|v_j - v_q\| \le \gamma_q. \end{aligned} \tag{2}$$

Here, γ_p is the interference range of v_p, which is defined as $\gamma_p = t_p \times \mu$; μ is the constant number and is generally estimated between $2 \le \mu \le 4$ [29]. Similarly, γ_q is the interference range of v_q. Eq. (2) indicates that $l_{p,q}$ interferes $l_{i,j}$ when v_i or v_j is located in the interference range of v_p or that of v_q.

IEEE 802.11 MAC protocol is based on RTS/CTS. On the other hand, IEEE 802.16 applies TDMA MAC protocol without relying on RTS/CTS. It is noted that the situation of the actual interference cannot be modeled exactly by the interference model in IEEE 802.16 [30], which assumes that an interference occurs only between nodes within two physical hop count. In this paper, we use RTS/CTS model since it is one of the most famous interference models and can define the interference between wireless links easily. We should note here that the interference model in IEEE 802.16 can also be applied to our approach.

3.2 Performance metric

In TDMA link scheduling [29], time slots are assigned to each wireless link proportionally to its link weight based on its traffic demand. When we assume that the link weight of $l_{i,j}$ is $d_{i\to j}$, the number of time slots $f_{i,j}$ assigned to $l_{i,j}$ is defined as follows:

$$f_{i,j} = \lceil \alpha \times d_{i\to j} \rceil \tag{3}$$

where α ($0 < \alpha \le 1$) is the quantization factor for suppressing the total amount of time slots, called frame length f, and $\lceil\cdot\rceil$ is a ceiling function which maps a real number to the next largest integer. f is the number of time slots needed for all wireless links to be assigned time slots proportional to their traffic demand. It requires a large time complexity for calculating f. When we use the link scheduling based on Ref. [29], the time complexity for calculating f becomes $O(n^3)$. From the time slots assigned to each link, the utilization u_i of a gateway node v_i is expressed as follows:

$$u_i = \sum_{v_j \in N_i} \frac{f_{i,j}}{f}. \tag{4}$$

From Eq. (4), the traffic rate on the wireless link $l_{i,j}$ (*wireless link throughput*) becomes $((f_{i,j}/f) \times s)$ [bps], where s [bps] is the wireless link capacity in the mesh network. Hence, the rate λ_i [bps] of the traffic which v_i can receive is expressed as

$$\lambda_i = u_i \times s.$$

Similarly, the rate ρ_i [bps] of the traffic that v_i can send to the wired network (*access link throughput*) cannot exceed the capacity c_i [bps] of the access link of v_i. Therefore, ρ_i is given as follows:

$$\rho_i = \min(\lambda_i, c_i).$$

In this paper, we propose a spanning tree construction algorithm to maximize the total amount of access link throughput on each gateway node, which is essentially the *system throughput* as follows:

$$\sum_{1 \le i \le m} \rho_i.$$

Table 1 List of notations for spanning tree construction algorithms.

Notation	Definition
G	Communication graph
V	Set of mesh nodes v_i ($1 \le i \le n$)
E	Set of wireless links $l_{a,b}$ ($\forall v_a, v_b \in V$ satisfying Eq. (1))
m	Number of gateway nodes ($1 \le m \le n$)
N_i	Set of v_i's neighboring nodes on the spanning tree
T_k	Tree whose root is the gateway node v_k
T	Set of trees T_k
g_i	ID of the gateway node through which the traffic from v_i passes
d_i	Traffic demand for v_i
r_i	Sum of the wireless link throughput of the links between v_i and $v_* \in N_i$
c_k	Access link capacity of the gateway node v_k
$x_{i,j}$	Path length from v_i to v_j
$y_{k,i}$	v_i's parent node in T_k

4. Spanning tree construction algorithms

In this section, we explain conventional spanning tree algorithms and our proposed spanning tree algorithm. Table 1 shows the notations used in this section.

4.1 Conventional algorithms

We explain the algorithm of shortest path tree construction and that of interference-aware tree construction as conventional spanning tree algorithms.

4.1.1 Shortest path tree construction

Algorithm 1 shows the algorithm of shortest path tree construction (SPTC). In SPTC, when there is more than one shortest path from a mesh node to different gateway nodes or to the same gateway node, a mesh node chooses one of paths at random.

When SPTC uses Dijkstra's algorithm [31] for calculating the shortest path tree, the time complexity of obtaining the path between a mesh node and a gateway node is $O(n^2)$. Therefore, the time complexity of SPTC is $O(n^2)$. In this paper, we do not use the Dijkstra's algorithm with heap which reduces the time complexity of SPTC by $O((n+e)\log n)$, where e is the number of wireless links.

4.1.2 Interference-aware tree construction

Wei *et al.* [26] proposed a spanning tree construction algorithm for routing in order to minimize radio interference in a mesh network. The authors define *blocking value* $b(i)$ of a mesh node v_i as the number of mesh nodes affected by interference from v_i's transmission. Then they define *blocking metric* $B(i,j)$ of the path from v_i to v_j as the sum of $b(k)$ of mesh nodes v_k along the path. Each mesh node selects the path that minimizes the blocking metric in order to decrease the radio interference and increase the wireless link throughput.

Algorithm 1 Shortest path tree construction (SPTC).

```
Input: $G = (V, E)$
Output: $T$
1: for all $v_i$ and $v_j$ such that $1 \le i \le m$ and $m+1 \le j \le n$ do
2:     Construct $T_i$ by calculating the shortest path from $v_i$ to $v_j$.
3: end for
4: for all $v_i$ such that $m+1 \le i \le n$ do
5:     $g_i = \arg\min_{1 \le k \le m}(x_{k,i})$.
6:     Set $v_i$'s parent node to $y_{i,g_i}$.
7: end for
```

Algorithm 2 Interference-aware tree construction (ITC).

```
Input: $G = (V, E)$
Output: $T$
1: for all $v_i$ such that $1 \le i \le n$ do
2:     Calculate $b(i)$.
3: end for
4: for all $v_i$ and $v_j$ such that $1 \le i \le m$ and $m+1 \le j \le n$ do
5:     Construct $T_i$ by calculating the path minimizing $B(i,j)$.
6: end for
7: for all $v_i$ such that $m+1 \le i \le n$ do
8:     $g_i = \arg\min_{1 \le k \le m}(B(k,i))$.
9:     Set $v_i$'s parent node to $y_{i,g_i}$.
10: end for
```

In this paper, we use a spanning tree construction algorithm that minimizes the blocking metric, that is interference-aware tree construction (ITC), for maximizing the wireless link throughput. Algorithm 2 shows the ITC algorithm.

$b(i)$ of mesh nodes v_i can be computed within $O(n^2)$. When the spanning tree is constructed by Dijkstra's algorithm with $b(i)$ as the link weight of $l_{i,j}$, the time complexity of obtaining the path becomes $O(n^2)$ (if ITC uses Dijkstra's algorithm with heap, the time complexity is $O((n+e)\log n)$). Consequently, the time complexity of ITC becomes $O(n^2)$.

4.2 Proposed algorithm

We propose load-balanced and interference-aware tree construction (LITC) algorithm.

LITC constructs a spanning tree for maximizing the system throughput while suppressing the time complexity as possible. To maximize the system throughput, it is the best way to directly use the system throughput as the metric for the parent node selection. However, this approach requires $O(n^5)$ time complexity to construct a spanning tree including the overhead for calculating the frame length, $O(n^3)$. Therefore, LITC tries to avoid calculating the frame length in the process of spanning tree construction as possible.

Algorithm 3 Load-balanced and interference-aware tree construction (LITC).

```
Input: $G = (V, E)$ and $h = 0$
Output: $T$
 1: Construct $T_i$ $(1 \leq i \leq m)$ by algorithm 2.
 2: Calculate $r_i$ $(1 \leq i \leq m)$.
 3: for all $v_i$ such that $1 \leq i \leq m$ do
 4:     Define $z_i$ for keeping the bottleneck information.
 5:     if $r_i > c_i$ then
 6:         $z_i = 0$. // The access link of $v_i$ is a bottleneck.
 7:     else
 8:         $z_i = 1$. // The access link of $v_i$ is not a bottleneck.
 9:     end if
10: end for
11: if $z_i$ of which the value is 0 exists then
12:     Determine the order of changing parent nodes by algorithm 4.
13:     do
14:         $D_h = T$.
15:         for all $v_i$ and $v_j$ such that $v_i \in V$ and $l_{i,j} \in E$ do
16:             if $\omega$ decreases and the increase in the hop count
                is less than $h$ when $v_j$ becomes $v_i$'s parent node
                then
17:                 Set $v_i$'s parent node to $v_j$.
18:             end if
19:         end for
20:         $h = h + 1$.
21:     while the system throughput increases.
22:     $T = D_{h-1}$.
23: end if
```

We show the algorithm of LITC in Algorithm 3. In order to increase the system throughput, we need to take account of both wireless link throughput and utilization of access link. When a mesh node selects a path that is longer than the shortest path to equalize the utilization of access link, the wireless link throughput decreases according to the increase in path length. It is important to suppress deterioration of the wireless link throughput as possible while making the utilization of the access link the same among gateway nodes.

First, LITC maximizes the wireless link throughput by using ITC. Then, if $r_i > c_i$ at gateway node v_i, the access link on v_i becomes a bottleneck. When there is more than one gateway node on which the access link is a bottleneck, LITC reconstructs the spanning tree so that the utilization of the access links among gateway nodes becomes the same. For this purpose, we define the bias of the utilization of access links as follows:

$$\omega = \frac{1}{m} \sum_{i=1}^{m} \left(d_i - \frac{c_i}{c} d \right)^2. \qquad (5)$$

Here, d is the sum of the traffic demand for gateway nodes, that is $\sum_{1 \leq i \leq m} d_i$, and c is the sum of the access link capacity of the gateway nodes, that is, $\sum_{1 \leq i \leq m} c_i$; d_i/c_i indicates the utilization of the access link on gateway node v_i. Eq. (5) indicates that each gateway node relays traffic whose volume is proportional to its access link capacity, that is, $(c_i/c)d$. Each mesh node selects its parent node as one which minimizes ω in order to equalize utilization of the access links among gateway nodes.

Algorithm 4 Determination of the order of mesh nodes switching their parent node.

```
Input: $G = (V, E)$.
Output: Order information.
1: for all $v_i$ such that $m + 1 \leq i \leq n$ do
2:     if $c_{g_i} \neq \max_{1 \leq k \leq m}(c_k)$ then
3:         Find the nearest gateway node $v_k$ that satisfies
           $c_k > c_{g_i}$.
4:         $h_i = x_{i,k}$.
5:     else
6:         $h_i = \infty$.
7:     end if
8: end for
9: Arrange the order of mesh nodes in ascending order of $h_i$.
```

At this time, we prevent deterioration of the wireless link throughput as possible. We first select mesh nodes relying on the gateway node with the highest utilization. Then, we arrange the mesh nodes in ascending order of hop count between the selected mesh node and the second-closet gateway node. Based on this order, we switch the parent nodes if system throughput increases. Algorithm 4 shows the algorithm determining the order of mesh nodes switching their parent nodes. In Algorithm 4, the order is determined according to the following two steps.

1. Ascending order of the hop count from the gateway node with larger access link capacity among mesh nodes selecting a gateway node with the non-maximum access link capacity.
2. The mesh nodes selecting the gateway node with the maximum access link capacity.

Note that this approach cannot guarantee the degree of the increase in the path length. To tackle this problem, LITC suppresses an increase in the hop count caused by changing the parent nodes up to h. LITC repeats tree construction from $h = 0$ to the value that achieves maximum system throughput since the optimal value of h varies according to the network environment and is difficult to obtain in advance.

The worst situation of time complexity for LITC occurs when the access link on a gateway node is a bottleneck. In this situation, the time complexity of LITC becomes $O(n^4)$ since LITC needs to calculate the system throughput of which the time complexity is $O(n^3)$ and repeat the tree construction up to n times before system throughput reaches maximum.

5. Performance evaluation

In this section, through several simulations, we demonstrate not only the effectiveness and but also the problems of the proposed algorithm.

Table 2 Parameter settings for performance evaluation

Area of deployment	1×1
Number of simulations	100
Number of gateway nodes	4
Location of gateway nodes	(see Fig. 4)
Number of mesh nodes except gateway nodes	20, 40, 60, 80, 100
Location of mesh nodes	Uniform distribution
Transmission range t_i	Minimum range establishing at least one path from a mesh node to a gateway node
Access link capacity	10, C_x [Mbps]
Wireless link capacity	11, 70 [Mbps]
Traffic demand $d_i^{(s)}$ for v_i 's connected stations	The size of Voronoi area of v_i (See the details in Section 3.1)
Quantization factor α	10^{-2}
Ratio μ of interference range to transmission range	2

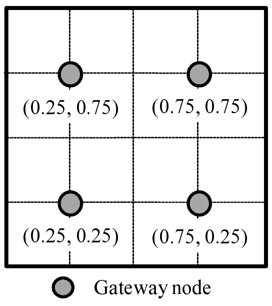
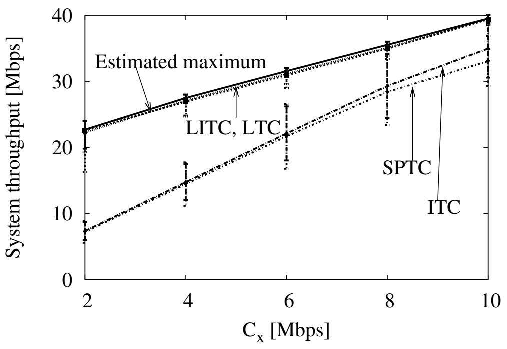
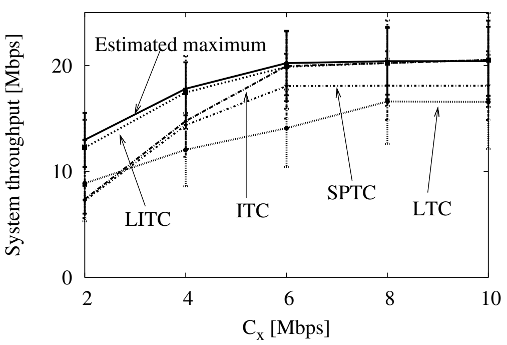


Fig. 4 Location of gateway nodes.



(a) Wireless link capacity = 70 Mbps



(b) Wireless link capacity = 11 Mbps

Fig. 5 Transitions of throughput when the access link capacity of gateway nodes varies.

5.1 Simulation model

Table 2 and Fig. 4 show the parameter settings used in the evaluation. We used IEEE802.11b (11 Mbps) and IEEE802.16 (70 Mbps) wireless link capacity which represents the situation where the wireless link is a bottleneck and the access link on a gateway node is a bottleneck, respectively, on the spanning tree constructed by SPTC. In addition, we deployed the gateway nodes on grids, as shown in Fig. 4. Two gateway nodes randomly selected among the four were assigned the access link with 10 Mbps and the remaining gateway nodes were assigned the access link with C_x Mbps. We varied C_x from 2 to 10.

In the following, we also evaluate a spanning tree construction algorithm that aims to equalize utilization of the access links on gateway nodes, that is gateway load-balanced tree construction (LTC), for comparison purpose. LTC first constructs the spanning tree by using SPTC. Then, a mesh node selects its parent node minimizing ω in the order of Algorithm 4. Although LITC considers the impact of the increase in the radio interference, LTC does not. Furthermore, as the estimated maximum system throughput under no time constraint, we also show the system throughput of the spanning tree construction algorithm which directly uses the system throughput as the metric for the parent node selection, that is throughput based tree construction (TTC). The order of calculating the system throughput is $O(n^3)$. In addition, a mesh node checks the mesh nodes within its transmission range in order to decide whether system throughput increases. Therefore, the time complexity of TTC becomes $O(n^5)$.

Hereafter, we show the results when the system throughput becomes the maximum under the condition that each gateway node does not receive grater traffic than its access link capacity.

5.2 System throughput

Fig. 5 shows averages with 95 % confidence intervals of the system throughput for each spanning tree construction algorithm when C_x varies and the number of mesh nodes is set to 100.

As shown in Fig. 5 (a), when the wireless link capacity is 70 Mbps, the system throughput of LITC and LTC becomes almost the same as the estimated maximum and are up to 3.1 times higher than those of SPTC and ITC. SPTC and ITC decrease their system throughput due to the bottleneck on the access link with C_x Mbps. In contrast, LITC and LTC can achieve system throughput close to the estimated maximum since these algorithms work to construct the spanning tree so that the gateway nodes receive traffic cor-

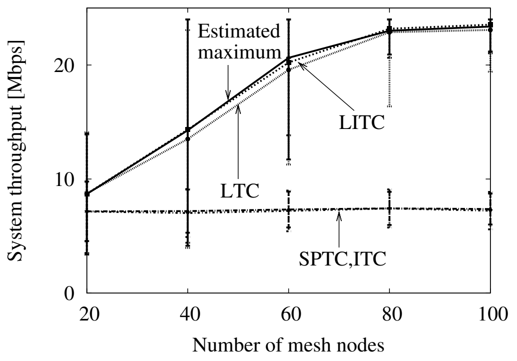
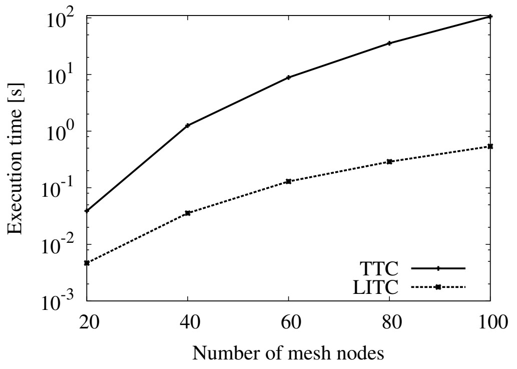


Fig. 6 Transitions of throughput when the number of mesh nodes varies (wireless link capacity = 70 Mbps, C_x = 2 Mbps).

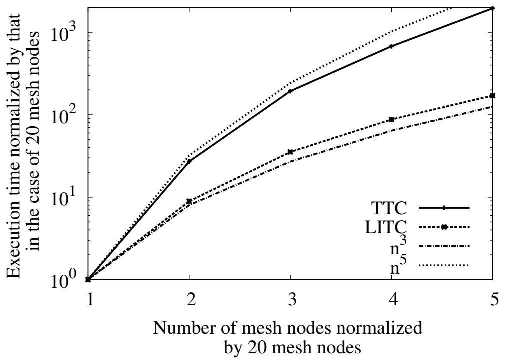
responding to their access link capacities.

As shown in Fig. 5 (b), when C_x is 2 Mbps and 4 Mbps, LITC achieves higher system throughput than do the other algorithms. In addition, the system throughput of ITC becomes high when C_x is larger than 6 Mbps. In Fig. 5 (b), the wireless link throughput is insufficient. Therefore, the system throughput of SPTC and LTC, which do not take account of the radio interference, deteriorates due to the bottleneck on the wireless link when C_x is larger than 6 Mbps. In contrast, ITC and LITC achieve high system throughput by reducing the radio interference during the tree construction. When C_x is 2 Mbps and 4 Mbps, the system throughput of SPTC and ITC decreases since the bottleneck occurs at the access link with C_x Mbps. At this time, the system throughput of LTC deteriorates because of the limited wireless link throughput. In such a situation, LITC can realize almost the same system throughput as the estimated maximum since LITC considers both the access link capacity of gateway nodes and the radio interference in the mesh network.

Fig. 6 shows averages with 95 % confidence intervals of the system throughput of each spanning tree construction algorithm when the number of mesh nodes varies, C_x is 2 Mbps, and the wireless link capacity is 70 Mbps. We find that LITC achieves almost the same system throughput as the estimated maximum independent of the number of mesh nodes. The system throughput of LITC and LTC deteriorates according to the decrease in the number of mesh nodes. The smaller the number of mesh nodes is, the smaller the number of candidate parent nodes for a mesh node. Thus, the number of mesh nodes switching their parent node decreases. In addition, the widths of the confidence intervals of these algorithms become wide when there is a small number of mesh nodes. The location of mesh nodes tends not to follow uniform distribution in this situation. As a result, the system throughput of these algorithms varies depending on the simulations. However, the system throughput of SPTC and ITC does not vary irrespective of the number of mesh nodes. The system throughput of these algorithms is suppressed by



(a) Transition of execution time.



(b) Transition of execution time normalized by the execution time for 20 mesh nodes.

Fig. 7 Execution time.

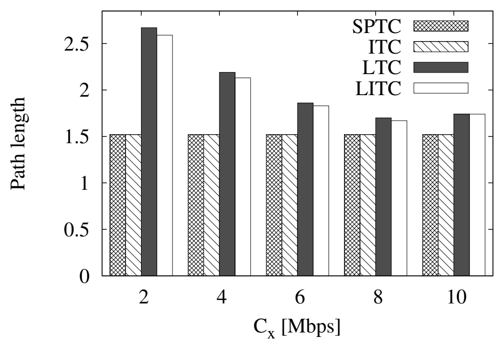
the bottleneck on the access link with C_x Mbps. Consequently, the system throughput of these algorithms becomes low without demonstrating their advantages.

5.3 Execution time

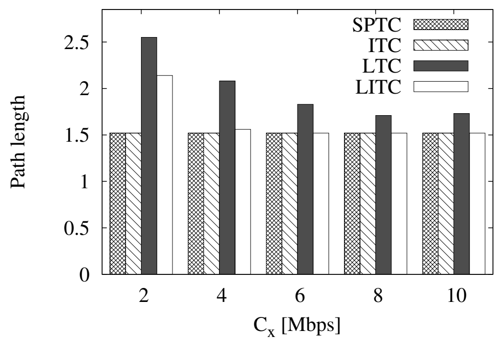
Fig. 7 (a) illustrates the average execution time according to the number of mesh nodes when C_x is 2 Mbps and the wireless link capacity is 70 Mbps. Fig. 7 (b) shows the execution time normalized by that in the case of 20 mesh nodes. These results were measured by a PC with a Pentium(R) D 3.4 GHz CPU and a 2 GB memory. Since the time complexity of LITC is smaller than that of TTC, the execution time of LITC is as much as 1/200 shorter than that of TTC. In addition, Fig. 7 (b) shows that the execution time of LITC is similar to $O(n^3)$ despite the fact that the theoretic time complexity of LITC is $O(n^4)$. Although the threshold of hop count h in LITC can become n in theory, h is as much as 6 in this simulation, which is much smaller than n. Hence, as a practical measure, LITC needs to repeat tree construction several times to maximize the system throughput, and the execution time of LITC remains at $O(n^3)$.

5.4 Path length

We demonstrated that LITC contributes to increasing



(a) Wireless link capacity = 70Mbps.



(b) Wireless link capacity = 11Mbps.

Fig. 8 Transitions of path length when the access link capacity of gateway nodes varies.

the system throughput in Section 5.2. However, it also tends to make the path length longer which results in the increase in transmission delay. In this subsection, we evaluate the transmission delay from the viewpoint of the path length.

Fig. 8 depicts how the average path length between a mesh node and a gateway node varies according to C_x when the number of mesh nodes is 100. The average path length of ITC is the same as that of SPTC, and the path length of LTC is longer than that of SPTC. In ITC, a mesh node selects the path minimizing the blocking metric. This means the path is set as short as possible. Therefore, the average path lengths of ITC and SPTC are the same. In contrast, LTC increases the path length because a mesh node selecting the gateway node with access link capacity C_x Mbps tries to select the gateway node with access link capacity 10 Mbps.

As shown in Fig. 8 (a), LITC increases the average path length up to 1.75 times compared to that of SPTC and it is almost the same path length as LTC since LITC tries to distribute the traffic on the gateway nodes with access link capacity C_x Mbps.

As shown in Fig. 8 (b), we find that the path length of LITC lengthens in the case of $C_x = 2$ Mbps. When $C_x = 2$ Mbps and 4 Mbps, the access link with C_x Mbps is a bottleneck. LITC attempts to construct a spanning tree to disperse the traffic on gateway nodes. Thus, the path length of LITC becomes long. However,

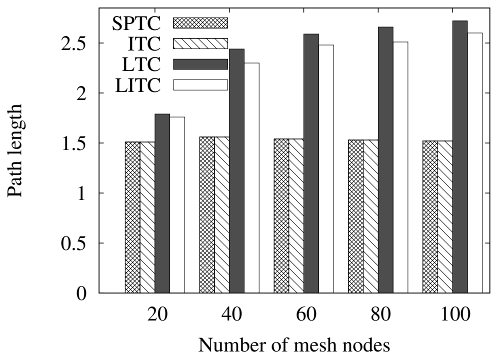


Fig. 9 Transitions of path length when the number of mesh nodes varies (wireless link capacity = 70 Mbps, $C_x = 2$ Mbps).

the path length of LITC is not so long when compared to that of LTC since LITC suppresses the increase in the hop count caused by changing the parent node up to h. The path length of LITC is similar to that of SPTC in a case of $C_x = 4$ Mbps. This is because the average of hop count threshold h is 0.19 whereas it is 1.95 in the case of $C_x = 2$ Mbps. Additionally, when C_x is more than 6 Mbps, LITC constructs a spanning tree that is similar to that constructed by ITC since the wireless link becomes a bottleneck. Therefore, the path length of LITC becomes almost the same as those of ITC and SPTC.

Fig. 9 depicts how the average path length between a mesh node and a gateway node varies according to the number of mesh nodes when C_x is 2 Mbps and the wireless link capacity is 70 Mbps. The path lengths of LITC and LTC lengthen according to the increase in the number of mesh nodes. When there is a small number of mesh nodes, the number of candidate parent nodes that can increase the system throughput may not exist. As a result, the path length does not change much. When increasing the number of mesh nodes, the path length lengthens according to the increase in the number of mesh nodes changing their parent nodes.

5.5 Trade-off between system throughput and realtime properties

We have shown that LITC can maximize the system throughput regardless of the bottleneck position but also makes the path length slightly longer than that of SPTC in the case of the bottleneck on an access link. The longer path length emerges from larger hop count threshold h and results in increase of transmission delay. The larger h also requires much time complexity. In this section, we reveal the trade-off between system throughput and realtime properties. Here, we focus on two kinds of realtime properties: path length and time complexity for tree construction.

Fig. 10 depicts the system throughput of LITC normalized by the estimated maximum and the path length of LITC normalized by that of SPTC according

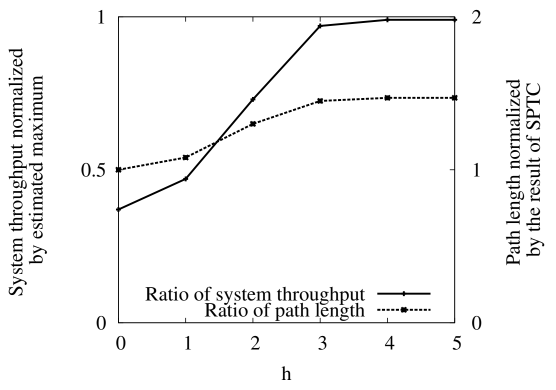


Fig. 10 Transitions of system throughput and path length when h varies (wireless link capacity = 70 Mbps, C_x = 2 Mbps).

to the upper bound of h when the wireless link capacity is 70 Mbps and C_x is 2 Mbps. Those parameters represent that the access link is a bottleneck and the wireless link throughput is enough. We increase h from 0 until whether the system throughput is maximum or the value of h reaches the upper bound. As shown in this figure, the larger h is, the larger system throughput becomes. The system throughput with $h = 5$ is 2.7 times higher than that with $h = 0$. The path length with $h = 5$ is also 1.5 times higher than that with $h = 0$. This indicates that there is a trade-off between system throughput and path length when the access link is a bottleneck. On the other hand, we obtained that average execution time varied from 0.113 [s] for $h = 0$ to 0.144 [s] for $h = 5$. The system throughput with $h = 4$ is almost the same as the estimated maximum. Therefore, LITC does not have to repeat tree construction many times and the execution time does not vary so much.

From above discussion, we need to set h properly depending on the application requirements. For example, for the applications sensitive to the transmission delay such as media streaming, h has to be set to low as possible while satisfying bit-rate constraint of the media streaming.

5.6 Summary

Table 3 summarizes the features of each spanning tree construction algorithm. In SPTC, each mesh node selects the shortest path to the gateway node. ITC constructs a spanning tree that minimizes radio interference in the mesh network and does not distribute the traffic on gateway nodes. Since decreasing radio interference is closely related to shortening the path length, the path length of ITC is almost the same as that of SPTC. However, SPTC and ITC decrease the system throughput when the access link on a gateway node becomes a bottleneck. LTC constructs a spanning tree that distributes the traffic on gateway nodes and does not consider radio interference in the mesh network. Hence, LTC decreases system throughput when the wireless link is a bottleneck. Furthermore, the path length of LTC becomes much longer than that of SPTC since there is a trade-off between the increase in path length and the distribution of traffic on gateway nodes. LITC considers both utilization of the access link of gateway nodes and the radio interference in the mesh network. As a result, LITC can maximize the system throughput regardless of the bottleneck position: either on the access link of a gateway node or on a wireless link. Furthermore, LITC suppresses the time complexity and cut down the time for constructing the spanning tree compared to the tree construction algorithm using directly system throughput. The path length of LITC becomes long as in LTC when LITC distributes the traffic on gateway nodes.

6. Conclusion and future work

In this paper, we proposed a spanning tree construction algorithm, called the load-balanced and interference-aware tree construction (LITC) algorithm, for a mesh network with multiple gateway nodes. The algorithm maximizes the traffic volume transferred between the mesh network and the Internet (system throughput). In LITC, each mesh node determines its parent node by considering both radio interference in the mesh network and the utilization of the access links at gateway nodes. Through performance evaluation, we showed that LITC achieved almost the maximum system throughput regardless of the bottleneck position: either the bottleneck on the access link of a gateway node or the bottleneck on a wireless link. We further showed that LITC could increase the system throughput up to 3.1 times higher than shortest path tree construction algorithm. Finally, LITC can reduce the time complexity for constructing the spanning tree by as much as 1/200 of that of TTC which is the tree construction algorithm directly using the system throughput as the metric.

As a remaining issue, we plan to design the protocol to achieve LITC on a real system. Since LITC requires the information on traffic load of gateway nodes and that on radio interference in the mesh network, we need some message exchanging mechanisms for this purpose. It is preferable that these mechanisms are decentralized to improve scalability to the number of mesh nodes.

We would also like to apply LITC to other wireless networks. For example, in a sensor network where the power consumption of nodes is a critical issue, it is necessary to distribute the traffic on not only sink nodes but also nodes near the sink node. We expect that energy efficient routing can be achieved by hierarchically applying LITC to each set of nodes which is grouped according to hop count from the sink node.

Table 3 Features of the spanning tree construction algorithms

Spanning tree construction algorithm		SPTC	ITC	LTC	LITC
System throughput	Bottleneck on a wireless link	low	high	low	high
	Bottleneck on an access link	low	low	high	high
Path length	Bottleneck on a wireless link	short	short	long	short
	Bottleneck on an access link	short	short	long	long
Time complexity		$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^4)$ (in practice, about $O(n^3)$)

Acknowledgement

This research was supported by a Grant-in-Aid for Young Scientists (B) 20700061 and a Grant-in-Aid for Scientific Research (B) 19360173 in Japan.

References

[1] L.F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," Computer Networks, vol.47, no.4, pp.445–487, March 2005.
[2] V. Navda, A. Kashyap, and S.R. Das, "Design and evaluation of iMesh: An infrastructure-mode wireless mesh network," Proceedings of IEEE World of Wireless Mobile and Multimedia Networks, pp.164–170, June 2005.
[3] IEEE 802.1D, "Standard for local and metropolitan area networks: media access control (MAC) bridges," 1998.
[4] IEEE 802.1S/D15, "Draft standard for local and metropolitan area networks: amendment 3 to 802.1Q virtual bridged local area networks: multiple spanning trees," 2002.
[5] MeshDynamics. available at `http://www.meshdynamics.com`.
[6] Tropos Networks. available at `http://www.tropos.com`.
[7] Roofnet. available at `http://pdos.csail.mit.edu/roofnet/doku.php`.
[8] BMN Lab wireless mesh networks research project. available at `http://www.ece.gatech.edu/research/labs/bwn/mesh`.
[9] MeshNetworks. available at `http://www.meshnetworks.com`.
[10] J. Janqeun and M.L.Sichitiu, "The nominal capacity of wireless mesh networks," IEEE Journal on Wireless Communications, vol.10, no.1, pp.8–14, Oct. 2003.
[11] The IEEE 802.11 Working Group for WLAN Standards. available at `http://grouper.ieee.org/groups/802/11/`.
[12] The IEEE 802.16 Working Group on Broadband Wireless Access Standards. available at `http://grouper.ieee.org/groups/802/16/`.
[13] IEEE 802.11 TGs, "Joint SEE-Mesh/Wi-Mesh Proposal to 802.11 TGs," Feb. 2006.
[14] C. Perkins, E.B. Royer, and S. Das, "Ad hoc on-demand distance vector (AODV)," RFC 3561, July 2003.
[15] S. Takeda, K. Yagyu, H. Aoki, and Y. Matsumoto, "Multi-interface oriented radio metric on-demand routing protocol for layer-2 mesh networks," IEICE Technical Report, pp.109–114, July 2005.
[16] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," Proceedings of ACM Mobile Computing and Networking, pp.114–128, Oct. 2004.
[17] T. Liu and W. Liao, "Capacity-aware routing in multi-channel multi-rate wireless mesh networks," Proceedings of IEEE Communications, pp.1971–1976, June 2006.
[18] C.E. Koksal and H. Balakrishnan, "Quality-aware routing metrics for time-varying wireless mesh networks," IEEE Journal on Selected Areas in Communications, pp.1984–1994, Nov. 2006.
[19] D.D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," IEEE Journal on Wireless Networks, pp.419–434, July 2005.
[20] L. Chen, Y. Tseng, D. Wang, and J. Wu, "Exploiting spectral reuse in routing, resource allocation, and scheduling for IEEE 802.16 mesh networks," IEEE Transactions on Vehicular Technology, March 2008.
[21] L.T. Nguyen, R. Beuran, and Y. Shinoda, "A load-aware routing metric for wireless mesh networks," Proceedings of IEEE Symposium on Computers and Communications, pp.429–435, July 2008.
[22] M.S. Kuran, G. Gur, T. Tugcu, and F. Alagoz, "Cross-layer routing-scheduling in IEEE 802.16 mesh networks," Proceedings of ACM MOBILe Wireless MiddleWARE, Operating Systems, and Applications, Feb. 2008.
[23] S. Lakshmanan, K. Sundaresan, and R. Sivakumar, "On multi-gateway association in wireless mesh networks," Proceedings of IEEE Wireless Mesh Networks, pp.64–73, Sept. 2006.
[24] F. Jin, A. Arora, J. Hwang, and H.A. Choi, "Routing and packet scheduling for throughput maximization in IEEE 802.16 mesh networks," submitted for publication, available at `http://www.seas.gwu.edu/~hchoi/publication/wireless/802.16mesh.pdf`.
[25] W. Jiao, P. Jiang, R. Liu, and M. Li, "Centralized scheduling tree construction under multi-channel IEEE 802.16 mesh networks," Proceedings of Global Telecommunication Conference, pp.4764–4768, Nov. 2007.
[26] H.Y. Wei, S. Genquly, R. Izmailov, and Z.J. Haas, "Interference-aware IEEE 802.16 WiMAX mesh networks," Proceedings of IEEE Vehicular Technology Conference, pp.3102–3106, May 2005.
[27] A. Okabe, B. Boots, and K. Sugihara, Spatial tessellations: concepts and applications of Voronoi diagrams, 1992.
[28] M. Alicherry, R. Bhatia, and L.E. Li, "Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks," IEEE Journal on Selected Areas in Communication, vol.24, no.11, pp.1960–1971, Nov. 2006.
[29] W. Wang, Y. Wang, X. Li, W. Song, and O. Frieder, "Efficient interference-aware TDMA link scheduling for static wireless networks," Proceedings of the 12th annual international conference on Mobile computing and networking, pp.262–273, Sept. 2006.
[30] H. Zhu and K. Lu, "On the interference modeling issues for coordinated distributed scheduling in IEEE 802.16 mesh networks," Proceedings of Broadband Communications, Networks and Systems, pp.1–10, Oct. 2006.
[31] E.W. Dijkstra, "A note on two problems in connection with graphs," Numerische Mathematik, vol.1, no.6, pp.269–270, 1959.

Hiroshi Tokito Hiroshi Tokito received the M.E. degree from Osaka University, Osaka, Japan, in 2009. He will be assigned as a researcher of Mitsubishi Electric Corp., Tokyo, Japan. His research interests include ubiquitous networking.

Masahiro Sasabe Masahiro Sasabe received the M.E. and Ph.D. degrees from Osaka University, Osaka, Japan, in 2003 and 2006, respectively. He is currently an Assistant Professor with the Department of Information and Communication Technology, Osaka University. From 2004 to 2007, he was an Assistant Professor with the Cybermedia Center, Osaka University. His research interests include P2P/overlay networking and ubiquitous networking. He is a member of the IEEE and IEICE.

Go Hasegawa Go Hasegawa received the M.E. and D.E. degrees from Osaka University, Osaka, Japan, in 1997 and 2000, respectively. From 1997 to 2000, he was a Research Assistant of Graduate School of Economics, Osaka University. He is currently an Associate Professor of Cybermedia Center, Osaka University. His research work is in the area of transport architecture for future high-speed networks. He is a member of the IEEE and IEICE.

Hirotaka Nakano Hirotaka Nakano received the B.E., M.E. and D.E. degrees in Electrical Engineering from Tokyo University, Tokyo Japan, in 1972, 1974 and 1977, respectively. He joined NTT Laboratories in 1977 and has been engaged in research and development of videotex systems, and multimedia-on-demand systems. He had been an executive manager of the Multimedia Systems Laboratory of the NTT Human Interface Laboratories from 1995 to 1999. Afterwards, he served as the head in the Multimedia Laboratory of the NTT DOCOMO until 2004, and now he is an Professor of Cybermedia Center, Osaka University. His research work is in the area of ubiquitous networks. He is a member of the IEEE and IEICE, and the institute of Image Information and Television Engineers of Japan.