On Designing Incentive-Compatible Routing and Forwarding Protocols in Wireless Ad-Hoc Networks[*] — An Integrated Approach Using Game Theoretical and Cryptographic Techniques

Sheng Zhong[⋆] Li (Erran) Li[†] Yanbin Grace Liu[‡] Yang Richard Yang[§]

[⋆] Department of Computer Science and Engineering, SUNY at Buffalo, Buffalo, NY 14260
[†]Networking Research Lab, Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974
[‡]Department of Computer Sciences, The University of Texas, Austin, TX 78712
[§]Department of Computer Science, Yale University, New Haven, CT 06520

ABSTRACT

In many applications, wireless ad-hoc networks are formed by devices belonging to independent users. Therefore, a challenging problem is how to provide incentives to stimulate cooperation. In this paper, we study *ad-hoc games*—the routing and packet forwarding games in wireless ad-hoc networks. Unlike previous work which focuses either on routing or on forwarding, this paper investigates both routing and forwarding. We first uncover an impossibility result—there does not exist a protocol such that following the protocol to always forward others' traffic is a *dominant action*. Then we define a novel solution concept called *cooperation-optimal protocols*. We present Corsac, a cooperation-optimal protocol consisting of a routing protocol and a forwarding protocol. The routing protocol of Corsac integrates VCG with a novel cryptographic technique to address the challenge in wireless ad-hoc networks that a link's cost (*i.e.*, its *type*) is determined by two nodes together. Corsac also applies efficient cryptographic techniques to design a forwarding protocol to enforce the routing decision, such that fulfilling the routing decision is the *optimal* action of each node in the sense that it brings the maximum utility to the node. Additionally, we extend our framework to a practical radio propagation model where a transmission is successful with a probability. We evaluate our protocols using simulations. Our evaluations demonstrate that our protocols provide incentives for nodes to forward packets.

[*]Sheng Zhong was supported in part by NSF grants ANI-0207399 and ANI-0238038. This work was done while Sheng Zhong was at Yale University. Yang Richard Yang was supported in part by NSF grants ANI-0207399 and ANI-0238038. Li (Erran) Li was partially supported by NSF NRT grant# ANI-0335244.

Categories and Subject Descriptors

C.2.1. [**Computer-Communication Networks**]: Network Architecture and Design – Wireless communication

General Terms

Algorithms, Economics, Security, Design

Keywords

Mechanism design, game theory, security, incentives, wireless ad-hoc network

1. INTRODUCTION

Many wireless ad-hoc networks are currently being designed or deployed, driven by the vision of any-time, any-where connectivity [27, 36, 42] and the wide availability of wireless communication devices such as PDAs, cell-phones, and 802.11 access points. The functioning of such ad-hoc networks depends on the assumption that nodes in the network forward each other's traffic. However, because forwarding packets consumes scarce resources such as battery power, when the nodes in the network belong to different users, they may not have incentives to forward each other's traffic.

To stimulate nodes to forward each other's traffic, many methods have recently been proposed and evaluated (*e.g.*, [3, 4, 5, 6, 22, 24, 30, 31, 39, 40, 47]). Given the complexity and the subtlety of the incentive issues, researchers start to formally apply game-theoretic techniques to analyze and design protocols in wireless ad-hoc networks, by modeling the nodes in the networks as selfish users whose goals are to maximize their own utilities (*e.g.*, [1, 3, 4, 5, 6, 39, 24, 29, 40, 44, 47]). Although much progress has been made in the last few years, several *fundamental* issues remain unaddressed.

A major lacking of previous studies is that each study focuses on a single component. Specifically, all previous studies focus either on the routing component (*e.g.*, [1, 44]) or the packet forwarding component (*e.g.*, [15, 24, 29, 47]). However, it is clear that both routing and packet forwarding are needed to build a complete system. The routing component determines a packet forwarding path from a source to a destination; it may also determine how many credits a node on the path will receive after forwarding each packet. However, because the nodes on the path should receive credits if and only if they actually forward packets, we also need the packet-forwarding component to verify that forwarding does happen. The designs of both the routing component and the forwarding component are challenging: the routing component should discover efficient packet forwarding paths (such as power-optimal paths) even when the nodes are selfish and thus may try to cheat to improve their utilities; the packet-forwarding component should address the fair exchange problem where no node wants to make a commitment before the others do [38]. Although both individual components are challenging, it is more challenging to design and analyze a complete system that integrates both routing and forwarding, given the interdependency of the two components. In the more general networking context, for scalable designs, many network systems are designed using a layered architecture; that is, an upper layer component relies on a lower layer component. Also, many network functions are implemented in multiple stages. However, there was no previous methodology in investigating the joint incentive properties of a system involving multiple components or stages where each component or stage needs to deal with incentive issues.

The integration of multiple components is particularly challenging in wireless ad-hoc networks because the wireless and ad-hoc nature may make it impossible to design protocols with strong incentive properties. Consider the forwarding protocol. An ideal forwarding protocol is one in which power-efficient paths are discovered; network nodes on the paths forward traffic; and following the protocol is a *dominant action* for each node [34]; that is, no matter what other nodes do, following the protocol always brings the maximum utility to a node. We call such a protocol a forwarding-dominant protocol. A forwarding-dominant protocol is more desirable than a protocol that achieves a Nash equilibrium, since typically there exist multiple Nash equilibria [13] and it is hard to make a system converge to a desirable Nash equilibrium in a distributed setting [26]. However, an issue that has not been investigated before is whether a forwarding-dominant protocol exists. If not, what is a good and feasible solution concept?

The unique properties of wireless ad-hoc networks also imply that tools from game theory may not be directly applicable or a direct application may result in incorrect results. Novel techniques are needed to adapt classic game theory tools to the new settings. Consider the classic VCG (Vickrey-Clark-Groves) mechanism [9, 20, 43], which has been applied to route discovery in wireless ad-hoc networks [1]. To discuss the challenge of applying VCG to wireless networks, we first briefly review the VCG mechanism as follows. Assume that each user has a *private* type (the notion of type in specific settings will be clear later). A user declares its type (which may or may not be the true type) to a social planner, who decides an outcome to optimize a social objective and a payment to each user. The outcome and the payments are determined in such a way that reporting type truthfully is a dominant action and thus the computed outcome is socially optimal. A classic application of the VCG mechanism is the second-price auction. In this problem, the type of each user is its internal value of a given item and the objective of the planner is to choose the user who values the item the most. Then according to the VCG mechanism, each user declares its value of the item (called a bid) to the planner, the planner assigns the item to the user who makes the highest bid, and this user pays the second highest bid. It can be shown that under this mechanism, declaring the true value of the item is a dominant action of each user; *i.e.*, regardless of the declarations of all other users, the best a user can do is to declare its true value.

Although the VCG mechanism has been applied to many networking problems in general (*e.g.*, [12, 33, 35]) and to routing protocols in particular (*e.g.*, [1, 11]), wireless ad-hoc networks pose unique challenges. Specifically, the VCG mechanism assumes that each user has a private type which is *internal* to the user. Therefore, to apply VCG directly, a user must be able to determine its type by itself. In wireless ad-hoc networks, for the problem of power-efficient routing, the type of a node includes the power levels to reach its neighbors. However, a node alone cannot determine these power levels because it needs *feedbacks* from its neighbors [27]. Since the nodes are non-cooperative, these feedbacks may allow one node to cheat its neighbors in order to raise its own welfare. Such mutually-dependent types have not been addressed before, neither in the game theory community nor in the networking community. Such mutual dependency is challenging to address; for example, the authors of [47] comment that VCG cannot be applied because there is no private type in wireless ad-hoc routing. Ignoring such mutual dependency may introduce serious flaws into protocols. For example, in Section 4.1, we show that the Ad-hoc VCG protocol [1] is flawed because it does not properly handle cheating in estimating power levels.

Last, the previous work (*e.g.*, [1]) on game design for routing and forwarding in wireless ad-hoc networks uses the binary link model where a packet is always received if the transmission power is above a threshold. Recent measurements suggest that a more realistic link model is that a packet is received with a probability [10, 16, 45, 46]. We refer to such links as *lossy links*. It is not known how to deal with lossy links in routing and forwarding when we consider incentives.

The objective of this paper is to address the above issues. Our contributions can be summarized as follows.

We first show that there does not exist a forwarding-dominant protocol; that is, in the context of wireless ad-hoc networks, there does not exist a protocol implementing both routing and packet forwarding such that under the protocol nodes always forward packets, and that following the protocol is a *dominant* action. A key reason for the impossibility result is that the success of packet forwarding depends on the cooperation of all nodes on a path. However, since the nodes in a wireless ad-hoc network are distributed, there are cases where it is impossible for the system to pinpoint the misbehaving node when a failure occurs. Thus it is infeasible to design a dominant protocol, because such a protocol requires that a node be cooperative even when some other node is not cooperative. Given the impossibility result and the previous misunderstanding of dominant actions in wireless ad-hoc networks, we need to search for a new, feasible solution concept in the context of wireless ad-hoc networks.

Then we define the novel concept of a *cooperation-optimal protocol* for non-cooperative selfish users in a wireless ad-hoc network. The concept of a cooperation-optimal protocol is novel in that it consists of two sub protocols for the two stages of a node's routing-and-forwarding behavior: the routing protocol and the forwarding protocol. The requirements of a cooperation-optimal protocol are “weaker” than those of a forwarding-dominant protocol. However, if feasible, it also stimulates cooperation. We show the feasibility of the concept of cooperation-optimal protocols by designing a cooperation-optimal protocol called Corsac, a Cooperation-optimal routing-and-forwarding protocol in wireless ad-hoc networks using cryptographic techniques. Specifically, the routing protocol of Corsac uses cryptographic techniques to prevent a node from cheating in the direction where the node can benefit. Thus, a combination of incentive consideration and security techniques allows us to provide a novel solution to the mutually-dependent-type problem. The routing protocol is also integrated with a novel data forwarding protocol based on cryptographic techniques to enforce the routing decision. The routing and forward protocols are integrated in such a way that fulfilling the routing decision is the *optimal action* of each node in the sense that it brings the maximum expected utility to the node.

Third, we present techniques that allow us to extend our results from the binary link model to lossy link models [2, 10, 16, 25, 45, 46]. In these models, packet reception is probabilistic and the probability is a function of transmission power.

We evaluate our protocols using simulations, taking into account the effects of MAC and radio propagation. We evaluate the relationship among credit balance, the total energy spent in forwarding each other's traffic, and the position of a node. We show that our protocols are fair in that nodes forwarding more packets receive more credits. We evaluate the relationship among Euclidean distance between the source and the destination of a session, the payment to the intermediate nodes, and the energy consumed by the intermediate nodes. We show that it is mainly the topology, instead of Euclidean distance, which determines the payment. We evaluate the effects of stopping a node from generating new packets when its credit balance is below a threshold. We also evaluate the effects of cheating and show that following our protocols brings higher utility.

The rest of the paper is organized as follows. We first describe our network model and an impossibility result in Section 2. Then we give our new solution concept in Section 3. We present the design and analysis of our routing and forwarding protocols in Sections 4 and 5, respectively. We extend our work to lossy links in Section 6. In Section 7 we present our evaluation results. We conclude in Section 8.

2. NETWORK MODEL AND AN IMPOSSIBILITY RESULT ON AD-HOC GAMES

2.1 A Model of Ad-hoc Games

Consider an ad-hoc network formed by a finite number of nodes $\mathcal{N} = \{1, 2, \ldots, N\}$. We assume that each node i has only a discrete set $\mathcal{P}_i$ of power levels at which it can send packets (*e.g.*, Cisco Aironet cards and Access Points can be configured with a few power levels such as 1 mW, 5 mW, 20 mW, 30 mW, 50 mW and 100 mW [8]).

For each (ordered) pair of nodes (i, j), we assume that there is a minimum power level $P_{i,j}$ at which node i can reach node j. That is, when node i sends a packet, node j receives the packet *if and only if* node i sends the packet at a power level greater than or equal to $P_{i,j}$. It is possible that $P_{i,j} = \infty$, which means that even if node i sends a packet at its maximum power level, node j still cannot receive the packet. The above transmission model is a binary model. In Section 6, we will extend our results to lossy link models.

As in previous approaches, we model routing and forwarding as uncooperative strategic games in game theory [34]. We call the games *ad-hoc games*. In an ad-hoc game, each player is a node who may participate in routing and packet forwarding. A node i chooses an *action* a_i. Given a communication protocol, the action a_i may or may not follow the protocol. Specifically, for each computational task the protocol requires node i to complete, a_i may replace the task with an arbitrary polynomial-time algorithm; for each message the protocol requires node i to send, a_i may either withhold the message or replace it with an arbitrary message and send the new message at an arbitrary power level. However, to simplify our model, we do not allow a_i to send more messages than it is supposed in the protocol. As a notational convention, we use a to denote the actions of all nodes, and a_{-i} the actions of all nodes except node i. Note that both a and a_{-i} are vectors. Sometime we write $a = (a_i, a_{-i})$ to denote that the action profile a where node i takes action a_i and the other nodes take actions a_{-i}. The action profile a of all nodes decides each node's *utility* in this game. A node i's utility u_i consists of two parts:

$$u_i = -c_i + p_i,$$

where c_i is node i's *cost*, and p_i node i's *payment*. In this paper, both cost and payment incur for data packets. We ignore the cost of control packets because control packets are in general smaller and are only generated at the beginning of a session.

We distinguish two cases in explaining cost and payment:

- If node i is outside the packet forwarding path, then clearly both c_i and p_i should be 0.
- If node i is on the packet forwarding path, then c_i stands for the energy cost consumed in forwarding data packets,[1] and p_i stands for the credit it receives from the system for forwarding the data packets. Whenever an intermediate node i forwards a data packet at power level l, the corresponding cost is $l \cdot \alpha_i$, where α_i is a cost-of-energy parameter. Here α_i reflects node i's internal states such as remaining battery and the valuation of each unit of power.

Note that both c_i and p_i are decided by the actions of all players:

$$c_i = c_i(a);$$

$$p_i = p_i(a).$$

DEFINITION 1. *In a non-cooperative strategic game, a* dominant action *of a player is one that maximizes its utility no matter what actions other players choose [34]. Specifically, a_i is node i's dominant action if, for any $a'_i \neq a_i$ and any a_{-i},*

$$u_i(a_i, a_{-i}) \geq u_i(a'_i, a_{-i}).$$

It is clear that an ideal ad-hoc network is a network where forwarding others' packets is a dominant action. More precisely we have the following definition for a forwarding-dominant protocol:

DEFINITION 2. *In an ad-hoc game, a* forwarding-dominant protocol *is a protocol in which 1) a subset of the nodes are chosen to form a path from the source to the destination; 2) the protocol specifies that the chosen nodes should forward data packets, and 3) following the protocol is a dominant action.*

2.2 Non-existence of Forwarding-Dominant Protocol

As a surprising result, we show that there is no forwarding-dominant protocol for ad-hoc games.

THEOREM 1. *There does not exist a forwarding-dominant protocol for ad-hoc games.*

PROOF. We prove by contradiction. Suppose that there exists a forwarding-dominant protocol. Then we consider a source node S, a destination D, and a node distribution in which there is a link (i, j) on the packet forwarding path such that

- $P_{i,j} < \infty$, which means that node j can receive packets sent by node i;

[1]We focus on transmission power consumption because receiving power consumption is generally fixed and thus can be included at a fixed value. There are also effective methods such IEEE 802.11 sleeping modes to reduce power consumption in idle states.

- $P_{i,l} = \infty$, for any $l \neq j$, which means that any other node cannot receive any packet sent by node i.[2]

Figure 1 shows the setup.

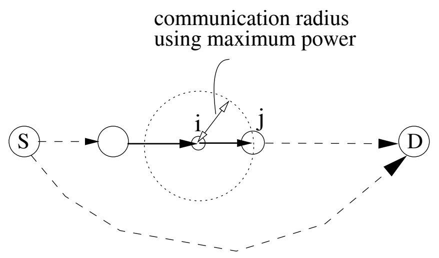


Figure 1: Illustration of the setup for the impossibility result.

We compare two action profiles. All nodes except node i have the same actions in both profiles. In both action profiles, any node except i, j follows the protocol faithfully. Also in both action profiles, j almost follows the protocol except that it behaves as if it did not receive the data packet with sequence number 0, even if it does receive the packet.[3] However, i has different actions in these two profiles: the action a_i means that i faithfully follows the protocol and forwards all packets; the action a'_i means that i follows the protocol except that it discards the data packet with sequence number 0. Obviously, by no means can the system distinguish these two action profiles, because packet 0 is always discarded and there is no way to know who discards it. Therefore, these two profiles bring the same payment to i:

$$p_i(a_i, a_{-i}) = p_i(a'_i, a_{-i}).$$

On the other hand, a_i has a greater cost than a'_i because it forwards one more packet:

$$c_i(a_i, a_{-i}) > c_i(a'_i, a_{-i}).$$

Thus we get

$$p_i(a_i, a_{-i}) - c_i(a_i, a_{-i}) < p_i(a'_i, a_{-i}) - c_i(a'_i, a_{-i}),$$

which is equivalent to

$$u_i(a_i, a_{-i}) < u_i(a'_i, a_{-i}).$$

This contradicts the definition of dominant action. □

Remark The above theorem applies only if each node is autonomous and has the freedom to choose its behavior. If, for example, the nodes' behavior is restricted by installed tamper-proof hardware, then a forwarding-dominant protocol can be designed. Specifically, consider the extreme situation in which each node is completely built on tamper-proof hardware — in this case, any protocol that forwards all packets is a dominant solution. However, ad-hoc networks formed by nodes with tamper-proof hardware may not be the common case.

[2]We can make sure that such (i, j) exists on the packet forwarding path by considering a situation in which every path from S to D contains a link that satisfies the two conditions. Therefore, no matter which path is chosen as the packet forwarding path, there is always a pair (i, j) on the packet forwarding path that satisfies these conditions.

[3]It can be the case that j's utility is lower if it pretends that it did not receive the packet when it does receive the packet; for example, see [47]. However, this is a valid action of j.

Remark The above theorem is valid in not only our model, but also *many alternative models*. For example, although our model assumes asymmetric links (i.e., $P_{i,j}$ is not necessarily equal to $P_{j,i}$), the above theorem is also valid with symmetric links (*i.e.*, $\forall(i, j)$, $P_{i,j} = P_{j,i}$), if reliable overhearing is not available. Even if we assume symmetric links plus reliable overhearing, the above theorem is still valid as long as the protocol cannot always use the maximum power level for transmission. Proofs under these models are similar.

3. THE CONCEPT OF A COOPERATION-OPTIMAL PROTOCOL

Given the surprising result that there is no forwarding-dominant protocol to ad-hoc games, we need to weaken the requirements so that feasible protocols can be designed and the protocols stimulate cooperation. Below we introduce the concept of a cooperation-optimal protocol for wireless ad-hoc networks with non-cooperative selfish users.

Specifically, the routing and forwarding behavior of a node occurs in two stages: the routing stage and the forwarding stage. Accordingly, each node's action in the ad-hoc game is divided into two parts: its subaction in the routing stage and its subaction in the packet forwarding stage. In the routing stage, the nodes' subactions jointly decide a *routing decision* — the content of this routing decision is all nodes' forwarding subactions, which specify what each node *is supposed to do* in the forwarding stage. In the forwarding stage, the routing decision (*i.e.*, what each node is supposed to do in this stage) and the nodes' forwarding subactions (*i.e.*, what each node really does in this stage) jointly decide each node's utility.

Formally, we have $a_i = (a_i^{(r)}, a_i^{(f)})$, where $a_i^{(r)}$ is node i's subaction in the routing stage, and $a_i^{(f)}$ is i's subaction in the forwarding stage. Let a denote the actions of all nodes, $a^{(r)}$ the routing subactions of all nodes, and $a^{(f)}$ the subactions of all nodes during packet forwarding.

A routing decision $\mathcal{R}$ is decided by the routing subactions of all nodes:

$$\mathcal{R} = \mathcal{R}(a^{(r)}).$$

Since a routing decision consists of all nodes' supposed forwarding subactions $\hat{a}^{(f)}$, we also write

$$\mathcal{R} = \hat{a}^{(f)}.$$

Finally, each node's utility u_i is decided by the routing decision $\mathcal{R}$ and the nodes' actual subactions $a^{(f)}$ in the forwarding game:

$$u_i = u_i(\mathcal{R}, a^{(f)}).$$

It is clear that utilities given as above are consistent with the original definition of utilities in ad-hoc games.

Remark The possibility of dividing a game into stages has also been suggested by Feigenbaum and Shenker in their PODC tutorial slides [14]. The definitions above divide an ad-hoc game into two stages.

3.1 Defining Solution Concept to the Routing Stage

DEFINITION 3. *Given a routing decision, the* prospective routing utility *of a node is the utility that it will achieve under the routing decision, if all nodes in the packet forwarding stage follow the routing decision (*i.e., *if each node takes the forwarding subaction* *designated by the routing decision). Formally, let $\mathcal{R}(=\hat{a}^{(f)})$ be a routing decision. Then node i's prospective routing utility is*

$$u_i^{(\mathcal{R})} = u_i(\mathcal{R}, \hat{a}^{(f)}).$$

Note that $u_i^{(p)}$ depends only on $\mathcal{R}$, and that $\mathcal{R}$ is decided by the routing subactions $a^{(r)}$. Therefore, $u_i^{(p)}$ is decided by $a^{(r)}$. Formally, we write

$$u_i^{(\mathcal{R})} = u_i^{(\mathcal{R})}(a^{(r)}).$$

DEFINITION 4. *In a routing stage, a* dominant subaction *of a potential forwarding node is one that maximizes its prospective routing utility no matter what subactions other players choose in this stage. Formally, $a_i^{(r)}$ is node i's dominant subaction in the routing stage if, for any $\bar{a}_i^{(r)} \neq a_i^{(r)}$, any $a_{-i}^{(r)}$*

$$u_i^{(\mathcal{R})}(a_i^{(r)}, a_{-i}^{(r)}) \geq u_i^{(\mathcal{R})}(\bar{a}_i^{(r)}, a_{-i}^{(r)}). \quad (1)$$

Remark In the above definition, note that $a_i^{(r)}, \bar{a}_i^{(r)}, a_{-i}^{(r)}$ are all program segments responsible for routing. Because these program segments might contain coin flips (due to using probabilistic algorithms), for practical purpose, we require only that Equation (1) be satisfied *with high probability*,[4] where the probability is computed over the coin flips in the involved program segments.

Also note that in the above definition we follow the convention of focusing on motivating nodes to forward traffic (*e.g.*, [1, 11, 21, 33]); therefore the definition applies only to the potential forwarding nodes.

DEFINITION 5. *A routing protocol is a* routing-dominant protocol to the routing stage *if following the protocol is a dominant subaction of each potential forwarding node in the routing stage.*

To finish defining the routing stage, we also need to decide who should do the route computation. To avoid an online central node to perform all computation, in our model, the destination of each session does the computation. Because the destination does not need to pay any node, neither will it receive any payment, it is likely to be reliable. This is particularly true in a hybrid architecture such as [24, 28, 39], where the destination is a base station. If there is a possibility that the destination is not trustworthy in computation, we can apply a sampling technique to validate the computation of the destination. That is, for a randomly chosen session, a node may initiate a validation session to check if the computation is valid. The node or a central authority collects the relevant information sent to the destination and verifies the computation. In the case that the central authority is not available online, if a node detects cheating, it can report all relevant information to the central authority offline. If cheating by a destination is detected, a high penalty is assessed (*e.g.*, the destination is removed from the system). To prevent potential denial of service attack on such a validation process, the number of sessions that can be sampled by a node should be limited.

3.2 Defining Solution Concept to the Forwarding Stage

The separation of routing and forwarding facilitates design. However, there was no previous study in the context of networking in analyzing the incentive issues of a system consisting of multiple interdependent protocols. To analyze such systems, we adopt the concept of extensive games.

Specifically, we consider an *extensive* game model. This model can be represented as a game tree: each vertex of the game tree corresponds to a wireless node (but each wireless node corresponds to multiple vertices in the game tree) and each edge going out of the vertex stands for a possible decision by this node in the forwarding stage. See Figure 2 for an example of game tree. Clearly, each subtree of the game tree corresponds to a subgame and each path from the root down to a leaf corresponds to a possible set of decisions by the wireless nodes in the forwarding stage. In classic game theory, such a path is said to be a *subgame perfect equilibrium* if it is a Nash equilibrium for every subgame.

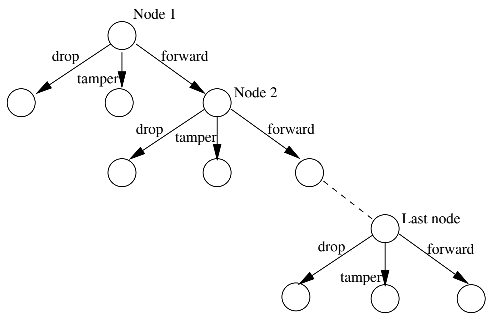


Figure 2: An example game tree.

DEFINITION 6. *A forwarding protocol is a* forwarding-optimal protocol to the forwarding stage *under routing decision $\mathcal{R}$ if all packets are forwarded to their destinations in this protocol and following the protocol is a subgame perfect equilibrium under routing decision $\mathcal{R}$ in the forwarding stage.*

3.3 Defining Solution Concept to the Ad-hoc Game

DEFINITION 7. *A protocol is a* cooperation-optimal protocol *to an ad-hoc game if*

- *its routing protocol is a routing-dominant protocol to the routing stage;*
- *for a routing decision $\mathcal{R}$ generated by the preceding routing subactions, its forwarding protocol is a forwarding-optimal protocol to the forwarding stage under $\mathcal{R}$.*

To find a cooperation-optimal protocol to an ad-hoc game, we first design a routing-dominant protocol to the routing stage and then an forwarding-optimal protocol to the forwarding stage. Combining these two protocols together, we design a cooperation-optimal to the ad-hoc game.

4. A ROUTING PROTOCOL FOR THE ROUTING STAGE

In this section, we present a protocol for the routing stage. The routing decision is based on the well-known VCG mechanism. However, we will first show that, a straightforward application of VCG to this problem (*e.g.*, the Ad-Hoc VCG protocol [1]) is *not* a dominant-subaction solution due to the special properties of wireless ad-hoc networks. Then, we construct a dominant-subaction solution by combining VCG with a novel cryptographic technique.

[4]High probability means 1 minus a negligible function, which decreases super-polynomially. See, *e.g.*, [19], for a formal definition of negligible functions.

4.1 VCG Payment

To motivate our design, we first briefly describe a straightforward application of VCG to this problem. (This is a simplified version of the Ad-Hoc VCG protocol [1]. We omit some details of [1] to make the presentation clearer.) Suppose that the destination collects the cost for each node to reach each of its neighbors (where a neighbor is a node that the node under discussion can reach at some power level $l \in \mathcal{P}$). Denote the lowest (claimed-)cost path from the source S to the destination D by $LCP(S, D)$; denote the lowest (claimed-)cost path from the source S to the destination D that does not include node i by $LCP(S, D; -i)$. Then the destination simply chooses $LCP(S, D)$ as the packet forwarding path from S to D, and the payment to node i is

$$p_i = cost(LCP(S, D; -i)) - cost(LCP(S, D) - \{i\}),$$

where the function $cost()$ sums the costs of all links on a path, $LCP(S, D) - \{i\}$ consists of the links on the LCP but with the link starting from node i removed, if node i is on the path.

The above description assumes that the cost of each link is known to the transmitter of the link. However, the transmitter of a wireless link needs the receiver's feedback to estimate the link cost, namely the required power level. Handling cheating in estimating link cost is a challenging task. Below we will show that the link-cost estimation scheme of the Ad-Hoc VCG protocol [1] is flawed; therefore their overall protocol does not preserve incentive compatibility.

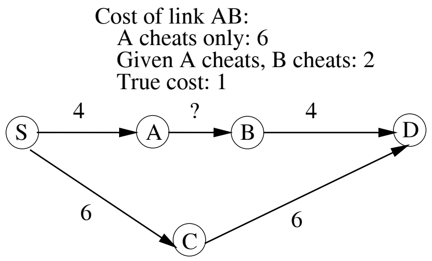


Figure 3: Illustration: VCG alone does *not* guarantee the existence of a dominant-subaction solution in routing.

Consider the link-cost estimation algorithm used in the Ad-hoc VCG protocol (see Equation (2) of [1]). The transmitter sends a pilot signal at a given power level P^{emit}; the receiver sends back the ratio R between received power level and target (minimal) power level; and then the transmitter determines its transmission power level $P = P^{emit}/R$ so that the operational power level is achieved at the receiver.

Given this protocol to determine link power level (*i.e.*, link cost), we have a simple example shown in Figure 3 to show that a straightforward application of VCG cannot be a dominant-subaction solution. Suppose that the real cost of link AB should be 1 (*e.g.*, $P^{emit} = 5$ and $R = 5$). Recall that a dominant subaction of B must be the best choice of B *no matter what subactions other nodes (such as A) choose*. Therefore, it is enough for us to consider the following specific subaction of A (with an attempt to overclaim its link cost): A sends at $P^{emit} = 5$; after receiving the feedback about the ratio R between received and target power level at the receiver, instead of claiming $5/R$, node A claims $5/R * 6$. Then,

- if B does not cheat, the claimed cost of link AB will be $5/5 * 6 = 6$;
- if B chooses a cheating subaction (to underclaim the cost by reporting R = 15), the claimed cost of link AB can be decreased back to 2.

With this subaction of A, if B does not cheat, then the LCP is the lower path in the figure, B receives zero payment and has a utility of 0. If B takes the above cheating subaction, it receives a payment of 12-4-2=6 which covers its cost of 4 on link BD and results in a positive utility of 2. Therefore, with this subaction of A, it is beneficial for B to cheat. Consequently, truthfully helping A to report the cost is *not* a dominant subaction of B *by the definition of dominant subaction.*[5]

Note that the above example uses a binary estimation scheme. We can show similar examples using other estimation schemes such as the well-known SNR based scheme.

We remark that the proof of Ad-Hoc VCG [1] is invalid in the case illustrated above. In the proof of their Lemma 2, they argue that a node like B in Figure 3 will not underclaim the cost of AB because, with the underclaimed cost, A will not be able to reach B in the data transmission phase. However, this argument becomes invalid *if A cheats*. Just as shown in Figure 3, with a cheating A, when B underclaims the cost of AB, the claimed cost ($= 2$) is still higher than the real cost ($= 1$) of the link. Therefore, A should still be able to reach B in this case, and so the proof is flawed.

The above problem is a direct consequence of mutually dependent types. With private types, this problem does not exist. To see this, we look at the same example. However, this time each node can determine the costs of its outgoing links by itself. Therefore, if the claimed cost of AB is 3 when A takes a cheating subaction and B does not cheat, then the claimed cost of AB is still 3 when A takes the same cheating subaction and B takes any cheating subaction. As a result, B's cheating is no longer beneficial. (To gain more insight, interested readers can refer to the proof in [11] that VCG mechanisms result in dominant action if each user has a private type.)

Consequently, the main remaining technical challenge is how to prevent neighbors from cheating in determining link cost. Below we present a cryptographic technique to address this issue.

4.2 Prevent Cheating in Determining Link Costs

Consider a node i and its neighbor j. There are two possibilities of cheating by node j regarding the cost $P_{i,j}$:

- Case (A): node j cheats by making $P_{i,j}$ greater.
- Case (B): node j cheats by making $P_{i,j}$ smaller.

In case (A), because we choose the lowest claimed-cost path, node j becomes less likely to be on the packet forwarding path (and thus less likely to get paid). Even if node j is still on the packet forwarding path, its payment will decrease. In summary, this kind of cheating is not beneficial to node j. Therefore, if we can find a way to prevent case (B), then we can prevent a neighbor from cheating.

We prevent case (B) using a cryptographic technique. Node i sends *pseudo-random* test signals at increasing power levels. Each test signal contains the cost information of node i at the corresponding power level. We require that node j report all the test signals it receives to the destination. Because test signals sent at lower power levels are not received by node j directly and other nodes who can hear i's signals can not relay them to j under our model in Section 2, node j has no way to report such a test signal to the destination.[6] Finally, the destination "translates" node j's reported test

[5]Note that this example does *not* involve any collusion, because a colluding group maximizes the group's overall utility in some sense (*e.g.*, sum of group members' utilities), while in our example, we only consider the utility of one single node, B.

[6]Note that this is a binary link model. We will consider a more general lossy link model in Section 6.

signals to derive the corresponding costs and selects the minimum cost for node i to reach node j.

To achieve the above goal, suppose that node i shares key $k_{i,D}$ with the destination D. Also suppose that the identifier of the session is (S, D, r), where r is a random number used to distinguish different sessions with source S and destination D. Then, for each available power level l (in increasing order), node i computes a test signal h_l by encrypting $[S, D, r, l, \alpha_i]$ (where α_i is a cost-of-energy parameter representing the cost of unit energy at node i) using key $k_{i,D}$ and attaching a Message Authentication Code (MAC) using the same key. Since only i and D know $k_{i,D}$, only i and D can compute these test signals (h_l's). Furthermore, h_l is protected by the MAC so that it is infeasible for any other node to tamper with h_l. Note that, in the above formula, S, D, and r cannot be omitted because we do not want different sessions to use the same h_l.

To set up a shared key $k_{i,D}$ between node i and destination D, we use the well-known Diffie-Hellman key exchange in cryptography: suppose that node i has a private key k_i and a public key $K_i = g^{k_i}$, and that D has a private key k_D and a public key $K_D = g^{k_D}$ (where g is a primitive root in a group where computing discrete logarithm is hard). Then we have $k_{i,D} = g^{k_i k_D} = (K_D)^{k_i} = (K_i)^{k_D}$. Note that node i can get $k_{i,D}$ by computing $(K_D)^{k_i}$ and D can get it by computing $(K_i)^{k_D}$. Readers who are not familiar with cryptography can read references such as [41] for details.

4.3 Protocol for the Routing Stage

Given the preceding solution, next we present our routing protocol. The protocol is an on-demand routing protocol in that the source initiates a route discovery after receiving a session from the application layer. (For ease of presentation, in the following protocol description we assume $\forall i$, $\mathcal{P}_i = \mathcal{P}$.)

4.3.1 Source node's test signals

- Source S starts a session of M packets. Source S divides the packets into $\lceil M/b \rceil$ blocks, where b is the number of packets in a block.
- Source S picks a random number r_0.
- Let H be a cryptographic hash function. S computes $r = H^{\lceil M/b \rceil}(r_0)$. (Note that r depends on the number of blocks in the session — this property will be useful in the packet forwarding protocol.)
- For each power level $l \in \mathcal{P}$ (in increasing order), S sends out (TESTSIGNAL, [S, D, r], [S, h_l]) at power level l, where h_l contains an encryption of $[S, D, r, l, \alpha_S]$ using key $k_{S,D}$ and a MAC of the encryption using the same key.

4.3.2 Intermediate node's test signals

Upon receiving (TESTSIGNAL, [S, D, r], [P, h]) from an upstream neighbor P, an intermediate node i does the following.

- Node i sends out (ROUTEINFO, [S, D, r], [P, i, h']) at power level P_{ctr} (where P_{ctr} is a power level for control messages such that the communication graph is connected when all links use power level P_{ctr} for transmission). Here h' is computed by encrypting h using key $k_{i,D}$. For integrity, this message is protected by a MAC using key $k_{i,D}$.
- If the TESTSIGNAL is the first one i receives for session (S, D, r), then for each $l \in \mathcal{P}$ (in increasing order), node i sends out (TESTSIGNAL, [S, D, r], [i, h'_l]) at power level l, where h'_l contains an encryption of $[S, D, r, l, \alpha_i]$ using the key $k_{i,D}$ and a MAC of the encryption using the same key.

4.3.3 Route information forwarding

Upon receiving (ROUTEINFO, [S, D, r], [P, i, h]), an intermediate node j does the following:

- If this ROUTEINFO is new to node j, then node j sends out (ROUTEINFO, [S, D, r], [P, i, h]) at power level P_{ctr}.

4.3.4 Destination protocol

Destination D maintains a cost matrix for each session (S, D, r). Each entry of this matrix is an array of power level and cost-of-energy parameter.

- Upon receiving (TESTSIGNAL, [S, D, r], h) from neighbor P, D decrypts h, verifies the MAC using the key $k_{P,D}$, and "translates" h to the corresponding power level l and cost-of-energy parameter α_P. D records (l, α_P) in the cost matrix's entry for link (P, D).
- Upon receiving (ROUTEINFO, [S, D, r], [P, i, h]), D decrypts h, verifies the packet's MAC using key $k_{i,D}$, and "translates" h to the corresponding power level l and cost-of-energy parameter α_P. D records (l, α_P) in the cost matrix's entry for link (P, i).

After collecting all link cost information, D checks, for each link, that the cost-of-energy parameter does not change. Then D chooses the minimum power level in record for each link, which determines the minimum link cost together with the cost-of-energy parameter. D computes the lowest cost path from S to D in this cost graph, using Dijkstra's algorithm. Denote the computed lowest cost path by $LCP(S, D)$. $LCP(S, D)$ is the chosen path for packet forwarding. Recall that the lowest cost path in the graph without node i by $LCP(S, D; -i)$. Then the *unit payment* (*i.e.*, the payment for one data packet) to node i is

$$p_i = cost(LCP(S, D; -i)) - cost(LCP(S, D) - \{i\}).$$

(Note that all the above computation can be finished in $O(N^3)$ time.)

4.4 Analysis of the Routing Protocol

THEOREM 2. *If the destination is able to collect all involved link costs, then the protocol given in Section 4.3 is a routing-dominant protocol to the routing stage.*

PROOF. Consider node i. Let $a_i^{(r)}$ be the subaction of node i in the routing stage that follows the protocol faithfully. Let $\bar{a}_i^{(r)} \neq a_i^{(r)}$ be a different sub-action. Let $a_{-i}^{(r)}$ be an arbitrary subaction profile of all other nodes in this stage. We will show that

$$u_i^{(\mathcal{R})}(a_i^{(r)}, a_{-i}^{(r)}) \geq u_i^{(\mathcal{R})}(\bar{a}_i^{(r)}, a_{-i}^{(r)}).$$

We note that the difference in node i's subaction ($\bar{a}_i^{(r)}$ versus $a_i^{(r)}$) can only lead to difference in the claimed costs of link (i, j)'s and/or link (j, i)'s (which, in turn, may influence the routing decision and the prospective utility). Because we are using VCG payment, the prospective utility of node i is independent of claimed costs of link (i, j)'s. So it is enough to consider the difference in claimed costs of link (j, i)'s. Note that our cryptographic technique prevents node i from reducing costs of link (j, i)'s (with high probability). Therefore, with $\bar{a}_i^{(r)}$, the claimed costs of link (j, i)'s can only be greater or unchanged. For simplicity, let us assume that the cost of only one link (j, i) is increased by $\bar{a}_i^{(r)}$. (If more than one such link costs are increased, we can prove the result similarly, by considering the change of one link cost at a time.) There are three cases:

(1) With $a_i^{(r)}$, node i is not on the packet forwarding path. In this case, with $\bar{a}_i^{(r)}$, node i is still not on the packet forwarding path, because increasing an upstream neighbor's cost to reach a node cannot move this node itself to the lowest cost path. Therefore,

$$u_i^{(\mathcal{R})}(a_i^{(r)}, a_{-i}^{(r)}) = u_i^{(\mathcal{R})}(\bar{a}_i^{(r)}, a_{-i}^{(r)}) = 0.$$

(2) With $a_i^{(r)}$, node i is on the packet forwarding path, but the link (j, i) is not (*i.e.*, node j is not the upstream neighbor of node i along this path). Then with $\bar{a}_i^{(r)}$ (*i.e.*, with increased cost of link (j, i)), the packet forwarding path is not changed. Because the link (j, i) is not on $LCP(S, D)$, $cost(LCP(S, D))$ is not changed. Because the link $(j, i))$ has an end point i, it cannot be on $LCP(S, D; -i)$; thus $cost(LCP(S, D; -i))$ is not changed. Therefore, p_i is not changed. Considering the cost of i is not changed as well, we know that i's prospective utility is not changed:

$$u_i^{(\mathcal{R})}(a_i^{(r)}, a_{-i}^{(r)}) = u_i^{(\mathcal{R})}(\bar{a}_i^{(r)}, a_{-i}^{(r)}).$$

(3) With $a_i^{(r)}$, node i is on the packet forwarding path, and so is the link (j, i) (*i.e.*, node j is the upstream neighbor of node i along this path). Then with $\bar{a}_i^{(r)}$, we will have a greater $cost(LCP(S, D))$. Therefore, p_i decreases and so does the prospective utility:

$$u_i^{(\mathcal{R})}(a_i^{(r)}, a_{-i}^{(r)}) > u_i^{(\mathcal{R})}(\bar{a}_i^{(r)}, a_{-i}^{(r)}).$$

Thus we finish the proof. $\square$

Remark Note that a routing-dominant protocol works in practice only if the computed payments can be enforced, and that the enforcement of payments is addressed in the forwarding stage.
Remark All the above analysis ignores the cost of control messages.
Remark Our proof covers all possible subactions, including claiming false power levels and claiming false cost-of-energy parameters.

5. A SECURE FRAMEWORK FOR THE PACKET FORWARDING STAGE

In the preceding section we have described our routing protocol, in this section we describe our packet forwarding protocol.

5.1 Design Techniques

We first describe our design techniques.

5.1.1 *Block confirmation using reversed hash chain*

For efficiency, data packets of a session with M packets are transmitted in blocks. Each block consists of b packets (except the last block in a session which may have fewer packets). After the transmission of each block, the destination gives the intermediate nodes a confirmation, which proves that they have succeeded in transmitting this block. Only after getting this confirmation will the intermediate nodes continue to forward the next block.

We give a very efficient way to implement block confirmations using reversed hash chain. Recall that H is a cryptographic hash function. Let r_0 be a random number selected by the source of a session. The source computes $r_m = H^m(r_0)$ for block m. Because there are altogether $\lceil M/b \rceil$ blocks, we let $r = r_{\lceil M/b \rceil}$. The source makes r public and computes $r_{\lceil M/b \rceil - m}$ as the confirmation of the m-th block. Therefore, it is very easy for any intermediate node (and any outsider) to verify this confirmation by checking

$$r = H^m(r_{\lceil M/b \rceil - m}).$$

On the other hand, it is infeasible for any intermediate node to forge the confirmation of any block that has not been successfully transmitted to the destination. Note that, when an intermediate node receives the confirmation of the m-th block, it can drop the confirmation of the $(m-1)$-th block because the m-th block's confirmation actually proves that all the first m blocks have been successfully transmitted.

5.1.2 *Mutual decision to resolve conflict*

It is still possible that the source and the intermediate nodes disagree about whether the "next block" (*i.e.*, the block immediately after the last one that has a confirmation) has been successfully transmitted. To eliminate the incentives to cheat, we use a technique called *mutual decision* [23]. That is, the source decides whether the intermediate nodes should be paid for the next block, while the intermediate nodes decide whether the source should be charged for this block. Note that no node will decide payment/charge to itself for this block. Therefore, every node has no incentive to cheat.

Specifically, the source sends the encrypted confirmation at the end of the corresponding block to the destination, and the destination releases the (decrypted) confirmation if it has received the block successfully. If an intermediate node has transmitted a block but does not get the confirmation, it submits the routing decision to the system (e.g. the central bank in [47]) so that the source is still charged for this block.

5.2 Protocol for Packet Forwarding

5.2.1 *Routing decision transmission phase*

Upon finishing the routing discovery phase, the destination D sends the routing decision

$$([S, D, r], LCP(S, D), P_S, \\ \{(P_i, p_i) \mid i \text{ is an intermediate node on } LCP(S, D)\}),$$

with a digital signature along the reversed path of $LCP(S, D)$, where P_i (resp., P_S) is the power level that node i should use to forward (resp., send) data packets and p_i is the unit payment node i should receive. Each intermediate node forwards the routing decision at a power level that can reach the upstream neighbor of the forward path of $LCP(S, D)$. For ease of explanation, we assume links are bidirectional in this section.

5.2.2 *Data transmission phase*

Upon receiving the signed routing decision, the source verifies the digital signature accompanied the decision. If the signature is valid, the source enters the data transmission phase. In this phase, the source and the intermediate nodes send data packets at the computed power levels (P_S or P_i in the routing decision, respectively). The source node sends out data packets in blocks. Recall that each block contains b packets. Together with the last data packet in the m-th block, the source sends out $r_{\lceil M/b \rceil - m} = H^{\lceil M/b \rceil - m}(r_0)$ (which is encrypted using key $k_{S,D} = K_D^{k_S}$). Then it waits for a confirmation before it sends the next block.

Once the source sends out packets in a block, the intermediate nodes forward them along $LCP(S, D)$ to the destination. After finishing a block, the intermediate nodes also wait for a confirmation before they start forwarding the next block. Once the destination receives all the packets in a block, it decrypts $r_{\lceil M/b \rceil - m}$. It sends $r_{\lceil M/b \rceil - m}$ in clear-text back along $LCP(S, D)$, as a confirmation of this block. Upon receiving the confirmation of the mth block, each intermediate node verifies that $r = H^m(r_{\lceil M/b \rceil - m})$. If this is correct, then the intermediate node saves the confirmation (which replaces the previously saved confirmation in this session) and forwards it back along $LCP(S, D)$.

Upon receiving the confirmation of one block, the source node starts sending the next block. Suppose that, in a session, the last confirmation saved by node i is $r_{\lceil M/b \rceil - m}$. Then all node j before i on the path gets a payment of $p_j \cdot b \cdot m$ from the source by submitting this confirmation to the system. If some packets in the $(m + 1)$-th block have been transmitted but the confirmation is never received, then the intermediate nodes submit the routing decision to the system so that the system charges the source node $\sum_i p_i \cdot b$ in addition. Note that this amount of credit does not go to the intermediate nodes, but goes to the system.

5.3 Analysis of the Packet Forwarding Protocol

THEOREM 3. *Suppose that $\mathcal{R}$ is a routing decision computed by the routing subactions designated by the protocol in Section 4.3. Assume that, for any node on the packet forwarding path, the computed payment is greater than the cost. Then the protocol presented in Section 5.2 is a forwarding-optimal protocol to the packet forwarding stage under $\mathcal{R}$.*

PROOF. We use a standard game-theoretic technique, *backward induction*, to give our proof. Using this technique, we start from the end of the forwarding stage and show that the intermediate nodes should forward the confirmation of the last block as specified in the protocol; then we go back in time and show that each node making a decision in the forwarding stage should follow the protocol; thus, we can conclude that following the protocol is a subgame perfect equilibrium.

It is clear that each intermediate node should forward the confirmation of the last block because we ignore the cost of control packets (no matter they forward control packets or not, they get the same amount of payment and have the same amount of cost). Going back one step in time, we find that, for the same reason, the destination should send out this confirmation. Furthermore, we note that the last node on the path before the destination should forward the last packet to the destination, because otherwise it would not get the confirmation and would lose the payment for the last block. Therefore, each node on the path should also forward the packet given that the nodes after it would forward the packet. A similar argument works for *every* packet in the last block. Then we go back to the last-but-one block and have a similar argument for this block. Finally, we go back to the routing decision transmission phase. In the phase, note that the intermediate nodes cannot tamper with the routing decision because it is protected by a digital signature. If an intermediate node drops or corrupts the routing decision packets, then the session stops and it has a utility of 0. Because the payment would be greater than the cost if the session does not stop, it is a better choice for the node to forward the routing decision packets. □

Remark The preceding theorem requires the condition that for any node on the packet forwarding path, the computed payment is greater than the cost. This condition is necessary to avoid the scenario that if the payment is equal to the cost, then a slight disturbance will cause the node to behave differently. This condition is practical in that it is unlikely that a node will cooperate if the payment is just enough to cover the cost.

THEOREM 4. *Our complete protocol, including the routing protocol in 4.3 and the packet forwarding protocol in 5.2, is a cooperation-optimal protocol to ad-hoc games under previous conditions.*

PROOF. This immediately follows from Theorems 2 and 3. □

6. EXTENSION TO LOSSY LINKS

In the preceding sections, we study ad-hoc games using the binary link model, which assumes that there exists a power threshold for each link, such that any packet sent at this power level or above can be received, and that any packet sent at any lower power level cannot be received. However, in some networks, we may need to consider lossy links (see, *e.g.*, [2, 10, 25, 16, 45, 46]). For such links, packets receptions are probabilistic in the sense that each packet is received with a probability, and the probability is decided by the power level at which the packet is sent. In this section, we show how to extend our work for the binary links to these lossy links. With such lossy links, there are three questions that need to be addressed: (1) For each link, how do we estimate the transmission success rate at each power level? (2) For each link, given the transmission success rate at each power level, how do we choose the power level at which data packets should be sent? (3) How do we adapt our protocols to lossy links, using the answers to (1) and (2)?

6.1 Estimating Transmission Success Rate

Consider a link (i, j). Suppose that, when node i sends a packet at power level l, the transmission success rate, (*i.e.*, the probability that node j receives this packet,) is $S_{i,j}(l)$. What we need to do is to estimate $S_{i,j}(l)$ for $l \in \mathcal{P}$. To estimate $S_{i,j}(l)$, we let node i send N_s packets. Suppose that node j receives N_r of them. If we simply estimate $S_{i,j}(l)$ based on these N_s packets, then clearly our estimate of $S_{i,j}(l)$ is

$$\hat{S}_{i,j}(l) = \frac{N_r}{N_s}.$$

However, we actually have more information that we can use for estimating $S_{i,j}(l)$: We know that $S_{i,j}(l)$ is a monotonically increasing function of l.[7] Therefore, we can use the following algorithm to estimate $S_{i,j}(l)$: (For notational simplicity, we present the algorithm for the case $\mathcal{P} = \{1, 2, \ldots, P_{max}\}$. It is straightforward to extend this algorithm to any power level set $\mathcal{P}$.)

- For $l = 1, \ldots, P_{max}$, set $x(l) = \frac{N_r}{N_s}$.
- Set $\hat{S}_{i,j}(1) = x(1)$.
- For $l = 2, \ldots, P_{max}$, if $x(l) \geq \max_{l' < l}\{x(l')\}$, then set $\hat{S}_{i,j}(l) = x(l)$; otherwise leave this entry empty, because this entry violates the knowledge that $S_{i,j}(l)$ must be increasing.
- For $l = 2, \ldots, P_{max}$, if $\hat{S}_{i,j}(l)$ is an empty entry, do the following:
  - Case A: There is a non-empty entry after $\hat{S}_{i,j}(l)$. Then suppose that the nearest non-empty entry before $\hat{S}_{i,j}(l)$ is $\hat{S}_{i,j}(l')$, and that the nearest non-empty entry after it is $\hat{S}_{i,j}(l'')$. We give an estimate of $S_{i,j}(l)$ using a linear interpolation based on these two values:

    $$\hat{S}_{i,j}(l) = ((l'' - l)\hat{S}_{i,j}(l') + (l - l')\hat{S}_{i,j}(l''))/(l'' - l').$$

[7] We may even know more than that. For example, certain analytical expressions can be derived for the transmission success rate in some link models [37]. However, because these models are still under investigation, we do not utilize such information.

- Case B: There is no non-empty entry after $\hat{S}_{i,j}(l)$. Then suppose that the nearest non-empty entry before $\hat{S}_{i,j}(l)$ is $\hat{S}_{i,j}(l')$. We give an estimate of $S_{i,j}(l)$ using a linear interpolation based on $\hat{S}_{i,j}(l')$ and an imaginary transmission success rate of 1 at power level $P_{max}+1$:

$$\begin{aligned}\hat{S}_{i,j}(l) = \quad & ((P_{max}+1-l)\hat{S}_{i,j}(l') \\ & +(l-l'))/(P_{max}+1-l').\end{aligned}$$

The above algorithm computes an optimistic estimate in the sense that it never underestimates the transmission success rate at any power level based on the transmission success rates at other power levels. This property will be useful when we design our routing protocol for lossy links.

6.2 Choosing Power Level

Given the estimated transmission success rates, we need to choose a power level for each link at which the data packets are sent. For each power level l, we consider the ratio $S_{i,j}(l)/l$; our choice is the power level that maximizes this ratio. Formally, we choose

$$L = \arg\max_{l} S_{i,j}(l)/l. \tag{2}$$

We have the following result:

LEMMA 1. *For a link (i,j), suppose that node i resends each data packet until it is received by node j.*[8] *Then sending data packets at power level L has the minimum expected power consumption.*

Remark At any node, the cost is proportional to the power consumption. Therefore, our choice of power level also minimizes the cost.

6.3 Adapting to Lossy Links

Using the results presented above, we can easily adapt our protocol to lossy link models. In the routing stage, we need to update the protocol as the following:

- When a source node S sends TESTSIGNAL, it sends N_s packets at each power level, where N_s is a constant. Specifically, for $l \in \mathcal{P}$, S sends out (TESTSIGNAL, $[S,D,r]$, $[S,h_{l,t}]$) (for $t=1,2,\ldots,N_s$) at power level l, where

  $$h_{l,t} = H(k_{S,D}, [[S,D,r,l,t],\sigma]).$$

  In the above, σ is a MAC of $[S,D,r,l,t]$ using key $k_{S,D}$.

- Similarly, when an intermediate node i sends TESTSIGNAL, it sends N_s packets at each power level. Specifically, for $l \in \mathcal{P}$, node i sends out (TESTSIGNAL, $[S,D,r]$, $[i,h'_{l,t}]$) (for $t=1,2,\ldots,N_s$) at power level l, where

  $$h'_{l,t} = H(k_{i,D}, [[S,D,r,l,t],\sigma]).$$

  In the above, σ is a MAC of $[S,D,r,l,t]$ using key $k_{i,D}$.

- If an intermediate node i receives TESTSIGNAL, no matter it is the first one from the upstream neighbor or not, node i sends out a corresponding ROUTEINFO packet.

- When the destination node D receives a TESTSIGNAL (no matter it is the first one from the upstream node or not) or ROUTEINFO, node D translates the pseudo-random value to the corresponding power level and records it.

  After collecting all the link cost information, D checks, for each link, that the cost-of-energy parameter does not change. Then D counts how many packets are received at each power level for each link, and estimates the transmission success rates as we show above. Node D picks a power level for each link as we show above, and proceeds to compute the VCG payment.

[8]We assume that the node j gives an acknowledgment signal such as CTS after it receives the packet. We ignore the cost of this acknowledge signal because it is very small.

THEOREM 5. *Suppose that the estimation algorithm of transmission success rates gives accurate results. (That is, the algorithm outputs accurate transmission success rates if the input were really the number of packets received.) If the destination is able to collect all link costs, then the updated protocol presented above is a dominant-subaction solution to the routing stage in the lossy link models.*

Remark The above theorem applies only if the estimation is accurate. If the estimation has certain errors, then theoretically the protocol is no longer a dominant-subaction solution. However, because a node normally does not have sufficient knowledge about the estimation errors in each particular session, it is unlikely that a node can benefit by exploiting such errors. Furthermore, it is possible to achieve approximate dominant-subaction in the sense that the benefit of cheating is small and bounded. When VCG payments or outcomes cannot be computed exactly due to computational or communication complexity, how to archive approximate dominant action is still under active study in algorithmic game theory [32]. We leave this to our future work.

To adapt our forwarding protocol to lossy links, we only need to require that each intermediate node keeps resending each data packet until it is received by the next hop node.

THEOREM 6. *Suppose that the estimating algorithm of transmission success rates gives accurate results. Let $\mathcal{R}$ be the routing decision computed by the routing subactions designated by the protocol in Section 4.3. Assume that, for any node on the packet forwarding path, the computed payment is always greater than the cost. Then, the updated forwarding protocol is a forwarding-optimal protocol to the packet forwarding stage under routing decision $\mathcal{R}$.*

7. EVALUATIONS

In this section we evaluate our protocols.

7.1 Simulation Setup

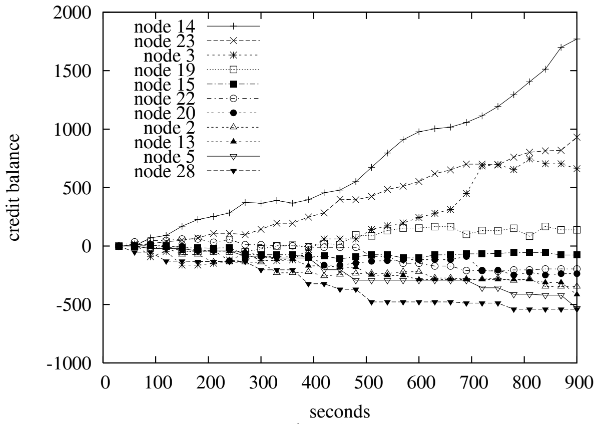
To perform the evaluations, we implement our protocols using the GloMoSim simulation package [17]. Our protocols are implemented in the application layer to allow maximum flexibility. We bypass the routing layer and use source routing. We use IEEE 802.11 (at 2 Mbps) as the MAC layer to capture contentions. We also modify the propagation and radio layer to be able to send at multiple power levels.

We perform simulations in various setups. In this paper we report the results from one typical setup to evaluate and illustrate the behaviors of our protocols.

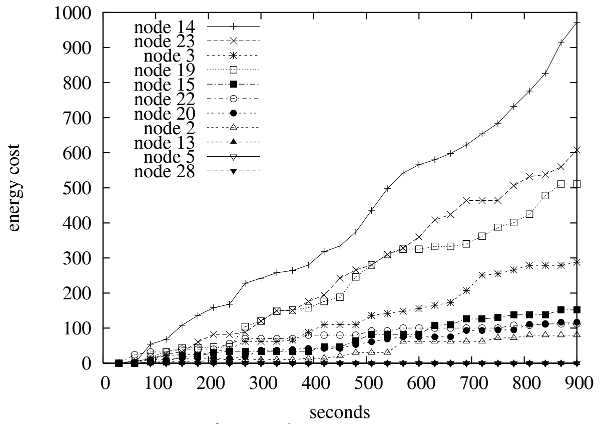
The setup consists of 30 nodes that are randomly distributed in an area of 2000 by 2000 meters. Each node has two transmission power levels at 7 and 14 dBm. The α value of each node is 1. The propagation model is free space model and adding noise does not change the results much. The connectivity of the nodes are shown in Figure 5.

We generate traffic randomly. The start of a session (namely a source-destination pair) at a node (in which this node is the source) is a Poisson arrival. The expected time interval between two sessions from the same node is 60 seconds. The destination of each session is picked uniformly from all nodes except the source. The number of packets in each session is uniformly distributed from 1 to 10, with packet size being 1024 bytes.

7.2 Evaluation Results



(a) credit balance



(b) forwarding energy cost

Figure 4: A network with 30 nodes running for 15 minutes.

We start our evaluation by observing the credit balance of the nodes (namely the total credit received by forwarding others' traffic minus the total credit paid in order to send one's own traffic). Figure 4 (a) shows the credit balances of the nodes for a duration of 15 minutes. The initial credit of each node is 0. We observe that the credit balances of some nodes increase monotonically while those of some other nodes decrease monotonically. Figure 4 (b) shows the accumulative energy the nodes spent in forwarding others' traffic. Comparing Figure 4 (a) with Figure 4 (b), we observe that the nodes accumulating more credits also spend more energy in forwarding others' traffic. Thus it shows that the protocols are fair.

Figure 5 investigates the relationship among credit balance, the total energy spent in forwarding others' traffic, and the position of a node. In this figure, we draw two circles at each node. The area of the solid circle represents the credit balance of a node (after shifting to make all credit balance non-negative), and that of the dashed

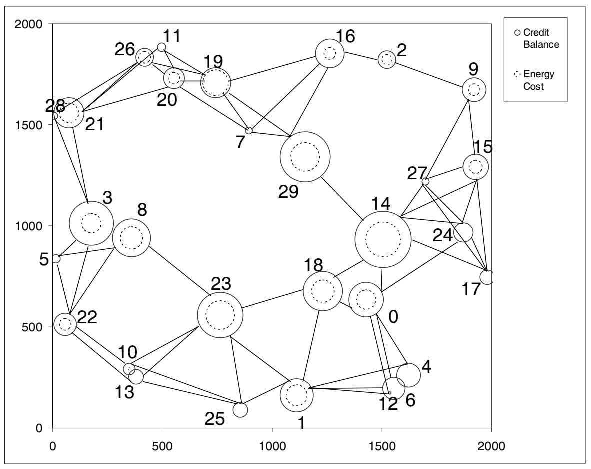
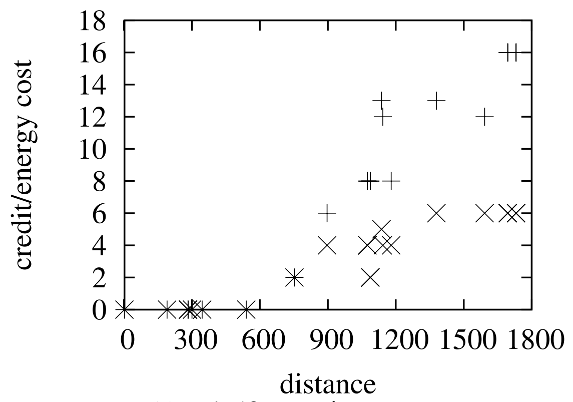


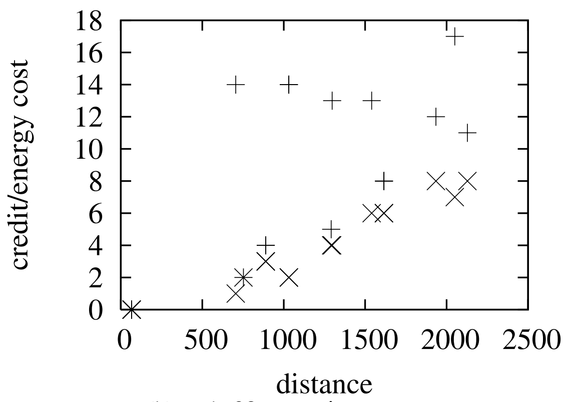
Figure 5: A network with 30 nodes. The ID's of the nodes are labeled. A link between two nodes indicates that they are neighbors. To avoid too many links, links between nodes at close locations are not drawn. The credit balance and forwarding energy cost at the end of 15 minutes are represented by the sizes of the circles.

circle shows the energy the node spent in forwarding others' traffic. We can observe that the position and connectivity of a node are the major factors which determines the number of packets a node forwards as well as the payment it receives for forwarding each packet. In general the nodes in the "center" of the network forward more packets, thus earning more credits. This can be observed from the figure since the larger circles are in general in the center of the network. However, nodes 1, 3, 21, although at the perimeter, also earn more credits because they are on the critical paths of some other nodes.

Figure 6 further investigates the relationship among Euclidean distance of the source and the destination of a session, the payment to the intermediate nodes, and the energy consumed by the intermediate nodes. In this figure, we plot two points for each session. One point has its x coordinate as the Euclidean distance from the source to the destination, and y coordinate as the total credits the source pays; the other point has its x coordinate as the Euclidean distance from the source to the destination, and y coordinate as the total cost the other nodes used to forward packets for this session. It is clear from this figure that payment is almost always higher than cost when there are intermediate nodes forwarding packets. We also observe that payment and forwarding energy cost can exhibit interesting behaviors. For the sessions with node 19 as the source, at short distance, payment and cost are both zero because node 19 can reach its destinations directly. Then, the further away the destination, the higher the forwarding energy cost other nodes spent, and the higher the payment to the intermediate nodes. On the other hand, for sessions with node 28 as the source, although the forwarding energy cost is in general increasing, the payment exhibits interesting behaviors. At low distance, the payment is either very low or very high. The explanation is that if the destination is at the lower half of the network, since node 3 is a critical point, then node 28 needs to make a high payment because the alternative path is



(a) node 19 as session source



(b) node 28 as session source

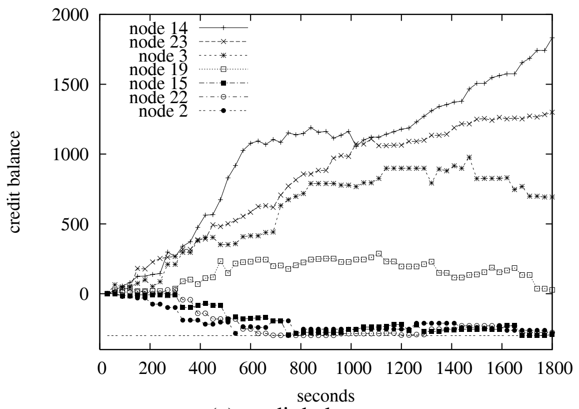
Figure 6: Relationship among Euclidean distance, payment, and forwarding energy cost. The points labeled with + are payment and those with x are forwarding energy cost.

the long path through the upper half of the network; on the other hand, if the destination is at the upper half of the network, the competition between nodes 21 and 26 reduces the payment. At long distance, namely for destinations at the opposite side of the network, node 28 has two alternative paths with similar energy costs; thus the payment can be even lower.

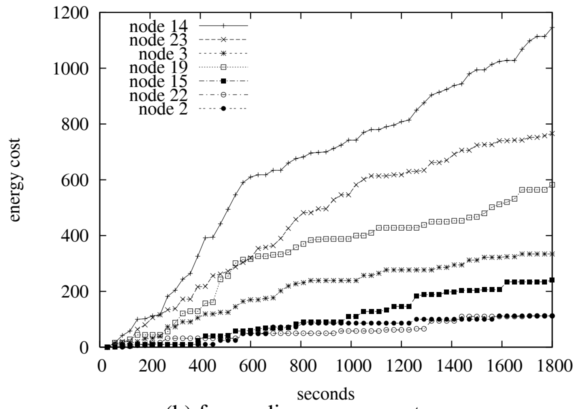
Our system assumes that each node will always forward packets if doing so can maximize its utility, and always generate packets if there is a request for communication from the application layer. One interesting experiment is that a node will no longer generate any new packets after its credit balance is below a threshold. This is reasonable since if a node can have very negative credit balance, then other nodes may not have incentives to forward its packets.

Parts (a) and (b) of Figure 7 show the evolution of credit balances and forwarding energy cost. The threshold is -300. We observe that the threshold prevents the credit balances of nodes 5, 22 and 2 from dropping below -300. As a result, nodes 5, 22 and 2 will stop generating new packets after their balances are below the threshold, forward others' packets to earn credits, and then generate their own packets after they have earned enough credits. A negative effect of this threshold, however, is that it may also reduce the total throughput of the network. Figure 8 verifies the reduction of the total packets delivered in the network. We observe that at the beginning the network with the threshold and that without the threshold achieve similar throughput. However, as time evolves, the threshold approach clearly slows down.

Finally we study the effects of cheating. Since we have already established the incentive-compatibility of our protocols, the results are mainly to illustrate the negative effects of cheating. Specifically, we study the effects when an intermediate node tries to cheat by falsely reporting the costs of the links from itself to its neighbors. This can be done by sending values that are either higher or lower than the true costs in the transmitted TESTSIGNAL's. Parts (a) and (b) of Figure 9 show the evolutions of credit balances and forwarding energy cost of node 3 in four different settings: node 3 cheats or is honest, and the other nodes cheat or are honest. In these evaluations, a node cheats by sending a cost that is higher than the true cost; the results for sending a cost that is lower than true cost are worse since packets may be dropped. For the settings where the other nodes cheat, a node cheats with probability 0.5. We observe from the figures that node 3 accumulates the highest amount of credits when it is honest and the others try to cheat. On the other

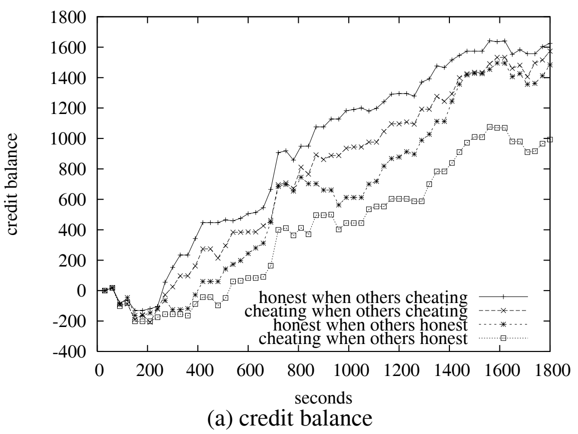


(a) credit balance

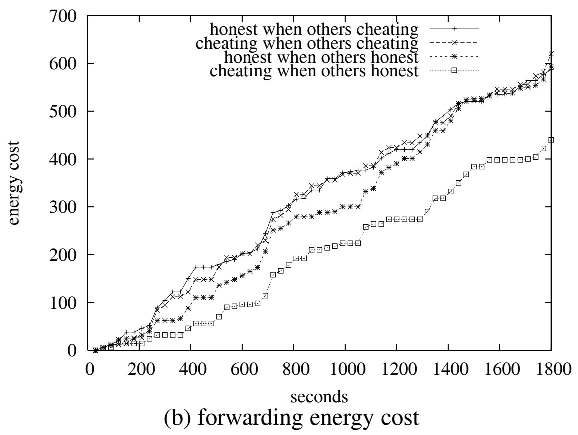


(b) forwarding energy cost

Figure 7: A network with 30 nodes running for 30 minutes with balance threshold.



(a) credit balance



(b) forwarding energy cost

Figure 9: Node 3 cheats or follows the protocols when the other nodes in the network cheat or follow the protocols.

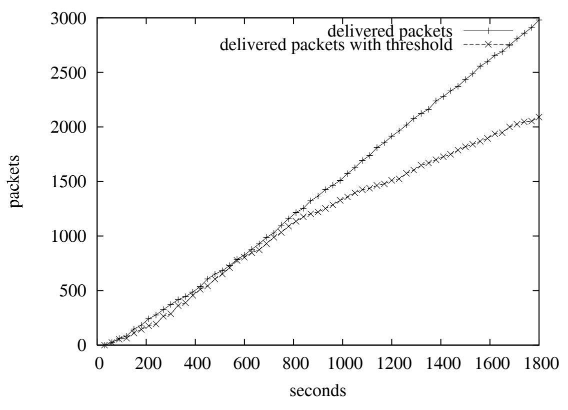


Figure 8: Reduction in throughput after using threshold.

hand, when it tries to cheat but the others are honest, node 3 accumulates the least amount of credits. It is clear that following the protocols brings the highest utility to node 3.

8. CONCLUSION AND FUTURE WORK

Wireless ad-hoc networks are often formed by nodes belonging to independent entities. These nodes do not have to cooperate unless they have incentives to do so. Therefore, both routing and packet forwarding become games. We propose the feasible notion of cooperation-optimal protocols and design the first incentive-compatible, integrated routing and forwarding protocol in wireless ad-hoc networks. Combining incentive mechanisms and security techniques to address the issue that a link's cost is not private but is determined by two nodes, we design novel routing protocols for both deterministic link models and probabilistic link models. We show that following the protocols is a dominant action for this stage. We also show that there does not exist a forwarding-dominant protocol. The implication of this result is that, forwarding others' traffic may not always result in maximal utility for a node. A node may choose not to forward in some cases in order to maximize its utility, depending on other nodes' actions. We propose an efficient forwarding protocol based on the use of hash chains in cryptography to deliver payments. Our simulation results demonstrate that our protocols provide incentives for node to forward packets.

There are many avenues for further exploration. In this paper, we considered only the integration of routing and forwarding. If a network is capacity limited, nodes may charge congestion price in addition to energy cost. Thus, we may have to deal with incentive issues in media access control (*e.g.*, [7]) and congestion control as well. Application layer may also have its incentive issues [18]. Of particular interest is a general model that can integrate solutions for these layers with the solution proposed in this paper. We have assumed fairly general node action space. A possible research direction is to design practical and efficient solutions with the support of secure hardware to limit a node's action space. Other interesting and related open questions include key management in ad-hoc network, issues related with node churns, mobility, intermittently reachable nodes, colluding selfish nodes, and malicious nodes. In particular, our use of cryptographic techniques to solve game theory problems in wireless ad-hoc networks is novel. This paper shows the first example and we believe that it is a promising direction and can be applied in many other settings such as congestion control games.

Acknowledgments

We thank the students at Yale who took CS434/534 in the spring of 2004 for their feedback to our protocols. We thank Joan Feigenbaum for giving us valuable comments. We are also grateful to the reviewers and the shepherd Michael Wallbaum for their valuable comments which improve the paper.

9. REFERENCES

[1] L. Anderegg and S. Eidenbenz. Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *Proceedings of the Ninth International Conference on Mobile Computing and Networking (Mobicom)*, San Diego, CA, Sept. 2003.

[2] H. Balakrishnan and R. Katz. Explicit loss notification and wireless web performance. In *Proceedings of IEEE Globecom Internet Mini-Conference*, 1998.

[3] S. Buchegger and J.-Y. Le Boudec. Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks. In *Proceedings of the Tenth Euromicro*

Workshop on Parallel, Distributed and Network-based Processing, 2002.

[4] S. Buchegger and J.-Y. Le Boudec. Performance analysis of the CONFIDANT protocol: Cooperation of nodes - fairness in dynamic ad-hoc networks. In *Proceedings of the Third ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Lausanne, Switzerland, June 2002.

[5] L. Buttyan and J. P. Hubaux. Enforcing service availability in mobile ad-hoc WANs. In *Proceedings of the First ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Boston, Massachusetts, Aug. 2000.

[6] L. Buttyan and J. P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM Journal for Mobile Networks (MONET), special issue on Mobile Ad Hoc Networks*, summer 2002.

[7] M. Cagalj, S. Ganeriwal, I. Aad, and J. P. Hubaux. On selfish behavior in csma/ca networks. In *Proceedings of IEEE INFOCOM*, Miami, Florida, USA, March 2005.

[8] Cisco Systems Inc. Data sheet for cisco aironet, 2004.

[9] E. Clarke. Multipart pricing of public goods. *Public Choice*, 11:17–33, 1971.

[10] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the Ninth International Conference on Mobile Computing and Networking (Mobicom)*, San Diego, CA, Sept. 2003.

[11] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. In *Proceedings of the 21st Symposium on Principles of Distributed Computing*, Monterey, CA, July 2002.

[12] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences (Special issue on Internet Algorithms)*, 63:21–41, 2001.

[13] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the Sixth International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, ACM Press, Sept. 2002.

[14] J. Feigenbaum and S. Shenker. Incentives and Internet algorithms. Tutorial given at PODC 2003. Available at `http://www.cs.yale.edu/~jf/PODC03.ppt`, July 2003.

[15] M. Felegyhazi, L. Buttyan, and J. P. Hubaux. Equilibrium analysis of packet forwarding strategies in wireless ad hoc networks – the static case. In *Proceedings of Personal Wireless Communications (PWC)*, Venice, Italy, 2003.

[16] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report UCLA/CSD-TR 02-0013, Computer Science Department, UCLA, July 2002.

[17] GloMoSim. `http://pcl.cs.ucla.edu/projects/glomosim/`.

[18] M. Goemans, L. E. Li, V. S. Mirrokni, and M. Thottan. Market sharing games applied to content distribution in ad-hoc networks. In *Proceedings of the Fifth ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, New York, NY, USA, 2004.

[19] O. Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, Aug. 2001.

[20] T. Groves. Incentives in teams. *Econometrica*, 41:617–663, 1973.

[21] J. Hershberger and S. Suri. Vickrey prices and shortest paths: What is an edge worth? In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, Las Vegas, Nevada, Oct. 2001.

[22] E. Huang, J. Crowcroft, and I. Wassell. Rethinking incentives for mobile ad hoc networks. In *Proceedings of ACM SIGCOMM Workshop on Practice and Theory of Incentives and Game Theory in Networked Systems*, Portland, OR, Sept. 2004.

[23] M. Jakobsson. Ripping coins for a fair exchange. In *Advances in cryptology, EUROCRYPT*, 1995.

[24] M. Jakobsson, J. P. Hubaux, and L. Buttyan. A micropayment scheme encouraging collaboration in multi-hop cellular networks. In *Proceedings of Financial Crypto*, La Guadeloupe, Jan. 2003.

[25] A. Konrad, B. Zhao, A. Joseph, and R. Ludwig. Explicit loss notification and wireless web performance. In *Proceedings the Fourth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Rome, Italy, July 2001.

[26] D. M. Kreps. *Game Theory and Economic Modeling*. Oxford Press, 1991.

[27] Y.-B. Lin and I. Chlamtac. *Wireless and Mobile Network Architectures*. John Wiley and Sons, 2000.

[28] H. Luo, R. Ramjee, P. Sinha, L. Li, and S. Lu. UCAN: A unified cellular and ad-hoc network architecture. In *Proceedings of the Ninth International Conference on Mobile Computing and Networking (Mobicom)*, San Diego, CA, Sept. 2003.

[29] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Experiences applying game theory to system design. In *Proceedings of ACM SIGCOMM Workshop on Practice and Theory of Incentives and Game Theory in Networked Systems*, Portland, OR, Sept. 2004.

[30] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the Sixth International Conference on Mobile Computing and Networking (Mobicom)*, Boston, MA, Aug. 2000.

[31] P. Michiardi and R. Molva. Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc network. In *Proceedings of Communications and Multimedia Security Conference (CMS)*, Portoroz, Sept. 2002.

[32] N. Nisan and A. Ronen. Computationally feasible VCG mechanisms. In *Proceedings of the ACM Symposium on Electronic Commerce (EC)*, Minneapolis, MN, Oct. 2000.

[33] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.

[34] M. J. Osborne and A. Rubenstein. *A Course in Game Theory*. The MIT Press, 1994.

[35] C. Papadimitriou. Algorithms, games, and the Internet. In *Proceedings of the 33rd Annual Symposium on Theory of Computing*, Heraklion, Crete, Greece, July 2001.

[36] C. Perkins. *Ad Hoc Networking*. Addison-Wesley, 2000.

[37] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2nd edition, Dec. 2001.

[38] I. Ray and I. Ray. Fair exchange in e-commerce. *SIGecom Exchange*, 3(2):9–17, 2002.

[39] N. B. Salem, L. Buttyan, J. P. Hubaux, and M. Jakobsson. A charging and rewarding scheme for packet forwarding in

multi-hop cellular networks. In *Proceedings of the Fourth ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Annapolis, MD, June 2003.

[40] V. Srinivasan, P. Nuggehalli, C.-F. Chiasserini, and R. Rao. Cooperation in wireless ad hoc networks. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, Apr. 2003.

[41] D. R. Stinson. *Cryptography: Theory and Practice*. CRC Press, 1995.

[42] C.-K. Toh. *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. Prentice Hall PTR, 2001.

[43] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.

[44] W. Wang, X.-Y. Li, and Y. Wang. Truthful multicast in selfish wireless networks. In *Proceedings of the Tenth International Conference on Mobile Computing and Networking (Mobicom)*, Philadelphia, PA, Sept. 2004.

[45] A. Woo and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angles, CA, Nov. 2003.

[46] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angles, CA, Nov. 2003.

[47] S. Zhong, J. Chen, and Y. R. Yang. Sprite, a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, Apr. 2003.