Yale University
Department of Computer Science

On Designing Incentive-Compatible Routing and Forwarding Protocols in Wireless Ad-Hoc Networks

— An Integrated Approach Using Game Theoretical and Cryptographic Techniques

Sheng Zhong[1] Li Li[2] Yanbin Liu[3]

Yang Richard Yang[4]

YALEU/DCS/TR-1286
March 15, 2004

[1]Computer Science Department, Yale University, New Haven, CT 06520-8285, USA. Email: sheng.zhong@yale.edu. Supported in part by NSF.

[2]Bell Labs, New Jersy, CT 06520-8285, USA. Email: li.li@bell-labs.com.

[3]Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712, USA. Email: ybliu@cs.yale.edu.

[4]Computer Science Department, Yale University, New Haven, CT 06520-8285, USA. Email: yry@cs.yale.edu. Supported in part by NSF grants ANI-0207399 and ANI-0238038.

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 Li† Yanbin Liu‡ Yang Richard Yang§

Abstract

In many applications, a wireless ad-hoc network is formed by devices belonging to independent users. Therefore, a challenging problem is 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, we conduct the first comprehensive study on both routing and forwarding. We first uncover a surprising 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 the concept of a *cooperation-optimal protocol* and present the design of such a protocol, which consists of a routing protocol and a forwarding protocol. Although our routing protocol also uses the VCG mechanism to compute a lowest cost path for routing, we integrate 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. We also apply 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 expected 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.

*Computer Science Department, Yale University, New Haven, CT 06520-8285, USA. Email: sheng.zhong@yale.edu. Supported in part by NSF.

†Bell Labs, New Jersy, CT 06520-8285, USA. Email: li.li@bell-labs.com.

‡Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712, USA. Email: ybliu@cs.yale.edu.

§Computer Science Department, Yale University, New Haven, CT 06520-8285, USA. Email: yry@cs.yale.edu. Supported in part by NSF grants ANI-0207399 and ANI-0238038.

Our evaluations demonstrate that our protocols provide incentives for nodes to forward packets.

1 Introduction

Many wireless ad-hoc networks are currently being designed or deployed, driven by the vision of any-time, any-where connectivity [27, 34, 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 other's traffic.

To stimulate nodes to forward others' traffic, many methods have recently been proposed (*e.g.*, [4–7,23,29,37,40,47]). 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.*, [2, 4–7, 23, 37, 40, 47]). Although much progress has been made in the last few years, several *fundamental* issues remain unaddressed.

First of all, the previous approaches focus either on the routing component (*e.g.*, [2]) or the packet forwarding component (*e.g.*, [16, 23, 47]). There is no previous system which integrates both routing and forwarding. 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 [36]. Although both individual components are challenging, it is more challenging to design and analyze a complete system that integrates both routing and forwarding.

Next consider the forwarding component. 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 [32]; 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 [14] and it is hard to make the system converge to a desirable Nash equilibrium in distributed settings [25]. 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?

Third, because wireless ad-hoc networks have their unique properties, 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 mechanism [9, 20, 43], which has been applied to route discovery in wireless ad-hoc networks [2]. 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.*, [13, 31, 33]) and to routing protocols in particular (*e.g.*, [2, 12]), 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 com-

munity 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 [2] is flawed because it does not properly handle cheating in estimating power levels.

Fourth, the previous work (*e.g.*, [2]) 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, 17, 45, 46]. We refer to such links as *lossy links*. It is not known how to deal with lossy links in routing and forwarding.

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

First, we 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 and thus 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 does not cooperate. Given the impossibility result and the previous misunderstanding of dominant actions in wireless ad-hoc networks, we need to search for a feasible solution concept in the context of wireless ad-hoc networks.

Second, we define the concept of a *cooperation-optimal protocol* for non-cooperative selfish users in a wireless ad-hoc network. A cooperation-optimal protocol 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 an efficient 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 an efficient 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 [3, 10, 17, 24, 45, 46]. In these models, packet reception is probabilistic and the probability is a function of the transmission power.

We also 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 others' 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 the highest 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 9.

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 [32]. 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 —the *communication program* used by this node which specifies what messages to generate, what messages to forward, what messages to discard, and at which power level to send each message, etc. As a notational convention, we use a_{-i} to denote the actions of all nodes except node i. Note that a_{-i} is a vector. Sometime we write (a_i, a_{-i}) to denote the action profile 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. Control packets can also be dealt with using a method such as that in [2].

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:

$$c_i = c_i(a);$$

$$p_i = p_i(a).$$

[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.

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 [32]. 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 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 3 *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;
- $P_{i,l} = \infty$, for any $l \neq j$, which means that any other node cannot receive any packet sent by node i.[2]

Fig. 1 shows the setup.

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

[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 an 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.

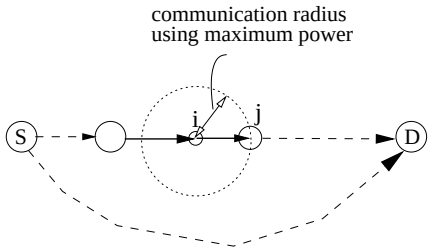

Figure 1: Illustration of the setup for the impossibility result.

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.

Remark The above theorem is valid not only in our model, but also *in many alternative models*. For example, although our model assumes asymmetric links, the above theorem is also valid with symmetric links, 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.

The routing and forwarding behavior of a node occurs in two stages: the routing stage and the forwarding stage. Thus we define two *inter-dependent* subgames: the *routing subgame* and the *forwarding subgame*. Accordingly, each node's action in the ad-hoc game is divided into two parts: its subaction in the routing subgame and its subaction in the packet forwarding subgame. In the routing subgame, 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 subgame. In the forwarding subgame, the routing decision (*i.e.*, what each node is supposed to do in this subgame) and the nodes' forwarding subactions (*i.e.*, what each node really does in this subgame) 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 subgame, and $a_i^{(f)}$ is i's subaction in the forwarding subgame. 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 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 idea of dividing a game into subgames has been suggested by Feigenbaum and Shenker in their PODC tutorial slides [15]. This paper provides the first concrete example showing the feasibility of the approach.

3.1 Defining Solution Concept to the Routing Subgame

Definition 4 *A node's* prospective routing utility *is the utility it will achieve under a routing decision if all nodes in the packet forwarding subgame follow the routing decision (*i.e.*, if each node takes the forwarding subaction designated by the routing decision). Formally, let $\mathcal{R}(=\hat{a}_i^{(f)})$ be a routing decision. Then node i's prospective routing utility is*

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

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

$$u_i^{(p)} = u_i^{(p)}(a^{(r)}).$$

Definition 5 *In a routing subgame, 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 subgame. Formally, $a_i^{(r)}$ is node i's dominant subaction in the routing subgame if, for any $\bar{a}_i^{(r)} \neq a_i^{(r)}$, any $a_{-i}^{(r)}$*

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

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 only require Equation (1) to hold *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 and focus on motivating nodes to forward traffic (*e.g.*, [2, 12, 21, 31]); therefore the definition applies only to the potential forwarding nodes.

Definition 6 *A routing protocol is a* routing-dominant protocol to the routing subgame *if following the protocol is a dominant subaction of each potential forwarding node in the routing subgame.*

To finish defining the routing subgame, 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,

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

it is likely to be reliable. This is particularly true in hybrid architectures such as [23, 28, 37], where the destination is a base station. If there is the 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, for example the destination is removed from the system. To prevent potential denial of service attack on such validation processes, the number of sessions that can be sampled by a node should be limited.

3.2 Defining Solution Concept to the Forwarding Subgame

Definition 7 *In a forwarding subgame, an* optimal subaction *of a node under routing decision $\mathcal{R}$ is one that maximizes its expected utility under routing decision $\mathcal{R}$. Formally, $a_i^{(f)}$ is node i's optimal subaction in the forwarding subgame if, for any $\bar{a}_i^{(f)} \neq a_i^{(f)}$, $E[u_i \| \mathcal{R}, a_i^{(f)}] \geq E[u_i \| \mathcal{R}, \bar{a}_i^{(f)}]$.*

Remark The above definition depends on the underlying probability distribution. While a dominant subaction always maximizes the utility no matter what others do, an optimal subaction maximizes only the expected value of the utility. Therefore, although dominant subactions may not exist for the forwarding subgame, there always exists an optimal subaction for a given probability distribution. For practical purposes, optimal subactions are good enough for packet forwarding.

Definition 8 *A forwarding protocol is a* forwarding-optimal protocol to the forwarding subgame *under routing decision $\mathcal{R}$ if all packets are forwarded to their destinations in this protocol and following the protocol is an optimal subaction of each player under routing decision $\mathcal{R}$ in the forwarding subgame.*

3.3 Defining Solution Concept to the Ad-hoc Game

Definition 9 *A protocol is a* cooperation-optimal protocol *to an ad-hoc game if*

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

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

4 A Routing Protocol for the Routing Subgame

In this section, we present a protocol for the routing subgame. The routing decision is based on the well-known Vickrey-Clark-Groves (VCG) mechanism. However, we will show that, a straightforward application of VCG to this problem (*e.g.*, the Ad-Hoc VCG protocol [2]) is *not* a dominant-subaction solution due to the special properties of wireless ad-hoc networks. To construct a dominant-subaction solution, we need to combine VCG with a novel cryptographic technique.

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 [2]. We omit some details of [2] 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 [2] is flawed; therefore their overall protocol does not preserve incentive compatibility.

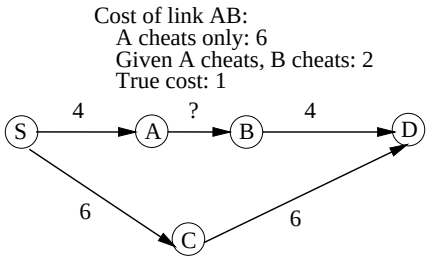

Figure 2: Illustration: VCG alone does *not* guarantee the existence of a dominant-subaction solution in routing.

Consider the link-cost estimation algorithm used in the Ad-hoc VCG protocol (see Equation (2) of [2]). 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 Fig. 2 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]

[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.

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 [2] is invalid in the case illustrated above. In the proof of their Lemma 2, they argue that a node like B in Fig. 2 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 Fig. 2, 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 [12] 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, node j has no way to report such a test signal to the destination.[6] Finally, the destination "translates" node j's reported test 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 node i computes, for each available power level l (in increasing order)

$$h_l = E(k_{i,D}; [[S, D, r, l, \alpha_i], \sigma]),$$

where E is a good symmetric encryption algorithm, and σ a Message Authentication Code (MAC) of $[S, D, r, l, \alpha_i]$ using key $k_{i,D}$. Clearly, only i and D can compute these test signals (h_l's). 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 Subgame

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.

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.

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

- 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

$$h_l = E(k_{S,D}; [[S, D, r, l, \alpha_S], \sigma]),$$

  where σ is a MAC of $[S, D, r, l, \alpha_S]$ using key $k_{S,D}$.

4.3.2 Intermediate node's test signals

Upon receiving (TESTSIGNAL, [S, D, r], [P, h]), an intermediate node i does the follows. (Suppose that this packet comes from upstream neighbor P.)

- If this is the first TESTSIGNAL received for session (S, D, r), node i does the following:

  – 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

$$h'_l = E(k_{i,D}; [[S, D, r, l, \alpha_i], \sigma]),$$

    where σ is a MAC of $[S, D, r, l, \alpha_i]$ using key $k_{i,D}$.

- If this is a TESTSIGNAL received from neighbor P for session (S, D, r), then i does the following:

  – Node i sends out (ROUTEINFO, [S, D, r], [P, i, h']) at power level P_{ctr}. Here h' is computed by encrypting h using key $k_{i,D} = (K_D)^{k_i}$. For integrity, this message is protected by a MAC using key $k_{i,D}$.

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 $k_{P,D} = (K_P)^{k_D}$, 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 $k_{i,D} = (K_i)^{k_D}$, 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 the 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. Denote 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

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

4.3.5 Messaging and computational complexity

It is clear that for each session, each node needs to announce TESTSIGNAL's at different power levels. In the worst case each node needs to forward each ROUTEINFO once. Thus in the worst case the total control message complexity is $O(|\mathcal{P}| \cdot E \cdot N)$, where $|\mathcal{P}|$ is the number of power levels, E the number of links, and N the number of nodes. The destination runs Dijkstra's algorithm, which has a computational complexity of $O(N^2 + E)$.

4.4 Analysis of the Routing Protocol

Theorem 10 *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 subgame.*

Proof: Consider node i. Let $a_i^{(r)}$ be the subaction of node i in the routing subgame that follows the protocol faithfully. Let $\bar{a}_i^{(r)} \neq a_i^{(r)}$ be a different subaction. Let $a_{-i}^{(r)}$ be an arbitrary subaction profile of all other nodes in this subgame. 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. ∎

5 A Secure Framework for the Packet Forwarding Subgame

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* [22]. 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 power level $\max \mathcal{P}$. (In this section, we assume that a node can always reach any of its upstream neighbors at power level $\max \mathcal{P}$. If this is not the case, we can use alternative approaches to forward the routing decision, for example by flooding.)

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 the node can get a payment of $p_i \cdot b \cdot m$ 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 node submits the routing decision to the system so that the system charges the source node $p_i \cdot b$ in addition. Note that this amount of credit does not go to the intermediate node, but goes to the system.

5.3 Analysis of the Packet Forwarding Protocol

We define $\mathcal{C}$ to be the set of (all possible) path costs:

$$\mathcal{C} = \{\sum_{i=1}^{n} C_i : C_i = 0 \text{ or } C_i = l \cdot \alpha_j, l \in \mathcal{P}, j \in \mathcal{N}\}.$$

Note that $\mathcal{C}$ is a finite set because the path length is bounded by n.

Theorem 11 *Suppose that $\mathcal{R}$ is a routing decision computed by the routing subactions designated by the protocol in Section 4.3. Assume that, with probability at least G a node follows the protocol. Also assume that for any node on the packet forwarding path, the computed payment is greater than the cost. If*

$$\frac{G^{n-1}}{1 - G^{n-1}} \geq \frac{\max \mathcal{P} \cdot \max_j \alpha_j}{\min\{C - C' : C, C' \in \mathcal{C}, C > C'\}},$$

then the protocol presented in Section 5.2 is an forwarding-optimal protocol to the packet forwarding subgame under $\mathcal{R}$.

Proof: Let $a_i^{(f)}$ be node i's forwarding subaction designated by the protocol in Section 5.2. We need to show that, for any $\bar{a}_i^{(f)} \neq a_i^{(f)}$, $E[u_i \| \mathcal{R}, a_i^{(f)}] \geq E[u_i \| \mathcal{R}, \bar{a}_i^{(f)}]$.

Consider the difference between $a_i^{(f)}$ and $\bar{a}_i^{(f)}$. Because we ignore the cost of sending control packets, we only need to consider the difference between these two

subactions in forwarding data packets. (That is, if these two subactions are different with respect to computing and sending control packets, we can always update $\bar{a}_i^{(f)}$ without changing $E[u_i \| \mathcal{R}, \bar{a}_i^{(f)}]$.) Therefore, without loss of generality, suppose that $a_i^{(f)}$ and $\bar{a}_i^{(f)}$ are only different with respect to forwarding one data packet. (If they are different with respect to forwarding more than one data packet, we consider one at a time.) With probability of at least G^{n-1}, $a_i^{(f)}$ leads to a payment. Let $a^{(r)}$ be the dominant subaction in the routing subgame that generates routing decision $\mathcal{R}$. In this case,

$$\begin{aligned} & u_i(a^{(r)}, a_i^{(f)}, a_{-i}^{(f)}) - u_i(a^{(r)}, \bar{a}_i^{(f)}, a_{-i}^{(f)}) \\ \geq \; & \min\{C - C' : C, C' \in \mathcal{C}, C > C'\}. \end{aligned}$$

On the other hand, with probability of at most $1 - G^{n-1}$, $a_i^{(f)}$ leads to no payment. In this case,

$$u_i(a^{(r)}, \bar{a}_i^{(f)}, a_{-i}^{(f)}) - u_i(a^{(r)}, a_i^{(f)}, a_{-i}^{(f)}) \leq \max \mathcal{P} \cdot \max_j \alpha_j.$$

In summary,

$$\begin{aligned} & E[u_i \| \mathcal{R}, a_i^{(f)}] - E[u_i \| \mathcal{R}, \bar{a}_i^{(f)}] \\ \geq \; & G^{n-1} \cdot \min\{C - C' : C, C' \in \mathcal{C}, C > C'\} \\ & - (1 - G^{n-1}) \cdot \max \mathcal{P} \cdot \max_j \alpha_j \geq 0. \end{aligned}$$

Equivalently, $E[u_i \| \mathcal{R}, a_i^{(f)}] \geq E[u_i \| \mathcal{R}, \bar{a}_i^{(f)}]$. ∎

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. The inequality condition is true in many ad-hoc network settings. This is because payment is integer and small. A node's cost-of-energy parameter, namely its α value, is also likely to be small. Therefore, if G is not too small, then the condition is true for a wide range of network confi gurations.

Theorem 12 *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 10 and 11. ∎

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.*, [3, 10, 17, 24, 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 $\hat{S}_{i,j}(1) = x(1)$.

[7] We may even know more than that. For example, certain analytical expressions can be derived for the transmission success rate in some link model [35]. However, because these models are still under investigation, we do not utilize such information.

- 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:
  $$\hat{S}_{i,j}(l) = ((l'' - l)\hat{S}_{i,j}(l') + (l - l')\hat{S}_{i,j}(l''))/(l'' - l').$$
  - 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 13 *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.

[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.

6.3 Adapting Protocol to Lossy Links

Using the results presented above, we can easily adapt our protocol to lossy link models. In the routing subgame, 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.

Theorem 14 *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.) Then the updated protocol presented above is a dominant-subaction solution to the routing subgame 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 benefi t by exploiting such errors. Furthermore, it is possible to achieve approximate dominant-subaction in the sense that the benefi t 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 [30]. 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 15 *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. If*

$$\frac{G^{n-1}}{1-G^{n-1}} \geq \frac{\max \mathcal{P} \cdot \max_j \alpha_j}{\min\{C - C' : C, C' \in \mathcal{C}, C > C'\}},$$

then the updated forwarding protocol is a forwarding-optimal protocol to the packet forwarding subgame 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 [18]. 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 Fig. 4.

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

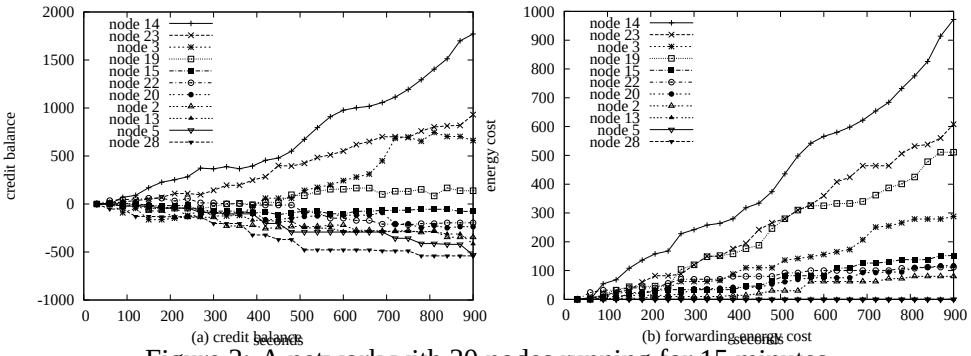

Figure 3: A network with 30 nodes running for 15 minutes.

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). Fig. 3 (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. Fig. 3 (b) shows the accumulative energy the nodes spent in forwarding others' traffic. Comparing Fig. 3 (a) with Fig. 3 (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.

Fig. 4 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 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,

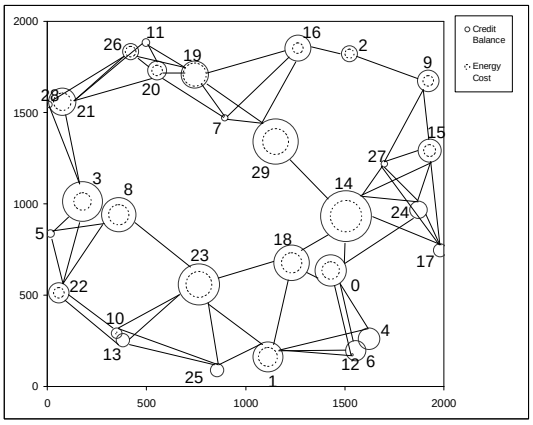

Figure 4: A network with 30 nodes. The ID's of the nodes are labeled. A link between two nodes indicates that they are neighbors. To avoid too many links, links between nodes at close locations are not drawn. The credit balance and forwarding energy cost at the end of 15 minutes are represented by the sizes of the circles.

nodes 1, 3, 21, although at the perimeter, also earn more credits because they are on the critical paths of some other nodes.

Fig. 5 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 the long path through the

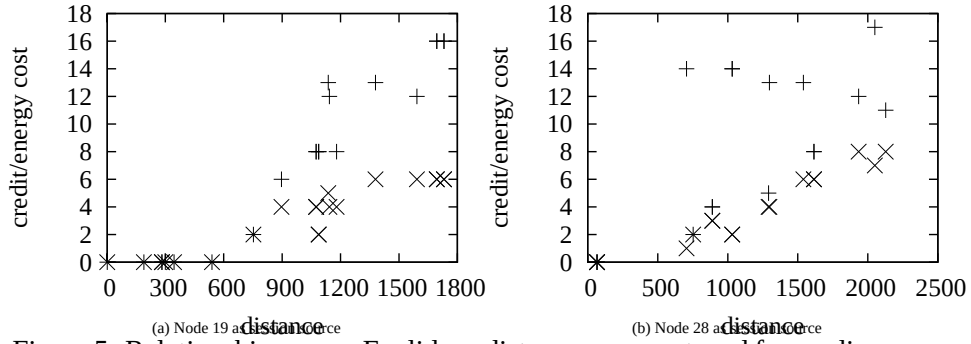

Figure 5: Relationship among Euclidean distance, payment, and forwarding energy cost. The points labeled with + are payment and those with x are forwarding energy cost.

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.

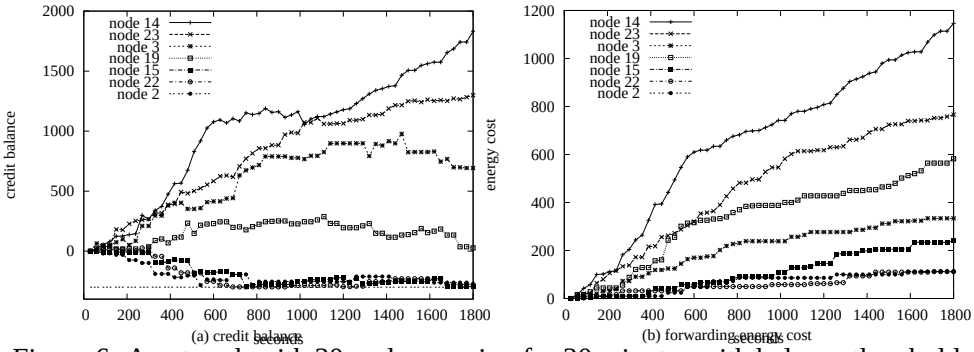

Figure 6: A network with 30 nodes running for 30 minutes with balance threshold.

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 the node may not have incentives to forward others' packets.

Parts (a) and (b) of Fig. 6 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,

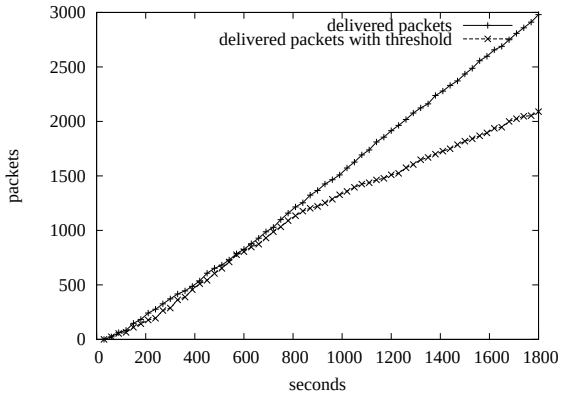

Figure 7: Reduction in throughput after using threshold.

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. Fig. 7 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.

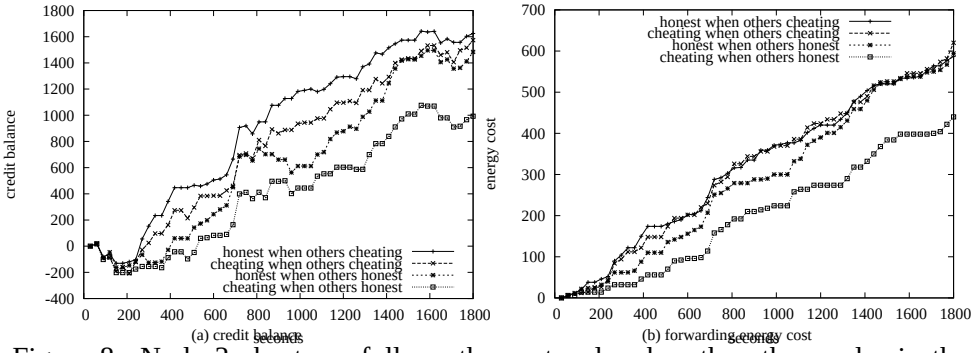

Figure 8: Node 3 cheats or follows the protocols when the other nodes in the network cheat or follow the protocols.

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 Fig. 8 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 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 Related Work

The problem of how to stimulate cooperation among selfish nodes in ad-hoc networks and multi-hop cellular networks has received significant attention recently. The related work can be classified roughly into two categories: reputation-based approaches and credit-based approaches.

Reputation-based approaches: Marti et al. [29] propose the first reputation-based approach in wireless ad-hoc networks. Their system considers misbehaving nodes, including both selfish and malicious nodes. In order to cope with misbehaving nodes, they propose two tools: a watchdog, which identifies misbehaving nodes, and a pathrater, which selects routes that avoid the identified nodes. They show that in their simulation setup these two tools can maintain the total throughput of an ad-hoc network at an acceptable level even with a large percentage of misbehaving nodes.

In [4,5], Buchegger and Le Boudec propose another reputation-based approach. In their CONFIDANT protocol, each node maintains a state machine to keep track the reputation of other nodes, and trust relationships and routing decisions are made based on experienced, observed, or reported routing and forwarding behavior of other nodes.

Although reputation-based approaches can perform well in some scenarios, there are several issues limiting their usage. First, there is no formal specification and analysis of the type of incentive provided by such approaches. Thus it is unclear how the approaches will perform in all scenarios. Second, due to the selfish nature of the nodes, it is unclear that nodes will cooperate in punishing misbehaving nodes. Third, some of the current approaches depend on the broadcast nature of wireless networks in order to monitor other nodes. Such monitoring, however,

may not always be feasible due to asymmetric links when nodes use power control. Furthermore, directional antennas [39,44], which are starting to be used in wireless networks in order to improve capacity, will also make monitoring less feasible.

In [40], Srinivasan et al. propose and study the Generous TIT-FOR-TAT (GTFT) strategy in wireless ad-hoc networks. The decision of the GTFT strategy is based on a node's relative performance compared with other nodes. They rigorously prove that GTFT results in a Nash equilibrium. Although GTFT as a mathematical framework has the potential of becoming a simple and effective method for prompting cooperation, further research is required to devise a practical algorithm to implement the strategy.

Credit-based approaches: Buttyan and Hubaux propose the first credit-based system [6, 7] in wireless ad-hoc networks in the Terminodes project. In [6], they propose the usage of nuglets, a virtual currency, to pay nodes to forward other's packets. Although the approach may work well in some scenarios, because the approach requires tamper-proof hardware, it may not be suitable in many other scenarios.

Motivated by the nuglet approach, several credit-based systems that do not require tamper-proof hardware are proposed. In [47], Zhong et al. propose Sprite, a credit-based system which uses a central authority to collect receipts from forwarding nodes. Charges and rewards are based on the receipts. Our packet forwarding protocol is motivated by Sprite. However, our protocol uses chained hash functions and thus is more efficient. Ben Salem et al. [37] propose a charging and rewarding scheme based on symmetric cryptography to make collaboration rational for selfish nodes. In [23], Jakobsson et al. propose a micro-payment scheme for multi-hop cellular networks to encourage collaboration in packet forwarding. A requirement of both [37] and [23] is that all data packets go through a base station. Thus there may be scenarios where the schemes may not apply.

The focus of the preceding credit-based approaches is mainly on packet forwarding. In [2], Anderegg and Eidenbenz discover the routing problem and apply the VCG mechanism to design a routing protocol that is guaranteed to find the most cost-efficient path. However, their result holds only for a restricted set of possible actions.

Other related game-theoretic results: Recently game theory has also been applied to many other problems in computer networks, *e.g.*, [1, 12, 14, 31, 33]. In wireless networks, Eidenbenz [11] proposes a topology control game that models user's selfish behavior in forming ad-hoc network topology. In [26], Lin et al. use game theory to propose an admission and rate control framework for CDMA data networks. These studies are orthogonal to our study, which focuses on routing and packet forwarding. There are some recent discussions in mechanism design on general solution concepts that are related with this paper. For example, Feigen-

baum and Shenker suggest the possibility of two-step games in their PODC 2003 tutorial (see page 89 of [15]). In [38], Shneidman and Parkes discuss using redundancy to improve the robustness of mechanism design. These discussions are in the context of general mechanism design and show that our two-step solution might extend to a more general setting.

9 Conclusion

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, one of the important network functions —routing and packet forwarding becomes a game. We uncover one important impossibility result —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.

Motivated by the above result, 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 subgame. We also 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. Of particular interest is the use of cryptographic techniques to solve game theory problems in wireless ad-hoc networks. 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.

References

[1] A. Akella, S. Seshan, R. Karp, and S. Shenker. Selfish behavior and stability of the internet: Game-theoretic analysis of TCP. In *Proceedings of ACM SIGCOMM '02*, Pittsburgh, PA, August 2002.

[2] 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, Sep 2003.

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

[4] S. Buchegger and J.-Y. Le Boudec. Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks. In *10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, 2002.

[5] 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 9-11 2002.

[6] 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, August 11 2000.

[7] 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.

[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, Sep 2003.

[11] S. Eidenbenz, V. S. A. Kumar, and S. Zust. Equilibria in topology control games for ad hoc networks. In *Proceedings of the 2003 Joint Workshop on Foundations of Mobile Computing*, pages 2–11, 2003.

[12] 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*, pages 173–182, 2002.

[13] 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.

[14] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 1–13. ACM Press, September 2002.

[15] 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.

[16] 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 '03)*, Venice, Italy, Sep 2003.

[17] 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.

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

[19] O. Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, August 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 Foudations of Computer Science*, pages 252–259, Las Vegas, Nevada, Oct. 2001.

[22] M. Jakobsson. Ripping coins for a fair exchange. In *Advances in cryptology, EUROCRYPT '95*, pages 220–230, 1995.

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

[24] A. Konrad, B. Zhao, A. Joseph, and R. Ludwig. Explicit loss notification and wireless web perfromance. In *Proceedings of ACM MSWIM*, 2001.

[25] D. M. Kreps. *Game Theory and Economic Modelling*. Oxford Press, 1991.

[26] H. Lin, M. Chatterjee, S. K. Das, and K. Basu. ARC: An integrated admission and rate control framework for cdma data networks based on non-cooperative games. In *Proceedings of The Ninth International Conference on Mobile Computing and Networking (Mobicom)*, pages 326–338, San Diego, CA, Sep 2003.

[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, Sep 2003.

[29] 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.

[30] N. Nisan and A. Ronen. Computationally feasible VCG mechanisms. In *Proceedings of the ACM Symposium on Electronic Commerce (EC' 00)*, pages 242–252, Minneapolis, MN, 2000.

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

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

[33] C. Papadimitriou. Algorithms, games, and the Internet. In *Proceedings of the 33rd Annual Symposium on Theory of Computing*, pages 749–753, 2001.

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

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

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

[37] 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 1-3 2003.

[38] J. Shneidman and D. C. Parkes. Using redundancy to improve robustness of distributed mechanism implementations. In *Proceedings of the ACM Symposium on Electronic Commerce (EC' 03)*, San Diego, CA, June 2003.

[39] A. Spyropoulos and C. Raghavendra. Energy effi cient communications in ad hoc networks using directional antennas. In *Proceedings of IEEE INFOCOM '02*, New York, NY, June 2002.

[40] V. Srinivasan, P. Nuggehalli, C.-F. Chiasserini, and R. Rao. Cooperation in wireless ad hoc networks. In *Proceedings of IEEE INFOCOM '03*, 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] J. E. Wieselthier, G. Nguyen, and A. Ephremides. Energy-limited wireless networking with directional antennas: The case of session-based multicasting. In *Proceedings of IEEE INFOCOM '02*, New York, NY, June 2002.

[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 '03*, San Francisco, CA, Apr. 2003.

.1 Notations

- a_i: action of node i
- a: actions of all nodes
- $a_i^{(r)}$: action of node i in the routing sub-game
- $a^{(r)}$: actions of all nodes in the routing sub-game
- $a_i^{(f)}$: action of node i in the forwarding sub-game
- $a^{(f)}$: actions of all nodes in the forwarding sub-game
- A_i: set of actions node i can choose from
- a_{-i}: actions of all nodes except i
- b: block size (the number of packets in a data block)
- c_i, $c_{i,j}$: cost of node i, and link i to j
- $cost$: function to sum the costs of links
- $\mathcal{C}$: set of all possible path costs
- D: destination node
- $D(key; cipher - text)$: decryption of cipher-text using key
- E: number of links
- $E(key; clear - text)$: encryption of clear-text using key
- $E[X]$: expected value of random variable X
- g: a primitive root in a group where Decisional Diffi e-Hellman (DDH) Assumption is valid
- G: the probability of a node following the protocol.
- h_l: pseudorandom value for test signal at power level l
- $H()$: an ideal hash function
- $H^l()$: the hash function $H()$ iterated for l times. Formally, we defi ne $H^l() = H(H^{l-1}())$

- i: node index
- j: node index
- k_i: node i's private key
- $K_i = g^{k_i}$: node i's public key
- $k_{i,D}$: the shared secret key between node i and destination D
- l: power level
- L: the power level chosen by our algorithm for the lossy link models
- LCP: lowest (power) cost path
- $LCP(S, D)$: lowest cost path from S to D
- $LCP(S, D; -i)$: lowest cost path from S to D without using node i
- $LCP(S, D) - \{i\}$: lowest cost path from S to D; if node i is on the path, remove the link from i to the next hop
- m: block id
- M: number of packets in the session
- n: upper bound of path length
- N: number of nodes
- $\mathcal{N}$: set of nodes
- p_i: payment to node i
- $P_{i,j}$: power level to reach from i to j
- P_{ctr}: a power level at which control signals are sent. Typically $P_{ctr} < P_{max}$. However, P_{ctr} should be large enough such that the ad-hoc network is strongly connected if every node sends at P_{ctr}
- P: parent node
- $\mathcal{P}$: the set of power levels; $\mathcal{P}_i$ is for node i only
- r_0: random number picked by source
- r_l: $H^l(r_0)$

- $\mathcal{R}$: route decision
- S: source node
- $S_{i,j}(l)$: the transmission success rate of link (i, j) when data packets are sent at power level l
- $\hat{S}_{i,j}(l)$: our estimate of $S_{i,j}(l)$
- u_i: the utility of node i
- $u_i^{(p)}$: the prospective utility of node i
- α_i: the cost-of-energy parameter of node i