Game-Based Broadcast over Reliable and Unreliable Wireless Links in Wireless Multihop Networks

Fu-Wen Chen, *Student Member*, *IEEE*, and Jung-Chun Kao, *Member*, *IEEE*

Abstract—This paper addresses the minimum transmission broadcast problem in wireless networks and presents efficient solutions, including an optimal broadcast scheme and a distributed game-based algorithm. Distinct from related work in the literature which typically assumes wireless links are reliable, we address the issue of broadcasting over both reliable wireless links and unreliable wireless links. Our main contributions are as follows: We first formulate the minimum transmission broadcast problems over reliable links and over unreliable links as two mixed integer linear programming (MILP) problems, respectively. This way, optimal broadcast schemes can be easily obtained using any existing MILP solver, for small-scale networks. For large-scale networks, we propose a distributed game-based algorithm and prove that the game-based algorithm achieves Nash Equilibrium. Using simulation, we confirm that compared with existing algorithms in the literature and optimal solutions obtained by our MILP techniques, the proposed game-based algorithm performs very well in terms of delivery ratio, the number of transmissions, and convergence speed.

Index Terms—Broadcast, wireless ad hoc networks, mixed integer linear programming, game theory

1 Introduction

Broadcasting, in which a node sends a message to all other nodes in the network, is a common and vital operation in wireless ad hoc networks. Broadcasting is required by many on-demand routing protocols such as AODV [1] in their route discovery processes. Besides, broadcasting is widely used for sending safety messages to nodes over the entire network or a certain region in vehicular ad hoc networks and wireless sensor networks.

Naive broadcast schemes are inefficient in wireless networks [2]. A representative example is flooding, in which each node rebroadcasts a message when receiving that message for the first time. Pure flooding often causes too many unnecessary packet transmissions and may lead to broadcasting storm [3]. To avoid the broadcasting storm problem, a crucial issue is to develop a broadcast scheme with the minimum number of transmissions. This problem is referred to as the minimum transmission broadcast (*MTB*) problem [2].

In the MTB problem, network models, particularly link models, play an important role and may affect performance significantly. There are two fundamental types of link models—the *reliable-link model* and the *unreliable-link model*. Packets transmitted over any *reliable link* are always delivered provided that there is no collision. On the contrary, packets transmitted from one end of an *unreliable link* reach the other end at a *probability*—an unreliable link sometimes delivers packets but sometimes does not. Since wireless links are inherently error-prone due to a number of dynamic factors such as noise, fading and interference, unreliable links are so pervasive in ad hoc radio networks. Although studying the MTB problems for both the models, this paper focuses on providing reliable broadcasting over *unreliable* wireless links.

Most of related work in the literature instead assumes the reliable-link model. Under the reliable-link model, the MTB problem is equivalent to the maximum leaf spanning tree (MLST) problem [4] and the minimum connected dominating set (MCDS) problem. Packets can be optimally broadcast along the constructed spanning tree or connected dominating set (CDS). Because the MLST and MCDS problems have been proven NP-hard [5], a number of approximation algorithms [6], [7], [8] and suboptimal broadcast schemes [9], [10], [11], [12], [13] have been proposed.

The MLST and MCDS problems have been extensively studied [6], [7], [8]. There are some approximation algorithms that can guarantee a constant approximation ratio. One of the best approximation algorithms is the three-approximation algorithm proposed by Lu in [6] for the MLST problem. And in [7], [8], a few distributed algorithms are developed for the MCDS problem. For example, Wan's algorithm [7] is an 8-approximation algorithm.

Among the suboptimal broadcast schemes, Wu and Li [9] proposed a CDS-based broadcast scheme in which a node belongs to the CDS if it has two neighbor nodes unconnected. Only nodes in the CDS need to rebroadcast the broadcast messages. In this scheme, each node requires knowing two-hop neighbor information to determine whether it belongs to the CDS.

Bako et al. [10] use two-hop neighbor information to know the number of potential forwarders, which in turn determines the forwarding probability. Both [9] and [10] require two-hop neighbor information and thus induce high overhead especially in highly dynamic networks, as claimed in [13].

• *The authors are with the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, R.O.C.*
E-mail: richarticle.chen@gmail.com, jungchuk@cs.nthu.edu.tw.

Manuscript received 21 July 2011; revised 24 Feb. 2012; accepted 18 May 2012; published online 31 May 2012.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2011-07-0405. Digital Object Identifier no. 10.1109/TMC.2012.133.

In [11], [12], [13], the knowledge of mere one-hop neighbors is leveraged for efficient broadcasting in mobile ad hoc networks. Whereas getting two-hop (or higher hop) neighbor information incurs higher overhead, one-hop neighbor information can be easily obtained. In Edge Forwarding [11], nodal transmission coverage is divided into six equal-size sectors. Upon receiving a broadcast message, the node will rebroadcast the message after overhearing the channel for a period of time, unless all of its six sectors have forwarders.

In [12], Liu et al. proposed an efficient broadcasting algorithm that achieves the local optimality by selecting the minimum number of forwarding nodes. The key idea is to take into consideration neighbor nodes' transmission coverage. If a neighbor node's coverage can be covered by other neighbor nodes, then the node would not be a forwarding node. Liu's algorithm can significantly reduce the number of forwarding nodes compared to Edge Forwarding, but there is still a room for further performance improvement.

In [13], Khabbazian and Bhargava proposed the responsibility-based scheme (RBS) to reduce the number of transmissions while guaranteeing full delivery under the reliable-link model. RBS is a receiver-based algorithm. In RBS, each node is responsible for rebroadcasting a message to the closest nodes which have not received the message. A node does not need to rebroadcast the message if all its neighbor nodes have already received the message or it is not responsible for any of its neighbor nodes, thus reducing the possibility of two nearby neighbors broadcasting the same message. This avoids most of redundant retransmissions and makes RBS effective, although RBS is simple and does not require knowing two-hop (or higher-hop) neighbor information. Khabbazian and Bhargava [13] show that RBS outperforms Edge Forwarding [11], Liu's algorithm [12], and the CDS-based broadcast scheme [9].

All of the above related work relies on the assumption of links being reliable; however, wireless links are inherently error-prone due to a number of dynamic factors such as noise, fading, and interference. The issue of how to ensure reliable data delivery to intended recipients over unreliable links has been extensively studied. A number of acknowledgment-based (ACK-based) retransmission schemes [14], [15], [16], [17], [18], [19], [20] and collision resolution strategies [21], [22] have been proposed.

Among the ACK-based retransmission schemes, Sheu et al. [16] propose the use of broadcast ACK pattern and backoff ACK window for notification purpose and overhead reduction: A node receiving a broadcast packet will randomly choose one of the minislots in the following DIFS time interval and send the broadcast ACK pattern in the chosen minislot. Lou and Wu [19] use a different approach to reducing acknowledgment overhead, in which ACKs are not sent out at all: Forwarding of broadcast messages at preselected forwarding nodes are regarded as acknowledgments instead; nonforwarding nodes are covered by at least two forwarding nodes to enhance the reliability. Besides, Impett et al. [15] address the problem of ACK implosion. To mitigate the ACK implosion problem, Impett et al. [15] and Banerjee et al. [17] suggest the use of negative acknowledgments.

Collision resolution strategies have also been well studied. For example, Gandhi et al. [21] propose a distributed collision-free broadcasting algorithm in which the transmit times for all nodes are scheduled such that collisions are avoided. In [22], Zhang and Shin propose a physical-layer collision resolution protocol which takes advantage of transmission diversity. The proposed technique can effectively decode overlapping packets in a symbol-level iterative manner, if the packets carry the same data. The idea is similar to the interference cancelation technique proposed in [23] which can significantly improve spatial reuse at a cost of small per-link performance degradation.

Based on reliable data delivery over unreliable wireless links via the concept of ACK-based retransmission and/or collision resolution, a number of efficient broadcast schemes [17], [20] applicable to the unreliable-link model have been proposed. Banerjee et al. [17] develop several centralized heuristics for constructing energy-efficient broadcast/multicast tree that helps decrease transmission energy. A delivery tree is constructed in such a way that the cost of a node takes into account the transmission power, including the first transmission and following retransmissions, consumed by the node to deliver messages to all its children nodes. The number of retransmissions depends on the error probability of outgoing links. In [20], Ros et al. propose a distributed broadcast protocol for vehicular scenarios. The proposed protocol combines a distributed construction of CDS and a neighbor elimination scheme [24] to improve efficiency. However, Ros et al. [20] do not consider link quality and retransmissions explicitly.

The aforementioned broadcast schemes are applicable to the unreliable-link model; however, because they are either centralized solutions or short of taking link quality into consideration, providing efficient broadcast over unreliable wireless links still remains an important research topic. From the theoretical perspective, the MTB problem under the unreliable-link model has not been formulated yet. From a practical point of view, distributed and link-quality-aware algorithms/heuristics designed for unreliable links have not been well investigated. This motivates our studies on providing reliable and efficient broadcasting over reliable and unreliable links. Our contributions include:

- We formulate the MTB problem under the two link models into two mixed integer linear programming (MILP) problems.
- We unify the two MTB problems and propose a unified, distributed algorithm which is a game-theoretic approach.

Game theory is a promising way to solve important problems in wireless networks such as routing [25], [26], coverage [27], channel allocation [28], sensor activation [29], security [30], and so on. However, to our best knowledge, this paper is the first attempt to use game theory to tackle the MTB problems. What follows explains the contributions mentioned above.

To obtain optimal broadcast schemes, we formulate the MTB problems under the reliable-link model and under the unreliable-link model as two MILP problems. This way, optimal broadcast schemes for reliable links and near-optimal broadcast schemes for unreliable links can be obtained by using any off-the-shelf MILP solver in a centralized manner.

To solve the MTB problems in a distributed manner and apply to self-configured wireless networks, we transform the MTB problems into a noncooperative game and

leverage game theory to solve the MTB problems. We prove convergence of the game to a Nash Equilibrium and develop a game-based algorithm that requires one-hop neighbor information only.

The remainder of this paper is organized as follows: Section 2 describes the network model in detail. Section 3 presents our MILP formulations of the MTB problems for reliable links and for unreliable links. Section 4 introduces theoretical and algorithmic parts of our proposed game-based approach. Simulation results are presented in Section 5. Finally, we present some concluding remarks in Section 6.

2 Network Model

We consider a wireless network in which there are a number of nodes. Nodes can be distributed arbitrarily, as long as the network remains connected. The set of all the nodes in the network is denoted by V. Same as [13], it is assumed that each node knows the information of its one-hop neighbors. This can be achieved, for example, by periodic hello messages.

Node u is a neighbor of node v if node u is within the transmission range of node v. For a given node, say node v, $N(v)$ is defined as the set of all its neighbor nodes. For a set of nodes, say S, $N(S)$ is defined as the union of the neighbor nodes of every node in S.

A node always fails to deliver packets to a node outside transmission range, while packet delivery within transmission range succeeds with a probability. For any two nodes u and v, the link reception probability that node v can successfully receive a packet sent from node u is denoted by p_{uv}. If the link quality is better, then p_{uv} is greater. Depending on the value of p_{uv}, the two link models—the reliable-link model and the unreliable-link model—are defined as follows:

- *The reliable-link model*: Any pair of neighbor nodes has a reliable link connecting each other. More precisely, for any two nodes u and v, $p_{uv} = 1$ if node v is a neighbor of node u and $p_{uv} = 0$ otherwise. Under this model, provided that there is no collision, packets transmitted within the transmission range are always delivered.
- *The unreliable-link model*: For simplicity of exposition, we call the model in [17] as the unreliable-link model. In this model, links connecting two nodes are not necessarily reliable. For any two nodes u and v, $0 < p_{uv} \leq 1$ if node v is a neighbor of node u; otherwise, $p_{uv} = 0$. Under this model, packets transmitted from one end of a link reach the other end with a link reception probability of p_{uv}.

A single transmission over an unreliable link may fail. To ensure delivery, a packet transmitted over an unreliable link may be retransmitted several times. For example, if $p_{uv} = 0.25$, then it takes $1/0.25 = 4$ transmissions on average until a packet is successfully delivered. Similarly, if a node intends to broadcast a packet to a subset of its neighbor nodes, it keeps broadcasting the same packet, perhaps several times, until all the intended recipients have received the packet. The expected number of needed transmissions will be modeled later in Section 3.2.

3 Problem Formulation and Solutions Obtained by MILP

The main objective of this section is to formulate the maximum transmission broadcast (MTB) problems for reliable links and for unreliable links as two MILP problems. This way, optimal broadcast schemes for reliable links and near-optimal broadcast schemes for unreliable links can be obtained by using off-the-shelf solvers such as CPLEX [31] and GLPK [32]. Our MILP formulations of the MTB problems under the reliable-link model and under the unreliable-link model are presented in detail in Sections 3.1 and 3.2, respectively.

For the reliable-link model, we generalize/relax some ideas from [33] and derive a MILP formulation with much fewer variables and constraints than what is proposed in [33]. Indeed, our formulation decreases the number of constraints by one order of magnitude and cuts the number of variables by half. Particularly, the number of integer variables is decreased by one order. More precisely, the number of constraints is decreased from $2|V|^2 + |V| - 2$ to $2|V| + 1$, the number of variables is decreased from $2|V|^2 - 2|V| + 1$ to $|V|^2$, and the number of integer variables is decreased from $|V|^2 - |V|$ to $|V|$. This is a significant improvement, considering that linear programming can be solved in polynomial time, whereas MILP is NP-hard.

For the unreliable-link model, to guarantee delivery, the objective function must consider retransmissions and thus is different from the objective function under the reliable-link model. To our best knowledge, we propose the first MILP formulation of the MTB problem under the unreliable-link model.

3.1 MILP Formulation under the Reliable-Link Model

The MTB problem aims to find a broadcast scheme with the minimum number of transmissions while guaranteeing full delivery. Under the reliable-link model, the MTB problem is equivalent to find an MLST, in which internal nodes need to broadcast the message once but leaf nodes do not. So what is left to do is the formulation of the MLST problem into a MILP problem.

To tackle the formulation, we first define some variables:

- Node r is the broadcasting source (the root).
- B_u is a binary variable representing whether node u is an internal node in the broadcast tree. $B_u = 1$ if node u is an internal node and $B_u = 0$ otherwise.
- D_{uv} is the number of downstream nodes of the directed edge (u, v) in the broadcast tree. Note that if the broadcast tree does not contain the edge (u, v), then $D_{uv} = 0$.

The downstream nodes of an edge are the node in the head of the edge and the descendants of that head node. A node is an internal node if its out-degree is at least 1; otherwise, it is a leaf node.

Since the goal is to find an optimal broadcast scheme which can guarantee full delivery, the *rooted spanning graph property* must be satisfied to ensure a constructed graph is a graph rooted at node r and spans over all other nodes. The rooted property corresponds to the fact that the broadcast

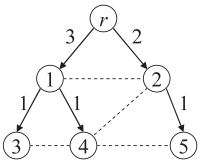

Fig. 1. This is an example satisfying the rooted spanning graph property. Node r is the source node. The number on each edge is the number of downstream nodes of the edge. In this example, nodes r, 1, and 2 are internal nodes, whereas nodes 3, 4, and 5 are leaf nodes.

message originates from the source node and the spanning property ensures full delivery to all the nodes. We denote the graph by $G(V)$, in which the directed edge (u, v) exists if and only if $D_{uv} > 0$. Theorem 1 proves that a graph satisfying the below three constraints is sufficient for its being a rooted spanning graph.

- The source node r must not be a downstream node of any other node, as formally expressed in (5).
- All other nodes are downstream nodes of node r. Therefore, the number of downstream nodes of node r is $|V| - 1$, as formally expressed in (6).
- The numbers of downstream nodes above and below a node other than r must differ by one, as formally expressed in (7), since the difference is that node itself.

Theorem 1. *For any connected graph, the rooted spanning graph property is satisfied if the constraint equations (5)-(7) are satisfied.*

Proof. We need to prove that through the graph $G(V)$, every node in the network is reachable from the source node r. In other words, for every node $v \in V \backslash \{r\}$, there is a directed path in $G(V)$ from node r to v. We prove it by contradiction. Suppose there exists a node u such that there is no directed path from node r to u. Define the set $R = \{v \in V |$ there is a directed path from node v to $u\}$. Obviously, for every node $v \in R$, there is no directed path from node r to v; otherwise, there would exist a directed path from node r to u. We also know that for every node $v \in R, D_{iv} > 0$ only if $i \in R$. With some mathematical manipulation, we have

$$\begin{aligned}\sum_{v\in R}\sum_{i\in N(v)} D_{iv} &= \sum_{v\in R}\sum_{i\in R\cap N(v)} D_{iv}\\ &= \sum_{i\in R}\sum_{v\in R\cap N(i)} D_{vi}\\ &\le \sum_{i\in R}\sum_{v\in R\cap N(R)} D_{vi}\\ &= \sum_{v\in R\cap N(R)}\sum_{i\in R} D_{vi}\\ &= \sum_{v\in R\cap N(R)}\sum_{i\in R\cap N(v)} D_{vi}\\ &\le \sum_{v\in R}\sum_{i\in N(v)} D_{vi},\end{aligned} \tag{1}$$

where the third line holds true because $N(i) \subseteq N(R)$ for any $i \in R$ and the fifth line holds true because $D_{vi} > 0$ only if $i \in N(v)$.

Objective Function:

$$\text{minimize} \sum_{u\in V} B_u \tag{4}$$

Subject to:

$$\sum_{u\in N(r)} D_{ur} = 0 \tag{5}$$

$$\sum_{u\in N(r)} D_{ru} = |V| - 1 \tag{6}$$

$$\sum_{v\in N(u)} D_{vu} - \sum_{v\in N(u)} D_{uv} = 1 \quad \forall u \in V \backslash \{s\} \tag{7}$$

$$B_u \ge \frac{1}{|V|}\sum_{v\in N(u)} D_{uv} \quad \forall u \in V \tag{8}$$

$$D_{uv} \ge 0 \quad \forall u \in V, v \in N(u) \tag{9}$$

$$B_u \in \{0,1\} \quad \forall u \in V \tag{10}$$

Fig. 2. MILP formulation under the reliable-link model.

Let $|R|$ denote the number of nodes in R. Using (7), we have

$$\sum_{v\in R}\left(\sum_{i\in N(v)} D_{iv} - \sum_{i\in N(v)} D_{vi}\right) = |R| \ge 1. \tag{2}$$

However, from (1), we know

$$\sum_{v\in R}\left(\sum_{i\in N(v)} D_{iv} - \sum_{i\in N(v)} D_{vi}\right) \le 0. \tag{3}$$

It is observed that (2) and (3) contradict each other. Therefore, we have proven this theorem. □

Any solution satisfying the above constraints satisfies the rooted spanning graph property, which in turn implies a valid broadcast scheme. On the other hand, for any broadcast scheme with full delivery guarantee, there always exist $\{D_{uv}\}$ satisfying the constraints, where $u \in V$ and $v \in N(u)$. This is obvious for a broadcast tree, because D_{uv} is set to the number of downstream nodes of edge (u, v). For a nontree, it can be proven by pruning some edges of the nontree to a tree, assigning $\{D_{uv}\}$ for the tree, and adding the pruned edges back with zero D_{uv}. Fig. 1 shows an example satisfying the rooted spanning graph property.

In addition to satisfying the rooted spanning graph property, a MLST must have the maximum number of leaf nodes. Equivalently, the objective is to minimize the number of internal nodes. To this end, we first use (8) to distinguish internal nodes from leaf nodes. The binary variable $B_u = 1$ if node u is an internal node and $B_u = 0$ otherwise. The objective function is the sum of all B_us over all nodes and should be minimized as (4) shows. Finally, the MILP formulation under the reliable-link model is shown in Fig. 2.

3.2 MILP Formulation under the Unreliable-Link Model

A single transmission over an unreliable link may fail. To guarantee delivery, a sender (i.e., internal node) needs to take a number of retransmissions until a message successfully reaches all of the intended recipients (i.e., children of the internal node). This can be achieved, for example, by utilizing ACK-based retransmission mechanisms [14], [15], [16], [17], [18], [19], [20]. Taking retransmissions into account, the MTB problem under the unreliable-link model

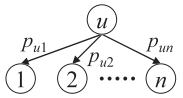

Fig. 3. Node u has n children. The message reception probability from nodes u to i is p_{ui}.

is defined to find the broadcast tree with the minimum *expected* number of transmissions until all the nodes have received the message successfully.

Due to linearity, the expected number of transmissions until all nodes in a broadcast tree have received a message is equal to the sum of the expected numbers of transmissions for each internal node to send the message to its children. Suppose node u is an internal node. As depicted in Fig. 3, we denote the number of node u's children by n and the reception probabilities at its children by p_{ui}, $i = 1, 2, \ldots, n$. Let T_u be the random variable representing the number of transmissions by node u until all its children have received the message. The cumulative distribution function of T_u can be expressed as

$$P(T_u \le t) = \prod_{i=1}^{n} (1 - (1 - p_{ui})^t). \tag{11}$$

$P(T_u \le t)$ can be further simplified and modeled in the same way as what is done in [28], in which high-order terms are ignored assuming $1 - p_{ui}$ is small. Doing so, an approximate $P(T_u \le t)$ can be obtained:

$$P(T_u \le t) \doteq 1 - \sum_{i=1}^{n} (1 - p_{ui})^t. \tag{12}$$

This approximation makes sense in practice. The reason is that links of relatively poor quality are unlikely to appear in an efficient broadcast tree since a poor-quality link incurs a high cost in terms of the number of needed transmissions. So, $\bar{T}_u$ can be modeled as follows:

$$\begin{aligned} \overline{T_u} &= \sum_{t=0}^{\infty} P(T_u > t) \\ &= 1 + \sum_{t=1}^{\infty} (1 - P(T_u \le t)) \\ &\doteq 1 + \sum_{t=1}^{\infty} \sum_{i=1}^{n} (1 - p_{ui})^t \\ &= 1 + \sum_{i=1}^{n} \frac{1 - p_{ui}}{p_{ui}}. \end{aligned} \tag{13}$$

Here, we give (13) an interpretation—1) its first term comes from the fact that node u is an internal node and thus sends out a message at least once, and 2) retransmissions to the children of node u contribute the second term of (13).

Knowing the expected number of transmissions (i.e., $\bar{T}_u$) between a single internal node and its children, the expected number of transmissions of all internal nodes can easily be calculated by summing them together. So, what is left is to identify the internal nodes and the children of each internal node. Since the variable B_u can indicate whether node u is an internal node, we only need to find a way to indicate the children of an internal node. To complete the MTB problem formulation under the unreliable-link model into a MILP problem, we define some new variables for the MILP formulation as follows:

Objective Function:

$$\text{minimize} \sum_{u \in V} \overline{T_u} \tag{14}$$

Subject to:

$$\sum_{u \in N(r)} D_{ur} = 0 \tag{15}$$

$$\sum_{u \in N(r)} D_{ru} = |V| - 1 \tag{16}$$

$$\sum_{v \in N(u)} D_{vu} - \sum_{v \in N(u)} D_{uv} = 1 \quad \forall u \in V \setminus \{r\} \tag{17}$$

$$E_{uv} \ge \frac{1}{|V|} D_{uv} \quad \forall u, v \in V \tag{18}$$

$$B_u \ge \frac{1}{|V|} \sum_{v \in N(u)} D_{uv} \quad \forall u \in V \tag{19}$$

$$\overline{T_u} = B_u + \sum_{v \in N(u)} \frac{1 - p_{uv}}{p_{uv}} E_{uv} \quad \forall u \in V \tag{20}$$

$$D_{uv} \ge 0 \quad \forall u \in V, v \in N(u) \tag{21}$$

$$E_{uv} \in \{0,1\} \quad \forall u \in V, v \in N(u) \tag{22}$$

$$B_u \in \{0,1\} \quad \forall u \in V \tag{23}$$

Fig. 4. MILP formulation under the unreliable-link model.

- $\bar{T}_u$ is the expected number of transmissions by node u. If node u is an internal node, $\bar{T}_u$ is modeled using (13); otherwise, $\bar{T}_u = 0$.
- E_{uv} is a binary variable. $E_{uv} = 1$ if node v is a child of node u in the broadcast tree and $E_{uv} = 0$ otherwise.

The MILP formulation for unreliable links is listed in Fig. 4. The objective function is to minimize the expected number of transmissions of all nodes, as shown in (14). Equations (15), (16), (17), and (19) are the same as what are described for the reliable-link model in Section 3.1. $D_{uv} \neq 0$ implies that node v is a child of node u, which in turns implies $E_{uv} = 1$, as described in (18). If a node u is an internal node, then $B_u = 1$ and (20) degrades to (13). Otherwise, $B_u = E_{uv} = 0$ and therefore $\bar{T}_u$ degrades to zero.

4 Game-Based Broadcast Tree Construction

Contrary to the optimal yet centralized solutions described in Section 3, this section presents a fully distributed approach: A unified model, called the *broadcast tree construction game*, is proposed to tackle the broadcast problems for both reliable links and unreliable links. A distributed algorithm, called the game-based broadcast tree construction (GB-BTC) algorithm, is developed, based on the noncooperative game. GB-BTC merely needs information about one-hop neighbors, thus causing practically negligible communication overhead. This section also proves that there exists at least one Nash Equilibrium in the broadcast tree construction game.

4.1 Broadcast Tree Construction Game

Since the MTB problems (under the reliable-link model and the unreliable-link model) are NP-hard, it is hard to find a solution with optimal result in polynomial execution time. Instead, we model a broadcast tree construction game to construct a distributed algorithm that solves the MTB problems with suboptimal result and fast convergence speed.

In game theory, a game consists of *players*, a set of actions (or *strategies*) available to the players, and a specification of utility functions (or *payoffs*) for all combinations of strategies. The total payoff is defined as the sum of payoffs to all players. In a *noncooperative* game, each player independently chooses the strategy maximizing its own payoff. Below, we explain the idea of modeling the MTB problems (for both reliable links and unreliable links) as a noncooperative game, the broadcast tree construction game.

Under the reliable-link model, the MTB problem is equivalent to the MLST problem, which aims to minimize the number of internal nodes. Regard the number of internal nodes as the total cost to be minimized. The total cost is added up by the costs of internal nodes; each internal node brings a unit cost. Imagine that such a unit cost is paid by the children nodes rather than the internal node itself. Provided that an internal node has c children, each child contributes a cost of $1/c$, or equivalently a payoff of $-1/c$. To be fully distributed, each nonroot node determines its parent independently to maximize its payoff.

For the unreliable-link model, the objective of the MTB problem is to minimize the cost—the sum of the expected numbers of transmissions for all the internal nodes. The cost contributed by an internal node can be computed using (13): Provided that an internal node u has n children, each child, say node v, shares the cost brought by its parent and contributes a cost of $1/c + (1 - p_{uv})/p_{uv}$, or a payoff of $-1/c - (1 - p_{uv})/p_{uv}$. Again, each nonroot node determines its parent by itself to maximize its own payoff.

Indeed, the reliable-link model can be regarded as a special case of the unreliable-link model in which $p_{uv} = 1$ for any reliable link $\overline{uv}$. From the above point of view, the MTB problem can be treated as a repeated and noncooperative game. In the game, all nodes except the source node are players, determining their parents (one parent for a node) selfishly such that their own payoffs are maximized.[1]

There are $|V| - 1$ players in total. Assume that players are enumerated and the set of players is denoted by P. Any player, say player i, adopts a strategy $s_i \in N(i)$ to determine its *parent*. The strategies of all players make a strategy profile $s = (s_1, s_2, \ldots, s_{|V|-1})$, and we denote the strategies of all players except player i by s_{-i}. For a strategy profile S, the number of children of node v is denoted by $c(v, s)$. We assume that each player knows the number of children of its neighbors through one-hop broadcasting when parent-children relationships change. According to the above argument, the payoff of player i is a function of s, denoted by the utility function $u_i(s)$:

$$u_i(s) = -\left(\frac{1}{c(s_i, s)} + \frac{1 - p_{s_i i}}{p_{s_i i}}\right). \tag{24}$$

Our objective function is the total payoff:

$$\text{maximize} \sum_{i \in P} u_i(s). \tag{25}$$

The broadcast tree construction game is a repeated game; each player chooses the *best response* in all iterations. In an iteration, each player considers the broadcast tree constructed in last iteration and chooses a neighbor to be its parent that maximizes its payoff. More precisely, given s_{-i} in last iteration, player i chooses the best strategy s_i^* such that $u_i(s_i^*, s_{-i})$ is maximized. The repeated game continues until no player can increase its payoff by changing only its own strategy.

Note that although players do not have conflict of interest in the broadcast tree construction game, players are encouraged to share the cost such that the total cost can be minimized. This is similar to the well-known global connection game (see [34, Chapter 19.3]). The global connection game is a cost-sharing game in which the cost of an edge is shared evenly by all players whose paths contain that edge. The broadcast tree construction game can be regarded as a generalization of the global connection game in the sense that both edge cost and node cost are encouraged to be shared.

4.2 Convergence

In general, there is no guarantee that best response converges to a stable state. Indeed, it is nontrivial whether best response converges in the broadcast tree construction game. This is because it is possible that during some iteration, a player increases its individual payoff by selfishly choosing a new parent but the total payoff adversely gets decreased.[2] In this section, we prove that the best response in the broadcast tree construction game eventually converges to the famous stable state, Nash Equilibrium.

Stated simply, players are in a Nash Equilibrium if the game can converge to a stable state in which no player can increase its payoff by changing only its own strategy unilaterally, while the other players keep theirs unchanged. Specifically, a strategy profile s^* is called a Nash Equilibrium if and only if the following inequality always holds true for each player i and any strategy s_i:

$$u_i(s^*) \geq u_i\big(s_i, s_{-i}^*\big). \tag{26}$$

To analyze the equilibrium property of the broadcast tree construction game, we first prove that the broadcast tree construction game is an *exact potential game* [35]. In game theory, a game is considered a *potential game* if the incentive of individual players to change their strategy can be expressed in one global function, the *potential function*. That is, the difference in individual payoffs for each player from individually changing one's strategy has the same (positive or negative) sign as the difference in values for the potential function. Specifically, if the differences have the same value, the game is an exact potential game.

Before proving the broadcast tree construction game is an exact potential game, we first explain the social meaning of the potential function. In the broadcast tree construction game, the total payoff or *social payoff* is the sum of all players' payoff which turns out being equal to the (expected) total number of transmissions until a broadcast message has successfully been delivered to all the nodes in the network. While the social payoff in an equilibrium reflects the efficiency of this equilibrium, the potential function reflects

1. Provided that there exists no directed cycle, the broadcast tree construction game guarantees to construct a rooted tree spanning over all reachable nodes. The algorithm that avoids forming cycles is discussed later in Section 4.3.

2. An example is that a player switches its parent from one internal node to another with more children and a slightly worse link connectivity to the player. In this situation, the individual payoff increases because the new parent has more children. However, the total payoff does not benefit at all by the switching because both the old and new parents remain internal nodes. Instead, the total payoff decreases due to the use of a worse link.

the difference in the social payoff for each player from individually changing one's strategy. Whenever a player changes its own strategy which turns out increasing (or decreasing) the social payoff, the potential function increases (or decreases) by the same amount.

The following theorem proves the broadcast tree construction game is an exact potential game.

Theorem 2. *The broadcast tree construction game is an exact potential game.*

Proof. To prove that the broadcast tree construction game is an exact potential game, we attempt to construct a potential function that takes both node cost and edge cost into consideration and then show that the change of any player's payoff is exactly the change of the potential function. The potential function is constructed as

$$\varphi(s) = -\left(\sum_{i\in V} H_{c(i,s)} + \sum_{i\in P} \frac{1-p_{s_i i}}{p_{s_i i}}\right), \tag{27}$$

where $H_n = \Sigma_{k=1}^{n} 1/k$ is the nth harmonic number. The edge cost is dealt with in the second term of (27), which simply sums up all edge costs involved in the broadcast tree. The node cost is dealt with in the first term of (27), using a harmonic number to cope with the node cost (which is either 0 or 1) shared by the children nodes.

Without loss of generality, we assume that in some iteration, player i changes its own strategy from s_i^1 to s_i^2. The strategy profile changes from $s^1 = (s_i^1, s_{-i})$ to $s^2 = (s_i^2, s_{-i})$. Because player i leaves the old parent s_i^1 and joins the new parent s_i^2, node s_i^1 loses a child, node s_i^2 gets a new child, and the parent-children relationships of other nodes are not affected. So, we know:

$$c\left(s_i^1, s^2\right) = c\left(s_i^1, s^1\right) - 1, \tag{28}$$

$$c\left(s_i^2, s^2\right) = c\left(s_i^2, s^1\right) + 1, \tag{29}$$

$$c(v, s^2) = c(v, s^1) \;\; \forall v \in V \backslash \left\{s_i^1, s_i^2\right\}. \tag{30}$$

Now, we prove that the change in player i's individual payoff is exactly equal to the change in values of the potential function as follows:

$$\begin{aligned}
&u_i(s^2) - u_i(s^1) = u_i\left(s_i^2, s_{-i}\right) - u_i\left(s_i^1, s_{-i}\right)\\
&= -\left(\frac{1}{c\left(s_i^2, s^2\right)} + \frac{1-p_{s_i^2 i}}{p_{s_i^2 i}}\right) + \left(\frac{1}{c\left(s_i^1, s^1\right)} + \frac{1-p_{s_i^1 i}}{p_{s_i^1 i}}\right)\\
&= -\left(H_{c(s_i^2,s^2)} - H_{c(s_i^2,s^2)-1} + \frac{1-p_{s_i^2 i}}{p_{s_i^2 i}}\right) + \left(H_{c(s_i^1,s^1)} - H_{c(s_i^1,s^1)-1} + \frac{1-p_{s_i^1 i}}{p_{s_i^1 i}}\right)\\
&= -\left(H_{c(s_i^2,s^2)} - H_{c(s_i^2,s^1)} + \frac{1-p_{s_i^2 i}}{p_{s_i^2 i}}\right) + \left(H_{c(s_i^1,s^1)} - H_{c(s_i^1,s^2)} + \frac{1-p_{s_i^1 i}}{p_{s_i^1 i}}\right)\\
&= -\left(\sum_{k\in\{s_i^1,s_i^2\}} H_{c(k,s^2)} + \frac{1-p_{s_i^2 i}}{p_{s_i^2 i}}\right) + \left(\sum_{k\in\{s_i^1,s_i^2\}} H_{c(k,s^1)} + \frac{1-p_{s_i^1 i}}{p_{s_i^1 i}}\right)\\
&= -\left(\sum_{k\in V} H_{c(k,s^2)} + \sum_{k\in P} \frac{1-p_{s_k^2 k}}{p_{s_k^2 k}}\right) + \left(\sum_{k\in V} H_{c(k,s^1)} + \sum_{k\in P} \frac{1-p_{s_k^1 k}}{p_{s_k^1 k}}\right)\\
&= \varphi(s^2) - \varphi(s^1),
\end{aligned} \tag{31}$$

where the fourth equality holds true because of (28) and (29), and the sixth equality holds true because of (30).

So far, we have proven that in the broadcast tree construction game, the change of any player's payoff is equal to the change of the potential function. Hence, the broadcast tree construction game is an exact potential game. □

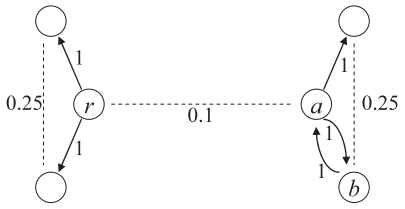

Fig. 5. This figure illustrates that without knowing the ancestor/descendant relationship, an *invalid* broadcast tree could be constructed in the broadcast tree construction game. In this figure, the solid arrows depict an invalid broadcast tree constructed in the game and the dotted lines are links not adopted in the game. The numbers along links are the link reception probabilities.

Since the broadcast tree construction game is a repeated game with an exact potential function, according to Rosenthal's well-known result [36], the best response dynamics in the broadcast tree construction game converges to a Nash Equilibrium. This proves the following theorem:

Theorem 3. *The broadcast tree construction game always converges to a Nash Equilibrium.*

4.3 Game-Based Broadcast Tree Construction Algorithm (GB-BTC)

In the last section, we have proven the convergence to a Nash Equilibrium in the broadcast tree construction game. However, without considering ancestor/descendant relationship and imposing corresponding constraints, the constructed broadcast tree may be invalid because it may contain undesired (directed) cycles and end up being disconnected or nontree.

Fig. 5 shows such an illustrative example in which node r is the source node (i.e., the root) of all broadcast messages. In this example, node a is supposed to choose node r to be its parent because there is only one possible path connecting nodes r and node a. However, without considering their relationship, node a would mistakenly select node b, one of its descendants, to be its parent in the game due to a higher payoff. As a result, a directed cycle is formed, causing the constructed graph (drawn in solid arrows) disconnected and nontree.

To solve the problem of causing directed cycles, each player must avoid choosing any of its descendant nodes as the strategy. However, maintaining and disseminating ancestor/descendant relationship could make a game-based algorithm unfavorably centralized. To be able to prevent a player from mistakenly choosing one of its descendants as its parent in a distributed approach, we develop a fully distributed algorithm, called the GB-BTC algorithm. GB-BTC prevents directed cycles from being formed by exploiting a local metric, called *rank*, (rather than using the ancestor/descendant relationship) and by setting some rank-based constraints for strategy selection.

4.3.1 Rank-Based Constraints

Each node/player has a *rank* attribute which is defined as the least hop count to the source node. The rank of node i is

denoted by $r(i)$. Ranks are computed locally and the values are exchanged in a one-hop and distributed manner: Initially, the rank of the source node is set to zero. Each node puts its own rank value into hello messages and disseminates hello messages one-hop away. After overhearing hello messages sent from neighboring nodes, each node updates its rank value to be the lowest rank of its neighbors plus one. The rank of the player's parent is also carried in hello messages and disseminated one-hop away.

There are multiple possible ways in which rank can be used to prevent a player from mistakenly selecting its descendants to be its parent. Although the simplest way is to force the rank of a player higher than the rank of its parent, this does not provide a satisfactory performance. It results in performance degradation because possible strategies become very limited. To solve the (directed) cycle problem while providing a satisfactory performance, GB-BTC uses rank-based constraints instead: A player, say player i, chooses its parent s_i to maximize its payoff, subject to the following rank-based constraints:

- The rank of its parent s_i cannot be higher than the rank of player i.
- If the rank of s_i is equal to the rank of s_i's parent, then the rank of s_i must be lower than the rank of player i.

Under the first constraint merely, it is possible that there still exists a directed cycle in which the ranks of all players are the same. To prevent such cycles, we apply the second constraint to make the rank increase within two hops. These constraints can also prevent long paths, which will increase the delay. With these two constraints, players need only one-hop neighborhood information to prevent cycles, and our proposed game-based algorithm, GB-BTC, can be run in a fully distributed manner.

4.3.2 GB-BTC Algorithm

Now, we describe the proposed GB-BTC algorithm. At beginning, a source node initiates a breadth-first traversal to establish an initial tree. In the breath-first traversal process, each node in the tree also computes the value of its rank, in the way described previously. After this process, each player's rank ends up being equal to its parent's rank plus one.

Then, each player can change its strategy (i.e., parent) to maximize its payoff under the constraints previously described. In the repeated game, if a player wants to change its strategy, it needs to send messages to inform both the old and new parents. In addition, the old and new parents need to broadcast HELLO messages immediately to inform their neighbors about the change of the number of their children. The repeated game keeps going until no player can increase its payoff by changing only its own strategy.

The pseudocode for the GB-BTC algorithm is summarized in Fig. 6, and we assume HELLO messages one-hop away, periodically and event triggered. The HELLO messages of a node, say node i, include the following information:

- s_i: Its strategy (i.e., its parent),
- $r(i)$: Its rank,
- $r(s_i)$: The rank of its parent,
- $c(i)$: The number of its children.

Both $r(i)$ and $r(s_i)$ are included because nodes need the information to check the rank-based constraints. $c(i)$ is included so that nodes can compute their utility functions and choose the best strategies.

Algorithm: GB-BTC algorithm for a given source node, r.

```
Node r initiates a breadth-first traversal to establish an initial broadcast tree and to compute the rank values.
while not converged do
  for each node i do
    if node i receives HELLO then
      update the neighbor information
    else if node i receives LEAVE from any neighbor v then
      c(i) ← c(i) − 1
      node i broadcast HELLO
    else if i receives JOIN from any neighbor v then
      c(i) ← c(i) + 1
      node i broadcast HELLO
    end if
    UpdateStrategy(i)
  end for
end while
// The final strategy profile s* is a Nash Equilibrium.
```

Procedure: UpdateStrategy(i)

```
u ← s_i
// Find a new parent that gives the highest payoff.
for each node v in N(i) do
  if ( u_i(v, s_-i) > u_i(u, s_-i) ) then
    // Make sure the new parent satisfies the rank-based constraints.
    if ( r(v) < r(i)  or  (r(v) = r(i) and r(s_v) < r(v)) ) then
      u ← v
    end if
  end if
end for
if u ≠ s_i then
  node i send LEAVE to node s_i
  node i send JOIN to node u
  s_i ← u
  node i broadcast HELLO
end if
```

Fig. 6. Pseudocode for the GB-BTC algorithm.

5 Performance Evaluation

In this section, we evaluate our proposed schemes under both the reliable-link and unreliable-link models, using an in-house simulator developed in C++. In particular, we compare the performance of our proposed GB-BTC algorithm with several heuristics/algorithms existing in the literature and the optimal solutions obtained by using our proposed MILP techniques. In addition, we evaluate the convergence speed of the GB-BTC algorithm through simulations.

The performance metrics of interest include the number of transmissions and the delivery ratio. Provided that the delivery ratio is the same, the number of transmissions is a measure of the efficiency of broadcast algorithms. The smaller the number of transmissions is, the more efficient the algorithm (or equivalently the constructed broadcast tree) is. Under the reliable-link model, the number of transmissions is equal to the number of internal nodes in a constructed broadcast tree. This is because transmissions within the transmission range never fail and leaf nodes do not forward messages. Under the unreliable-link model, a node may need to broadcast the same message multiple times to ensure delivery to all of the intended recipients among its neighbors. There is a tradeoff between the number of transmissions and the delivery ratio.

TABLE 1
Simulation Parameters Used for the Reliable-Link Model

Parameter	Value
Number of nodes	50
Node density	40-200 nodes/km^2
Transmission range	200m
Collision model	collision-free

The delivery ratio is defined as the percentage of nodes which receive the broadcast message. Under the reliable-link model, the delivery ratio for any algorithm is always 100 percent, as long as the broadcast scheme spans over all nodes in the network. Under the unreliable-link model, the delivery ratio is an indicator of reliability. The larger the delivery ratio, the more reliable the broadcast scheme.

Note that Section 5.1 shows simulation results in terms of the number of transmissions, but not the delivery ratio. This is because under the reliable-link model, 100 percent delivery ratio is guaranteed in all of the compared algorithms. For the unreliable-link model, simulation results in terms of both the delivery ratio and the number of transmissions are presented in Section 5.2.

5.1 Simulations for the Reliable-Link Model

The first set of simulations is for the reliable-link model. In this simulation setup, we consider a collision-free environment where packets reach everywhere within a fixed transmission radius and there is no collisions. In each simulation run, 50 nodes are distributed uniformly over a square region with a node density ranging from 40 to 200 nodes/km^2. The node density is controlled by adjusting the area of the square region. Each simulation result (i.e., a point in a figure) is averaged over 100 instances. Important simulation parameters are listed in Table 1.

We compare our schemes (MILP and GB-BTC) with three existing schemes—Lu's, Wan's, and RBS schemes. The Lu's algorithm [6] is a centralized three-approximation algorithm designed for the MLST problem. The Wan's algorithm [7] is a distributed eight-approximation algorithm for the (equivalent) MCDS problem. The third scheme, RBS [13], is a distributed and efficient broadcast scheme using only one-hop neighbor information.

As shown in Fig. 7, our proposed MILP method achieves the best performance. This is because MILP guarantees optimality in terms of the number of transmissions. Although MILP is a centralized scheme and thus is not very suitable to apply to large-scale wireless networks in practice, it can be treated as a performance bound when a number of extra constraints are imposed in reality.

As can be seen in Fig. 7, the performance of our proposed GB-BTC algorithm is closest to the optimal performance and thus is better than the three existing algorithms (Lu's, Wan's, and RBS). Among Lu's, Wan's, and RBS algorithms, RBS performs worst in this simulation setup. However, one advantage of RBS is the lowest overhead, since RBS needs neither tree construction nor global information.

5.2 Simulations for the Unreliable-Link Model

For the unreliable-link model, we run simulations by adding a random noise and considering the effect of propagation loss. The additive white Gaussian noise is

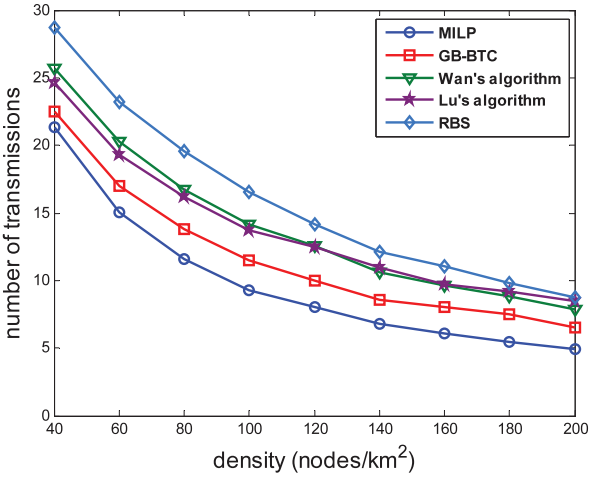

Fig. 7. The number of transmissions versus various node densities, under the reliable-link model.

added to the wireless channel. The propagation loss is modeled using a free space model and the received signal power in free space at distance d from the transmitter is

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L},$$

where P_t is the transmitted signal power. G_t and G_r are the antenna gains of transmitter and receiver, respectively. L is the system loss, and λ is the wavelength. It is common to select $G_t = G_r = 1$ and $L = 1$. Packets are transmitted at a data rate of 2 Mbps using the BPSK modulation. Bit error rate (BER) of BPSK can be computed using the well-known formula:

$$BER = Q\left(\sqrt{2E_b/N_0}\right),$$

where E_b is received signal energy per bit and N_0 is the noise power spectral density. With BER and packet length L, the link reception probability that node v can successfully receive a packet sent from node u is $p_{uv} = (1 - BER)^L$. Important simulation parameters are listed in Table 2. In such an environment, a longer distance to receiver implies a lower reception probability.

In this simulation setup, our goal is to evaluate RBS, flooding, and our proposed schemes (MILP and GB-BTC) in terms of both delivery ratio and the number of transmissions. Lu's and Wan's algorithms are not evaluated in this

TABLE 2
Simulation Parameters Used for the Unreliable-Link Model

Parameter	Value
Number of nodes	35
Node density	20-160 nodes/km^2
Propagation model	free space model
Carrier frequency	2.4 GHz
Collision model	collision-free
Noise model	AWGN
Noise power spectral density N_0	-140 dBm/Hz
Transmit power P_t	0.1 Watts
Data rate	2 Mbps
Modulation	BPSK
Packet length	1000 bits

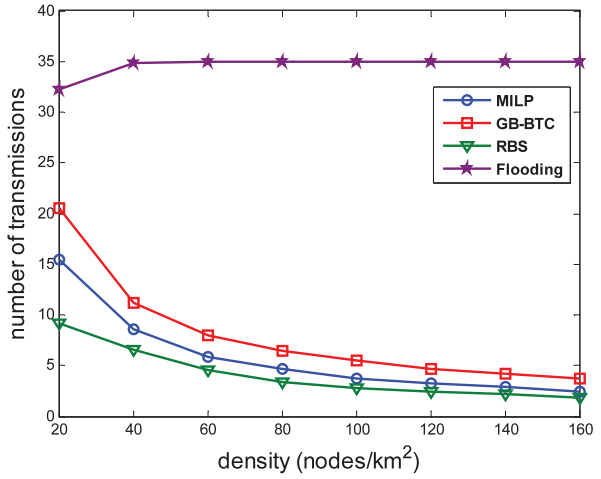

Fig. 8. The number of transmissions versus various node densities, under the unreliable-link model. (Nodes are uniformly distributed.)

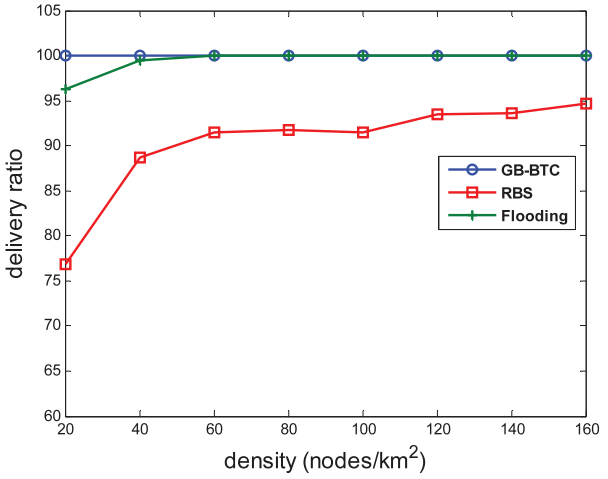

Fig. 10. The delivery ratio versus various node densities, under the unreliable-link model. (Nodes are distributed in accordance with a 2D Gaussian distribution.)

simulation setup because they are not designed for the unreliable model.

We first investigate the number of transmissions of our proposed GB-BTC and MILP algorithms. Both of them guarantee full delivery. As shown in Fig. 8, GB-BTC performs very well because this fully distributed algorithm only takes slightly more transmissions compared to the centralized MILP technique. In other words, the GB-BTC algorithm can construct efficient broadcast trees in a distributed fashion.

Although it can be observed in Fig. 8 that RBS takes fewer transmissions than GB-BTC and MILP do, RBS does not really outperform GB-BTC and MILP. This is because GB-BTC and MILP guarantee full delivery, but RBS does not. For GB-BTC and MILP, internal nodes in the constructed broadcast tree may transmit the same broadcast message multiple times until all of their children nodes have received the message. On the contrary, for RBS, some nodes might not receive broadcast messages and end up being disconnected, since each node forwards the received message at most once. The lower the delivery ratio, the smaller the number of transmissions tends to be. As shown in Fig. 9, RBS cannot achieve 100 percent delivery ratio. Particularly, when the node density is small, the delivery ratio of RBS degrades significantly. On the contrary, flooding can achieve nearly 100 percent delivery ratio but the number of redundant transmissions is extremely high.

We extend the simulation to consider one more scenario. We change the node distribution from a uniform distribution to a 2D-Gaussian distribution. For the 2D-Gaussian distribution, the x-coordinates and y-coordinates of all nodes are both Gaussian random variables with a mean equal to half the edge of the square region and with a standard deviation equal to quarter the edge of the square region. In this simulation setup, the node population is denser around the center and sparser around the edges. Compared to the scenario of uniform distribution, the delivery ratio of RBS and flooding in this 2D-Gaussian distribution scenario becomes higher, as shown in Fig. 10, since more nodes are spread around the center. For the same reason, the average number of transmissions becomes lower. The performance of GB-BTC algorithm is still close to that of MILP as shown in Fig. 11.

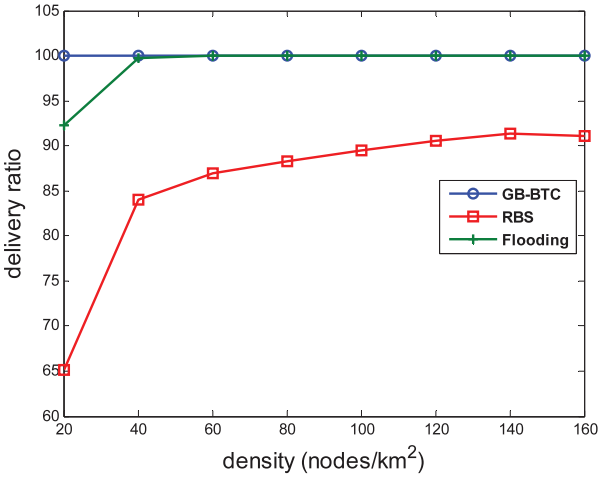

Fig. 9. The delivery ratio versus various node densities under the unreliable-link model. (Nodes are uniformly distributed.) The result for MILP is not shown here because it achieves 100 percent delivery ratio as the GB-BTC algorithm does.

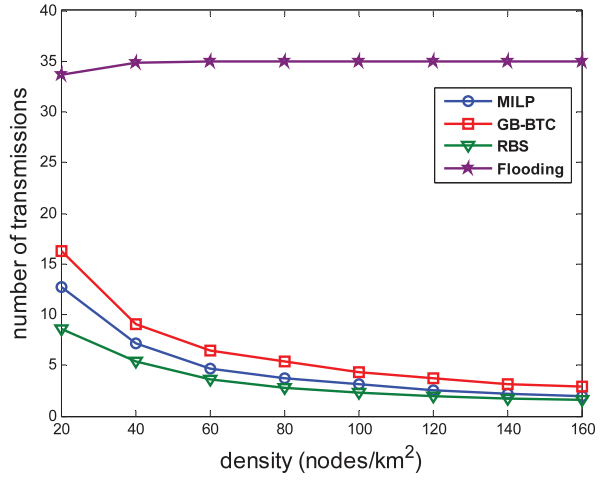

Fig. 11. The number of transmissions versus various node densities, under the unreliable-link model. (Nodes are distributed in accordance with a 2D Gaussian distribution.)

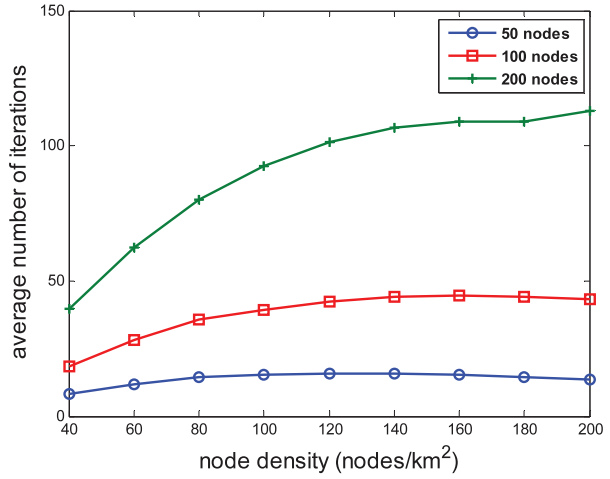

Fig. 12. The average number of iterations versus various node densities, under the reliable-link model. The result shows that each player needs to change its strategy about 0.5 times.

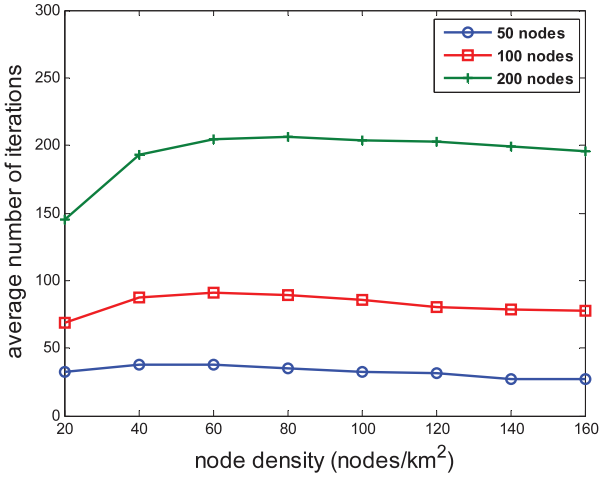

Fig. 13. The average number of iterations versus various node densities, under the unreliable-link model. The result shows that each player needs to change its strategy about one time.

5.3 Analysis of the Convergence Speed

The GB-BTC algorithm has been proven in Section 4 to converge to a Nash Equilibrium in a finite number of iterations. In this section, we run simulations to analyze the convergence speed. The GB-BTC algorithm starts with an initial tree built by using a breadth-first traversal in a distributed manner; after that, each player in each iteration attempts to maximize its payoff selfishly. To analyze the convergence speed, we restrict that only one player can change its strategy in one iteration, and the convergence speed is defined as to the inverse of the average number of iterations until a state of convergence is achieved. The more the number of iterations, the slower the convergence speed.

For both the reliable-link and unreliable-link models, we run the simulations to learn the convergence speeds. Both the number of nodes and the node density vary in this set of simulations. For the reliable-link model, as shown in Fig. 12, the average number of iterations is roughly equal to half number of nodes; that is, each player only needs to change its strategy 0.5 times on average. This implies a fast convergence speed. For the unreliable-link model, the average number of iterations is roughly the same as the number of nodes as shown in Fig. 13. This number is roughly twice as many as the number of transmissions under the reliable-link model. The reason that it takes more iterations in the unreliable-link case is because under the unreliable-link model, the total payoff may be decreased after a player changes its strategy selfishly. Nevertheless, each player only needs to change its strategy once on average—it is still a fast convergence speed.

6 Conclusion

In this paper, we have addressed the minimum transmission broadcast (MTB) problems under the reliable-link model and under the unreliable-link model. Under the reliable-link model, the MTB problem is formulated as a MILP problem, with only few variables and constraints. Moreover, we tackle the error-prone nature of wireless links and provide the first MILP formulation of the MTB problem under the unreliable-link model. Having the MILP formulations, optimal broadcast schemes can be obtained using any existing MILP solver in a centralized manner and can be treated as a performance bound.

To solve the MTB problems in a fully distributed manner, we have also developed a game-based algorithm. The MTB problems for reliable links and unreliable links are consolidated—modeling as a broadcast tree construction game. A fully distributed algorithm, the GB-BTC algorithm, is developed based on the game. We have proven that the broadcast tree construction game converges to a Nash Equilibrium in a finite number of iterations. Simulation results show that our proposed GB-BTC algorithm performs very well in terms of both delivery ratio and the number of transmissions; meanwhile, the convergence speed is very fast.

Acknowledgments

The authors thank the anonymous reviewers for their many helpful suggestions. This research was supported in part by National Science Council, Taiwan, under grants NSC 98-18-E-007-007, NSC 98-2219-E-007-012, NSC 99-2219-E-007-005, and NSC 100-2219-E-007-003.

References

[1] C.E. Perkins and E.M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," *Proc. Second IEEE Workshop Mobile Computing Systems and Applications (WMCSA)*, pp. 90-100, 1999.

[2] J. Hong, W. Li, S. Lu, J. Cao, and D. Chen, "Sleeping Schedule Aware Minimum Transmission Broadcast in Wireless Ad Hoc Networks," *Proc. 14th IEEE Int'l Conf. Parallel and Distributed Systems (ICPADS)*, pp. 399-406, Dec. 2008.

[3] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," *Proc. ACM MobiCom*, pp. 151-162, Aug. 1999.

[4] T. Fujie, "An Exact Algorithm for the Maximum Leaf Spanning Tree Problem," *Computers Operations Research*, vol. 30, pp. 1931-1944, Nov. 2003.

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[6] H.-I. Lu and R. Ravi, "Approximating Maximum Leaf Spanning Trees in almost Linear Time," *J. Algorithms*, vol. 29, pp. 132-141, Oct. 1998.

[7] P.-J. Wan, K.M. Alzoubi, and O. Frieder, "Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM*, pp. 1597-1604, June 2002.

[8] S. Funke, A. Kesselman, U. Meyer, and M. Segal, "A Simple Improved Distributed Algorithm for Minimum CDS in Unit Disk Graphs," *ACM Trans. Sensor Networks*, vol. 2, pp. 444-453, Aug. 2006.
[9] J. Wu and H. Li, "On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks," *Proc. Int'l Workshop Discrete Algorithms and Methods for Mobile Computing and Comm.*, pp. 7-14, Aug. 1999.
[10] B. Bako, F. Kargl, E. Schoch, and M. Weber, "Advanced Adaptive Gossiping Using 2-Hop Neighborhood Information," *Proc. IEEE Global Comm. Conf. (GlobeCom)*, pp. 1-6, 2008.
[11] Y. Cai, K.A. Hua, and A. Phillips, "Leveraging 1-Hop Neighborhood Knowledge for Efficient Flooding in Wireless Ad Hoc Networks," *Proc. 24th IEEE Int'l Performance, Computing, and Comm. Conf.*, pp. 347-354, Apr. 2005.
[12] H. Liu, P. Wan, X. Jia, X. Liu, and F. Yao, "Efficient Flooding Scheme Based on 1-Hop Information in Mobile Ad Hoc Networks," *Proc. IEEE INFOCOM*, pp. 1-12, Apr. 2006.
[13] M. Khabbazian and V.K. Bhargava, "Efficient Broadcasting in Mobile Ad Hoc Networks," *IEEE Trans. Mobile Computing*, vol. 8, no. 2, pp. 231-245, Feb. 2009.
[14] E. Pagani, "Providing Reliable and Fault Tolerant Broadcast Delivery in Mobile Ad-Hoc Networks," *Mobile Networks and Applications*, vol. 4, pp. 175-192, Oct. 1999.
[15] M. Impett, M.S. Corson, and V. Park, "A Receiver-Oriented Approach to Reliable Broadcast in Ad Hoc Networks," *Proc. IEEE Wireless Comm. Networking Conf.*, pp. 117-122, Sept. 2000.
[16] S.-T. Sheu, Y. Tsai, and J. Chen, "A Highly Reliable Broadcast Scheme for IEEE 802.11 Multi-Hop Ad Hoc Networks," *Proc. IEEE Int'l Conf. Comm.*, pp. 610-615, Apr. 2002.
[17] S. Banerjee, A. Misra, J. Yeo, and A. Agrawala, "Energy-Efficient Broadcast and Multicast Trees for Reliable Wireless Communication," *Proc. IEEE Wireless Comm. and Networking (WCNC)*, pp. 660-667, Mar. 2003.
[18] W. Lou and J. Wu, "A Reliable Broadcast Algorithm with Selected Acknowledgements in Mobile Ad Hoc Networks," *Proc. IEEE Global Comm. Conf. (GlobeCom)*, pp. 3536-3541, Dec. 2003.
[19] W. Lou and J. Wu, "Toward Broadcast Reliability in Mobile Ad Hoc Networks with Double Coverage," *IEEE Trans. Mobile Computing*, vol. 6, no. 2, pp. 148-163, Feb. 2007.
[20] F.J. Ros, P.M. Ruiz, and I. Stojmenovic, "Acknowledgment-Based Broadcast Protocol for Reliable and Efficient Data Dissemination in Vehicular Ad Hoc Networks," *IEEE Trans. Mobile Computing*, vol. 11, no. 1, pp. 33-46, Jan. 2012.
[21] R. Gandhi, A. Mishra, and S. Parthasarathy, "Minimizing Broadcast Latency and Redundancy in Ad Hoc Networks," *IEEE/ACM Trans. Networking*, vol. 16, no. 4, pp. 840-851, Aug. 2008.
[22] X. Zhang and K.G. Shin, "Chorus: Collision Resolution for Efficient Wireless Broadcast," *Proc. IEEE INFOCOM*, pp. 1-9, Mar. 2010.
[23] D. Halperin, T. Anderson, and D. Wetherall, "Taking the Sting Out of Carrier Sense: Interference Cancellation for Wireless LANs," *Proc. ACM MobiCom*, pp. 339-350, Sept. 2008.
[24] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating Sets and Neighbor Elimination-Based Broadcasting Algorithms in Wireless Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 13, no. 1, pp. 14-25, Jan. 2002.
[25] R. Kannan and S.S. Iyengar, "Game-Theoretic Models for Reliable Path-Length and Energy-Constrained Routing with Data Aggregation in Wireless Sensor Networks," *IEEE J. Selected Areas in Comm.*, vol. 22, no. 6, pp. 1141-1150, Aug. 2004.
[26] X. Zhang and B. Li, "Dice: A Game Theoretic Framework for Wireless Multipath Network Coding," *Proc. ACM MobiHoc*, pp. 293-302, May 2008.
[27] X. Ai, V. Srinivasan, and C.-K. Tham, "Optimality and Complexity of Pure Nash Equilibria in the Coverage Game," *IEEE J. Selected Areas in Comm.*, vol. 26, no. 7, pp. 1170-1182, Sept. 2008.
[28] Q. Yu, J. Chen, Y. Fan, X. Shen, and Y. Sun, "Multi-Channel Assignment in Wireless Sensor Networks: A Game Theoretic Approach," *Proc. IEEE INFOCOM*, pp. 1-9, Mar. 2010.
[29] V. Krishnamurthy, M. Maskery, and G. Yin, "Decentralized Adaptive Filtering Algorithms for Sensor Activation in an Unattended Ground Sensor Network," *IEEE Trans. Signal Processing*, vol. 56, no. 12, pp. 6086-6101, Dec. 2008.
[30] M. Kodialam and T.V. Lakshman, "Detecting Network Intrusions via Sampling: A Game Theoretic Approach," *Proc. IEEE INFOCOM*, pp. 1880-1889, Mar. 2003.
[31] CPLEX, http://www.cplex.com, 2013.
[32] GLPK, http://www.gnu.org/software/glpk, 2013.
[33] G. Song and O.W. Yang, "Minimum-Energy Multicast Routing in Static Wireless Ad Hoc Networks," *Proc. IEEE Vehicular Technology Conf.*, vol. 6, pp. 3989-3993, Sept. 2004.
[34] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani, *Algorithmic Game Theory*. Cambridge Univ., pp. 494-502, 2007.
[35] D. Monderer and L. Shapley, "Potential Games," *Games and Economic Behavior*, vol. 14, pp. 124-143, May 1996.
[36] R.W. Rosenthal, "A Class of Games Possessing Pure-Strategy Nash Equilibria," *Int'l J. Game Theory*, vol. 2, pp. 65-67, Jan. 1973.

Fu-Wen Chen received the BS degree from the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 2008. One year after his master's study in the same department, he was transferred to the PhD program due to academic excellence. He is currently working toward the PhD degree. His research interests include wireless ad hoc and sensor networks, cooperative communications, network coding, distributed algorithms, and performance evaluation. He is a student member of the IEEE.

Jung-Chun Kao received the BS degree in electrical engineering from National Taiwan University, Taipei, in 1999, the MS degree in electrical engineering from the University of Southern California, Los Angeles, in 2003, and the PhD degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, in 2008. He joined the faculty of the Department of Computer Science, National Tsing Hua University, Taiwan, in August 2008. His research interests include analysis, modeling and optimization techniques for wireless networks, cyber-physical systems, and high-speed communication and protocols. He is a member of the IEEE and the IEEE Computer Society.