Compatibility Study for Optimal Tree-based Broadcast Routing

Chuan Han and Yaling Yang

Abstract—Broadcast routing is a critical component in the routing design. While there are plenty of routing metrics and broadcast routing schemes in current literature, it remains an unsolved problem as to which metrics are compatible to a specific broadcast routing scheme. In particular, in the wireless broadcast routing context where transmission has an inherent broadcast property, there is a potential danger of incompatible combination of broadcast routing algorithms and metrics. This paper shows that different broadcast routing algorithms have different requirements on the properties of broadcast routing metrics. The metric properties for broadcast routing algorithms in both undirected network topologies and directed network topologies are developed and proved. They are successfully used to verify the compatibility between broadcast routing metrics and broadcast routing algorithms. This work provides important criteria in broadcast routing metric design.

Index Terms—Broadcast routing, Routing protocols, Routing metric design.

1 Introduction

Broadcast routing is a critical component of network administration and management. It delivers and updates various network control information. Without effective broadcast routing, the whole network may degrade into chaos. Therefore, designing effective broadcast routing protocols is an important part in routing research and design.

The definition of "effective" broadcast routing, however, varies according to the application scenarios. In an energy-critical sensor network, broadcast routing may need to minimize energy consumption or maximize network lifetime. In an information-sensitive military network, broadcast routing may need to guarantee timely and secure message delivery. In many other application scenarios, multiple performance factors, such as delay, packet loss rate, bandwidth, etc., may need be jointly considered for effective broadcast routing [1]. All these different performance requirements for broadcast routing are usually reflected in the design of routing metrics [2], which guide the broadcast tree calculation algorithms to prefer one broadcast tree over the others.

However, many existing broadcast tree calculation algorithms [3], [4] are only known to find the optimal broadcast tree when the routing metric defines the weight of a broadcast tree as the simple sum of all the link weights. Many routing metrics [2] that accurately capture network performance requirements are not such simple link weight aggregation. Therefore, there arises the fundamental question of whether a broadcast tree calculation algorithm is guaranteed to find the optimal broadcast tree based on a certain routing metric definition. Some existing works [5], [6], [7] have shown that, for unicast routing, if metrics are inappropriately combined with incompatible path calculation algorithms, the optimality of unicast routing can be compromised. Similar compatibility issues also exist in broadcast routing as shown by a simple example as follows.

One important design goal of broadcast routing in sensor networks is to save transmission energy. A common metric capturing the total broadcast routing energy consumption [8] is

$$w(T) = \sum_{i \in N_t(T)} \max_{(i,j) \in E(T)} \varepsilon_{ij}, \quad (1)$$

where $N_t(T)$ is the set of transmitting nodes in the broadcast tree T, $E(T)$ is the set of links in T, and ε_{ij} is the minimum energy consumption of transmitting a packet from node i to node j.

While the metric definition in (1) is simple, finding the optimal broadcast tree based on this definition is very challenging. Consider the undirected network topology G consisting of four nodes r, i, j and k in Fig.1 (I), where the source node r of a broadcast session is marked as the black node, and the number associated with each edge is the ε_{ij} in (1). By definition (1), the minimum energy broadcast tree is $T_1 = \{ri, rj, rk\}$ in Fig.1 (II), whose total energy consumption is 3.5. However, following Prim's algorithm [3], which is a well-known algorithm for calculating the minimum spanning tree (MST) for undirected graphs, the broadcast tree is $T_2 = \{ri, ij, jk\}$ in Fig.1 (III), whose total energy consumption is 4.1. Hence, the combination of Prim's algorithm and the broadcast routing metric (1) fails to find the optimal broadcast routing tree.

The above example demonstrates the strong impact of routing metrics on the optimality of broadcast routing.

- *Chuan Han and Yaling Yang are with the ECE department, Virginia Tech, Blacksburg, VA, 24061.*
E-mail: hanc@vt.edu

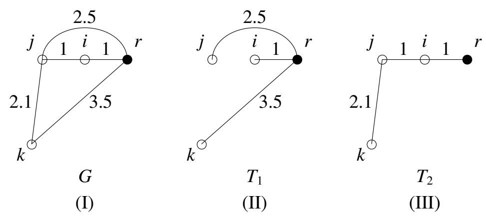


Fig. 1. The network topology example

Unfortunately, in current literature, there is a lack of systematic understanding about this incompatibility problem. Hence, it is very easy for network engineers to combine incompatible routing metrics with optimal broadcast algorithms, causing unexpected problems later.

In this paper, we fill in this critical technical void. Our work provides important guidelines to network designers so that the incompatibility issue between routing metrics and optimal routing protocols can be resolved at design stage. Our unique contribution is two-fold. Firstly, we propose a novel broadcast routing algebra that is a framework used to investigate the broadcast routing compatibility problem. Our algebra is different from Sobrinho's routing algebra [5]. Sobrinho's routing algebra can only operate over paths and hence can only capture unicast routing. Our broadcast routing algebra, on the other hand, can operate over arbitrary graphs and capture broadcast routing. Our routing algebra is also highly flexible. It not only can capture traditional routing metrics that define broadcast tree weight as linear link weight aggregation, but also can capture more complex non-linear routing metrics. Secondly, using the broadcast routing algebra, we identify the necessary and sufficient routing metric properties for optimal broadcast tree (OBT) calculation algorithms, including Prim's algorithm, the generic edge-adding algorithm, the generic edge-deleting algorithm and Edmonds' algorithm. These properties are used to judge the compatibility between these algorithms and plausible broadcast routing metrics.

The notation used in this paper is summarized in Table 1. The remaining part of this paper is organized as follows. In Section 2, the broadcast routing algebra is introduced. Sections 3 and 4 provide the necessary and sufficient metric properties required by broadcast routing for undirected network topologies and directed network topologies, respectively. A distributed optimal broadcast routing protocol is proposed in Section 5, and the root-independence properties of OBTs are discussed in Section 6. The applications of the derived broadcast routing metric properties are presented in Section 7. Finally, Section 8 concludes the whole paper.

2 Broadcast Routing Algebra

The analysis of the routing metric properties is based on our broadcast routing algebra that formulates the routing problem in an algebraic manner. This section introduces the broadcast routing algebra.

TABLE 1
Notation

Symbol	Definition
$=$	Equivalent to
$A^-(n)$	The set of arcs that terminate at node n
$E(G)$	The edge set of graph G
$G=(N(G),E(G))$	The graph whose node set is $N(G)$ and edge set is $E(G)$
$G_1 \ominus G_2$	Remove the common edges of network topologies G_1 and G_2 from G_1
$G_1 \oplus G_2$	Merge of two network topologies G_1 and G_2
$N(G)$	The node set of graph G
$\partial(S)$	The edge cut of node set S
$\prec$	Lighter than
$\preceq$	Lighter than or equivalent to
$\succ$	Heavier than
a_{ij}	The arc emanating from node i and terminating at node j
$d^-(n)$	The indegree of node n
e_{ij}	The edge whose two end nodes are i and j
r	The root node of graph
$w(T)$	The weight of network topology T
$\|S\|$	The total number of elements in the set S

2.1 Definition

Definition 1: The broadcast routing algebra is defined as

$$A=(\Sigma,\preceq,\oplus,\ominus,w(\cdot)), \tag{2}$$

where

- Σ is the set of signatures describing the characteristics of all the subgraphs of an original graph G. These characteristics may include each link's capacity, energy consumption, etc..
- The symbol $\preceq$ is the preference order operator, where $w(T_1) \preceq w(T_2)$ indicates that topology T_1 is better than or equivalent to topology T_2 under weight function $w(\cdot)$.
- The symbol $\oplus$ denotes the operator that joins two network topologies. For any edge $e_1 \in E(G_1)$ and node $n_1 \in N(G_1)$ in topology G_1, and any edge $e_2 \in E(G_2)$ and node $n_2 \in N(G_2)$ in topology G_2, we have e_1, e_2 in the edge set $E(G_1 \oplus G_2)$ and n_1, n_2 in the node set $N(G_1 \oplus G_2)$.
- The symbol $\ominus$ denotes the operator that removes the common edges of the left and the right operand topologies from the left operand topology. For any edge $e_1 \in E(G_1), e_1 \notin E(G_2)$, and node $n_1 \in N(G_1)$ in topology G_1, we have e_1 in the edge set $E(G_1 \ominus G_2)$ and n_1 in the node set $N(G_1 \ominus G_2)$.
- The symbol $w(\cdot)$ denotes the weight function over the signature of network topologies. With this routing algebra, any routing metric is mathematically represented by the weight function $w(\cdot)$.

We further define $w(a) \prec w(b)$ as $w(a) \preceq w(b)$ and $w(a) \neq w(b)$, and let $w(a) \succ w(b)$ mean $w(b) \prec w(a)$. Any OBT algorithm essentially builds the best subgraph using the $\oplus$ and/or $\ominus$ operator. With the broadcast

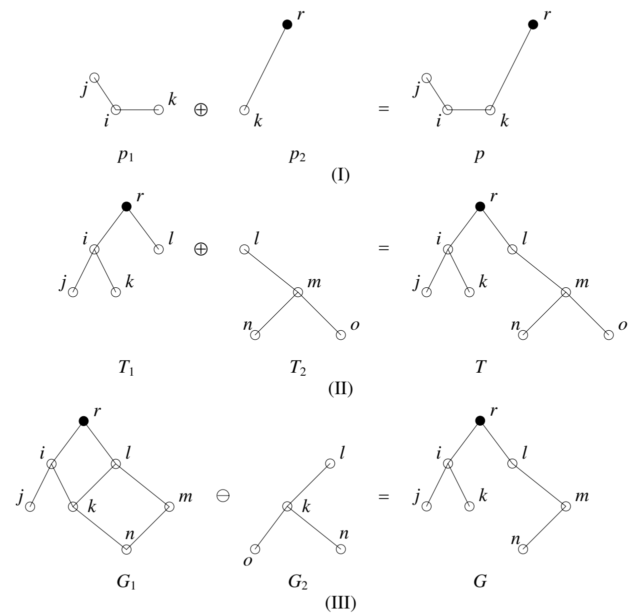


Fig. 2. The broadcast routing algebra example

routing algebra, OBT algorithms can be expressed in an algebraic manner.

It is important to point out that, our broadcast routing algebra is different from Sobrinho's routing algebra. In Sobrinho's routing algebra, the $\oplus$ operation can only capture simple path concatenation as depicted in Fig.2 (I). In our broadcast routing algebra, the $\oplus$ operation is the more general graph union operation as depicted in Fig.2 (II), where given two broadcast trees T_1 and T_2 that share a common node l, the $\oplus$ operation joins the two trees and the resulting tree is $T = T_1 \oplus T_2$. Moreover, the broadcast routing algebra also has an unique $\ominus$ operation as shown by the example in Fig.2 (III), where $\ominus$ removes from its left operand topology the common edges between its left and right operand topologies.

2.2 Properties

The broadcast routing algebra has the following properties:

Σ is closed under $\oplus$: $a \oplus b \in \Sigma$ for any $a, b \in \Sigma$;

Σ is closed under $\ominus$: $a \ominus b \in \Sigma$ for any $a, b \in \Sigma$;

$\preceq$ is complete: for any $a, b \in \Sigma$, either $w(a) \preceq w(b)$ or $w(b) \preceq w(a)$ (or both);

$\preceq$ is transitive: for $a, b, c \in \Sigma$, if $w(a) \preceq w(b), w(b) \preceq w(c)$, then $w(a) \preceq w(c)$;

$\oplus$ is idempotent: $a \oplus a = a$ for any $a \in \Sigma$;

$\oplus$ is commutative: $a \oplus b = b \oplus a$ for any $a, b \in \Sigma$;

$\oplus$ is associative: $(a \oplus b) \oplus c = a \oplus (b \oplus c)$ for any $a, b, c \in \Sigma$;

$\ominus$ is non-commutative: $a \ominus b \neq b \ominus a$ for some $a, b \in \Sigma$;

$\ominus$ is only left-associative but non-associative: $a \ominus b \ominus c = (a \ominus b) \ominus c$ for any $a, b, c \in \Sigma$, but $(a \ominus b) \ominus c \neq a \ominus (b \ominus c)$ for some $a, b, c \in \Sigma$.

3 Optimal Broadcast Routing in Undirected Network Topologies

In our analysis of OBT algorithms' requirements on routing metrics, there are two different models of the underlying networks: the directed graph model and the undirected graph model. The undirected graph model is appropriate if all the links in the network are bidirectional and the two directions have the same signatures (a.k.a. characteristics). The directed graph model is used to capture more complicated cases where there are asymmetric links. Different OBT algorithms need to be used for undirected and directed network topologies, and these algorithms have different requirements on routing metric design. In this section, we focus on OBT algorithms for undirected network topologies. In the next section, we study OBT algorithms for directed network topologies.

Without loss of generality, we formulate the problem of optimal broadcast routing in undirected network topologies as the MST problem for undirected graphs. In the remainder of this section, we develop and prove the necessary and sufficient metric properties for which Prim's algorithm, the generic edge-adding algorithm and the generic edge-deleting algorithm guarantee optimality. Prim's algorithm is based on subtrees, the generic edge-adding algorithm is based on subforests, and the generic edge-deleting algorithm is based on connected subgraphs. Many well-known MST algorithms, such as Kruskal's algorithm [3], Boruvka's algorithm [4], the GHS algorithm [9] and the reverse-delete algorithm [10], etc., are special cases of the two generic algorithms.

3.1 Prim's Algorithm

3.1.1 Algorithm Overview

Prim's algorithm starts by treating the root node r, which is the broadcast source node, as the initial partial spanning tree T_r. Then it progressively grows the partial spanning tree T_r by adding the best edge from the edge cut of the current T_r, until T_r spans the entire graph. The calculation of the best edge e^* can be generalized based on the binary operation $\oplus$ and the order relation $\preceq$ as follows:

$$e^* = \arg\min_{e \in \partial(T_r)} \{w(T_r \oplus e)\}, \tag{3}$$

where $\partial(T_r)$ is the edge cut of T_r [11]. Here, the edge cut $\partial(T_r)$ is the set of edges with one end in $N(T_r)$ and the other end in $N(G) - N(T_r)$. Note that the original Prim's algorithm is only defined for linear metrics based on linear link weight aggregation. The generalized computation of minimum weight edge in (3) extends Prim's algorithm to cover both linear and non-linear metric designs.

3.1.2 Metric Properties and Proof

The required routing metric property for the generalized Prim's algorithm can be expressed by the following definition.

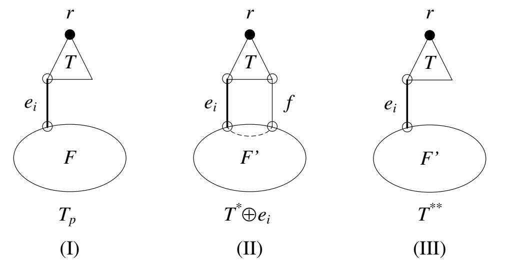


Fig. 3. Sufficient proof illustration for Prim's algorithm

Definition 2: Right $\oplus$-isotonicity for trees: A weight function $w(\cdot)$ of trees is said to be isotonic for $\oplus$ over a graph G if for any tree $T \subset G$, we have

$$w(T \oplus e) \preceq w(T \oplus e') \Rightarrow w(T \oplus e \oplus F) \preceq w(T \oplus e' \oplus F), \quad (4)$$

for any edge $e, e' \in \partial(T)$ and forest F such that both $T \oplus e \oplus F$ and $T \oplus e' \oplus F$ are still trees.

Theorem 1: Given any connected and undirected network topology G whose root node is r, Prim's algorithm produces the MST, if and only if the broadcast routing metric $w(\cdot)$ is right $\oplus$-isotonic for trees.

Proof:

Sufficient condition:

Let T_p be the Prim tree that is generated by Prim's algorithm. Denote $e_1, \ \ldots, \ e_{|N(G)|-1}$ as the order in which Prim's algorithm selects edges, where $|N(G)|$ is the total number of nodes in graph G. We next prove that if the broadcast routing metric satisfies the property in (4), Prim's algorithm produces a MST.

Following the order $e_1, \ \ldots, \ e_{|N(G)|-1}$, compare each edge in $E(T_p)$ with edges of a MST. If all the edges in $E(T_p)$ are the same as the edges in the MST, then Prim's algorithm produces the MST for the given metric. Otherwise, T_p starts to differ with the MST at some edge. Denote the MST that shares the largest number of consecutive common edges with T_p as T^* and the first edge that T_p differs with T^* as e_i. By Prim's algorithm, the edge set $\{e_1, \ e_2, \ \ldots, \ e_{i-1}\}$ is a tree. Denote this tree as T as shown in Fig. 3 (I). Tree T is a subgraph for both T_p and T^*. Consider adding edge e_i to T^* as shown in Fig. 3 (II). Then, there must exist a cycle containing e_i, and within the cycle there exists an edge $f \in \partial(T), \ f \in T^*, \ f \neq e_i$. By Prim's algorithm, it follows $w(T \oplus e_i) \preceq w(T \oplus f)$. Substituting f with e_i, the resulting subgraph $T^{**} = T^* \oplus e_i \ominus f$ in Fig. 3 (III) is still a spanning tree. Since $w(T \oplus e_i) \preceq w(T \oplus f)$, by the property in (4), it follows that $w(T^{**}) \preceq w(T^*)$. Since T^* is a MST, T^{**} is also a MST. The fact that T^{**} has one more common edge e_i with T_p than T^* contradicts the definition of T^*. Hence, T_p is also a MST.

Necessary condition:

We need to prove that if Prim's algorithm produces the MST on any network topology, then the metric satisfies the property in (4). This can be proved by showing that its contrapositive is correct, i.e., if a metric does not satisfy the property in (4), then there is at least one network topology for which the algorithm does not guarantee optimality.

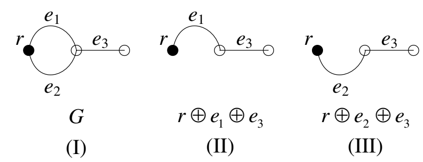


Fig. 4. The necessary condition proof example for Prim's algorithm

Consider the network topology G in Fig. 4 (I). Suppose $w(r \oplus e_1) \preceq w(r \oplus e_2)$ and the metric does not satisfy the property in (4). Note that $r \oplus e_1 \oplus e_3$ and $r \oplus e_2 \oplus e_3$ are trees, as shown in Fig. 4 (II) and (III), respectively. Since for the given metric the property in (4) does not hold, it is possible that $w(r \oplus e_1 \oplus e_3) \succ w(r \oplus e_2 \oplus e_3)$. Meanwhile, the tree produced by Prim's algorithm is $T_p = r \oplus e_1 \oplus e_3$ which is not the MST. Hence, for the given metric, Prim's algorithm does not produce the MST for this particular network topology. Therefore, for Prim's algorithm to guarantee optimality on any network topology, the metric must satisfy the property in (4). □

3.2 Generic Edge-adding Algorithm

There is a generic edge-adding algorithm on undirected graphs [3], [4]. It covers typical forest-based MST algorithms, e.g., Kruskal's algorithm, Boruvka's algorithm, and the GHS algorithm, etc.. These algorithms only differ in their edge-adding orders.

3.2.1 Algorithm Overview

The generic edge-adding algorithm starts by initializing forest F as $F = F_0 = (N(G), \emptyset)$, where $\emptyset$ is the empty set. In each of the following steps, it picks one edge e^* to add to F until F is a spanning tree of G. The edge e^* is determined by first finding a node set S such that F does not have any edge that belongs to the edge cut of S. Here, the edge cut of S is the set of edges in G with one end in S and the other end in $N(G) - S$. The edge e^* is then chosen from the edge cut of S [11], denoted as $\partial(S)$, as follows:

$$e^* = \arg\min_{e \in \partial(S)} \{w(F \oplus e)\}, \quad (5)$$

where $\partial(S) \cap E(F) = \emptyset$ and $S \subset N(G)$.

The edge-adding process can be illustrated by an example. Consider the undirected graph G in Fig.5 (I) and the initial forest $F_0 = (N(G), \emptyset)$. In the first step shown in Fig.5 (II), e_1 is chosen from $\partial(S_1)$, i.e., the edge cut of the node set S_1. This results in a forest $F = F_0 \oplus e_1$. Note that in the next step, the node set S_2 shown in Fig.5 (III) cannot be chosen since $\partial(S_2) \cap E(F_0 \oplus e_1) = \{e_1\} \neq \emptyset$. Instead, one can choose the node set S_3 within $\partial(S_3)$ edge e_2 as shown in Fig.5 (IV). This process continues until the resulting forest is a spanning tree of G.

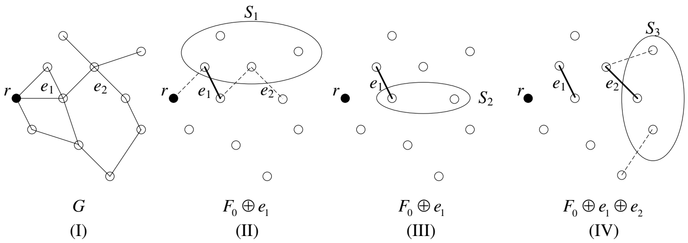


Fig. 5. The generic edge-adding algorithm example

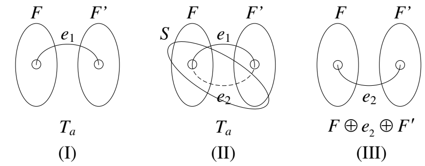


Fig. 6. The necessary condition proof example for the generic edge-adding algorithm

3.2.2 Metric Properties and Proof

The required routing metric property for the generic edge-adding algorithm can be expressed by the following definition.

Definition 3: Right $\oplus$-isotonicity for forests: A weight function $w(\cdot)$ of forests is said to be $\oplus$-isotonic for forests over a graph G if for any forest $F \subset G$, given any node set $S \subset N(G)$ satisfying $\partial(S) \cap E(F) = \emptyset$, we have

$$w(F \oplus e) \preceq w(F \oplus e') \Rightarrow \\ w(F \oplus e \oplus F') \preceq w(F \oplus e' \oplus F'), \quad (6)$$

for any edge $e,\ e' \in \partial(S)$ and forest $F' \subset G$ such that $F \oplus e \oplus F'$ and $F \oplus e' \oplus F'$ are forests.

Theorem 2: Given any connected and undirected network topology G whose root node is r, the generic edge-adding algorithm produces the MST, if and only if the broadcast routing metric $w(\cdot)$ is right $\oplus$-isotonic for forests.

Proof:

Sufficient condition:

Let T_a be the spanning tree generated by the generic edge-adding algorithm. Denote $e_1, e_2, ..., e_{|N(G)|-1}$ as the order of adding edges, where $|N(G)|$ is the total number of nodes in the graph G. Suppose T_a is not a MST. Denote T^* as the MST that shares the largest number of consecutive common edges with the edge sequence $e_1, e_2, ..., e_{|N(G)|-1}$ of T_a. Denote the first edge in the sequence that differs from T^* as e_i. Let F be the forest consisting of edges $e_1, e_2, ..., e_{i-1}$, i.e., $F = \{e_1, e_2, ..., e_{i-1}\}$. Consider adding edge e_i into the spanning tree T^*. Then, there must exist a cycle containing e_i. By the generic edge-adding algorithm and the tree property of T^*, there must exist an edge $e'_i \in E(T^*)$ in the cycle, such that $w(F \oplus e_i) \preceq w(F \oplus e'_i), e_i, e'_i \in \partial(S), \partial(S) \cap E(F) = \emptyset, S \subset N(G)$, where S is the selected node set before adding edge e_i when running the algorithm. Deleting e'_i from $T^* \oplus e_i$ generates another spanning tree. By the property in (6), it follows that $w(T^* \oplus e_i \ominus e'_i) = w(F \oplus e_i \oplus (T^* \ominus F \ominus e'_i)) \preceq w(F \oplus e'_i \oplus (T^* \ominus F \ominus e'_i)) = w(T^* \oplus e'_i \ominus e'_i) = w(T^*)$. Since T^* is a MST, $T^* \oplus e_i \ominus e'_i$ is also a MST and it contains edge e_i. Since $T^* \oplus e_i \ominus e'_i$ contains more consecutive common edges with T_a than T^*, this contradicts the definition of T^*. Hence, T_a is also a MST.

Necessary condition:

We need to prove that if for any topology the spanning tree T_a generated by the generic edge-adding algorithm is a MST then the metric satisfies the property in (6). We prove it by showing that its contrapositive is correct, i.e., if there is at least one metric that does not satisfy the property in (6), there is at least one network topology for which T_a is not a MST.

Suppose, for a network topology, T_a is the spanning tree generated by the generic edge-adding algorithm, as shown in Fig.6 (I). By the generic edge-adding algorithm, it follows that $w(F \oplus e_1) \preceq w(F \oplus e_2), e_1, e_2 \in \partial(S), \partial(S) \cap E(F) = \emptyset$, where F is the partially generated forest before adding edge e_1 and S is the selected node set before adding edge e_1. Let $F' = T_a \ominus e_1 \ominus F$ as shown in Fig.6 (II). Since for the given metric the property in (6) is not guaranteed, it is possible that $w(T_a) = w(F \oplus e_1 \oplus F') \succ w(F \oplus e_2 \oplus F')$ as shown in Fig. 6 (III). That is T_a cannot be the MST of the original network topology. Hence, if T_a is a MST, then the metric satisfies the property in (6). □

Note that the above proof can be applied to any specific order of adding edges. Therefore, Theorem 2 also holds for typical edge-adding algorithms, e.g., Kruskal's algorithm, Boruvka's algorithm and the GHS algorithm.

3.3 Generic Edge-deleting Algorithm

Similar to the generic edge-adding algorithm, there is a generic edge-deleting algorithm on undirected graphs. Common edge-deleting algorithms, e.g., the reverse-deleting algorithm, are only special cases of the generic edge-deleting algorithm with particular edge deleting orders.

3.3.1 Algorithm Overview

The generic edge-deleting algorithm starts by initializing a connected graph F as $F = G$. In each of the following steps, it deletes one edge e^* from F, while still maintaining the connectivity of F. The edge deleting process produces a tree when there is no edge to be deleted in F any more. At each step, the algorithm first finds a cycle C in the current connected graph F. Then, the edge e^* is selected as follows:

$$e^* = \arg \min_{e \in E(C)} \{w(F \ominus e)\}, \quad (7)$$

where $C \subseteq F$.

The edge-deleting process can be illustrated by an example. Consider the undirected graph G in Fig.7 (I). In the first step, as shown in Fig.7 (II), edge e_1 within the cycle C_1 in G is chosen and deleted. In the second

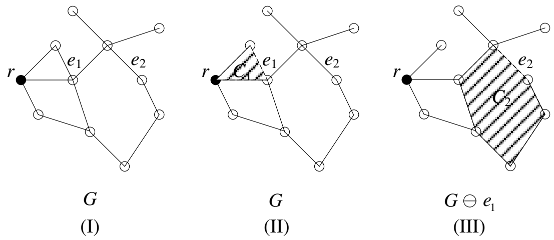


Fig. 7. The generic edge-deleting algorithm example

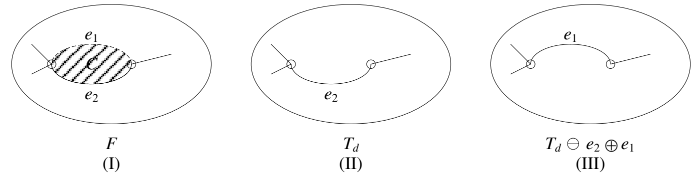


Fig. 8. The necessary condition proof example for the generic edge-deleting algorithm

step, as shown in Fig.7 (III), edge e_2 within the cycle C_2 in $G \ominus e_1$ is chosen and deleted. This process continues until there are no cycles in the graph.

3.3.2 Metric Properties and Proof

The required routing metric property for the generic edge-deleting algorithm can be expressed by the following definition.

Definition 4: Right $\ominus$-isotonicity for connected graphs: A weight function $w(\cdot)$ of connected graphs is said to be right $\ominus$-isotonic for connected subgraphs over a graph G if for any connected subgraph $F \subset G$, given any cycle C in F, we have

$$w(F \ominus e) \preceq w(F \ominus e') \Rightarrow \\ w(F \ominus e \ominus F') \preceq w(F \ominus e' \ominus F'), \quad (8)$$

for any edge $e, e' \in E(C)$ and subgraph F' in G such that both $F \ominus e \ominus F'$ and $F \ominus e' \ominus F'$ are connected subgraphs.

Theorem 3: Given any connected and undirected network topology G whose root node is r, the generic edge-deleting algorithm produces the MST, if and only if the broadcast routing metric $w(\cdot)$ is right $\ominus$-isotonic for connected subgraphs.

Proof:

Sufficient condition:

Let the spanning tree generated by the generic edge-deleting algorithm be T_d. Denote $e_1, e_2, ..., e_{|E(G)|-|N(G)|+1}$ as the order of deleting edges, where $|E(G)|$ is the total number of edges in the graph G and $|N(G)|$ is the total number of nodes in the graph G. Suppose T_d is not a MST. Denote T^* as the MST such that $E(G) - E(T^*)$ shares the largest number of consecutive common edges with the sequence $e_1, e_2, ..., e_{|E(G)|-|N(G)|+1}$. Denote the first edge that differs be e_i, and let F be the subgraph generated by deleting $e_1, e_2, ..., e_{i-1}$ from the graph G. Note that $e_i \in E(T^*)$. By the generic edge-deleting algorithm and the tree property of T^*, there must exist an edge $e'_i \notin E(T^*)$ such that $w(F \ominus e_i) \preceq w(F \ominus e'_i), e_i, e'_i \in E(C), C \subseteq F$, where C is the selected cycle in F before deleting e_i when running the algorithm. By deleting e_i from T^* and adding e'_i to $T^* \ominus e_i$, we get another spanning tree $T^* \ominus e_i \oplus e'_i$. By the property in (8), it follows that $w(T^* \ominus e_i \oplus e'_i) = w(F \ominus e_i \ominus (F \ominus T^* \ominus e'_i)) \preceq w(F \ominus e'_i \ominus (F \ominus T^* \ominus e'_i)) = w(T^*)$. Since T^* is a MST, $T^* \ominus e_i \oplus e'_i$ is also a MST and it does not contain edge e_i. This contradicts the definition of T^*. Hence, T_d is also a MST.

Necessary condition:

We need to prove that if any spanning tree T_d generated by the generic edge-deleting algorithm is a MST, then the metric satisfies the property in (8). We prove it by showing that its contrapositive is correct, i.e., if a metric does not satisfy the property in (8), there is at least one network topology for which T_d is not a MST.

Suppose, for a network topology, F is the partially generated subgraph by the generic edge-deleting algorithm as shown in Fig.8 (I). Let e_1 be the edge to be removed by the generic edge-deleting algorithm, and T_d be the spanning tree finally generated by the generic edge-deleting algorithm. Then, it follows that $w(F \ominus e_1) \preceq w(F \ominus e_2), e_1, e_2 \in E(C)$, where $C \subset F$. Since for the given metric the property in (8) is not guaranteed, it is possible that $w(T_d) = w(F \ominus e_1 \ominus (F \ominus T_d \ominus e_1)) \succ w(F \ominus e_2 \ominus (F \ominus T_d \ominus e_1)) = w(T_d \ominus e_2 \oplus e_1)$ as shown in Fig.8 (II) and (III). That is T_d cannot be a MST. Hence, if T_d is a MST, then the metric satisfies the property in (8). □

Again, note that the above proof can be applied to any specific order of deleting edges. Therefore, Theorem 3 holds for all edge-deleting algorithms, e.g., the reverse-delete algorithm.

4 Optimal Broadcast Routing in Directed Network Topologies

In a directed network topology, the link from node n_1 to node n_2 may have different characteristics comparing to the link from node n_2 to node n_1. The problem of optimal broadcast routing in directed network topologies can be formulated as the minimum weight spanning r-arborescence problem for directed graphs. In graph theoretic terminology, we define spanning r-arborescence and partition subgraph of a directed graph as follows.

Definition 5: Given a connected and directed graph D whose root node is r, a spanning r-arborescence is a directed subgraph of D such that there exists exactly one directed path from the root node r to any other non-root node.

Definition 6: Given a directed graph D whose root node is r, a partition subgraph is a subgraph D' of D such that the indegree of the root node r is equal to 0 and the indegrees of other nodes are less than or equal to 1, i.e., $d^-(r) = 0, d^-(n) \leq 1, \forall\, n \in N(D'), n \neq r$.

The most well-known algorithm for solving the minimum weight spanning r-arborescence problem is Edmonds' algorithm [12]. In this section, for directed network topologies, the necessary and sufficient metric property of the generalized Edmonds' algorithm is developed and proved.

4.1 Edmonds' Algorithm

4.1.1 *Overview of Edmonds' Algorithm*

The core idea of Edmonds' algorithm is summarized as follows. Given a directed graph D, whose root node is r, remove all the inbound arcs of root node r, and denote the resultant subgraph as D_0. Edmonds' algorithm proceeds to build the minimum r-arborescence spanning D_0 following a three-phase procedure.

Initialization: Send D_0 to Phase I as Phase I's input.

Phase I: Given the input graph D_i for the i_{th} iteration of Phase I, create a directed graph T_i that includes all nodes in $N(D_i)$ and no arcs between nodes. Each node in T_i, hence, is a separate arborescence. For each non-root node n whose indegree is 0, a new arc $a^* \in E(D_0)$ is selected to be included in T_i. The new arc a^* is selected as follows:

$$a^* = \arg \min_{a \in \mathcal{A}(n)} \{w(a)\}, \qquad (9)$$

where $\mathcal{A}(n) = \{a | a \in A^-(n), n \in N(D_i), n \neq r,\ d^-(n) = 0\}$, $A^-(n)$ is the set of arcs that terminate at node n, and $d^-(n)$ is the indegree of node n in T_i.

The above arc adding process ends when $d^-(n) = 1$ for any $n \in N(T_i), n \neq r$, i.e, the indegree of any non-root node in T_i is 1. If there is no circuit in T_i after the arc adding process, then go to phase III with T_i as its input. If there are circuits in T_i, then go to phase II and let T_i and D_i be its input.

Phase II: Given the input graph T_i and D_i, pick a circuit C_i in T_i to eliminate as follows. Find a pair of arcs $(a^*_{c'c}, a^*_{nc})$ that can be used to break C_i by replacing $a^*_{c'c}$ with a^*_{nc}, where $a^*_{c'c} \in C_i$, $a^*_{nc} \in D_i$ and $a^*_{nc} \notin T_i$. Both $a^*_{c'c}$ and a^*_{nc} are inbound arcs to node $c \in N(C_i)$. Arcs $a^*_{c'c}$ and a^*_{nc} are selected based on the following equation:

$$(a^*_{c'c}, a^*_{nc}) = \arg \min_{\substack{a_{nc} \notin T_i \\ a_{c'c} \in C_i}} \{w(C_i \ominus a_{c'c} \oplus a_{nc})\}, \qquad (10)$$

where $a_{c'c}$ is any arc in C_i, $C_i \ominus a_{c'c}$ is the path generated by deleting arc $a_{c'c}$ from C_i, and a_{nc} is an inbound arc to node c. Denote P^*_i as the path generated by breaking the circuit C_i, i.e., $P^*_i = C_i \ominus a^*_{c'c} \oplus a^*_{nc}$.

This circuit breaking operation can be illustrated by an example. Consider the circuit C_i in Fig.9 (I), where the circuit is the solid part of the graph. The circuit C_i is broken by replacing $a_{c'c}$ by a_{nc} as shown in Fig.9 (II), where the resultant arborescence is the solid part of the graph. The optimal circuit-breaking scheme is shown in Fig.9 (III), where the resultant arborescence P^*_i is the solid-line part of the graph.

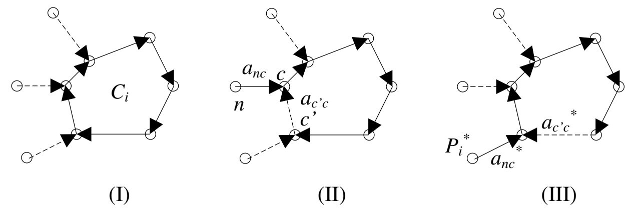


Fig. 9. Illustration for Edmonds' algorithm

With $(a^*_{c'c}, a^*_{nc})$ identified, shrink the circuit C_i in D_i to a pseudo-node n_i. Replace $N(C_i)$ in D_i by n_i. Consider the end nodes of the outbound arcs going out of nodes in C_i. Their signatures remain unchanged. Set the inbound arc of n_i as a^*_{nc} and modify its signature to be the signature of P^*_i, i.e., the signature of the path used to replace the original circuit. Denote this newly generated graph through the shrinking operation as D_{i+1}. Go to the beginning of Phase I and let D_{i+1} be Phase I's input.

Phase III: Given the input T_l after l iterations of phase I and II, expand the pseudo-node n_i to P_i in the reverse order ($i = l, l-1, \ldots, 1$) of their generation sequence in phase II. The resulting subgraph T_e is the minimum weight spanning r-arborescence of D_0.

Note that the original Edmonds' algorithm is only defined for linear link metric aggregation. The algorithm discussed in this section is extended to cover both linear and nonlinear link metric aggregation.

4.1.2 *Metric Properties and Proof*

The required routing metric property for Edmonds' algorithm can be expressed by the following definition.

Definition 7: Right $\oplus$-isotonicity for partition subgraphs: A weight function $w(\cdot)$ of partition subgraphs is said to be right $\oplus$-isotonic for partition subgraphs over a directed graph D if for any partition subgraphs D_1, D_2 of D, we have

$$w(D_1) \preceq w(D_2) \Rightarrow w(D_1 \oplus D_3) \preceq w(D_2 \oplus D_3), \qquad (11)$$

where D_3 is also a partition subgraph of D such that $D_1 \oplus D_3, D_2 \oplus D_3$ are still partition subgraphs of D.

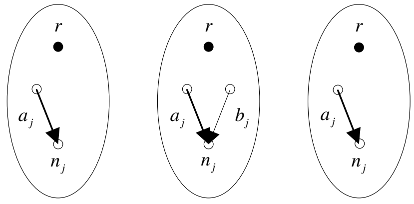
Theorem 4: Given any connected and directed network topology D whose root node is r, Edmonds' algorithm produces the minimum weight r-arborescence spanning D, if and only if the broadcast routing metric is right $\oplus$-isotonic for partition subgraphs.

Proof:

Sufficient condition:

Consider a connected and directed network topology D whose root node is r. We are going to prove that if the broadcast routing metric satisfies the property in (11), Edmonds' algorithm produces the minimum weight r-arborescence spanning D.

First, we prove that if there is no circuit generated after the arc adding procedure in phase I, phase I produces the minimum weight spanning r-arborescence T_i of its input graph D_i. Let T^* be a minimum weight spanning r-arborescence and T_e be the arborescence generated



T_e $T^* \oplus a_j$ $T^* \ominus b_j \oplus a_j$

(I) (II) (III)

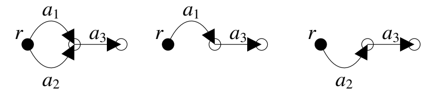
Fig. 10. Sufficient proof illustration for Edmonds' algorithm

by phase I of Edmonds' algorithm. Assume $T_e \neq T^*$. There must exist some nodes $n_i, i = 1, \ldots, m$ whose inbound arc a_i in T_e is different from the inbound arc b_i in T^* as shown in Fig. 10. By Edmonds' algorithm, $w(a_i) \preceq w(b_i)$. Starting from T^*, replace b_1 in T^* by the corresponding a_1 in T_e. Denote the generated subgraph as T_1^*. By the property in (11), it follows that $w(T_1^*) = w(T^* \ominus b_1 \oplus a_1) \preceq w(T^* \ominus b_1 \oplus b_1) = w(T^*)$. Similarly, replacing b_2 by a_2 in T_1^*, we get T_2^* that satisfies $w(T_2^*) \preceq w(T_1^*)$. By continuing this arc replacing process, we get a series of graphs $T_1^*, T_2^*, \cdots T_m^* = T_e$ that satisfy $w(T_{i+1}^*) \preceq w(T_i^*), i = 1, \ldots, m-1$. Hence, it follows that $w(T_e) \preceq w(T^*)$. Since T^* is a minimum weight spanning r-arborescence, $w(T_e) = w(T^*)$. T_e is also a minimum weight spanning r-arborescence.

Denote T_i^e as the subgraph generated after expanding the pseudo node n_i to P_i^* in phase III. We next prove that T_i^e is the minimum weight spanning r-arborescence of D_i through induction hypothesis. First, for the base case, note that T_l is the minimum weight spanning r-arborescence of D_l and our hypothesis is satisfied. Then, assume that T_{i+1}^e is the minimum weight spanning r-arborescence of D_{i+1}. We next show that T_i^e is the minimum weight spanning r-arborescence of D_i.

Let T_i^0 be an arbitrary spanning arborescence of D_i. By the fact that T_i^0 is a spanning arborescence, there must exist an arc a_{nc} that emanates from some node $n \in N(D_i) - N(C_i)$ and terminates at some node $c \in N(C_i)$. Let $P_i^n = C_i \ominus a_{c'c} \oplus a_{nc}$ be the subgraph generated by replacing arc $a_{c'c}$ by arc a_{nc}, where $a_{c'c} \in C_i$. Denote the subgraph consisting of the inbound arcs of $N(C_i)$ in T_i^0 as F_i. If $P_i^n = F_i$, then $w(P_i^n) = w(F_i)$. If $P_i^n \neq F_i$, there must exist some nodes in $N(C_i)$ whose inbound arc a_j in F_i is different from the inbound arc b_j in P_i^n, $j = 1, 2, ..., t$. By Edmonds' algorithm, $w(b_j) \preceq w(a_j)$. Starting from F_i, replace a_1 in F_i by the corresponding b_1 in P_i^n. Let the generated subgraph be F_i^1. By the property in (11), it follows that $w(F_i^1) = w(F_i \ominus a_1 \oplus b_1) \preceq w(F_i \ominus a_1 \oplus a_1) = w(F_i)$. Similarly, replacing a_2 by b_2 in F_i^1, we get F_i^2 that satisfies $w(F_i^2) \preceq w(F_i^1)$. By continuing this arc replacing process, we get a series of graphs $F_i^1, F_i^2, \cdots F_i^t = P_i^n$ that satisfy $w(F_i^{j+1}) \preceq w(F_i^j), j = 1, \ldots, t-1$. Hence, it follows that $w(P_i^n) \preceq w(F_i)$.

Let $T_i^1 = T_i^0 \ominus F_i \oplus P_i^n$ be the subgraph generated by replacing F_i in T_i^0 by P_i^n. Since $w(P_i^n) \preceq w(F_i)$, it follows



(I) (II) (III)

Fig. 11. The necessary condition proof example for Edmonds's algorithm

that

$$w(T_i^1) = w(T_i^0 \ominus F_i \oplus P_i^n) \preceq w(T_i^0 \ominus F_i \oplus F_i) = w(T_i^0). \quad (12)$$

Shrink P_i^n in T_i^1 into n_i to get a new graph T_i^2. By the fact that T_i^0 is a r-arborescence of D_i, T_i^2 is still a r-arborescence of D_{i+1}. Since T_{i+1}^e is the minimum weight spanning r-arborescence of D_{i+1}, we have

$$w(T_{i+1}^e) \preceq w(T_i^2). \quad (13)$$

By Edmonds' algorithm, the shrink and expansion operations do not change the weight of the signature of a graph, i.e., $w(T_i^e) = w(T_{i+1}^e)$ and $w(T_i^2) = w(T_i^1)$. Hence, from (12) and (13), we get

$$w(T_i^e) \preceq w(T_i^0). \quad (14)$$

Since T_i^0 is an arbitrary spanning arborescence of D_i, T_i^e is the minimum weight spanning r-arborescence of D_i. Hence, by induction hypothesis, T_i^e is the minimum weight spanning r-arborescence of D_i. Hence, after expanding all the l circuits, Edmonds's algorithm produces the minimum weight r-arborescence of D_0.

Necessary condition:

Given any connected and directed network topology D, whose root node is r, we next prove that if Edmonds' algorithm produces the minimum weight r-arborescence spanning D, then the broadcast routing metric satisfies the properties in (11). It can be proved by showing its contrapositive is correct, i.e., if a broadcast routing metric does not satisfy the properties in (11), then there exists at least one network topology where Edmonds' algorithm does not guarantee optimality.

Consider the directed network topology D in Fig.11 (I), where node r is the root node. Suppose $w(a_1) \preceq w(a_2)$. Assume the metric does not satisfy the property in (11). Since $w(a_1) \preceq w(a_2)$, it is possible that $w(a_1 \oplus a_3) \succ w(a_2 \oplus a_3)$ as shown in Fig.11 (II) and (III). Meanwhile, the output of Edmonds' algorithm is $T_e^2 = a_1 \oplus a_3$, which is not optimal. Hence, Edmonds' algorithm fails to produce the minimum weight spanning r-arborescence. Therefore, the property in (11) is the necessary metric property for the statement that Edmonds' algorithm guarantees optimality on any network topology. □

5 DISTRIBUTED OPTIMAL BROADCAST ROUTING

Theorems 1, 2, 3, and 4 provide the necessary and sufficient metric properties for which Prim's algorithm, the generic edge-adding algorithm, the generic edge-deleting algorithm and Edmonds' algorithm guarantee optimality, respectively. It is important to note that although all these algorithms require knowledge of global topology, optimal broadcast routing protocols based on these algorithms do not need to be centralized. In the following, we outline a simple example of such distributed broadcast routing protocols.

Similar to link-state routing protocols such as OSPF [13], every node periodically advertises its local connectivity to the entire network. In this way, every node can learn the global topology. For each possible broadcast root node, a node runs one of the OBT algorithms examined in the previous sections to compute the broadcast tree. In the broadcast tree, the node identifies its children and stores them as the outgoing links in its routing table. When each non-root node receives a broadcast routing packet, it checks the source address of the packet and forwards the packet to the outgoing links according to its routing table. Since each node has the same view of the topology and runs the same OBT algorithm, the forwarding information stored in each node's routing table is consistent with the same unique broadcast tree and hence routing loops are avoided.

The uniqueness of the broadcast tree can be guaranteed, since each node follows the same order of adding/deleting edges/arcs. For instance, for the generic edge-adding algorithm, all the nodes follow the Kruskal algorithm's edge-adding order. For the generic edge-deleting algorithm, all the nodes follow the reverse-delete algorithm's edge-deleting order. For Edmonds' algorithm, circuits are lexicographically ordered by their node identities, and all nodes break circuits following the same lexicographic order. In all these algorithms, when there are multiple choices in adding edges, deleting edges, adding arcs or breaking circuits, the same tie breaking rule is applied at each node. In this case, the resultant broadcast tree is the same at each node, even if they run algorithms separately. Therefore, distributed optimal broadcast routing is achieved.

6 Root-independence Property

In optimal broadcast routing, each optimal broadcast tree has its own root node. In general, the optimal broadcast tree for one root node is different from the optimal broadcast tree for another root node. The computation overhead and storage overhead on the routers can be huge, if there are many broadcasting source nodes in the network. Therefore, we are interested in developing metric properties for which the computed broadcast trees are root independent, i.e., the MST or minimum weight spanning arborescence rooted at one node is the same as a MST or minimum weight spanning arborescence rooted at another node.

In the following, we first study the root-independence property of the MST, and then study the root-independence property of the minimum weight spanning arborescence. Before the formal analysis, we define the root-independence of the metric and root-independence of the MST.

Definition 8: The metric $w(\cdot)$ is root independent, if $w(F_1) = w(F_2)$, where F_1 and F_2 are subgraphs that have different root nodes and satisfy $N(F_1) = N(F_2), E(F_1) = E(F_2)$.

Definition 9: A MST $T = (N(T), E(T))$ based on metric $w(\cdot)$ is root independent, if it is a MST no matter which of its node is set as the root node.

Theorem 5: Given any connected and undirected network topology G whose root node is r, the MST is independent of root node r, if and only if the metric definition is independent of root node r.

Proof:

Sufficient condition:

Let T_1^* be one MST of the connected and undirected network topology G_1 whose root node is r_1. It follows that $w(T_1^*) \preceq w(T_1)$, where T_1 is any spanning tree rooted at r_1. For network topology G_1, change the root node from r_1 to $r_2 \neq r_1$. Denote the new network topology as G_2. Note that any spanning tree rooted at r_1 can also be a spanning tree rooted at r_2, and vice verse. Let T_2^* be the spanning tree generated by changing the root node of T_1^* from r_1 to r_2. Since the metric definition is independent of root node, for any r_1-rooted spanning tree T_1 of G_1, we have a r_2-rooted spanning tree T_2 of G_2 such that $N(T_1) = N(T_2), E(T_1) = E(T_2), w(T_1) = w(T_2)$. It follows that $w(T_2^*) = w(T_1^*) \preceq w(T_1) = w(T_2)$, i.e., T_2^* is a MST of G_2. Therefore, the MST is independent of the root node.

Necessary condition:

Given any connected and undirected network topology with a root node, we need to prove that if the MST is independent of the root node then the metric definition is independent of the root node. This can be proved by showing that its contrapositive is correct, i.e., for a metric whose weight computation is related to the identity of the root node, there exists at least one topology, where the MST is dependent on the root node.

Suppose, for a network topology G_1 whose root node is r_1, T_1 is a MST rooted at r_1. For network topology G_1, change the root node from r_1 to $r_2 \neq r_1$. Denote the new network topology as G_2. Change the root node r_1 of T_1 to r_2, and let the generated spanning tree as T_2. Since the metric definition is dependent on the root node, it is possible $w(T_1) \neq w(T_2)$. Therefore, the MST is dependent on the root node. □

Theorem 6: Given any connected and directed network topology D whose root node is r, the minimum weight spanning arborescence is dependent on the root node r.

Proof:

In directed network topologies, the definition of minimum weight spanning arborescence is dependent on the root node r. For two nodes n_1 and n_2, the weight of the link from n_1 to n_2 is generally different from the weight of the link from n_2 to n_1. Moreover, the link between n_1 and n_2 may be unidirectional rather than bidirectional.

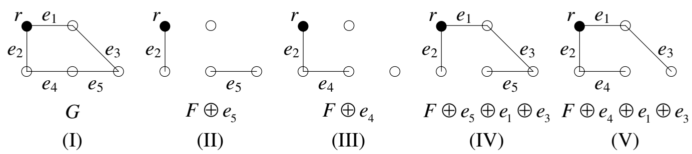


Fig. 12. The generic edge-adding algorithm example for the minimum tree depth routing

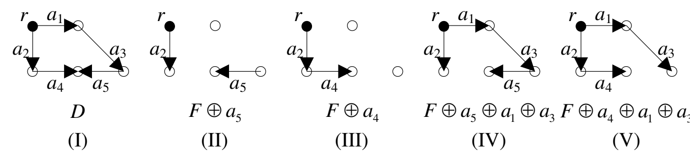


Fig. 13. The Edmonds algorithm example for the minimum tree depth routing

For some node, there may not exist a spanning arborescence rooted at this node. Therefore, in the general case, the root-independence property does not hold for the minimum weight spanning arborescence. □

7 Case Study Based on Compatibility Analysis

In this section, we illustrate by examples about how to apply our analytical results in Sections 3 and 4 to judge the compatibility between broadcast routing metrics and OBT algorithms.

7.1 Case 1: Minimum Tree-depth Broadcast Routing

Consider the minimum tree-depth broadcast routing that aims at finding the minimum depth broadcast tree rooted at a broadcast source node to minimize message delivery delay. In order to cover more general topologies, we extend the depth metric to forests, connected subgraphs and partition subgraphs. The depth of a subgraph F is defined as follows:

$$w(F) = \max_{T \subset F} depth(T), \tag{15}$$

where T is a component of F, and $depth(T)$ is the depth of component T. The depth of a component T is defined as the maximum number of edges along the shortest simple path from the root node to a leaf node. If broadcast source node does not belong to the tree T, then the node in T that can minimize the depth of T is picked as the local root node. If the broadcast source node belongs to T, then this broadcast source node is the root node.

7.1.1 Compatibility with Prim's Algorithm

It is easy to show that the depth metric definition in (15) satisfies the property in (4). Therefore, by Theorem 1, Prim's algorithm can find the minimum depth broadcast tree.

7.1.2 Compatibility with the Generic Edge-adding Algorithm

The depth metric definition in (15) does not satisfy the property in (6). Consider the network topology G as shown in Fig.12 (I). Let $F = (N(G), \{e_2\})$. Note that $w(F \oplus e_5) = 1 \prec w(F \oplus e_4) = 2$ as shown in Fig.12 (II) and (III), but $w(F \oplus e_5 \oplus e_1 \oplus e_3) = 3 \succ w(F \oplus e_4 \oplus e_1 \oplus e_3) = 2$ as shown in Fig.12 (IV) and (V). Therefore, it does not satisfy the property in (6). Therefore, by Theorem 2, the generic edge-adding algorithm cannot guarantee finding the minimum depth broadcast tree. This can be verified by noticing that one possible tree generated by the generic edge-adding algorithm is $F \oplus e_5 \oplus e_1 \oplus e_3$, which is not optimal.

7.1.3 Compatibility with the Generic Edge-deleting Algorithm

It is clear that the depth metric in (15) satisfies the property in (8). Therefore, by Theorem 3, the generic edge-deleting algorithm can find the minimum depth broadcast tree.

7.1.4 Compatibility with Edmonds' Algorithm

The depth metric in (15) does not satisfy the property in (11). Consider the network topology D as shown in Fig.13 (I). Let $F = (N(D), \{a_2\})$. Note that $w(F \oplus a_5) = 1 \prec w(F \oplus a_4) = 2$ as shown in Fig.13 (II) and (III), but $w(F \oplus a_5 \oplus a_1 \oplus a_3) = 3 \succ w(F \oplus a_4 \oplus a_1 \oplus a_3) = 2$ as shown in Fig.13 (IV) and (V). Therefore, the depth metric does not satisfy the property in (11). Therefore, by Theorem 4, Edmonds' algorithm cannot guarantee producing the minimum depth broadcast arborescence. This can be verified by noticing that one possible Edmdonds' algorithm output is $F \oplus a_5 \oplus a_1 \oplus a_3$, which is not optimal.

7.2 Case 2: Most Reliable Widest Bandwidth Broadcast Routing

Consider a most reliable widest bandwidth metric whose design is based on the following two observations. First, to maximize the capacity of links over the broadcast tree T, it is desirable to design a metric maximizing the bandwidth of the tree. The bandwidth of the tree is defined as the minimum bandwidth among all the links of the tree. Second, to ensure successful delivery of a broadcast message, a proper broadcast routing metric should minimize the probability that the broadcast message is lost by some nodes in the broadcast tree T. This probability can be calculated as

$$1 - \prod_{i \in E(T)} (1 - p_i) \approx \sum_{i \in E(T)} p_i, \tag{16}$$

where p_i is the estimated packet loss rate for a link in the broadcast tree, and the approximation is based on the assumption that each link's packet loss rate is small enough. This is valid for most scenarios. Hence, it is reasonable to consider the following lexicographic metric

$$(b, p), \tag{17}$$

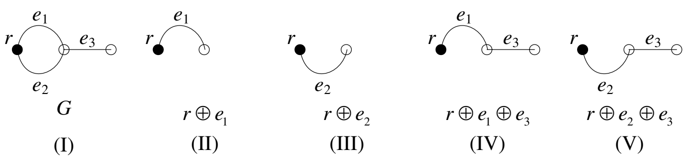


Fig. 14. The Prim algorithm example for the most reliable widest bandwidth broadcast routing

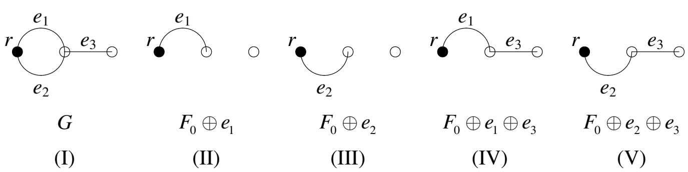


Fig. 15. The generic edge-adding algorithm example for the most reliable widest bandwidth broadcast routing

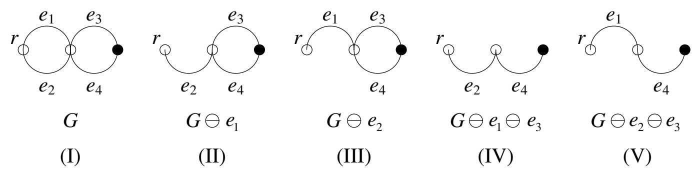


Fig. 16. The generic edge-deleting algorithm example for the most reliable widest bandwidth broadcast routing

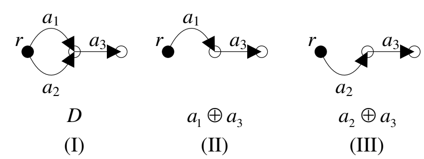


Fig. 17. The Edmonds algorithm example for the most reliable widest bandwidth broadcast routing

where b is the bandwidth of the tree, p is the packet loss rate of the tree. We have $w(b_1, p_1) \preceq w(b_2, p_2)$ if either $b_1 > b_2$ or $b_1 = b_2, p_1 \leq p_2$. The objective of the most reliable widest bandwidth broadcast routing is to find the lowest packet loss rate spanning tree within all widest bandwidth spanning trees.

7.2.1 Compatibility with Prim's Algorithm

The metric in (17) does not satisfy the required property in (4). Consider the network topology in Fig.14 (I). Let $w(e_1) = (10, 0.03), w(e_2) = (8, 0.01), w(e_3) = (5, 0.05)$. Note that $w(r \oplus e_1) = (10, 0.03) \prec w(r \oplus e_2) = (8, 0.01)$ but $w(r \oplus e_1 \oplus e_3) = (5, 0.08) \succ w(r \oplus e_2 \oplus e_3) = (5, 0.06)$, where $r \oplus e_1, r \oplus e_2, r \oplus e_1 \oplus e_3, r \oplus e_2 \oplus e_3$ are shown in Fig.15 (II), (III), (IV), (V), respectively. Therefore, the most reliable widest bandwidth metric does not satisfy the property in (4). By Theorem 1, Prim's algorithm cannot guarantee producing the MST. This can be easily shown by noticing that the algorithm output is $r \oplus e_1 \oplus e_3$, which is not optimal.

7.2.2 Compatibility with the Generic Edge-adding Algorithm

The metric in (17) does not have the required property in (6). Consider the network topology G in Fig.15 (I). Let $w(e_1) = (10, 0.03), w(e_2) = (8, 0.01), w(e_3) = (5, 0.05)$. Note that $w(F_0 \oplus e_1) = (10, 0.03) \prec w(F_0 \oplus e_2) = (8, 0.01)$ but $w(F_0 \oplus e_1 \oplus e_3) = (5, 0.08) \succ w(F_0 \oplus e_2 \oplus e_3) = (5, 0.06)$, where $F_0 = (N(G), \emptyset)$, and $F_0 \oplus e_1, F_0 \oplus e_2, F_0 \oplus e_1 \oplus e_3, F_0 \oplus e_2 \oplus e_3$ are shown in Fig.15 (II), (III), (IV), (V), respectively. Therefore, the most reliable widest bandwidth metric does not satisfy the property in (6). By Theorem 2, the generic edge-adding algorithm cannot guarantee producing the MST. This can be easily shown by noticing that the algorithm output is $F_0 \oplus e_1 \oplus e_3$, which is not optimal.

7.2.3 Compatibility with the Generic Edge-deleting Algorithm

The metric in (17) does not have the required property in (8). Consider the network topology G in Fig.16 (I). Let $w(e_1) = (5, 0.04), w(e_2) = (10, 0.07), w(e_3) = (5, 0.05), w(e_4) = (10, 0.06)$. Then, we have $w(G \ominus e_2) = (5, 0.15) \prec w(G \ominus e_1) = (5, 0.18), w(G \ominus e_4) = (5, 0.16) \prec w(G \ominus e_3) = (5, 0.17)$. Note that although $w(G \ominus e_2) = (5, 0.15) \prec w(G \ominus e_1) = (5, 0.18)$, $w(G \ominus e_2 \ominus e_3) = (5, 0.1) \succ w(G \ominus e_1 \ominus e_3) = (10, 0.13)$, where $G \ominus e_1, G \ominus e_2, G \ominus e_1 \ominus e_3, G \ominus e_2 \ominus e_3$ are shown in Fig.16 (II), (III), (IV), (V), respectively. Therefore, the most reliable widest bandwidth metric does not satisfy the property in (8). By Theorem 3, the generic edge-deleting algorithm cannot guarantee optimality. This can be shown by noticing that the MST of the given network topology is $G \ominus e_1 \ominus e_3$, while the output of the algorithm is $G \ominus e_2 \ominus e_4$, which is not a MST.

7.2.4 Compatibility with Edmonds' Algorithm

The metric in (17) does not have the required property in (11). Consider the network topology D shown in Fig.17 (I). Let $w(a_1) = (10, 0.03), w(a_2) = (8, 0.01), w(a_3) = (5, 0.05)$. Note that $w(a_1) \prec w(a_2), w(a_1 \oplus a_3) = (5, 0.08) \succ w(a_2 \oplus a_3) = (5, 0.06)$, where $a_1 \oplus a_3, a_2 \oplus a_3$ are shown in Fig.17 (II), (III), respectively. Therefore, the most reliable widest bandwidth metric does not satisfy the property in (11). By Theorem 4, Edmonds' algorithm does not guarantee optimality. This can be shown by noticing that the algorithm output $a_1 \oplus a_3$ is not the minimum weight spanning arborescence.

7.3 Case 3: Minimum Energy Broadcast Routing

Let us consider the minimum energy broadcast routing, whose metric is defined in (1).

7.3.1 Compatibility with Prim's Algorithm

The metric in (1) does not satisfy the property in (4). Consider the network topology G shown in Fig.18 (I). The numbers associated with the edges are the weights of these edges. Let the partial spanning tree be $T_r = (\{r, i\}, \{e_{ri}\})$ as shown by the bold part of Fig.18 (I). Note that $w(T_r \oplus e_{ij}) = 2 \prec w(T_r \oplus e_{rj}) = 2.5$ as shown

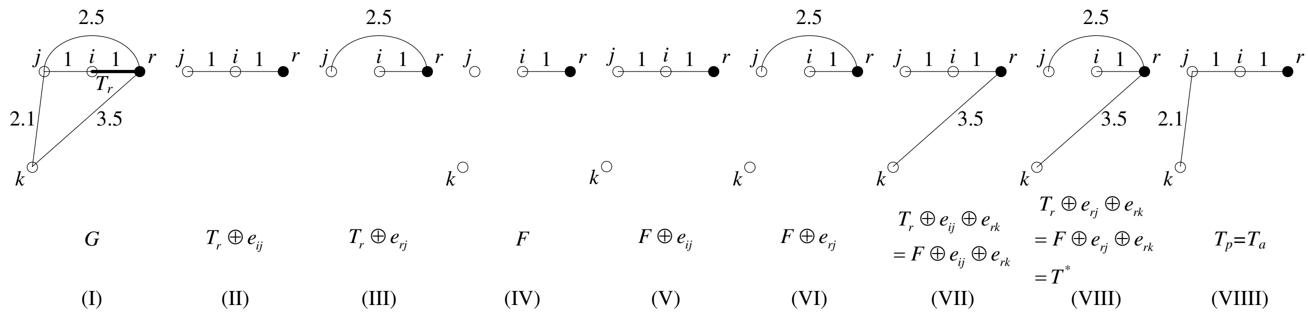


Fig. 18. The Prim algorithm and generic edge-adding algorithm example for minimum energy broadcast routing

in Fig.18 (II) and (III), but $w(T_r \oplus e_{ij} \oplus e_{rk}) = 4.5 \succ w(T_r \oplus e_{rj} \oplus e_{rk}) = 3.5$ as shown in Fig.18 (VII) and (VIII). Hence, the metric in (1) does not satisfy the property in (4). Therefore, by Theorem 1, Prim's algorithm cannot guarantee producing the minimum energy broadcast routing tree based on the metric in (1). This can be verified by noticing that $w(T_p) = 4.1 \succ w(T^*) = 3.5$, where T_p is the Prim tree generated by Prim's algorithm as shown in Fig.18 (VIIII) and T^* is the minimum energy broadcast tree as shown in Fig.18 (VIII).

7.3.2 Compatibility with the Generic Edge-adding Algorithm

The metric in (1) does not satisfy the property in (6). Again, consider the network topology G shown in Fig.18 (I). Suppose the partial spanning forest generated by the generic edge-adding algorithm is $F = (N(G), \{e_{ri}\})$ as shown in Fig.18 (IV). Note that $w(F \oplus e_{ij}) = 2 \prec w(F \oplus e_{rj}) = 2.5$ as shown in Fig.18 (V) and (VI), but $w(F \oplus e_{ij} \oplus e_{rk}) = 4.5 \succ w(F \oplus e_{rj} \oplus e_{rk}) = 3.5$ as shown in Fig.18 (VII) and (VIII). Hence, the metric does not satisfy the property in (6). Therefore, by Theorem 2, the generic edge-adding algorithm cannot guarantee producing the minimum weight broadcast routing tree based on the metric in (1). This can be verified by noticing that, for the generic edge-adding tree T_a and the minimum weight broadcast routing tree T^*, $w(T_a) = 4.1 \succ w(T^*) = 3.5$, as shown in Fig.18 (VIIII) and (VIII).

7.3.3 Compatibility with the Generic Edge-deleting Algorithm

Note that the metric is based on trees and property definition in Theorem 3 is based on connected graphs which may contain cycles. It is impossible to find a reasonable definition of the metric in (1) for a graph containing cycles. Therefore, the metric in (1) and the property definition in Theorem 3 are not compatible. Hence, the generic edge-deleting algorithm cannot find the minimum energy broadcast routing tree.

7.3.4 Compatibility with Edmonds' Algorithm

The metric in (1) does not satisfy the required property in (11). Consider the connected and directed network topology D whose root node is r as shown in Fig.19

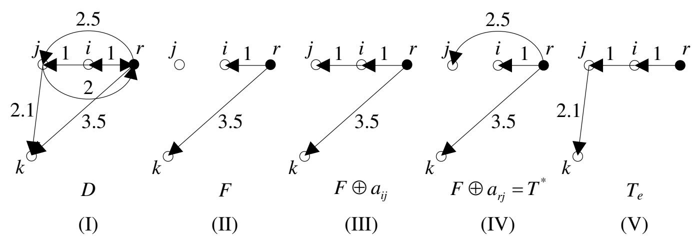


Fig. 19. The Edmonds algorithm example for the minimum energy broadcast routing

(I). The numbers associated with arcs are the weights of these arcs. Let F be a partial spanning arborescence forest as shown in Fig.19 (II). Note that $w(a_{ij}) \prec w(a_{rj})$ and $w(F \oplus a_{ij}) = 4 \succ w(F \oplus a_{rj}) = 3$ as shown in Fig.19 (III) and (IV). Hence, the minimum energy broadcast routing metric in (1) does not satisfy the property in (11). Therefore, Edmonds' algorithm does not produce the minimum weight broadcast routing arborescence based on the metric in (1). This conclusion can be verified by noticing that $w(T_e) = 4 \succ w(T^*) = 3$, where T_e is the Edmonds arborescence generated by Edmonds' algorithm and T^* is the minimum energy broadcast routing arborescence.

7.4 Case 4: Lexicographically Optimal Broadcast Routing based on Packet Loss Rate

Consider a broadcast routing metric finding the lexicographically optimal broadcast tree in terms of packet loss rate on each link. The metric is a sorted list of link packet loss rates of the considered network topology. Elements of the list are sorted in a decreasing order. Mathematically, the metric can be expressed as: $w(a) = (a_1, a_2, ..., a_m)$, where m is the total number of links in the network topology a, and $a_1 \geq a_2 \geq ... \geq a_m$ are the packet loss rates of links. Consider two network topologies a and b, and let $w(a) = (a_1, a_2, ..., a_m), w(b) = (b_1, b_2, ..., b_n)$. Define $w(a) = w(b)$, if $m = n, a_i = b_i, i = 1, 2, ..., m$. Define $w(a) \prec w(b)$, if there exists a number l such that $a_i = b_i, 1 \leq i \leq l-1$, and either $l-1 < \min\{m, n\}, a_l < b_l$, or $l-1 = m, m < n$. We further define $w(a) \preceq w(b)$ if either $w(a) = w(b)$ or $w(a) \prec w(b)$.

7.4.1 Compatibility with Prim's Algorithm

It is clear that the metric satisfies the property in (4). Therefore, by Theorem 1, Prim's algorithm can find the optimal broadcast tree.

7.4.2 Compatibility with the Generic Edge-adding Algorithm

It is clear that the metric satisfies the property in (6). Therefore, by Theorem 2, the generic edge-adding algorithm can find the optimal broadcast tree.

7.4.3 Compatibility with the Generic Edge-deleting Algorithm

It is clear that the metric satisfies the property in (8). Therefore, by Theorem 3, the generic edge-deleting algorithm can find the optimal broadcast tree.

7.4.4 Compatibility with Edmonds' Algorithm

It is clear that the metric satisfies the property in (11). Therefore, by Theorem 4, Edmonds' algorithm can find the optimal broadcast tree.

7.5 Case 5: Maximum Network Lifetime Broadcast Routing

Consider the broadcast routing metric which maximizes the network lifetime [14]. It is defined as

$$\min_{i \in N(T)} \frac{\varepsilon_i}{\max\limits_{(i,j) \in E(T)} P_{ij}}, \tag{18}$$

where $N(T)$ is the node set of the broadcast tree T, ε_i is the residual energy of node i, $E(T)$ is the link set of the broadcast tree T, and P_{ij} is the minimum power consumption of transmission from node i to node j. The broadcast routing metric definition given in (18) is essentially the minimum lifetime of all the links in the broadcast tree T. Also note the fact that the underlying network topology is undirected when the residual energy of each node is equal, and the underlying network topology is directed when the residual energy of each node is different [14].

7.5.1 Compatibility with Prim's Algorithm

Note that the property definition in Theorem 1 is based on undirected network topologies. Therefore, when the residual energy of each node is different, the metric in (18) is not compatible with the property definition in Theorem 1. When the residual energy of each node is equal, we further check if the metric in (18) satisfies the property in (4). Clearly, the minimization operation of the metric satisfies the property in (4). Therefore, we have the following conclusions.

- If the residual energy of each node is different, Prim's algorithm cannot find the maximum network lifetime broadcast tree based on the metric in (18).
- If the residual energy of each node is equal, Prim's algorithm produces the maximum network lifetime broadcast tree based on the metric in (18).

7.5.2 Compatibility with the Generic Edge-adding Algorithm

Since the property definition in Theorem 2 is based on undirected network topologies, the metric in (18) is not compatible with the property definition in Theorem 2 when the residual energy of each node is different. When the residual energy of each node is equal, we further check if the metric in (18) satisfies the property in (6). Clearly, the minimization operation of the metric satisfies the property in (6). Therefore, we have the following conclusions.

- If the residual energy of each node is different, the generic edge-adding algorithm cannot find the maximum network lifetime broadcast tree based on the metric in (18).
- If the residual energy of each node is equal, the generic edge-adding algorithm produces the maximum network lifetime broadcast tree based on the metric in (18).

7.5.3 Compatibility with the Generic Edge-deleting algorithm

Since the property definition in Theorem 3 is based on undirected network topologies, the metric in (18) is not compatible with the property definition in Theorem 3 when the residual energy of each node is different. When the residual energy of each node is equal, we further check if the metric in (18) satisfies the property in (8). It is clear that the minimization operation of the maximum network lifetime metric satisfies the property in (8). Therefore, we have the following conclusions.

- If the residual energy of each node is different, the generic edge-deleting algorithm cannot find the maximum network lifetime broadcast tree based on the metric in (18).
- If the residual energy of each node is equal, the generic edge-deleting algorithm produces the maximum network lifetime broadcast tree based on the metric in (18).

7.5.4 Compatibility with Edmonds' Algorithm

Since the property definition in Theorem 4 is based on directed network topologies and undirected topologies can be transformed into directed network topologies, the metric in (18) is compatible with the property definition in Theorem 4. Next, we check if the metric in (18) satisfies the property in (11). Clearly, the minimization operation of the maximum network lifetime metric satisfies the property in (11). Therefore, we have the following conclusions.

- If the residual energy of each node is different, Edmonds' algorithm produces the maximum network lifetime broadcast arborescence based on the metric in (18).
- If the residual energy of each node is equal, Edmonds' algorithm also produces the maximum network lifetime broadcast arborescence based on the metric in (18).

8 Conclusion and Future Work

In this paper, the potential incompatibility between broadcast routing metrics and optimal broadcast tree calculation algorithms is identified. To theoretically model and analyze this incompatibility problem, we propose a novel broadcast routing algebra. Our algebra can operate on arbitrary network topologies and capture broadcast routing. Using our broadcast routing algebra, we develop and prove the necessary and sufficient properties of broadcast routing metrics which guarantee optimal broadcast routing in both undirected network topologies and directed network topologies. These properties are used to identify compatibility between broadcast routing metrics and broadcast routing algorithms. The results in this work serve as important guidelines for broadcast routing design.

References

[1] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Comput. Netw. ISDN Syst.*, vol. 47, no. 4, pp. 445–487, 2005.

[2] R. Baumann, S. Heimlicher, M. Strasser, and A. Weibel, "A survey on routing metrics," ETH Zrich, Tech. Rep. TIK Report 262, 2006.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. Cambridge, MA, USA: MIT Press, 2001.

[4] D. Jungnickel, *Graphs, Networks and Algorithms*. Springer, 2007.

[5] J. L. Sobrinho, "Algebra and algorithms for QoS path computation and hop-by-hop houting in the internet," in *Proc. IEEE INFOCOM*, 2001.

[6] ——, "Network routing with path vector protocols: theory and applications," in *Proc. ACM SIGCOMM*, 2003.

[7] Y. Yang and J. Wang, "Design guidelines for routing metrics in multihop wireless networks," in *Proc. IEEE INFOCOM*, 2008.

[8] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks," in *Proc. IEEE INFOCOM*, 2000.

[9] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 5, pp. 66–77, 1983.

[10] J. Kleinberg and va Tardos, *Algorithm Design*. Addison Wesley, 2005.

[11] J. Bondy and U. Murty, *Graph Theory*. Springer, 2008.

[12] A. Gibbons, *Algorithmic Graph Theory*. Cambridge University Press, 1985.

[13] J. Moy, "OSPF Version 2," RFC 2328, April 1998.

[14] I. Kang and R. Poovendran, "Maximizing network lifetime of broadcasting over wireless stationary ad hoc networks," *Mob. Netw. Appl.*, vol. 10, no. 6, pp. 879–896, 2005.