On the Construction of a Strongly Connected Broadcast Arborescence with Bounded Transmission Delay

Yingshu Li * My T. Thai † Feng Wang † Ding-Zhu Du †

Abstract

Energy conservation is an important concern in wireless networks. Many algorithms for constructing a broadcast tree with minimum energy consumption and other goals have been developed. However, no previous research work considers the total energy consumption and transmission delays of the broadcast tree simultaneously. In this paper, based on (α, β)-tree, a novel concept to wireless networks, we define a new Strongly connected Broadcast Arborescence with bounded Transmission delay (SBAT) problem and design the Strongly connected Broadcast Arborescence (SBA) algorithm with linear running time to construct a strongly connected broadcast tree with bounded total power, while satisfying the constraint that the transmission delays between the

*Department of Computer Science, Georgia State University, 34 Peachtree Street, Atlanta, GA, 30303. Email: {yli}@cs.gsu.edu.

†Department of Computer Science and Engineering, University of Minnesota, 200 Union Street, Minneapolis, MN, 55455. Email: {mythai,fwang,dzd}@cs.umn.edu.

source and the other hosts are also bounded. We also propose the distributed version of the SBA algorithm. The theoretical analysis and simulation results show that the SBA algorithm gives a proper solution to the SBAT problem.

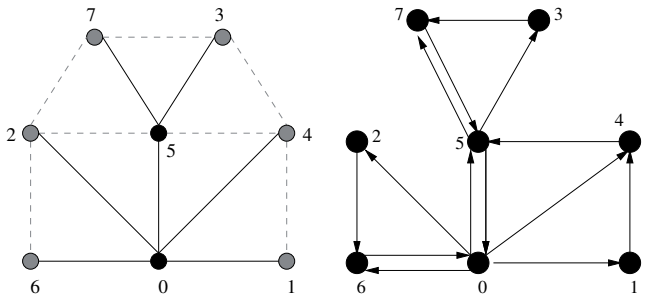
Keywords: Wireless networks, Routing protocols, Network management, Distributed applications, Algorithm design and analysis .

1 Introduction

Wireless networks are bringing more and more benefits to us and more effort are being invested to this field. Wireless networks do not have pre-defined or fixed infrastructures. Hosts in a wireless network communicate via a shared medium either through a single hop or multihop routing. The transmission power of each host greatly affects the network topology. If the transmission ranges of two hosts are big enough for them to reach each other, then they can communicate directly. However, hosts may not always transmit at the maximum transmission power level due to the concern of energy conservation in wireless networks. Therefore, people spend much effort in studying how to adjust the transmission power of each host for variety of issues, such as network connectivity, link quality and routing protocols. In this paper, we study how to adjust the transmission power of each host for the broadcast routing in wireless networks.

Broadcast is one of the important and primary communication methods in wireless networks for the dissemination of control packets, topology update packets or route discovery

packets. If a host wants to do a broadcast (we call it a source node), it is impractical for each host to just blindly forward every packet it receives to all of its neighbors, since the network capacity will be greatly reduced and the *broadcast storm* may happen. Therefore, a broadcast tree rooted at the source node that spans all the hosts in the network needs to be constructed so that the packets can be delivered to every host along the tree. For example, in Figure 1(a), node 0 needs to broadcast a message. All the solid edges form a broadcast tree. Then the message can be broadcast along the paths in this tree until every host in the network receives this message. Here, only node 5 needs to relay the message and all the other leaf nodes of the tree do not need to relay this message.



(a) A broadcast tree in a network (b) Strongly connected broadcast tree

Figure 1: An example for constructing a SBA.

The broadcast tree construction problem in wireless networks is to decide a transmission power level for each host so that a tree can be constructed based on the resultant network topology. In this paper, we construct a broadcast tree with the following constraints:

A. Bound the total power consumption of all the hosts

The total power consumption of all the hosts needs to be bounded for energy conservation.

Hosts in a wireless ad hoc network are battery-powered, and in a battle field, it is difficult or impossible to recharge or replace the batteries. Sensor nodes in a sensor network have very limited power, so power must be used efficiently. The driving force of this constraint comes from the above applications.

B. Bound the transmission delay between the source node and the other hosts

Transmission delay refers to the time for messages to travel across the network. It is highly dependent on the length of the path between two communicating hosts. Therefore, to obtain a small transmission delay between a pair of hosts, it is expected that the path from the source to the destination should be as short as possible. In Figure 1(a), if we assume that each edge has a unit length. Path1 $(0, 1, 4, 3)$ is longer than path2 $(0, 5, 3)$. If the length of the path between a pair of communicating hosts is the main factor that dominates the transmission delay, then the transmission delay for a message to be sent along path2 is less than the one along path1. In a battle field where each soldier carries a laptop, they form a wireless ad hoc network. The commander needs to quickly disseminate orders to all the soldiers. In a sensor network where a sink needs to quickly broadcast user queries to all the sensor nodes, the result also needs to be reported back to users as soon as possible. In all of these applications, small transmission delays are highly expected.

C. Preserve network connectivity

The network connectivity must be maintained at any time to support peer-to-peer communications in wireless networks. Moreover, a message may need to be delivered through several hops before reaching its final destination, thus, it is important to guarantee that

every host is reachable from any other one. Therefore, for each host, there should exist a path to any other host in the network. Such a network is said to be *strongly connected* [8] if the network is modelled as a directed graph. In Figure 1(b), a strongly connected broadcast tree is illustrated where there exists a path between any ordered pair of hosts. Then the source node can send messages to any other host and any other host can also send messages back to the source node.

Many algorithms for constructing broadcast trees have been proposed. However, most of them modelled the network as an undirected graph and did not consider both the total power of the tree and the transmission delays between the source and all the other hosts. In this paper, we construct a broadcast tree with both of these two concerns and try to balance them so that the broadcast tree can better serve the network. We give a formal definition of this problem in section 3. It is called Strongly Connected Broadcast Arborescence with Bounded Transmission Delay (SBAT) Problem and we give out the solution by constructing a strongly connected broadcast tree with bounded total power while the transmission delays between the source and all the other hosts are also bounded. With such a broadcast tree, the requirements for energy conservation and transmission delay are both satisfied. We evaluate the solution through theoretical analysis and simulations. It is shown that the proposed algorithm gives a proper solution to the SBAT problem.

The rest of this paper is organized as follows. Section 2 briefly describes some related research works. The wireless communication model and some preliminaries are illustrated in Section 3. Section 4 gives the solution to our SBAT problem, and the analysis of the

proposed algorithm is presented in Section 5. The simulations are introduced in Section 6 and a distributed algorithm is shown in Section 7. Section 8 ends this paper with a conclusion and some future research work.

2 Related work

The broadcast problem in wireless networks has received significant attention and most of the previous work addressed the issue of energy-efficiency aiming at minimizing the total energy consumption of the whole broadcast tree. This problem is shown to be NP-hard in [9] and [19]. Some near-optimal heuristics are given in [20] and [21]. The BLU algorithm in [20] first establishes a minimum-cost path from the source to each destination, and the broadcast tree is formed by the superposition of these unicast paths. However, the aggregation of these individual paths does not guarantee that the resultant broadcast tree has the minimum total cost. The BIP algorithm in [20] is known to be the best globalized solution. It is similar to Prim's MST algorithm where the nodes are added into the tree one at a time on a minimum cost basis, where the cost is defined as the cost of the link connecting the new added node to the tree. In BIP, the nodes are added to the tree in a similar way. The difference is that the cost is defined as the incremental cost to add a new node to the tree, so the node resulting the smallest total incremental cost is added at each step. BIP does not guarantee to provide a minimum-cost tree, even though the objective in [20] is the minimum total energy. In [5], the authors designed a localized protocol where each node only needs to know the distances to all of its neighbors and distances between its paired neighbors. Some other research works

that consider the broadcast tree construction in wireless networks are [1], [15], [3]. In [1], the proposed Hitch-Hiking model takes advantage of the physical layer to combine partial signals containing the same data in order to decode a complete message. The authors in [15] proposed a globalized algorithm which builds a 2-node-connected graph and assume an arbitrary energy model. Three different integer programming models are illustrated in [3] to provide an optimal solution of the minimum power broadcast/multicast problem in wireless networks.

The purpose of all the above algorithms is to minimize the total energy consumption of a broadcast tree. While in [7] and [6], the authors tried to maximize the network lifetime and balance the energy consumptions among all the hosts in tradeoff minimizing the total energy consumption. They focused on balancing the energy consumptions of all the nodes in a network. The network lifetime is defined as the earliest time that a node runs out of power in [7]. By delaying the earliest time that a node runs out of power in terms of balancing the energy consumptions of all the nodes, the network lifetime can be extended as much as possible.

None of the above research work considers to bound the transmission delay and minimize the total energy consumption simultaneously. To the best of our knowledge, no other research work addresses this problem. In this paper, we propose a novel method to solve this problem with the help of (α, β)-tree.

The (α, β)-tree was firstly introduced into wired networks by Khuller and Raghavachari in [13]. They noticed that it is almost impossible to consider minimizing the total weight

and minimizing the distance from the root to each node simultaneously using just a minimum spanning tree or just a shortest path tree, since there is a tradeoff between these two minimization goals. They proposed the LAST algorithm to construct a spanning tree that can simultaneously approximates a shortest-path tree and a minimum spanning tree. The motivation is to balance the minimization purposes of the total link cost and the cost of a message to be sent from each host to the root of the tree. In wired networks, energy conservation is not a great concern. However, in wireless networks, energy is a cherished resource that all the algorithms and protocols must take into account. Therefore, the cost in LAST is defined based on the distance of each neighboring hosts, while the cost in our algorithm should be the energy spent to transmit from the sender to the receiver. The authors in [18] mentioned to employ a (α, β)-tree for multicasting in wireless Ad Hoc networks. However, their communication model was still modelled as a wired one and they ignored the fact that the power needed for the communication between two nodes is not the distance between them.

The given example in [13] is illustrated in Figure 2. The minimum spanning tree in Figure 2(a) cannot guarantee that the distance from the root to each other node is the shortest and the shortest tree in Figure 2(b) makes it heavy (the total cost of the tree is larger). Figure 2(c) illustrated a reasonable tree that approximates both a minimum spanning tree and a shortest path tree.

BSMA in [22] is an algorithm in traditional wired networks. It is used to construct minimum-cost multicast trees with delay constraints. The BSMA structure begins with a

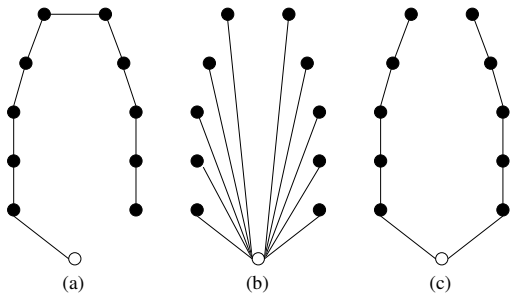


Figure 2: Approximating a minimum spanning tree and a shortest path tree.

shortest path tree and progressively works towards a minimum spanning tree to minimize the cost. The cost in BSMA is defined as the total bandwidth utilization. Our cost is defined as the total energy consumption. Unlike traditional wired networks where quality of service is important, for wireless networks, energy conservation is of primary concern which is especially true for sensor networks [2].

3 Wireless communication model and preliminaries

In this paper, a wireless network is modelled as a directed weighted connected graph, where each node represents a host in the wireless network. The weight of a directed edge (u, v) is equal to the transmission power needed to transmit from u to v. We assume that the received signal power at v is proportional to $d^{-\gamma}$, where γ is a parameter between 2 and 4, depending on the communication medium; d is the distance between u and v as defined in [16]. So the power needed to transmit from u to v is $p_{uv} = C_v \cdot d_{uv}^{\gamma}$ where C_v is a constant number determined by the signal detection threshold which is highly related to the environment.

Therefore, when deciding the transmission power level of a node, not only the distance is considered, but also the environment parameters, such as interference, communication medium, are also considered. In wireless networks, edge weights are not always symmetric between a pair of neighboring nodes. For example, if node u and v have different signal detection thresholds, the actual transmission power needed for node u to reach its direct neighbor v can be different from that in the reverse direction. The reason for two nodes to have different signal detection thresholds is that they may exist in environments with different noise levels. In this case, the network is modelled as a directed graph with different weights of the edges between a pair of neighboring nodes. Usually, a tree is undirected, therefore we use *arborescence* instead in this paper. We also assume that all the nodes are deployed in a 2-dimensional plane.

We now formally define a wireless network as a directed weighted connected graph $G = (V, E)$ where V is the set of the hosts in this network and E includes all the directed links in the network. Each directed edge $e = (u, v)$ is assigned a non-negative *weight* $w(e) = C_v d_{uv}^{\gamma}$ (or denoted as $w(u, v)$) which is the transmission power needed to transmit from u to v. Then the total weight of an arborescence A is $w(A) = \sum_{e \in A} w(e)$. A Minimum Spanning Arborescence (MSA) of G is an arborescence including all the nodes in G and this arborescence has the minimum weight.

In wireless network, all the neighbors within a sender's transmission range can receive the messages from the sender, unlike in the wired networks where a sender needs to send to every neighbor separately. It is called *Wireless Multicast Advantage* (WMA) [20]. Then for

each node u, if it transmits at power level $p(u) = max\{p_{uv} : (u, v) \in G\}$, this transmission can be received by all of its neighbors. Let A be a spanning arborescence of G and the *power* of this arborescence A (denoted as $p(A)$) is $p(A) = \sum_{u \in V} p(u)$.

A Shortest Path Tree (SPT) in G is a tree including all the nodes in G and in this SPT, all the nodes preserve the shortest paths from the root to itself. Let $d(e)$ be the length of an edge e. Let $d(G) = \sum_{e \in E} d(e)$ denotes the total length of a graph G. Clearly, $d(SPT)$ is the minimum compared to all the other trees.

If there is a communication path between any ordered pair of nodes in the broadcast arborescence, then the arborescence is *strongly connected*. We now formally define the Strongly connected Broadcast Arborescence with bounded Transmission delay problem as follows:

Definition 3.1 ***Strongly connected Broadcast Arborescence with bounded Transmission delay (SBAT) Problem***

Given a set of hosts in a wireless network, construct a strongly connected broadcast arborescence (SBA) so that $d(SBA)$ is bounded by a user-specified threshold and $p(SBA)$ is minimized.

In this paper, we introduce (α, β)-arborescence into wireless networks to solve the SBAT problem. It approximately satisfies the power constraint and the transmission delay constraint by constructing a single strongly connected SBA. Based on the definition of (α, β)-tree in wired networks [13], we give the definition of (α, β)-arborescence in wireless networks as follows:

Definition 3.2 *For $\alpha \geq 1$ and $\beta \geq 1$, a spanning arborescence A of G meeting the following two requirements is called an (α, β)-arborescence rooted at r.*

1. *(Distance) For every vertex u, the distance between r and u in A is at most α times the shortest distance from r to u in G.*

2. *(Power) The power of A is at most β times the power of an optimal solution to the SBAT problem.*

We add the strongly connectivity constraint to the (α, β)-arborescence. The weight in [13] is the distance between a pair of nodes. But in this paper, the weight is defined as the energy needed to transmit between a pair of nodes in wireless networks.

4 The SBA construction algorithm

Given a weighted, directed and strongly connected graph $G = (V, E)$, a root node r and an $\alpha > 1$, the SBA construction algorithm (SBA-algorithm) constructs an (α, β)-SBA satisfying the constraints that the total power of the SBA and the path lengths between the root r and the other nodes are bounded. The general idea of SBA-algorithm is as follows. First, it constructs two MSAs A and A'. A is rooted at r where there exists a path from r to every other node. A' is sinked at r where there exists a path from every other node to r. The superposition of A and A' results a strongly connected graph. Then based on the connectivity information of the network, the SBA-algorithm constructs a SPT H. H includes all the shortest paths from r to every other node. Next, A will be traversed in a depth-first

manner. When visiting a node u, if the length of the path from r to u in A is larger than a user-defined threshold α, then this path is replaced by the path from r to u in H. For A', the same process is conducted. After SBA-algorithm ends, the SBAT problem is solved.

We use our previously proposed algorithms MWIA in [7] and MWIA-TC in [14] to construct two MSAs. MWIA-TC constructs a Minimum Weight Incremental Arborescence (MWIA) A and a reverse MWIA A'. One more task that needs to be finished is that during the construction procedure of an MWIA, each node needs to record which node is its parent and which nodes are its children. A MWIA rooted at r is obtained as follows based on [7]: beginning from r, a node is added to the arborescence on a minimum weight basis i.e., find an edge with the minimum weight from a vertex in the arborescence to a vertex not in the arborescence. The arborescence is grown one node at a time until all nodes are included. When adding a node u and the edge (v, u) to the arborescence, node u's parent $\pi[u]$ in this MWIA is set to node v and node u becomes one of node v's children, i.e., adding u to v's children list. A reverse MWIA is obtained in [14] by first exchanging the weight of edge (x, y) and the weight of edge (y, x) in G, which gives a new graph G', then the above MWIA algorithm is employed to obtain a MWIA A'' on graph G'. When adding a node u and the edge (v, u) to A'', node u's parent $\pi'[u]$ in this MWIA is still set to node v and node u becomes one of node v's children, i.e., adding node u to v's children list. By changing the direction of each edge in A'' to its reverse direction, a sink-arborescence-like subgraph A' of the original graph G is generated and this is a reverse MWIA of G. Here A and A' are two desired MSAs.

Dijkstra's algorithm can be used to solve the single-source shortest-paths problem [8]. Therefore, we use Dijkstra's algorithm to construct a SPT H. H includes all the shortest paths from the root r to all the other nodes.

To construct and maintain the SBA, each node u needs to have a parent pointer $\pi[u]$ and an upper bound $d[u]$ on the distance from node u to the root r. Let $D_G(u, v)$ denote the length of the path from u to v. We use the $INITIALIZE$ and $RELAX$ algorithms in [13] to initialize and maintain both of these attributes as shown in Algorithm 1 and Algorithm 2:

Algorithm 1 INITIALIZE(G, r)

1: **for** each vertex $v \in V$ **do**
2: $d[v] \leftarrow \infty$
3: $\pi[v] \leftarrow NIL$
4: **end for**
5: $d[r] \leftarrow 0$

Algorithm 2 RELAX(u, v)

1: **if** $d[v] > d[u] + |(u, v)|$ **then**
2: $d[v] = d[u] + |(u, v)|$
3: $\pi[v] \leftarrow u$
4: **end if**

The SBA-algorithm then traverse the MSA A in a depth first manner beginning from the root r along the paths from r to all the other nodes. This is possible since when

constructing A, each node u's parent and children have already been recorded. We use the LAST algorithm in [13] to construct a directed LAST (dLAST) T rooted at r. When node u is reached for the first time, if $d[u]$ is greater than $\alpha D_{SPT}(r,u)$, then the shortest P_{ru} in H is added to A and $d[u]$ and $\pi[u]$ are updated. After this, node u's parent v needs to be checked if the updated path from r to u will result in shortening the path from r to v. If so, then v's parent will be checked and so on until the root r is reached. For A', the same procedure is conducted to obtain another dLAST T'. The LAST algorithm is shown in Algorithm 5.

Algorithm 3 ADD-PATH(v)

1: **if** $d[v] > D_{SPT}(r,v)$ and $parent_{SPT}(v)$!= NIL **then**
2: ADD-PATH($parent_{SPT}(v)$)
3: RELAX($parent_{SPT}(v), v$)
4: **end if**

Algorithm 4 DFS(u)

1: **if** $d[u] > \alpha D_{SPT}(r,u)$ **then**
2: ADD-PATH(u)
3: **end if**
4: **for** each child v of u in MSA **do**
5: RELAX(u, v)
6: DFS(v)
7: RELAX(v, u)
8: **end for**

Algorithm 5 LAST(MSA, SPT, r, α)

1: INITIALIZE(MSA, r)

2: DFS(r)

3: return SBA B

The union $B = T \cup T'$ is the desired SBA satisfying the constraints that both $p(B)$ and $d(B)$ are bounded. The SBA-algorithm is given in Algorithm 6.

Algorithm 6 SBA(MSA A, MSA A', SPT H, ROOT r, α)

1: T = LAST(A, H, r, α)

2: T' = LAST(A', H, r, α)

3: return SBA $B = T \cup T'$

To better understand the SBA algorithm, an example is illustrated in Figure 3. Figure 3(a) is the connectivity information of a network where the number on each edge denotes the physical distance between a pair of nodes. G in Figure 3(b) is the weighted graph after considering the transmission power of each node. G' in Figure 3(c) is the weighted graph by exchanging the weight of edge (x, y) and the weight of edge (y, x) in G. The shortest path tree H of G is given in Figure 3(d) and the two MSAs M and M' are illustrated in Figure 3(e) and Figure 3(f). Suppose A is the root and $\alpha = 2$. M is first traversed. When visiting D, the length of the path (A, B, C, D) in M is $d_M(A, D) = 23$, and the length of the path (A, D) in SPT H is $d_H(A, D) = 11$, therefore $d_M(A, D) > \alpha d_H(A, D)$. Thus, edge (A, D) is added and $\pi[u]$ is changed from C to A. Note that edge (C, D) is not deleted (represented as dotted line) since we need to go back along edge (C, D) to retrieve D's original parent C to check if the new added path (edge (A, D) in this example) will also shorten the path

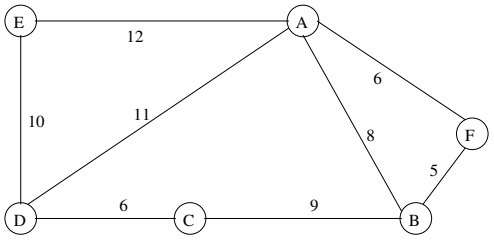
from A to C. In this example, the path from A to C will not be replaced. This backward check of a node's original parent happens whenever the path to this node is replaced. A dLAST T is obtained after traversing M, as shown in Figure 3(g). The same procedure is then employed on M' and when visiting D, edge (D, A) is added. Figure 3(h) gives another resultant dLAST T' after traversing M'. Finally, the desired SBA $B = T \cup T'$ is illustrated in Figure 3(i). The number in the brackets represent the physical distance of a pair of nodes, otherwise, the numbers represent the weight of this edge.

5 Analysis

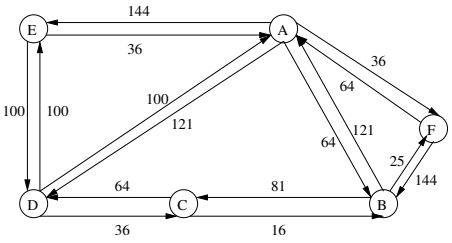
In this section, we evaluate the correctness of the SBA-algorithm through examining if the two constraints in Definition 3.2 are satisfied. α is a user-defined input and β is related to α. We derive a relationship between α and β so that the relationship between the powers of the constructed SBA and the optimal solution to the SBAT problem can be obtained. We also analyze the time complexity of the SBA-algorithm.

We employ the SBA-algorithm in a bidirectional graph $G = (V, E)$ to construct a SBA. For $p_{uv} = C_v \cdot d_{uv}^{\gamma}$, we set γ to 2 and C_v to 1 for every node v. We define $p(opt)$ to be the total power consumption of an optimal solution *opt* to the SBAT problem in G. Let T_{opt} be the directed broadcast arborescence rooted at root r of *opt* and T'_{opt} be the directed broadcast arborescence sinked at r.

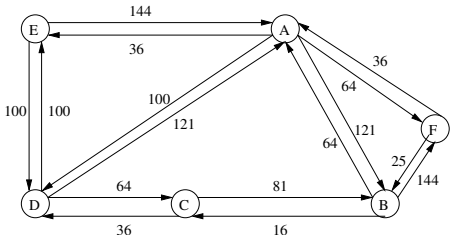
Lemma 1 *For every node u, the distance between r and u in T is at most α times the shortest distance from r to u in G.*



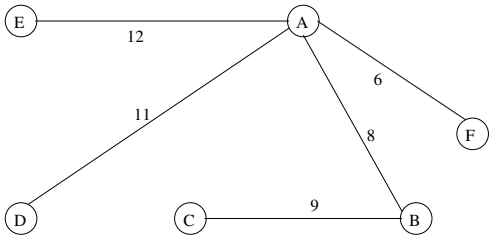
(a) Connectivity information of the network



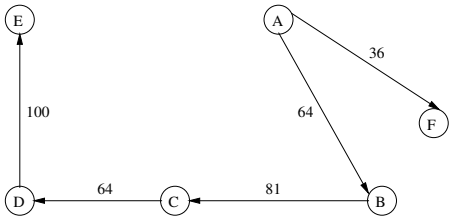
(b) Weighted graph G



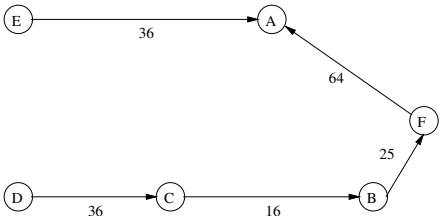
(c) Weighted graph G'



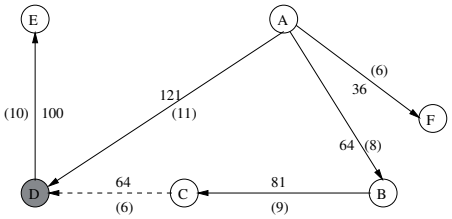
(d) A SPT H for G



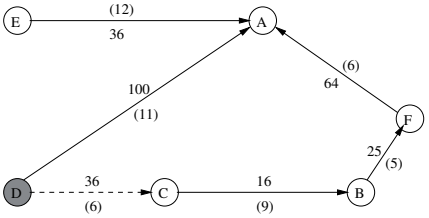
(e) An MSA M rooted at A



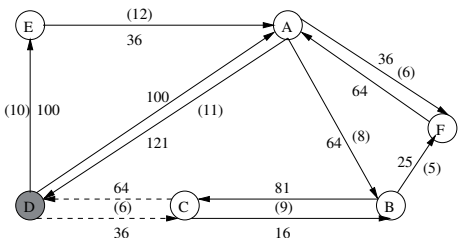
(f) An MSA M' sinked at A



(g) A directed-LAST T rooted at A



(h) A directed-LAST T' sinked at A



(i) A SBA $B = T \cup T'$

Figure 3: An example for constructing a SBA.

Proof. During the construction of T, for every node u, if $d[u] > \alpha d_{SPT}(r,u)$, then the shortest path between r and u is added to replace the old one by calling $ADD-PATH$. This added shortest path will never be replaced by other paths. Therefore, the lemma is true. □

Note that with WMA in wireless networks, $p(G)$ and $w(G)$ may be different and we have the following lemma.

Lemma 2 *For a bidirectional graph G, $p(G) \leq w(G)$.*

Proof. For a bidirectional graph G,

$$\begin{aligned} p(G) &= \textstyle\sum_{u \in V} p(u) \\ &= \textstyle\sum_{u \in V} max_{(u,v) \in E} w(u,v) \\ &\leq \textstyle\sum_{u \in V} \sum_{(u,v) \in E} w(u,v) \\ &= w(G) \end{aligned}$$

□

Lemma 3 *Given a SPT T and a MSA A, $w(T) \leq n(\frac{2}{\alpha-1})^2 w(A)$ and $w(T) \leq n(\frac{2}{\alpha-1})^2 w(A')$, where n is the number of the nodes in a network.*

Proof. It has been proved in [13] that $d(T) \leq \frac{2}{\alpha-1} d(A)$ and $d(T) \leq \frac{2}{\alpha-1} d(A')$. Then we have $w(T) \leq d^2(T) \leq (\frac{2}{\alpha-1})^2 d^2(A) \leq n(\frac{2}{\alpha-1})^2 w(A)$. It is the same for $w(T) \leq n(\frac{2}{\alpha-1})^2 w(A')$. □

Lemma 4 *$w(T_{opt}) \leq \Delta \cdot p(opt)$ and $w(T'_{opt}) \leq p(opt)$, where Δ is the maximum node degree in G.*

Proof. For the first inequality, for each node u on the arborescence rooted at the root, $\sum_{(u,v)\in E} w(u,v) \leq \Delta \cdot p_u$, thus $w(T_{opt}) \leq \Delta \cdot p(opt)$. For the second inequality, each non-root node u on the arborescence sinked at the root has only one outgoing edge (u,v), thus for each vertex u, p_u is at least $w(u,v)$. Hence $w(T'_{opt}) \leq p(opt)$. □

Theorem 1 *Let B be the SBA obtained by employing the SBA-algorithm, $p(B) \leq (1 + n(\frac{2}{\alpha-1})^2)(\Delta+1)p(opt)$.*

Proof. Let T be a SPT, A be a MSA rooted at the root and A' be a MSA sinked at the root. From Lemma 2 through Lemma 4 and the definition of MSA, we have

$$\begin{aligned} p(B) &\leq w(B) \\ &\leq w(A) + w(T) + w(A') + w(T) \\ &\leq (1 + n(\tfrac{2}{\alpha-1})^2)(w(A) + w(A')) \\ &\leq (1 + n(\tfrac{2}{\alpha-1})^2)(w(T_{opt}) + w(T'_{opt})) \\ &\leq (1 + n(\tfrac{2}{\alpha-1})^2)(\Delta \cdot p(opt) + p(opt)) \\ &\leq (1 + n(\tfrac{2}{\alpha-1})^2)(\Delta+1)p(opt) \end{aligned}$$

□

We believe that the SBAT problem is NP-complete, and $\beta = (1 + n(\frac{2}{\alpha-1})^2)(\Delta+1)$ is the performance ratio of our SBA algorithm. In a dense network where $\Delta = O(n)$, $\beta = O(n^2)$ and this is the worst case. Note that if for a pair of nodes (u,v), $w(u,v) = w(v,u)$, then $p(T) \leq 2(1 + n(\frac{2}{\alpha-1})^2)p(opt)$. The proof is similar to the proof of Theorem 1 except that $w(T_{opt}) = w(T'_{opt}) \leq p(opt)$.

For the general cases where γ is between 2 and 4 and C_v is a constant number in the formula $p_{uv} = C_v \cdot d_{uv}^{\gamma}$, by Jensen's Inequality [23], $d^{\gamma}(A) \leq n^{\gamma-1} * w(A)$. As a result, $\beta = (1 + n^{\gamma-1}(\frac{2}{\alpha-1})^2)(\Delta + 1)$.

Theorem 2 *The time complexity of the SBA algorithm is* $O(n^2)$.

Proof. If an MSA and a SPT are given as the input, the running time of the SBA algorithm is dominated by how many times the $RELAX$ function is executed. The execution time for function $RELAX$ is $O(1)$. When visiting a node u, every edge on the path from r to u (denoted as $P(r, u)$) is relaxed. Then when visiting u's child v, all of these edges need to be relaxed again. It is the same for all of u's children. This will result up to $O(n)$ relaxations. Totally, there are n nodes, therefore, the execution times of function $RELAX$ is $O(n^2)$. □

6 Simulation results

In section 5, we evaluate the SBA algorithm through theoretical analysis. In this section, we evaluate its performance by conducting simulations to measure the total power and the transmission delays of the constructed SBA. We also examine how α will affect the performance of the SBA algorithm.

We conduct the simulations for the networks of sizes from 10 to 150 nodes. All these networks are randomly generated in a fixed 1000*1000 region. For each network size, 200 network instances are investigated and the results averaged. For each edge $e = (u, v)$, we

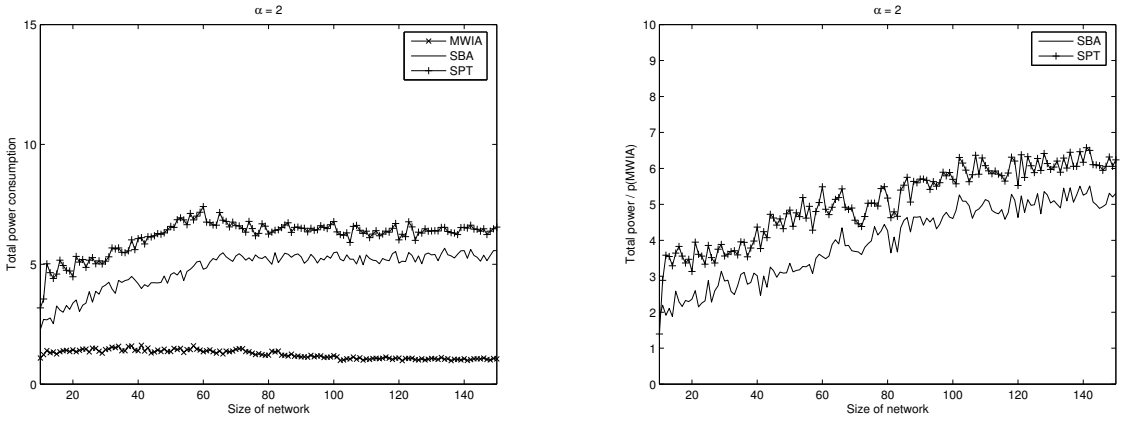
set its weight $w(u, v) = C_v \cdot d_{uv}^{\gamma}$, where d_{uv} is the Euclidean distance between u and v, and γ is fixed to 2, which is a typical value for unobstructed environment, and C_v is a random constant.

For all the simulations, we compare the results of MWIA algorithm, SBA algorithm and SPT algorithm. MWIA algorithm constructs an arborescence with the minimum total power and SPT constructs a tree with the shortest paths between each node and the root. Through the simulation results, we can see how the power and the transmission delays are balanced.

6.1 Simulations for total power consumption

The purpose of this simulation is to evaluate β and β measures the total power consumption. We compare the total powers of a SBA, a MWIA and a SPT. When computing the total power, we take the advantage of WMA which is mentioned in Section 3. Thus the total power is the sum of each node's maximum transmission power. In this simulation, α is set to 2. The results are shown in Figure 4.

Figure 4(a) compares the total power consumptions of the broadcast arborescence constructed by MWIA algorithm, SBA algorithm and SPT algorithm. It is shown that for all the networks, the broadcast arborescence constructed by MWIA has the smallest total power consumption. On average, $p(SPT)$ is 5.07 times of $p(MWIA)$ and $p(SBA)$ is 3.98 times of $p(MWIA)$. The comparison between $\frac{p(SBA)}{p(MWIA)}$ and $\frac{p(SBA)}{p(MWIA)}$ is given in Figure 4(b). From this simulation, it can be seen that SBA has less total power consumption than SPT does. The reason is that for a node u, SBA only adds the path from u to r in the SPT under the



(a) Total power consumption (b) β

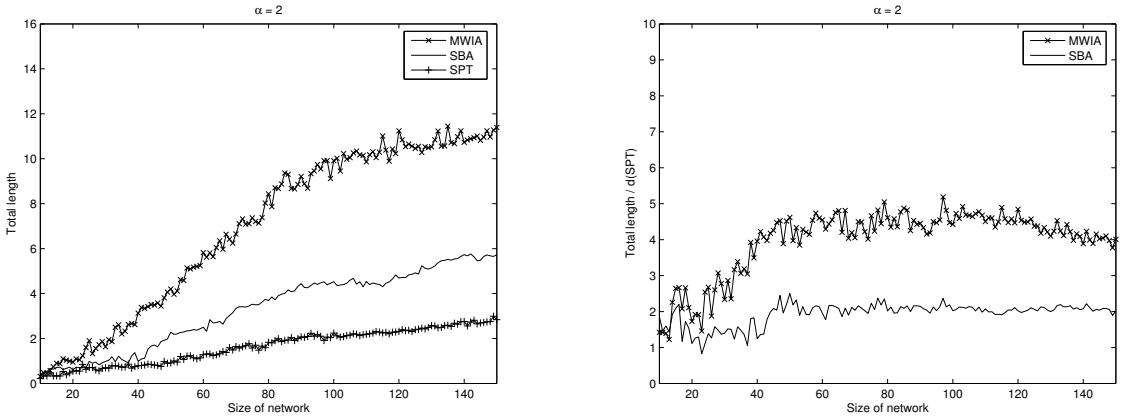
Figure 4: Simulations for total power consumption.

condition that the weight of the path from u to r in the MSA is greater than 2 times of the weight of the one in the SPT. This limits the total power consumption and complies with our expectation. Another observation is that as the size of the network increases, the total power consumption does not increase much. This is because as more and more nodes are employed into the fixed area, two neighboring nodes are getting closer. Therefore, the total power consumption does not increase much.

6.2 Simulations for total length

The purpose of this simulation is to evaluate the total length of the paths in the constructed arborescence since transmission delay is highly related to the length of each path that messages traverse. The SBAT problem requires the strong connectivity, for a node u, both of the path from r to u and the path from u to r need to be evaluated. Note that in a SPT, the path from the root to a node and the path from the node back to the root are the same.

But in MWIA and SBA, these two paths may be different. In this simulation, α is set to 2. The results are shown in Figure 5.



(a) Total length (b) Results compared to SPT

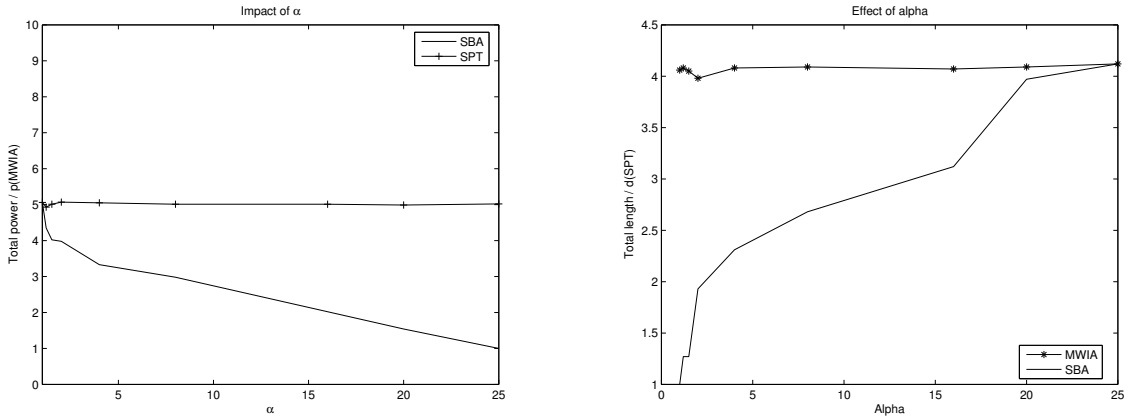
Figure 5: Simulations for transmission delay.

Figure 5(a) compares the total lengths of all the paths from the root to every other node and also the paths from each node to the root in each broadcast arborescence constructed by MWIA algorithm, SBA algorithm and SPT algorithm. It is shown that for all the networks, the broadcast arborescence constructed by SPT has the smallest total length. $d(MWIA)$ is 3.98 times of $d(SPT)$ and $d(SBA)$ is 1.94 times of $d(SPT)$ on average. The comparison between $\frac{d(MWIA)}{d(SPT)}$ and $\frac{d(SBA)}{d(SPT)}$ is illustrated in Figure 5(b).

From Figure 4 and Figure 5, SBA has shown the ability to balance the advantages and shortcomings of MWIA and SPT. This right satisfies the requirement of the SBAT problem to bound both of the total power consumption and the transmission delays of the broadcast arborescence.

6.3 Simulations based on different α

In the above simulations, α is set to 2. In this section, we conduct the simulations with different values of α. We study the relationship between α and the total power and the relationship between α and transmission delays. The results are shown in Figure 6.



(a) Total length (b) Results compared to SPT

Figure 6: Simulations based on different α.

In Figure 6(a), each line represents the ratio of the total power consumption of the constructed broadcast arborescence over $p(MWIA)$. When α is set to 1, SBA constructs a SPT. Therefore, the two lines meet at the same point as shown in Figure 6(a). As α increases, the ratio for SPT remains at around 5. However, the ratio for SBA decreases as α increases and finally, the ratio converges to 1. This is because when α is set to a large number, the transmission delay is not a consideration when constructing a broadcast arborescence and only the total power consumption is the great concern, thus SBA constructs a broadcast arborescence with the smallest total power consumption which is a MWIA.

In Figure 6(b), each line represents the ratio of the total length of the constructed broad-

cast arborescence over $d(SPT)$. When α is set to 1, SBA constructs a SPT. Therefore, the ratio for SBA is equal to 1 as shown in Figure 6(b). As α increases, the ratio for MWIA remains at around 4. However, the ratio for SBA increases as α increases and finally, the two lines meet at the same point. This is because of the decreasing concern of the transmission delays. When α is set to a large number, the transmission delay is not a consideration when constructing a broadcast arborescence and only the total power is the great concern, thus SBA constructs a broadcast arborescence with the smallest total power which is a MWIA.

7 Distributed SBA algorithm

In this section, we introduce the distributed SBA (dSBA) algorithm. dSBA is executed at every node. There are two phases in this algorithm. In the first phase, 3 existing distributed algorithms dSPT [4], dMSA [12] and dDFS [17] are executed to obtain a SPT, a MSA and a DFS traversal order. All of these algorithms are initiated and terminated by a *Manager*. The Manager can be the root node. Then a SBA is constructed in a distributed manner. In the second phase, all the edges are reversed at each node by changing the directions of the edges associated to each node and the operations in phase 1 are conducted again. By reversing the directions of the edges in the obtained SBA, a reverse SBA is generated. The union of the SBA generated in phase 1 and the reverse SBA generated in phase 2 is the expected solution. During the construction of a SBA, we include two messages *REVERSE* and *ADD-FINISH* and a field *initiator* for some messages for the synchronization purpose. Nodes communicate with each other through exchanging the following messages:

• $WAKEUP$. If node u sends this message to node v, this indicates that u has been visited in the DFS order and v becomes the current being visited node.

• ADD. If node u sends this message to node v, this indicates that the path from r to u in the SPT will be added and u is telling its parent v in the SPT this fact. This message should record u as the initiator of this ADDPATH operation.

• $ADD-FINISH$. This message is initiated by the root r after it receives an ADDPATH request and this message is transferred by all the nodes on the being added path. Note that since each ADD message has an initiator, the $ADD-FINISH$ message also needs to have an initiator which is the same as the one in the corresponding ADD message.

• $UPDATE$. If node u sends this message to node v, this indicates that edge (u, v) needs to be RELAXed.

•. $REVERSE$. If a node receives this message, all its outgoing edges become incoming edges and all its incoming edges become outgoing edges.

A distributed algorithm for constructing a single source SPT is proposed in [4]. Both of the time complexity and the message complexity of this algorithm are $O(n^2)$, where n is the number of the nodes in a network. This is an algorithm in parallel computing. The nodes in the network is partitioned into disjoint subsets and the number of subsets corresponds to the number of the available processors. In our case, the number of processors is equal to the number of the nodes in a network since each node has computation capability in wireless networks, therefore, each node is only responsible for itself. Via message exchanging, each processor can decide each node's shortest path to the root. The distributed algorithm is

initiated and terminated by a *Manager*. The Manager can be the root node. By exchanging information only with neighbors, each node u can decide the distance $d_{SPT}(r, u)$ between the root r and itself. After the execution of this algorithm, a SPT is constructed and each node knows its parent, child/children in the SPT and the distance $d_{SPT}(r, u)$. We call this procedure dSPT.

In [12], a distributed algorithm is proposed for constructing a minimum weighted directed spanning arborescence. Both of the time complexity and the message complexity of this algorithm are $O(n)$, where n is the number of the nodes in a network. Their goal is to find n MSAs, one rooted at each node. After the execution of this algorithm, each node knows which edges are part of the MSAs. Here we only need the MSA rooted at the given root node. First employ this algorithm to find a MSA A rooted at the root r. Then reverse the direction of all the edges at each node and the above distributed MSA algorithm is employed again to obtain another MSA A'' rooted at the root r. By changing the direction of each edge at in A'' to its reverse direction at each node, a sink-arborescence-like subgraph A' of the original graph G is generated and this is the reverse MSA of G. We call this procedure dMSA. After the execution of dMSA, each node knows the distances between root r and itself in both MSA A and reverse MSA A', and each node also knows its parents and children in both MSA A and reverse MSA A'.

In [17], a distributed Depth-First-Search (DFS) algorithm is proposed. Both of the time complexity and the message complexity of this algorithm are $O(n^2)$, where n is the number of the nodes in a network. After employing this algorithm, the order of the DFS traversal of

a graph can be decided. Each node knows its predecessor and successor in this order. The algorithm requires that each node has a list including all its adjacent neighbors and a source node to initialize this algorithm. Since we can construct two MSAs as talked above, each node does have the local connectivity information in the two MSAs. If the root r serves as the source node, then the DFS order to traverse each MSA can be decided in a distributed way. We call this procedure dDFS.

To construct a SBA, the dSPT, dMSA and dDFS talked above are executed at each node firstly. Since the initiation and termination of each distributed algorithm is conducted by the root r, all the nodes in the network can be synchronized and these algorithms can be employed in a serialized distributed fashion. A DFS traverse is first conducted on the MSA A according to the order obtained in the dDFS procedure. Each node u uses $d[u]$ to record the current distance between the root r and itself and uses $\pi[u]$ to record its parent in the desired SBA. Initially, $d[u]$ is set to $d_A(r, u)$ for each node u. The root r initiates this traverse by sending out a $WAKEUP$ message to its successor in the order obtained from the dDFS. Then the following operations may be conducted at each node:

- upon receiving a $WAKEUP$ message from node v, a node u becomes the currently visited node. u will first relax the edge (v, u) and update $d[u]$ and $\pi[u]$ accordingly. Then u needs to check if $d[u] > \alpha d_{SPT}(r, u)$. If so, the path from r to u in the SPT, denoted as $P_{SPT}(r, u)$, will be added to A. To add $P_{SPT}(r, u)$, u sends an ADD message to its parent w in the SPT. Here u is the initiator of this ADD message. Then u waits for the reply of this ADD message with initiator u.

• upon receiving an ADD message with initiator i from node u, a node w other than the root node sends an ADD message with initiator i to its parent x in the SPT. At the same time, w records u as a new child.

• upon receiving an ADD message with initiator i from node u, the root node r records u as a new child and sends an $ADD - FINISH$ message back to node u. Since each ADD message includes an initiator, this $ADD - FINISH$ message also needs to include an initiator which is the same as the initiator in the ADD message from node u.

• upon receiving an $ADD - FINISH$ message with initiator i from node w, a node u relaxes the edge (w, u). Note that $d[u]$ and $\pi[u]$ need to be updated if necessary. If u is not equal to i, then u sends an $ADD - FINISH$ with initiator i to the node that has sent u the ADD message with initiator i. Otherwise, u sends a $WAKEUP$ message to its successor obtained from the dDFS. If $\pi[u]$ is changed from x to y, u sends an $UPDATE$ message with initiator u to x.

• upon receiving an $UPDATE$ message from u, a node x will relax the edge (u, x) and update its $d[x]$ and $\pi[x]$ if necessary. Again, if $\pi[x]$ is changed from v to w, x sends an $UPDATE$ message with initiator x to v.

• if the last node u in the order obtained from the dDFS receives an $ADD - FINISH$ message with initiator u, it reverses the directions of all of its associated edges and sends a $REVERSE$ message to its parent in A.

• upon receiving a $REVERSE$ message, each node reverses the directions of all of its associated edges and sends a $REVERSE$ message to its parent in A.

After receiving a $REVERSE$ message and an $UPDATE$ message with initiator u which is marked as the last node in the order obtained from the dDFS, the root r reverses the directions of all of its associated edges and this is the end of the first phase. r then initiates phase 2 by send out a $WAKEUP$ message to its successor in the order obtained from the dDFS. Then the above operations are executed again. The algorithm terminates after the root r receivs an $UPDATE$ message with initiator u which is marked as the last node in the order obtained from the dDFS.

Theorem 3 *The time complexity of the dSBA algorithm is $O(n^2)$, and the message complexity of the dSBA algorithm is $O(n^2)$.*

Proof. The dSBA is executed at each node simultaneously. The execution time is thus dominated by the $ADD-PATH$ procedure. The $ADD-PATH$ function executes $RELAX$ at most once and the execution time for function $RELAX$ is $O(1)$. Therefore, the execution time of function $ADD-PATH$ is proportional to the number of the relaxed edges. The worst case is that the longest path of length $O(n)$ in a network needs to be added. Thus the execution time of function $ADD-PATH$ is $O(n)$. Since the time complexities of dSPT, dMSA and dDFS are all $O(n^2)$, the time complexity of dSBA is $O(n^2)$.

In each of the above operations, each node sends a certain message to just one node. The $WAKEUP$ and $REVERSE$ messages are sent twice at each node. The ADD, $ADD-FINISH$ and $UPDATE$ messages maybe sent up to $O(n)$ times. Thus the message complexity of dSBA is $O(n^2)$. □

If we assume that it takes one unit power to transmit a message, then the message com-

plexity decides the power consumption during the construction procedure which is $O(n^2)$. By Theorem 1, the total power consumption of a broadcast session by employing the constructed broadcast tree is is at most $(1 + n(\frac{2}{\alpha-1})^2)(\Delta + 1)p(opt)$. For a good broadcast tree construction approach, the power consumed for constructing this tree should be less than the total power consumption to conduct a broadcast based on the constructed broadcast tree. Therefore, if $p(opt) \geq O(\frac{n}{\Delta})$, which is reasonable for most cases, it is acceptable to construct an SBA.

8 Conclusion

In this paper, we investigate a new SBAT problem of how to construct a strongly connected broadcast arborescence with bounded total power consumption and transmission delays in wireless networks. We use an (α, β)-arborescence, a novel concept to wireless networks, to design the SBA algorithm with linear running time to solve this SBAT problem. The theoretical analysis and simulation results show that the SBA algorithm gives a proper solution to the SBAT problem. The SBAT problem studies how to construct a single broadcast tree for one source node. We will further investigate the broadcast tree construction problem when multiple source nodes present.

We also propose a distributed version of the SBA algorithm (dSBA) with time complexity of $O(n^2)$ and message complexity of $O(n^2)$. For the dSBA algorithm, only local information is used at each node. However, this algorithm consists of several phases and it is a serialized algorithm, which means that the initiation and termination of each phase needs to be coor-

dinated by a specific manager node. Thus, it is possible that a node may need to wait for a long time to receive such information. Since several arborescence-structures are employed in the dSBA algorithm, it makes the maintenance of the broadcast arborescence difficult. For example, hosts can turn off and on at any time. Once these happen, the arborescence crashes and new arborescence need to be constructed. Mobility is another concern in wireless networks since hosts may move around at any speed. Based on these factors, it is our interest to further develop the maintenance mechanism of the constructed SBA in both centralized and distributed manners. Simulations to evaluate the execution time and memory usage of the distributed algorithm should also be developed. We will also consider to employ this (α, β)-arborescence and the SBA algorithm to solve other problems in wireless networks.

References

[1] M. Agarwal, J. H. Cho, L. Gao and J. Wu, Energy Efficient Broadcast in Wireless Ad hoc Networks with Hitch-hiking, In Proc. IEEE INFOCOM, 2004.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, Wireless sensor networks: a survey, Computer Networks, 38:393-422, 2002.

[3] A. K. Das, R. J. Marks, M. El-Sharkawi, P. Arabshahi and A. Gray, Minimum Power Broadcast Trees for Wireless Networks: Integer Programming Formulations, In Proc. IEEE INFOCOM, 2003.

[4] Lubos Brim, Ivana Cerna, Pavel Krcal and Radek Pelanek, Distributed shortest paths for directed graphs with negative edge lengths, Techmical report FIMU-RS-2001-01, Faculty of Informatics, Masaryk University, http://www.fi.muni.cz/informatics/reports, 2001.

[5] J. Cartigny, D. Simplot and I. Stojmenovic, Localized minimum-energy broadcasting in ad-hoc networks, In Proc. IEEE INFOCOM, 2003.

[6] J.-H. Chang and L. Tassiulas, Energy Conserving Routing in Wireless Ad-hoc Networks, In Proc. IEEE INFOCOM, pp.22-31, 2000.

[7] M. X. Cheng, J. Sun, M. Min, Y. Li and W. Wu, Energy-efficient Broadcast and Multicast Routing in Multihop Ad Hoc Wireless Networks, To appear in Wireless Communications and Mobile Computing (WCMC), 2005.

[8] C. E. Leiserson, R. L. Rivest, T. H. Cormen and C. Stein, Introduction to Algorithms, MIT Press and McGraw-Hill Book Company, 1976.

[9] O. Egecioglu and T. F. Gonzalez, Minimum-energy broadcast in simple graphs with limited node power, In Proc. IASTED international Conference on Parallel and Distributed Computing and Systems (PDCS 2001), Anaheim, CA, pp.334-338, August 2001.

[10] M. L. Fredman and R. E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, Journal of the ACM, 34(3):596-615, 1987.

[11] H. N. Gabow, Z. Galil, T. H. Spencer and R. E. Tarjan, Efficient algorithms for finding minimum spanning trees in undirected and directed graphs, Combinatorica, 6(2):109-122, 1986.

[12] P. A. Humblet, A distributed algorithm for minimum weight directed spanning trees, Rep. LIDS-P-1149-A, Laboratory for Information and Decision Systems, Massachusetts Inst. of Technology, Cambridge, Mass., Sept. 1981.

[13] S. Khuller, B. Raghavachari and N. Young, Balancing minimum spaning and shortest path trees, Algorithmica, 120(4):305-321, 1994.

[14] Y. Li, M. X. Cheng and D.-Z. Du, Optimal Topology Control for Balanced Energy Consumption in Ad Hoc Wireless Networks, Journal of Parallel and Distributed Computing (JPDC), 65(2):124-131, Feb, 2005.

[15] E. L. Lloyd, R. Liu, M. V. Marathe, R. Ramanathan and S. S. Ravi, Algorithmic aspects of topology control problems for ad hoc networks, In Proc. Annual Workshop on Mobile and Ad Hoc Networking and Computing (MobiHoc), Lausanne, Switzerland, 2002.

[16] T. S. Rappaport, Wireless communications: principles and practice, Prentice Hall, 1996.

[17] M. B. Sharma, S. S. Iyengar and N. K. Mandyam, An optimal distributed depth-first-search algorithm, In Proc. of the seventeenth annual ACM conference on Computer science: Computing trends in the 1990's, Louisville, Kentucky, pp.287-294, 1989.

[18] Y. Ben-Shimol, A. Dvir and M. Segal, SPLAST: A novel approach for multicasting in mobile wireless ad hoc networks, manuscript, 2004.

[19] P.-J. Wan, G. Calinescu, X.-Y. Li and O. Frieder, Minimum-energy broadcast routing in static ad hoc wireless networks, In Proc. IEEE INFOCOM, pp.1162-1171, 2001.

[20] 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, pp.585-594, 2000.

[21] J. E. Wieselthier, G. D. Nguyen and A. Ephremides, Algorithms for energy-efficient multicasting in static ad hoc wireless netoworks, In Mobile Networks and Applications (MONET)6, pp.251-263, 2001.

[22] Q. Zhu, M. Parsa and J. J. Garcia-Luna-Aceves, A source-based algorithm for delay-constrained minimum-cost multicasting, In Proc. IEEE INFOCOM, Boston, MA, April 1995.

[23] A. Zygmund, Trigonometric Series (Third Edition), pp.21-22, Cambridge University Press, 2002.