Truthful Multicast in Selfish Wireless Networks

Submitted to MobiCom 2004.

ABSTRACT

In wireless network, it is often assumed that each individual wireless terminal or link will faithfully follow the prescribed protocols without any deviation– except, perhaps, for the faulty or malicious ones. Wireless terminals or links, often owned by individuals, will likely do what is most beneficial to their owners – act "selfishly". Thus, it is more reasonable to expect that each selfish terminal will try to manipulate the algorithms or protocols for its owners' benefit, instead faithfully follow the designed protocols. Therefore, an algorithm or protocol intended for selfish wireless terminals or links must be designed.

In this paper, we specifically study how to conduct efficient multicast in *selfish* wireless networks. We assume that each wireless terminal or communication link (called agent) will incur a cost when it transits some data, and the cost is known to the wireless terminal or communication link itself.

For each of the widely used structures for multicast, we design a strategyproof multicast mechanism without using the well known VCG mechanism such that each agent has to truthfully report its cost to maximize its profit.

Extensive simulations are conducted to study the practical performances of the proposed protocols regarding the actually network cost and total payment.

1. INTRODUCTION

Recent years saw a great amount of researches in wireless networks on various important problems such as routing, Quality of Service, security, power management, and traffic and mobility modelling. However, there are still many challenges left.

While unicast in wireless network has been studied extensively in literatures and deployed in practice for years, several important issues about multicast over wireless networks haven't been explored fully. In practice, multicasting is a more efficient way to support group communication than unicasting or broadcasting, as it can transmit packets to destination using fewer network resource. Typical wireless multicast application including group-oriented mobile commerce, military command and control, distance education, and intelligent transportation systems. For multicast routing, usually a tree among sources and receivers is used because it requires less network resource than other structures. Most often, these tree structures are based on shortest paths from the sender to the receivers or the minimum spanning tree. During the process of building the multicast tree, most works intrinsically assumed that each individual wireless terminals or wireless links (possibly owned by selfish users) will follow prescribed protocols without deviation, which is known as cooperative. However, this assumption is not always true. The limitation of energy supply and scarce resources of these mobile devices raise concerns about this traditional belief. Following the common belief in neoclassic economics, it is more reasonable to assume all wireless terminals or links are *rational*: they try to maximize their benefit instead of conforming to the existing protocols. Thus, we need to design some mechanisms to assure these *rational* wireless terminals or links will conform to our protocol without deviation.

How to achieve cooperation among terminals in network was previously addressed in [4, 12, 14, 3, 5, 17, 18]. The key idea behind these approaches is that terminals providing a service should be remunerated, while terminals receiving a service should be charged. Each terminal maintains a counter, called *nuglet counter*, in a tamper resistant hardware module, which is decreased when the terminal sends a packet as originator and increased when the terminal forwards a packet. Both of these methods belong to so called *credit based method*. Usually, they are heuristic and need some special hardware. In recent years, *incentive based methods* have been proposed to solve the non-cooperative problem. The most well-known and widely used *incentive based method* is so called VCG mechanism family by Vickrey [20], Clarke [6], and Groves [10]. Nisan and Ronen [15] provided a mechanism belonging to VCG family to assure the cooperation for unicast problem in general network. Unfortunately, as we will show later, if we apply VCG mechanism to those commonly used multicast tree structures, it can't guarantee wireless agents' conforming to our protocol. So in this paper, we study how to design non-VCG truthful mechanisms for multicast in *selfish* wireless networks.

The rest of the paper is organized as follows. First, we introduce some preliminaries and related works in Section 2. We also present our communication model and the problems to be solved in this paper. We study the strategy-proof mechanism for link weighted network in Section 3 and node weighted network in Section 4. Simulation results are presented in Section 5. We conclude our paper in Section 6 by pointing out some possible future work.

2. PRELIMINARIES AND PRIORI ART

2.1 Preliminaries

In designing efficient, centralized or distributed algorithms and network protocols, the computational agents are typically assumed to be either *correct/obedient* or *faulty* (also called adversarial). Here agents are said to be *correct/obedient* if they follow the protocol correctly; agents are said to be *faulty* if (1) they stop working, or (2) they drop messages, or (3) they act arbitrarily, which is also called *Byzantine failure*, i.e., they may deviate from the protocol in arbitrary ways that harm other users, even if the deviant behavior does not bring them any obvious tangible benefits.

In contrast, economists design market mechanisms in which it is assumed that agents are *rational*. The rational agents respond to well-defined incentives and will deviate from the protocol only if it improves their gain. A rational agent is neither correct/obedient nor adversarial.

A standard economic model for the design and analysis of scenarios in which the participants act according to their own self-interests is as follows. Assume that there are n agents, which could be the wireless devices in a wireless ad hoc networks, the computers in a peer-to-peer networks, the network links in a network, or the bidders in an auction. Each agent i, for $i \in \{1, \cdots, n\}$, has some private information t_i, called its *type*. Here, the type t_i could be its cost to forward a packet in a network environment; could be a monetary value that it is willing to pay for a good in an auction environment. Then the set of n agents define a type vector $t = (t_1, t_2, \cdots, t_n)$.

A mechanism defines, for each agent i, a set of strategies A_i. For each strategy vector $a = (a_1, \cdots, a_n)$, i.e., agent i plays a strategy $a_i \in A_i$, the mechanism computes an *output* $o = o(a_1, \cdots, a_n)$ and a *payment* vector $p = (p_1, \cdots, p_n)$, where $p_i = p_i(a_1, \cdots, a_n)$. Here the payment p_i is the money given to the participating agent i and it depends on the strategies used by all agents. If $p_i < 0$, it means that the agent has to pay $-p_i$ to participate in the action.

For each possible output o, agent i's preferences are given by a valuation function v_i that assigns a real monetary number $v_i(t_i, o)$ to output o. Everything in the scenario is public knowledge except the type t_i, which is a private information to agent i. For example, in an instance of unicast routing, there are n network terminals, which are n agents. Agent i's type is its cost c_i of forwarding a given data. The space of feasible outputs consists of all paths that connect the source and target terminal. The valuation of terminal k for a path connecting source and target is $-c_k$ if terminal k is on the path and 0 otherwise. Let $u_i(t_i, o(a), p_i(a))$ denote the *utility* of agent i at the outcome of the game, given its preferences t_i and strategies profile $a = (a_1, \cdots, a_n)$ selected by agents. Let $a_{-i} = (a_1, \cdots, a_{i-1}, a_{i+1}, \cdots, a_n)$ denote the vector of strategies of all other agents except i. A strategy a_i is called *dominant strategy* if it maximizes the utility for all possible strategies of all other agents, i.e.,

$$u_i(t_i, o(a_i, b_{-i}), p_i(a_i, b_{-i})) \geq u_i(t_i, o(a_i', b_{-i}), p_i(a_i', b_{-i}))$$

for all $a_i' \neq a_i$ and all strategies b_{-i} of agents other than i. A strategy vector a is called *Nash Equilibrium* if it maximizes the utility when the strategies of all agents are fixed, i.e.,

$$u_i(t_i, o(a_i, a_{-i}), p_i(a_i, a_{-i})) \geq u_i(t_i, o(a_i', a_{-i}), p_i(a_i', a_{-i}))$$

for all i, and $a_i' \neq a_i$.

A very common assumption in mechanism design, and one which we will follow in this paper, is that agents are *rational* and have quasi-linear utility functions. The utility function is *quasi-linear* if $u_i(t_i, o) = v_i(t_i, o) + p_i$. An agent is called *rational*, if agent i always tries to maximize its utility u_i by finding its best strategy.

The system-wide goal in mechanism design is defined by a *social choice function* $g()$, which, given agent types, selects the *optimal* outcome. Given mechanism with outcome function $o()$, we say that a mechanism implements social choice function $g()$ if the outcome computed with equilibrium agent strategies is a solution to the social choice function for all possible agent preferences. An output function o of a mechanism is *allocatively-efficient* if it maximizes the summation of valuations of all agents, i.e., $\sum_{i=1}^{n} v_i(t_i, o) \geq \sum_{i=1}^{n} v_i(t_i, o')$ for all possible types t. A mechanism is *efficient* if it implements an allocatively-efficient social choice function.

A direct-revelation mechanism is a mechanism in which the only actions available to agents are to make direct claims about their preferences v_i to the mechanism. An *incentive compatible* (IC) mechanism is a direct-revelation mechanism in which agents report their valuations v_i to the mechanism truthfully so as to maximize its utility. Incentive-compatibility captures the essence of designing a mechanism to overcome the self-interest of agents: in an incentive compatible mechanism an agent will choose to report its private information truthfully in order to maximize its utility. A direct-revelation mechanism is strategy-proof if truth-revelation is a dominant-strategy equilibrium. Let $t|^i b = (t_1, \cdots, t_{i-1}, b, t_{i+1}, \cdots, t_n)$, i.e., each agent $j \neq i$ reports its type t_j except that the agent i reports type b. Then, in a direct-revelation strategy-proof mechanism, the payment function should satisfy that, for each agent i,

$$v_i(t_i, o(t)) + p_i(t) \geq v_i(t_i, o(t|^i b)) + p_i(t|^i b).$$

The strategy-proof mechanism wants each agent to report its private type truthfully by providing incentives to agents. Another very common requirement in the literature for mechanism design is so called *individual rationality* or *voluntary participation*: the agent's utility of participating in the output of the mechanism is not less than the utility of the agent if it did not participate.

For unicast routing, the set of strategies A_k for a terminal k in a direct revelation mechanism is the set of possible costs that terminal k could declare. The utility of a terminal k on a path connecting source and target is the payment p_k for terminal k minus its cost c_k.

Arguably the most important positive result in mechanism design is what is usually called the generalized Vickrey-Clarke-Groves (VCG) mechanism by Vickrey [20], Clarke [6], and Groves [10]. The VCG mechanism applies to maximization problems where the objective function is simply the sum of all agents' valuations. A maximization mechanism design problem is called *utilitarian* if it implements a social choice function $g(o, t) = \sum_i v_i(t_i, o)$. A direct revelation mechanism $m = (o(t), p(t))$ belongs to the VCG family if (1) the output $o(t)$ computed based on the type vector t maximizes the objective function $g(o, t) = \sum_i v_i(t_i, o)$, and (2) the payment to agent i is $p_i(t) = \sum_{j \neq i} v_j(t_j, o(t)) + h_i(t_{-i})$. Here $h_i()$ is an arbitrary function of t_{-i}. It is proved by Groves [10] that a VCG mechanism is truthful. Green and Laffont [9] proved that, under mild assumptions, VCG mechanisms are the *only* truthful implementations for utilitarian problems.

An output function of a VCG mechanism is required to maximize the objective function. This makes the mechanism computationally intractable in many cases. Notice that replacing the optimal algorithm with non-optimal approximation usually leads to untruthful mechanisms if VCG payment method is used. In their seminal paper on algorithmic mechanism design, Nisan and Ronen [15] add computational efficiency to the set of concerns that must be addressed in the study of how privately known preferences of a large group of selfish agents can be aggregated into a "social choice" that results in optimal allocation of resources.

In summarize, we want to design strategy-proof multicast for a selfish wireless network with the following properties. 1)*Incentive Compatibility (IC)*: an agent will reveal its true cost to maximize its utility no matter what the other agents do. 2)*Individual Rationality (IR)*: an agent is guaranteed to have non-negative utility if it reports its type truthfully no matter what other agents do. 3)*Polynomial Time Computability (PC)*: all computations (the computation of the output and the payment) are done in polynomial time.

2.2 Priori Arts on Selfish Routing

Routing has been an important part of the algorithmic mechanism-design from the very beginning. Nisan and Ronen [15] provided a polynomial-time strategyproof mechanism for optimal unicast route selection in a centralized computational model. In their formulation, the network is modelled as an abstract graph $G = (V, E)$. Each edge e of the graph is an agent and has a private type t_e, which represents the cost of sending a message along this edge. The mechanism-design goal is to find a Least Cost Path (LCP) $\mathsf{LCP}(x, y)$ between two designated nodes x and y. The valuation of an agent e is $-t_e$ if the edge e is part of the path $\mathsf{LCP}(x, y)$ and 0 otherwise. Nisan and Ronen used the VCG mechanism for payment. The payment to agent e is $D_{G-\{e\}}(x, y) - D_G(x, y)$, where $D_{G-\{e\}}(x, y)$ is the cost of the LCP through G when edge e is not presented and $D_G(x, y)$ is the cost of the least cost path $\mathsf{LCP}(x, y)$ through G. Clearly, there must have two link disjoint paths connecting x and y to prevent the monopoly. The result in [15] can be easily extended to deal with wireless unicast problem for arbitrary pair of terminals.

Feigenbaum *et. al* [7] then addressed the truthful low cost routing in a different network model. They assume that each node k incurs a transit cost c_k for each transit packet it carries. For any two nodes i and j of the network, $T_{i,j}$ is the intensity of the traffic (number of packets) originating from i and destined for node j. Their strategyproof mechanism again is essentially the VCG mechanism. They gave a distributed method such that each node i can compute a payment $p_{ij}^k > 0$ to node k for carrying the transit traffic from node i to node j if node k is on the LCP $\mathsf{LCP}(i, j)$. Anderegg and Eidenbenz [1] recently proposed a similar routing protocol for wireless ad hoc networks based on VCG mechanism again. They assumed that each link has a cost and each node is a selfish agent.

For multicast flow, Feigenbaum *et. al* [8] assumed that there is a fixed multicast infrastructure, given any set of receivers $Q \subset V$, connects the source node to the receivers. Additionally, for each user $q_i \in Q$, they assumed a *fixed* path from the source to it, determined by the multicast routing infrastructure. Then for every subset R of receivers, the delivery tree $T(R)$ is merely the union of the fixed paths from the source to the receivers R. They also assumed that there is a link cost associated with each communication link in the network and the link cost is *known* to everyone. For each receiver q_i, there is a valuation w_i that this user values the reception of the data from the source. This information w_i is only known to q_i. User q_i will report a number w_i', which is the amount of money he/she is willing to pay to receive the data. The source node then selects a subset $R \subset Q$ of receivers to maximize the difference $\sum_{i \in R} w_i' - C(R)$, where $C(R)$ is the cost of the multicast tree $T(R)$ to send data to all nodes in R. The approach of fixing the multicast tree is relatively simple to implement but could not model the greedy nature of all network terminals in the network.

There is a vast literature on the mechanism design or implementation paradigm in which some mechanisms are designed to achieve the socially desirable outcomes in spite of users' selfishness. Some of these approaches use Nash equilibrium rather than dominant-strategy. That is, they assumed that simultaneous selfish play leads to a self-consistent Nash equilibrium, in which no agent can improve its utility by deviating from its current strategy when other agents keep their strategies. Notice that since Nash equilibrium has a weak requirement on the strategies used by the agents, it often can achieve a much wider variety of outcomes.

2.3 Communication Model

In this paper, as did in the literature, we study two different models of wireless networking: link weighted and node weighted networking. For both models, usually the communication links are needed to be symmetric due to the following requirement: each receiver has to send an acknowledgment packet directly to the sender after it received the data. Thus, in this paper, we consider all communication links as undirected. Actually, our results can apply to case when the link is directed with some minor modification.

In a link weighted network, each communication link incurs a cost when a message is sent over it and the communication link is an agent, e.g., the marginal cost of this link transmitting the data. For example, in a cellular networks, it could be the cost of using the channel. For node weighted network each communication terminal will incur a cost when it has to relay a message for other node. Typical example of a node weighted network is the wireless ad hoc network with fixed transmission range. In a wireless ad hoc network (or sensor network) each node has some computation power and an omni-directional antenna. This is attractive for a single transmission of a node can be received by all nodes within its vicinity. The main communication cost in wireless networks is to send out the signal while the receiving cost of a message is neglected here. Throughout this paper, we always assume that our network is **bi-connected**, which implies that if we remove the agent the network is still connected. This assumption is necessary to prevent some nodes from being monopoly and charging arbitrary cost, in addition to increase network robustness.

It is well known that finding the minimum cost multicast tree is NP-hard for both link weighted networks and node weighted networks. So several multicast structures have been proposed in the literature to approximate the minimum cost multicast tree. In practice, there are two types of multicast structures to meet the different requitement of different application: *source based multicast tree* and *share based multicast tree*. For those applications like online movie, they usually has one or only a few senders and lots of receivers. Therefore, we can use a source based multicast tree in which receivers only receive messages but do not send them. On the other hand, many applications have lots of active senders, such as distributed interactive simulation applications, and distributed video-gaming (where most receivers are also senders). Due to scalability reason, share based tree has been used instead of source based tree.

In this paper, we study how to design truthful payment schemes for the most widely used multicast trees, including source based trees and shared trees for both edge weighted and node weighted networks.

2.4 Problem Statement

Consider any communication network $G = (V, E, c)$, where $V = \{v_1, \cdots, v_n\}$ is the set of communication terminals, $E = \{e_1, e_2, \cdots, e_m\}$ are the set of links, and c is the cost vector of all agents. Here agents are terminals in a node weighted network and are links in a link weighted network. Given a set of receivers $Q = \{q_0, q_1, q_2, \cdots, q_{r-1}\} \subset V$, the multicast problem is to find a tree $T \subset G$ spanning all receiving terminals Q. For simplicity, we assume that $s = q_0$ is the sender of the multicast if exist. All terminals or links are require to declare a cost of relaying the message. Based on the declared cost profile d, we should construct the multicast tree and decide the payment for the agents. The utility of an agent is its payment received, minus its cost if it is selected in the multicast tree. Instead of reinventing the wheels, we will still use the previously proposed structures for multicast as the output of our mechanism. Given a multicast tree, we will study the designing of strategyproof payment schemes based on this tree.

Given a network H, we use $\omega(H)$ to denote the total cost of all agents in this network. If we change the cost of any agent i (link e_i or node v_i) to c_i', we denote the new network as $G' = (V, E, c|^i c_i')$, or simply $c|^i c_i'$. If we remove one agent i from the network, we denote it as $c|^i\infty$. Denote $G\backslash e_i$ as the network without link e_i, and denote $G\backslash v_i$ as the network without node v_i and all its incident links. For the simplicity of notation, we will use the cost vector c to denote the network $G = (V, E, c)$ if no confusion is caused.

3. MULTICAST IN LINK WEIGHTED COMMUNICATION NETWORKS

In this section, we discuss how to conduct truthful multicast when the network is modelled by a link weighed communication graph. We assume the communication network is modelled by a undirected graph $G = (V, E, c)$. Here, the value of c_i is only known to individual link e_i.

We specifically study the following three structures: least cost path star (LCPS), pruning minimum spanning tree (PMST), and link weighted Steiner tree (LST). Notice that the first and the third structure belong to the family of the source based multicast tree, while the second structure belongs to the share based multicast tree.

3.1 Least Cost Path Star

In practice, this is the most widely used multicast distribution tree. Notice that, although we only discuss the using of least cost path star for the link weighted network (i.e., the link will incur a cost when transmitting data), all results we presented in this subsection can be extended to the node weighted scenario without any difficulty.

3.1.1 Constructing LCPS

For each receiver $q_i \neq s$, we compute the shortest path (least cost path), denoted by $\mathsf{LCP}(s, q_i, d)$, from the source s to q_i under the reported cost profile d. The union of all least cost paths from the source to receivers is called *least cost path star*, denoted by $LCPS(d)$. Next we discuss how to design a truthful payment scheme while using LCPS as the output.

3.1.2 VCG mechanism on LCPS is not strategyproof

Intuitively, we would use the VCG payment scheme in conjunction with the LCPS tree structure as follows. The payment $p_k(d)$ to each link e_k is

$$p_k(d) = \omega(LCPS(d|^k\infty)) - \omega(LCPS(d)) + d_k.$$

We show by an example that the above payment scheme is not strategyproof. In other words, if we simply apply VCG scheme on LCPS, a link may have incentives to lie about its cost. Figure 1 illustrates such an example where link sv_3 can lie its cost to improve its utility.

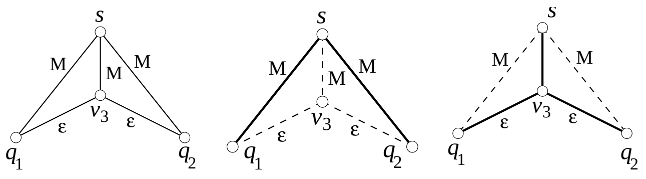


Figure 1: The cost of links are $c(sq_1) = c(sq_2) = c(sv_3) = M$, and $c(q_1v_3) = c(q_2v_3) = \epsilon$. Here, q_1 and q_2 are the receiving terminals.

The payment to link sv_3 is 0 and its utility is also 0 if it reports its cost truthfully. The total payment to link sv_3 when sv_3 reported a cost $d_3 = M - 2\epsilon$ is $\omega(LCPS(c|^3\infty)) - \omega(LCPS(c|^3d_3)) + d_3 = 2M - (M - 2\epsilon + 2\epsilon) + M - 2\epsilon = 2M - 2\epsilon$ and the utility of link sv_3 becomes $u_3(c|^3d_3) = 2M - 2\epsilon - (M + \epsilon) = M - 3\epsilon$, which is larger than $u_3(c) = 0$, when $0 < \epsilon < M/3$.

3.1.3 Strategyproof mechanism on LCPS

Now, we describe our strategyproof mechanism that does not rely on VCG payment. For each receiver $q_i \neq s$, we compute the least cost path from the source s to q_i, and compute a payment $p_k^i(d)$ to every link e_k on the $\mathsf{LCP}(s, q_i, d)$ using the scheme for unicast

$$p_k^i(c) = d_k + |\mathsf{LCP}(s, q_i, d|^k\infty)| - |\mathsf{LCP}(s, q_i, d)|.$$

Here $|\mathsf{LCP}(s, q_i, d)|$ denotes the total cost of the least cost path $\mathsf{LCP}(s, q_i, d)$. The final payment to link e_k is then

$$p_k(d) = \max_{q_i \in Q} p_k^i(d) \qquad (1)$$

THEOREM 1. *Payment (1) based on LCPS is truthful and it is minimum among all truthful payments based on LCPS.*

PROOF. Clearly, when link e_k reports its cost truthfully, it has non-negative utility, i.e., the payment scheme satisfies the IR property. In addition, since payment scheme for unicast is truthful, so e_k cannot lie its cost to increase its payment $p_k^i(c)$ based on $\mathsf{LCP}(s, q_i, d)$. Thus, it cannot increase $\max_{q_i \in Q} p_k^i(c)$ by lying its cost. In other words, our payment scheme is truthful.

We then show that the above payment scheme pays the minimum among all strategyproof mechanism using LCPS as output. Before showing the optimality of our payment scheme, we give some definitions first. Consider all paths from sender s to receiver q_i, they can be divided into two categories: with edge e_k or not. The path having the minimum length among these paths with edge e_k is denoted as $\mathsf{LCP}_{e_k}(s, q_i, d)$; and the path having the minimum length among these paths without edge e_k is denoted as $\mathsf{LCP}_{-e_k}(s, q_i, d)$.

Assume there is another payment scheme $\tilde{p}$ that pays less for a link e_k in a network G under cost profile d. Let $\delta = p_k(d) - \tilde{p}_k(d)$, then $\delta > 0$. Without loss of generality, assume that $p_k(d) = p_k^i(d)$. Thus, link e_k is on $\mathsf{LCP}(s, q_i, d)$ and the definition of $p_k^i(d)$ implies that

$$|\mathsf{LCP}_{-e_k}(s, q_i, d)| - |\mathsf{LCP}(s, q_i, d)| = p_k(d) - d_k.$$

Then consider another cost profile $d' = d|^k(p_k(d) - \frac{\delta}{2})$ where the true cost of link e_k is $p_k(d) - \frac{\delta}{2}$. Under profile d', since $|\mathsf{LCP}_{-e_k}(s, q_i, d')| = |\mathsf{LCP}_{-e_k}(s, q_i, d)|$, we have

$$\begin{aligned}
|\mathsf{LCP}_{e_k}(s, q_i, d')| &= |\mathsf{LCP}_{e_k}(s, q_i, d|^k 0)| + p_k(d) - \frac{\delta}{2} \\
&= |\mathsf{LCP}_{e_k}(s, q_i, d)| + p_k(d) - \frac{\delta}{2} - d_k \\
&= |\mathsf{LCP}(s, q_i, d)| + p_k(d) - \frac{\delta}{2} - d_k \\
&= |\mathsf{LCP}_{-e_k}(s, q_i, d)| - \frac{\delta}{2} \\
&< |\mathsf{LCP}_{-e_k}(s, q_i, d)| = |\mathsf{LCP}_{-e_k}(s, q_i, d')|
\end{aligned}$$

Thus, $e_k \in LCPS(d')$. From the following Lemma 2, we know that the payment to link e_k is the same for cost profile d and d'. Thus, the utility of link e_k under profile d' by payment scheme $\tilde{p}$ becomes $\tilde{p}_k(d') - c_k = \tilde{p}_k(d) - c_k = \tilde{p}_k(d) - (p_k(d) - \frac{\delta}{2}) = -\frac{\delta}{2} < 0$. In other words, under profile d', when link e_k reports its true cost, it gets a negative utility under payment scheme $\tilde{p}$. Thus, $\tilde{p}$ is not strategyproof. This finishes our proof. □

LEMMA 2. *If a mechanism based on a tree T with payment function $\tilde{p}$ is truthful, then for every agent a_k in network, if $a_k \in T$ then payment function $\tilde{p}_k(d)$ should be independent of d_k.*

PROOF. We prove it by contradiction. Suppose that there exists a truthful payment scheme such that $\tilde{p}_k(d)$ depends on d_k. There must exist two valid declared costs x_1 and x_2 such that $x_1 \neq x_2$ and $\tilde{p}_k(d|^k x_1) \neq \tilde{p}_k(d|^k x_2)$. Without loss of generality we assume that $\tilde{p}_k(d|^k x_1) > \tilde{p}_k(d|^k x_2)$. Now consider agent a_k with actual cost $c_k = x_2$. Obviously, it can lie its cost as x_2 to increase his utility, which violates the incentive compatibility (IC) property. □

3.1.4 Computational Complexity

Assume there are r receivers, for every terminal q_i, we calculate the payment for all nodes $v_k \in \mathsf{LCP}(s, q_i, c)$ based on $\mathsf{LCP}(s, q_i, c)$ using the fast payment scheme for unicast problem [21]. This will take $O(n \log n + m)$ time. So for all terminals, it will take $O(rn \log n + rm)$. Note that we can construct the least cost path star in time $O(n \log n + m)$. A very natural question is whether we can reduce the time complexity from $O(rn \log n + rm)$ to $O(n \log n + m)$. We leave it as an open question.

3.2 Pruning Minimum Spanning Tree

3.2.1 Constructing PMST

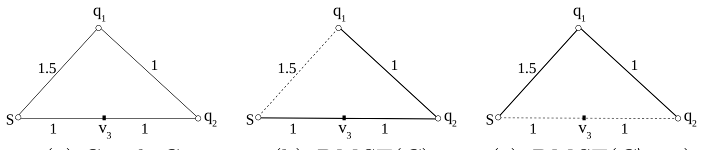
First we construct the minimum spanning tree $MST(G)$ on the graph G. We then root the tree $MST(G)$ at sender s, prune all subtrees that do not contain a receiver. The final structure is called Pruning Minimum Spanning Tree (PMST).

3.2.2 VCG mechanism on PMST is not strategyproof

Intuitively, we would use the VCG payment scheme in conjunction with the PMST structure. The payment to an edge $e_k \in PMST(G)$ based on VCG would be as follows

$$p_k(d) = \omega(PMST(d|^k\infty)) - \omega(PMST(d)) + d_k.$$

We show by an example that the above payment scheme is not strategyproof. Figure 2 illustrates such an example where link q_1v_1 has a negative utility when it reveals its true cost.



(a) Graph G (b) $PMST(G)$ (c) $PMST(G \backslash sv_3)$

Figure 2: Terminals S is the send and q_1,q_2 are receivers; $c(sq_1) = 1.5$ and $c(q_1q_2) = c(sv_3) = c(v_3q_2) = 1$.

If sv_3 reveals its true cost, then the payment to link sv_3 is $\omega(PMST(G\backslash sv_3)) - \omega(PMST(G) + c(sv_3) = 2.5 - 3 + 1 = 0.5$ and the utility of link sv_3 becomes -0.5, which violates IR.

3.2.3 Strategyproof mechanism on PMST

We now discuss our strategyproof payment scheme using PMST as the output. Instead of applying the VCG mechanism on PMST, we apply VCG mechanism on the MST. The payment for edge $e_k \in PMST(d)$ is

$$p_k(d) = \omega(MST(d|^k\infty)) - \omega(MST(d)) + d_k. \qquad (2)$$

For edge $e_k \notin PMST(d)$, its payment is 0.

THEOREM 3. *Our payment scheme (2) is truthful and minimal among all truthful payment schemes based on PMST.*

PROOF. For link $e_k \in PMST(d)$ or $e_k \notin MST(d)$, the payment is exactly the payment based on MST structure. Notice the payment based on MST belongs to VCG mechanism, so it is truthful. Thus, if $e_k \in PMST(d)$ or $e_k \notin MST(d)$, it don't have the incentive to lie. Now considering when $e_k \in MST(d) - PMST(d)$. If e_k lies its cost such that $e_k \notin MST(d)$, then it still gets utility 0; else the MST will keep unchanged which implies that e_k is still not in $PMST$. Thus, e_k also don't have the incentive to lie in this case. So our payment scheme (2) is truthful.

For $e_k \in PMST(d)$ our payment is same as the payment for MST, which is a VCG mechanism. Thus, our payment is minimal among all truthful payment scheme if the output is PMST. Detailed proof is omitted here due to space limit. □

3.2.4 Computational Complexity

Obviously, we can construct the PMST in time $O(n \log n + m)$. We then analyze the time complexity of computing all links' payment in PMST. Let $G \backslash MST(G)$ be the graph after removing the edges of $MST(G)$ from G. Call the minimum spanning tree of $G \backslash MST(G)$ the second minimum spanning tree, denoted by $MST_2(G)$. It was shown that the total payment to all links in the MST equals to the actual cost of the $MST_2(G)$ in [2]. Also, it is not difficult to calculate payment for every link in PMST in time $O(n \log n + m)$, which is optimal.

3.3 Link Weighted Steiner Tree (LST)

It is well-known [16, 19] that it is NP-hard to find the minimum cost multicast tree when given an arbitrary link weighted graph G. For LCPS and PMST structure, while they usually work well in practice, in some extreme situation, the cost of these structures could be arbitrary larger than the optimal cost. From a computer science view, it is desirable that we can find a structure such that even in worst case, the cost of structure is at most α times of the optimal. In literature, this structure is said to have a α-approximation of the optimal and α is called approximation ratio.

Takahashi and Matsuyama [19] first gave a polynomial time algorithm that can output 2-approximation of the minimum cost Steiner tree (MCST). Then a series of results have been developed to improve the approximation ratio. The current best result is due to Robins and Zelikovsky [16], in which the authors presented a polynomial time method with approximation ratio $1+\frac{\ln 3}{2}$. Takahashi and Matsuyama's algorithm is simpler and can be implemented in a distributed way, which fits the need of wireless networks. Thus, we use this algorithm instead of the algorithm with the best approximation ratio to construct multicast tree.

3.3.1 Constructing the LST

We first review the algorithm by Takahashi and Matsuyama:

ALGORITHM 1. *(Takahashi and Matsuyama [19])*

Repeat the following steps until no receiver remains:

1. Find one of the remaining receiver, say q_i, that is closest to the source s, i.e., the $\mathsf{LCP}(s, q_i, d)$ has the least cost among the shortest paths from s to all receivers.

2. Connect q_i to s using the least cost path between them and contract this least cost path to one virtual vertex. Remove some edges during contracting if necessary. This is virtual source terminal for next round.

For each iteration in Algorithm 1, we call it a round. Let P_i be the path found in round i, and t_i be the receiver it connects with the virtual source terminal. Given r receivers, the method terminates in r rounds. Hereafter, let $LST(d)$ be the final tree constructed by Algorithm 1. The authors of [19] proved that $\omega(LST(d)) \leq 2\omega(MCST(d))$.

3.3.2 VCG mechanism on LST in not strategy-proof

Given a tree $LST(d)$ approximating the minimum cost Steiner tree, a natural payment scheme would be to pay each edge based on VCG scheme, i.e., the payment to an edge $e_k \in LST(G)$ is

$$p_k(d) = \omega(LST(d|^k\infty)) - \omega(LST(d)) + d_k.$$

We give an example to show that this payment scheme does *not* satisfy IR property, i.e., it is possible that some edges have negative utility. Figure 3 illustrates the example with

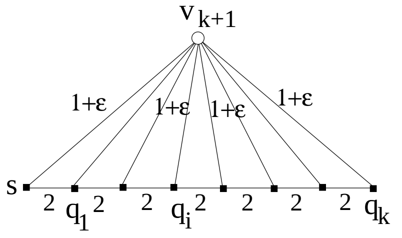


Figure 3: Terminals q_i, $1 \leq i \leq k$ are receivers; the cost of each link $v_{k+1}q_i$ and $v_{k+1}s$ is $1+\epsilon$, where ϵ is a sufficiently small positive real number. The cost of each link q_iq_{i+1} and sq_1 is 2.

terminal s being the source terminal. It is not difficulty to show that, in the first round, link sq_1 is selected to connect terminals s and q_1 with cost 2; in round r, we will select link $q_{r-1}q_r$ to connect to q_r with cost 2. Thus, the tree $LST(G)$ will be just the path $sq_1q_2\cdots q_k$, whose cost is $\sum_{i=1}^{k-1} c(q_iq_{i+1}) + c(sq_1) = 2k$.

When link $e_1 = q_1q_2$ is not used, it is easy to see that the final tree $LST(G\backslash e_1)$ will only use terminal v_{k+1} to connect all receivers with total cost $(k+1)(1+\epsilon)$. Thus, the utility of link $e_1 = sq_1$ is $\omega(LST(G\backslash e_1)) - \omega(LST(G)) = (k+1)(1+\epsilon) - 2k = k\epsilon - k + 2$, which is negative when $\epsilon < \frac{k-2}{k}$. Thus, the payment to link sq_1 does not satisfy the incentive rationality property.

3.3.3 Strategy-proof mechanism based on LST

In this subsection we describe our strategyproof mechanism (without using VCG) based on LST. Instead of paying the wireless link based on the final structure LST, we will calculate a payment for each round and choose the maximum as the final payment. Let $w_i(d)$ be the cost of the path P_i selected in the ith round if the cost profile is d.

ALGORITHM 2. *Truthful payment to e_k based on LST*

1. Use Algorithm 1 to find $LST(d|^k\infty)$. The graph used in the beginning of round i is denoted as $G_i^{-e_k}$.

2. For every round i, considering graph $G_i^{-e_k} \bigcup e_k$, find LCP from s to every remaining receivers and choose the LCP with the minimum weight. For simplicity, we denote this LCP as $\mathsf{P}'_i(d)$.

3. Define the payment for edge e_k in round i as
$$p_k^i(d) = w_i(d|^k\infty) - |\mathsf{P}'_i(d)| + d_k$$

4. The final payment to link e_k on $LST(d)$ is
$$p_k(d) = \max_{i=1}^{r} p_k^i(d) \tag{3}$$

THEOREM 4. *Our payment scheme based on LST is strategy-proof and minimum among all truthful payment schemes based on LST.*

PROOF. First, for every round i, the payment scheme $p_k^i(d)$ belongs to VCG mechanism, so e_k gets maximum and non-negative utility from round i if reveals its true cost c_k. Notice the final payment scheme is the maximal of $p_k^i(d)$ over all round i, so e_k gets maximum and non-negative under payment scheme (3) when it reveals its true cost c_k. Thus, our payment scheme is strategy-proof.

Now we prove the optimality of our payment scheme. We prove by contradiction. Suppose there exists a payment scheme $\tilde{p}$ such that for profile d, $\tilde{p}_k(d) < p_k(d_k)$, which equals $\tilde{p}_k(d) = p_k(d_k) - \delta$ ($\delta > 0$). From the IR property, we can assure that e_k is selected under profile d. Here we argue that if $d_k < p_k(d_k)$, then $e_k \in LST(d)$. Without loss of generality, we can assume $p_k(d_k) = p_k^i(d_k)$. If e_k is selected before round i, then done. Else, in round i, we have $d_k < p_k(d_k) = p_k^i(d_k) = w_i(d|^k\infty) - |\mathsf{P}'_i(d|^k 0)|$. This implies that $w_i(d|^k\infty) > |\mathsf{P}'_i(d|^k 0)| + d_k$, which guarantees that e_k is selected in round i. Considering profile $d|^k p_k(d_k) - \frac{\delta}{2}$ with e_k's true cost $c_k = p_k(d_k) - \frac{\delta}{2}$. From lemma 2, e_k's payment under $\tilde{p}$ equals to $\tilde{p}_k(d|^i p_k(d_k) - \frac{\delta}{2}) = p_k(d_k) - \delta$, which is smaller than the true cost $c_k = p_k(d_k) - \frac{\delta}{2}$ of link e_k. This violates the assumption that payment scheme $\tilde{p}$ is truthful, which finishes our proof. □

3.3.4 Computational Complexity

For every round, the payment $p_k^i(d)$ could be calculated in time $O(n \log n + m)$. There are r rounds, where r is the number of receivers, so overall complexity is $O(rn \log n + rm)$. The question left unsolved is can we reduce the time complexity to $O(n \log n + m)$, which should be optimal if we can achieve that.

4. MULTICAST IN NODE WEIGHTED COMMUNICATION NETWORKS

In this section, we discuss in detail how to conduct truthful multicast when the network is modelled by a node weighed communication graph. We specifically study the following two structures: virtual minimum spanning tree (VMST) and node weighted Steiner tree (NST). Although LCPS is a very commonly used structure in node weighted wireless networks, but its construction and strategy-proof payment scheme are nearly the same as in the link weighted networks, so we omit the discussion of this structure here. Notice both VMST and NST are source based multicast trees, which implies that the receivers could also be the sender. In practice, for those shared based trees, receivers/senders in the same multicast group are usually belong to the same organization or company, so their behavior can be expected to be cooperative instead of uncooperative. Thus, we assume every receiver will relay the packet for free.

4.1 Virtual Minimum Spanning Tree

4.1.1 Constructing the VMST

We first describe our method to construct the virtual minimum spanning tree.

ALGORITHM 3. ***Virtual MST Algorithm***

1. *First, calculate the pairwise least cost path* $\mathsf{LCP}(q_i, q_j, d)$ *between any two terminals* $q_i, q_j \in Q$ *when the cost vector is* d.

2. *Construct a virtual complete link weighted network* $K(d)$ *using* Q *as its terminals, where the link* $q_i q_j$ *corresponds to the least cost path* $\mathsf{LCP}(q_i, q_j, d)$*, and its weight* $w(q_i q_j)$ *is the cost of the path* $\mathsf{LCP}(q_i, q_j, d)$*, i.e.,* $w(q_i q_j) = |\mathsf{LCP}(q_i, q_j, d)|$.

3. *Build the minimum spanning tree (MST) on* $K(d)$*. The resulting MST is denoted as* $VMST(d)$.

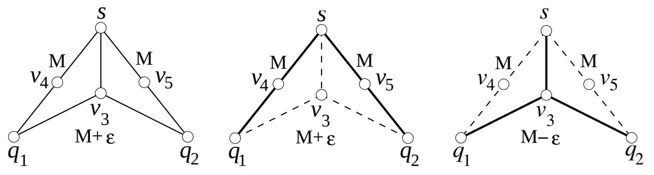
4. *For every virtual link* $q_i q_j$ *in* $VMST(d)$*, we find the corresponding least cost path* $\mathsf{LCP}(q_i, q_j, d)$ *in the original network. Combining all these paths can generate a subgraph of* G*, say* $VMSTO(d)$.

5. *In* $VMSTO(d)$*, build a LCPS rooted at* s *and spanning all receivers. The final tree is called* $LCPS - VMST(d)$.

Notice terminal v_k is in $LCPS - VMST(d)$ iff v_k is on some virtual links in the $VMST(d)$, so we can focus our attention on these terminals in $VMST(d)$.

4.1.2 VCG mechanism on VMST is not strategy-proof

In this subsection, we show that a simple application of VCG mechanism on VMST is not strategy-proof. Figure 4 illustrates such an example where terminal v_3 can lie its cost to improve its utility when output is VMST. The payment to terminal v_3 is 0 and its utility is also 0 if it reports its cost truthfully. The total payment to terminal v_3 when v_3 reported a cost $d_3 = M - \epsilon$ is $\omega(VMST(c|^3\infty)) - \omega(VMST(c|^3 d_3)) + d_3 = 2M - (M - \epsilon) + M - \epsilon = 2M$ and the utility of terminal v_3 becomes $u_3(c|^3 d_3) = 2M - (M + \epsilon) = M - \epsilon$, which is larger than $u_3(c) = 0$. Thus, VCG mechanism based on VMST is not strategy-proof.



(a) Graph G (b) VMST (c) VMST with lie

Figure 4: The cost of terminals are $c_4 = c_5 = M$ **and** $c_3 = M + \epsilon$**.**

4.1.3 Strategyproof Mechanism on VMST

Before discussing the strategyproof mechanism based on VMST, we give some related definitions first. Given a spanning tree T and a pair of terminals p and q on T, clearly there is a unique path connecting them on T. We denote such path as $\Pi_T(p, q)$, and the edge with the maximum length on this path as $LE(p, q, T)$. For simplicity, we use $LE(p, q, d)$ to denote $LE(p, q, VMST(d))$ and use $LE(p, q, d|^k d'_k)$ to denote $LE(p, q, VMST(d|^k d'_k))$.

Following is our truthful payment scheme when the output is the multicast tree $VMST(d)$.

ALGORITHM 4. *Truthful payment scheme based on VMST*

1. For every terminal $v_k \in V \setminus Q$ in G, first calculate $VMST(d)$ and $VMST(d|^k\infty)$ according to the terminals' declared costs vector d.

2. For any edge $e = q_iq_j \in VMST(d)$ and any terminal $v_k \in \mathsf{LCP}(q_i, q_j, d)$, we define the payment to terminal v_k based on the virtual link q_iq_j as follows:

$$p_k^{ij}(d) = |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d)| + d_k.$$

   Otherwise, $p_{ij}^k(d)$ is 0. The final payment to terminal v_k based on $VMST(d)$ is

$$p_k(d) = \max_{q_iq_j \in VMST(d)} p_k^{ij}(d). \quad (4)$$

THEOREM 5. *Our payment scheme (4) is strategyproof and minimum among all truthful payment schemes based on VMST structure.*

Instead of proving Theorem 5, we prove Theorem 6, Theorem 9 and Theorem 11 in the remaining of this subsection.

Before the proof of Theorem 5, we give some related notations and observation. Considering the graph $K(d)$ and a node partition $\{Q_i, Q_j\}$ of Q, if an edge's two end nodes belong to different node set of the partition, we call it a *bridge*. All bridges q_sq_t over node partition Q_i, Q_j in the graph $K(d)$ satisfying $v_k \notin \mathsf{LCP}(q_s, q_t, d)$ form a bridge set $B^{-v_k}(Q_i, Q_j, d)$. Among them, the bridge with the minimum length is denoted as $MB^{-v_k}(Q_i, Q_j, d)$ when the nodes' declared cost vector is d. Similarly, All bridges q_sq_t over node partition Q_i, Q_j in the graph $K(d)$ satisfying $v_k \in \mathsf{LCP}(q_s, q_t, d)$ form a bridge set $B^{v_k}(Q_i, Q_j, d)$. The bridge in $B^{v_k}(Q_i, Q_j, d)$ with the minimum length is denoted as $BM^{v_k}(Q_i, Q_j, d)$. Obviously, we have

$$BM(Q_i, Q_j, d) = \min\{BM^{v_k}(Q_i, Q_j, d), BM^{-v_k}(Q_i, Q_j, d)\}.$$

We then state our main theorems for the payment scheme discussed above.

THEOREM 6. *Our payment scheme satisfies IR.*

PROOF. First of all, if terminal v_k is not chosen as relay terminal, then its payment $p_k(d|^kc_k)$ is clearly 0 and its valuation is also 0. Thus, its utility $u_k(d|^kc_k)$ is 0.

When terminal v_k is chosen as a relay terminal when reveals its true cost c_k, from the following observation 1 about MST we have $|LE(q_i, q_j, d|^k\infty)| \geq |\mathsf{LCP}(q_i, q_j, d|^kc_k)|$. The lemma immediately follows from

$$p_{ij}^k(d|^kc_k) = |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^kc_k)| + c_k > c_k.$$

This finishes the proof. $\square$

OBSERVATION 1. *For any cycle C in graph G, assume e_c is the longest edge in the cycle, then $e_c \notin MST(G)$.*

From the definition of the incentive compatibility (IC), we assume the d_{-k} is fixed throughout this proof. For our convenience, we will use $G(d_k)$ to represent the graph $G(d|^kd_k)$. We first prove a series of lemmas that will be used to prove that our payment scheme satisfies IC.

LEMMA 7. *If $v_k \in q_iq_j \in VMST(d)$, then $p_k^{ij}(d)$ doesn't depends on d_k.*

PROOF. Remember that the payment based on link q_iq_j is $p_k^{ij}(d) = |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d)| + d_k$. The first part $LE(q_i, q_j, d|^k\infty)$ is the longest edge of the unique path from q_i to q_j on tree $VMST(d|^k\infty)$. Clearly, it is independent of d_k. Now considering the second part $\mathsf{LCP}(q_i, q_jd) - d_k$. From the assumption we know that $v_k \in \mathsf{LCP}(q_i, q_j, d)$, so the path $\mathsf{LCP}(q_i, q_j, d)$ remains the same regardless of v_k's declared cost d_k. Thus, the summation of all terminals' cost on $\mathsf{LCP}(q_i, q_j, d)$ except terminal v_k equals to

$$|\mathsf{LCP}(q_i, q_j, d|^k0)| = |\mathsf{LCP}(q_i, q_j, d)| - d_k.$$

In other word, the second part is also independent of d_k. Now we can write the payment to a terminal v_k based on edge q_iq_j as following:

$$p_k^{ij}(d) = |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^k0)|,$$

Here terminal $v_k \in \mathsf{LCP}(q_i, q_j, d)$ and $q_iq_j \in VMST(d)$. $\square$

If a terminal v_k lies its cost c_k upward, we denote the lied cost as $\overline{c_k}$. Similarly, if terminal v_k lies its cost c_k downward, we denote the lied cost as $\underline{c_k}$. Let $E_k(d_k)$ be the set of edges q_iq_j such that $v_k \in \mathsf{LCP}(q_i, q_j, d)$ and $q_iq_j \in VMST(d)$ when terminal v_k declares a cost d_k. From the lemma 7 the non-zero payment to v_k is defined based on $E_k(d_k)$. Following lemma reveals the relationship between d_k and $E_k(d_k)$:

LEMMA 8. *$E_k(d_k) \subseteq E_k(\underline{d_k'})$ when $d_k' \leq d_k$.*

We now state the proof that payment scheme 4 satisfies IC.

THEOREM 9. *Our payment scheme satisfies the incentive compatibility (IC).*

PROOF. For terminal v_k, if it lies its cost from c_k to $\overline{c_k}$, then $E_k(\overline{c_k}) \subseteq E_k(c_k)$, which implies that payment

$$\begin{aligned} p_k(d|^k\overline{c_k}) &= \max_{q_iq_j \in E_k(\overline{c_k})} p_k^{ij}(d|^k\overline{c_k}) \\ &\leq \max_{q_iq_j \in E_k(c_k)} p_k^{ij}(d|^kc_k) = p^k(d|^kc_k). \end{aligned}$$

Thus, terminal v_k won't lies it cost upward, so we focus our attention on the case when terminal v_k lies its cost downward.

From Lemma 8, we know that $E_k(c_k) \subseteq E_k(\underline{c_k})$. Thus, we only need to consider the payment based on edges in $E_k(\underline{c_k}) - E_k(c_k)$. For edge $e = q_iq_j \in E_k(\underline{c_k}) - E_k(c_k)$, let $q_I^kq_J^k = LE(q_i, q_j, d|^k\infty)$ in the spanning tree $VMST(d|^k\infty)$. If we remove the edge $q_I^kq_J^k$, we have a vertex partition $\{Q_I^k, Q_J^k\}$, where $q_i \in Q_I^k$ and $q_j \in Q_J^k$. In the graph $K(d)$, we consider the bridge $BM(Q_I^k, Q_J^k, d)$ whose weight is minimum when the terminals cost vector is d. There are two cases need to be considered about $BM(Q_I^k, Q_J^k, d)$: 1) $v_k \notin BM(Q_I^k, Q_J^k, d|^kc_k)$ or 2) $v_k \in BM(Q_I^k, Q_J^k, d|^kc_k)$. We discuss them individually.

Case 1: $v_k \notin BM(Q_I^k, Q_J^k, d|^kc_k)$. In this case, edge $q_I^kq_J^k$ is the minimum bridge over Q_I^k and Q_J^k. In other words, we have $|LE(q_i, q_j, |^k\infty)| \leq |\mathsf{LCP}(q_i, q_j, d|^kc_k)|$. Consequently

$$\begin{aligned} p_k^{ij}(d|^k\underline{c_k}) &= |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^k\underline{c_k})| + \underline{c_k} \\ &= |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^kc_k)| + c_k \\ &\leq c_k, \end{aligned}$$

which implies v_k will not get benefit from lying its cost downward.

Case 2: $v_k \in BM(Q_I^k, Q_J^k, d|^k c_k)$. From the assumption that $q_iq_j \notin VMST(G(d|^k c_k))$, we know edge q_iq_j cannot be $BM(Q_I^k, Q_J^k, d|^k c_k)$. Thus, there exists an edge $q_sq_t \neq q_iq_j$ such that $v_k \in \mathsf{LCP}(q_s, q_t, d|^k c_k)$ and $q_sq_t = BM(Q_I^k, Q_J^k, d|^k c_k)$. This guarantees that $q_sq_t \in VMST(d|^k c_k)$.

Obviously, q_sq_t can't appear in the same set of Q_I^k or Q_J^k. Thus, $q_I^k q_J^k$ is on the path from q_s to q_t in graph $VMST(d|^k\infty)$, which implies that $|\mathsf{LCP}(q_I^k, q_J^k, d|^k\infty)| = |LE(q_i, q_j, d|^k\infty)| \leq |LE(q_s, q_t, d|^k\infty)|$. Using Lemma 8, we have $\mathsf{LCP}(q_s, q_t, d|^k \underline{c_k}) \in VMST(d|^k \underline{c_k})$). Thus,

$$\begin{aligned} p_k^{ij}(d|^k \underline{c_k}) &= |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^k \underline{c_k})| + \underline{c_k} \\ &= |LE(q_i, q_j, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^k c_k)| + c_k \\ &\leq |LE(q_s, q_t, d|^k\infty)| - |\mathsf{LCP}(q_i, q_j, d|^k c_k)| + c_k \\ &\leq |LE(q_s, q_t, d|^k\infty)| - |\mathsf{LCP}(q_s, q_t, d|^k c_k)| + c_k \\ &= p_k^{st}(d|^k c_k) \end{aligned}$$

This inequality concludes that even if v_k lies its cost downward to introduce some new edges in $E_k(\underline{c_k})$, the payment based on these newly introduced edges is no larger than the payment on some edges already contained in $E_k(c_k)$. In summary, node v_i don't have the incentive to lie its cost upward or downward, which proves the IC. □

Before proving Theorem 11, we prove the following lemma regarding all truthful payment schemes based on VMST.

LEMMA 10. *If $v_k \in VMST(d|^k c_k)$, then as long as $d_k < p_k(d|^k c_k)$ and d^{-k} fixed, $v_k \in VMST(d)$.*

PROOF. Again, we prove it by contradiction. Assume that $v_k \notin VMST(d)$. Obviously, $VMST(d) = VMST(d|^k\infty)$. Assume that $p_k(d|^k c_k) = p_k^{ij}(d|^k c_k)$, i.e., its payment is computed based on edge q_iq_j in $VMST(d|^k c_k)$. Let q_Iq_J be the $LE(q_i, q_j, d|^k\infty)$ and $\{Q_i, Q_j\}$ be the vertex partition introduced by removing edge q_Iq_J from the tree $VMST(d|^k\infty)$, where $q_i \in Q_i$ and $q_j \in Q_j$. The payment to terminal v_k in $VMST(d|^k c_k)$ is $p_k(d|^k c_k) = |\mathsf{LCP}(q_I, q_J, d|^k\infty)| - c_{ij}^{v_k}$, where $c_{ij}^{v_k} = |\mathsf{LCP}(q_i, q_j, d|^k 0|$. When v_k's declare its cost as d_k, the length of the path $\mathsf{LCP}(q_i, q_j, d)$ becomes $c_{ij}^{v_k} + d_k = |\mathsf{LCP}(q_I, q_J, d|^k\infty)| - p_k(d|^k c_k) + d_k < |\mathsf{LCP}(q_I, q_J, d|^k\infty)|$.

Now consider the spanning tree $VMST(d)$. We have assumed that $v_k \notin VMST(d)$, i.e., $VMST(d) = VMST(d|^k\infty)$. Thus, among the bridge edges over Q_i, Q_j, edge q_Iq_J has the least cost when graph is $G\backslash v_k$ or $G(d|^k d_k)$. However, this is a contradiction to we just proved: $|\mathsf{LCP}(q_i, q_j, d|^k d_k)| < |\mathsf{LCP}(q_I, q_J, d|^k\infty)|$. This finishes the proof. □

We now ready to show that our payment scheme is optimal among all truthful mechanisms using VMST.

THEOREM 11. *Our payment scheme is the minimum among all truthful payment schemes based on VMST structure.*

PROOF. We prove it by contradiction. Assume that there is another truthful payment scheme, say $\mathcal{A}$, based on VMST, whose payment is smaller than our payment for a terminal v_k under cost profile d. Assume that the payment calculated by $\mathcal{A}$ for terminal v_k is $\tilde{p}_k(d) = p_k(d) - \delta$, where $p_k(d)$ is the payment calculated by our algorithm and $\delta > 0$.

Now consider another profile $d|^k d_k'$, where terminal has the true cost $c_k = d_k' = p^k(d) - \frac{\delta}{2}$. From Lemma 10, we know that v_k is still in $VMST(d|^k d_k')$. Using Lemma 2, we know that the payment for terminal v_k using algorithm $\mathcal{A}$ is $p_k(c) - \delta$, which is independent of terminal v_k's declared cost. Notice that $d_k = p_k(d) - \frac{\delta}{2} > p_k(d) - \delta$. Thus, terminal v_k has a negative utility under payment scheme $\mathcal{A}$ when it reveals it true cost under cost profile $d|^k d_k'$, which violates the incentive compatibility (IC). This finishes the proof. □

By summarizing Theorem 6, Theorem 9 and Theorem 11, we get Theorem 5.

4.1.4 Computational Complexity

We now discuss how to compute the payment to every relay terminal efficiently. Assume that the original communication graph G has n vertices and m edges.

One naive method of computing the payment works as follows. We first construct the complete graph $K(d)$ and then construct the spanning tree $VMST(d)$ on $K(d)$. It is easy to show the overall time complexity to construct $VMST(d)$ is $O(r^2 + rn\log n + rm) = O(rn\log n + rm)$, where r is the number of receivers. In order to calculate the payment for terminal $v_k \in \mathsf{LCP}(q_i, q_j, d) \in VMST(d)$, we should construct the tree $VMST(d|^k\infty)$, which will take time $O(rn\log n + rm)$. Finding the longest edge $LE(q_i, q_j, d|^k\infty)$ takes only $O(r)$ time. In the worst case, terminal v_k may appear on $O(r)$ edges of $VMST(d)$. Thus, we can calculate the payment for the single terminal v_k in time $O(r^2) + O(rn\log n + km) = O(rn\log n + rm)$. In the worst case, there could be $O(n)$ terminals on $VMST(d)$, so we can calculate the payment for all relay terminals in tree $VMST(G)$ in time $O(rn^2\log n + rmn)$.

Our improvement uses the fast payment for unicast as a subroutine. For a pair of terminals q_iq_j, we calculate the path $\mathsf{LCP}(q_i, q_j, d|^k\infty)$ for every terminal $v_k \in \mathsf{LCP}(q_i, q_j, d)$, which can be done in time $O(n\log n + m)$. It takes $O(r^2 n\log n + r^2 m)$ to find the complete graph $K(d|^k\infty)$ for every terminal v_k. Finding the MST on each such complete graph takes time $O(r^2)$. Thus, we can construct VMSTs for all these n complete graphs in time $O(r^2 n)$. Based on these n VMSTs, it takes $O(r^2)$ to calculate the payment for one terminal. Thus, in the worst case, it also takes $O(r^2 n)$ to calculate the payment to every relay terminal. Overall, the time complexity of this approach is $O(r^2 n\log n + r^2 m) + O(r^2 n) + O(r^2 n) = O(r^2 n\log n + r^2 m)$. When $r = o(\sqrt{n})$, this approach outperforms the naive approach with time complexity $O(n^2\log n + mn)$. When r is a constant, the time complexity of the above approach becomes $O(n\log n + m)$, which is optimum.

4.2 Node Weighted Steiner Tree (NST)

Compared with LST in link weighted network, the structure of node-weighted Steiner tree (NST) in a node weighted network is even tough. It is well-known [11, 13] that it is NP-hard to find the minimum cost multicast tree when given an arbitrary node weighted graph G, and it is at least as hard to approximate as the set cover problem. Klein and Ravi [13] showed that it can be approximated within $O(\ln r)$, where r is the number of receivers.

4.2.1 Constructing NST

We review the method used in [13] to find a NST. We first introduce some definitions that are essential to construct the NST. A *spider* is defined as a tree having at most one node of degree more than two. Such a node (if exists) is called the center of the spider. If the degree of the center node is m, then we call this spider is a m spider. It is easy to observe that a m spider has m leaves. Each path from the center to a leaf is called a *leg*. The *cost* of a spider S is defined as the sum of the cost of all nodes in spider S, denotes as $\omega(S)$. The number of terminals or *legs* of the spider is denoted by $t(S)$, and the ratio of a spider is defined as

$$\rho(S) = \frac{\omega(S)}{t(S)}.$$

Contraction of a spider is the operation of contracting all terminals of the spider to form one virtual terminal. Contracted virtual terminal has zero weight, and we make it a terminal with degree one.

Algorithm 5. ***Construct NST***

Repeat the following steps until no receivers left and there is only one virtual terminal left.

1. Find the spider S with the minimum $\rho(S)$ that connect some receivers and virtual terminals.[1]

2. Contract the spider S by treating all nodes in it as one virtual terminal. We call this as one *round*.

All nodes belong to the final unique virtual terminal form the NST.

Theorem 12. *[13] The tree constructed above has cost at most* $2 \ln k$ *times of the optimal.*

4.2.2 VCG mechanism on NST in not strategy-proof

Again, we may want to pay terminals based on VCG scheme, i.e., the payment to a terminal $v_k \in NST(d)$ is

$$p_k(d) = \omega(NST(d|^k\infty)) - \omega(NST(d)) + d_k.$$

We show by an example that the payment scheme does *not* satisfy IR property: it is possible that some terminal has negative utility. Figure 5 illustrates such an example. It

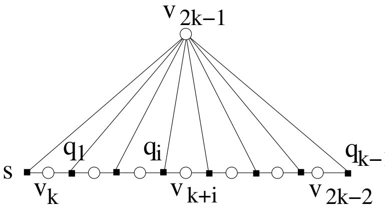


Figure 5: Terminals q_i, $1 \le i < k$ are receivers; the cost of terminal v_{2k-1} is 1. The cost of each terminal v_i is $\frac{2}{k+1-i} - \epsilon$, where ϵ is a sufficiently small positive number.

is not difficulty to show that, in the first round, terminal v_k is selected to connect terminals s and q_1 with cost ratio $\frac{1}{k} - \frac{\epsilon}{2}$ (while all other spiders have cost ratio at least $\frac{1}{k}$). Then terminals s, v_k and q_1 form a virtual terminal. At the beginning of round r, we have a virtual terminal, denoted by V_r formed by terminals v_{k+i-1}, $1 \le i \le r-1$, and receivers q_i, $1 \le i \le r$; all other receivers q_i, $r < i < k$ are the remaining terminals. It is easy to show that we will select terminal q_{k+r-1} at round r to connect V_r and q_{r+1} with cost ratio $\frac{1}{k+1-r} - \frac{\epsilon}{2}$. Thus, the total cost of the tree $NST(G)$ is $\sum_{i=1}^{k-1}(\frac{2}{k+1-i} - \epsilon) = 2H(k) - 2 - (k-1)\epsilon$.

When terminal v_k is not used, it is easy to see that the final tree $NST(G\backslash u_1)$ will only use terminal v_{2k-1} to connect all receivers with cost ratio $\frac{1}{k}$ when $\frac{1}{k-1} - \frac{\epsilon}{2} > \frac{1}{k}$. Notice that this condition can be trivially satisfied by letting $\epsilon = \frac{1}{k^2}$. Thus, the utility of terminal v_k is $p_1(d) - c(v_k) = \omega(NST(G\backslash v_k)) - \omega(NST(G)) = -2H(k) + 3 + (k-1)\epsilon$, which is negative when $k \ge 8$, and $\epsilon = 1/k^2$.

4.2.3 Strategy-proof mechanism based on NST

Notice, the construction of NST tree is by rounds. Following, we show that if terminal v_k is selected as part of the spider with minimum ratio under cost profile d in a round i, then v_k is selected before or in round i under cost profile $d' = d|^k d'_k$ for $d'_k < d_k$. We prove this by contradiction, which assumes terminal v_k won't appear before round $i+1$. Notice that the graph remains the same for round i after the profile changes, so spider $S_i(d)$ under cost profile d is still a valid spider under cost profile d'. Its ratio becomes $\omega_i^k(d) - d_k + d'_k < \omega_i^k(d)$ while all other spiders' ratio keeps the same if they don't contain v_k. Thus, spider $S_i^k(d)$ has the minimum ratio among all spiders under cost profile d', which is a contradiction. So for terminal v_k, there exists a real value $B_k^i(d_{-k})$ such that terminal v_k selected before or in round i iff $d_k < B_k^i(d_{-k})$. If they are r rounds, we have an increasing sequence

$$B_k^1(d_{-k}) \le B_k^2(d_{-k}) \le \cdots \le B_k^r(d_{-k}) = B_k(d_{-k})$$

Obviously, terminal v_k is selected in the final multicast tree iff $d_k < B_k(d_{-k})$. Following is our payment scheme based on NST. For a node v_k, if v_k is selected then it gets payment

$$p_k(d) = B_k(d_{-k}). \tag{5}$$

Otherwise, it gets payment 0.

Regarding this payment we have the following theorem:

Theorem 13. *Our payment scheme (5) is truthful, and among all truthful payment schemes for multicast tree based on NST, our payment is minimal.*

Proof. From our conclusion that v_k is selected iff $d_k < B_k^i(d_{-k})$, we have $u_k(d) = B_k(d_{-k}) - d_k > 0$, which implies IR. Now we prove our payment scheme (5) satisfies IC by cases. Notice when v_k is selected, its payment doesn't depend on d_k, so we only need to discuss the following two cases:

Case 1: When v_k declares c_k, it is selected. What happens if it lies its cost upward as d_k to make it not selected? From the IR property, v_k gets positive utility when it reveals its true cost while it gets utility 0 when it lies it cost as d_k. So v_k has better not to lie.

Case 2: When v_k declares c_k, it is not selected. What happens if it lies its cost downward as d_k to make it selected? When v_k reveals c_k, it has utility 0, after lying it has utility $B_k(d_{-k}) - c_k$. From the assumption that v_k is not selected under cost profile $d|^k c_k$, we have $B_k(d_{-k}) \le c_k$. Thus, v_k will get non-positive utility if it lies, which ensures v_k revealing its true cost c_k.

So overall, v_k will always choose to reveal its actual cost to maximize its utility (IC property).

[1]For simplicity of the proof, we assume there doesn't have two spiders with the same ratio. Dropping the assumption won't change our results.

Next we prove that our payment is minimal. We prove it by contradiction, suppose there exists such payment scheme $\tilde{P}$ such that for terminal v_k under cost profile d, the payment to $\tilde{P}_i(d)$ is smaller than our payment. Notice in order to satisfies the IR, the terminal must be selected, so we assume $\tilde{P}_i(d) = B_k(d_{-k}) - \delta$, while δ is a positive real number. Now considering the profile $d' = d|^k(B_k(d_{-k}) - \frac{\delta}{2})$ with v_k's actual cost $c_k = B_k(d_{-k}) - \frac{\delta}{2}$. Obviously, v_k is selected, from lemma 2 the payment to v_k is $B_k(d_{-k}) - \delta$. Thus, the utility of v_k becomes $u_k(d') = B_k(d_{-k}) - B_k(d_{-k}) - \delta + \frac{\delta}{2} = -\frac{\delta}{2} < 0$, which violates the IR. This finishes our proof. □

With Theorem 13, we only need focus our attention on how to get the value $B_k^i(d_{-k})$. Before we present our algorithm to find $B_k^i(d_{-k})$, we first review in details how to find the minimum ratio spider. In order to find the spider with the minimum ratio, we find the spider centered at terminal v_j with the minimum ratio over all terminals $v_j \in V$ and choose the minimum among them. The algorithm is as follows.

ALGORITHM 6. ***Find the minimum ratio spider***

Do the following process for all $v_j \in V$:

1. *Calculate the shortest path tree rooted at v_j and spanning all terminals. We call each shortest path a* branch. *The weight of the branch is defined as the length of the shortest path. Notice that the weighted of the shortest path doesn't include the weight of the center node v_j of the spider.*
2. *Sort the branches according to their weights.*
3. *For every pair of branches, if they have terminals in common then remove the branch with larger weight. Assume the remaining branches are*
$$L(v_j) = \{L_1(v_j), L_2(v_j), \cdots, L_r(v_j)\}$$
sorted in ascending order according to their weights.
4. *Find the minimum ratio spider with center v_j by linear scanning: the spider is formed by the first t branches such that $\frac{c_j + \sum_{k=1}^{t} L_k}{t} \le \frac{c_j + \sum_{k=1}^{h} L_k}{h}$ for any $h \neq t$.*

Assume that the spider with minimum ratio centered at terminal v_j is $S(v_j)$ and its ratio is $\rho(v_j)$. Then the spider with minimum ratio is $S = \{S(v_j) | v_j \in V$ and $\rho(v_j) = \min_{v_i \in V} \rho(v_i)\}$.

In Algorithm 6, $\omega(L_i(v_j))$ is defined as the sum of the terminals' cost on this branch, and $\Omega_i(L(v_j)) = \sum_{s=1}^{i} \omega(L_s(v_j)) + c_j$. If we remove node v_k, the minimum ratio spider centered at v_j is denoted as $S^{-v_k}(v_j)$ and its ratio is denoted as $\rho^{-v_k}(v_j)$. Let $L_1^{-v_k}(v_j), L_2^{-v_k}(v_j), \cdots, L_r^{-v_k}(v_j)$ be those branches in ascending order before linear scan.

From now on, we fix d_{-k} and graph G to study the relationship between minimum ratio of spider centered at v_j $\rho(v_j)$ and d_k. First, we have the following observation.

OBSERVATION 2. *The number of the legs of minimum ratio spider decreases over d_k.*

If the minimum ratio spider with terminal v_k has t legs, then its ratio will be a line with slope of $\frac{1}{t}$. So the ratio-cost function is several line segments. From Observation 2, these line segments have decreasing slopes and thus it has at most r segments, where r is the number of receivers. So given a real value y, we can find corresponding cost of v_k in time $O(\log r)$. The algorithm to find these line segments is as follows.

ALGORITHM 7. ***Find the ratio-cost function*** $y = \mathcal{R}_{v_j}(x)$

If $j = k$ then apply the following procedures:

1. *Apply step $1, 2, 3$ of algorithm 6 to get $L(v_k)$.*
2. *Set number of legs to $t = 2$, lower bound $lb = 0$ and upper bound $ub = 0$.*
3. *While $t < r$*
   - *(a) $ub = t \times \omega(L(v_k)) - \Omega_t(L(v_k))$.*
   - *(b) $y = \frac{\Omega_t(L(v_k)) + x}{t}$ for $x \in [lb, ub)$.*
   - *(c) Set $lb = ub$ and $t = t + 1$.*
   - *(d) $y = \frac{\Omega_r(L(v_k)) + x}{r}$ for $x \in [lb, \infty)$.*

Otherwise, we do as follows:

1. *Remove terminal v_k, apply algorithm 6 to find $S^{-v_k}(v_j)$.*
2. *Find the shortest path with terminal v_k from v_j to every receiver, sort these paths according to their length in a descending order, say sequence*
$$L^{v_k}(v_j) = \{L_1^{v_k}(v_j), L_2^{v_k}(v_j), \cdots, L_r^{v_k}(v_j)\}.$$
Here, r is the number of terminals, and $\omega(L_i^{v_k}(v_j))$ is the sum of terminals on path $L_i^{v_k}(v_j)$ excluding *terminal v_k.*
3. *t is the index for branches in $L^{v_k}(v_j)$ and l is the index for paths in $L^{-v_k}(v_j)$.*
4. *For $L_t^{v_k}(v_j)$ $(1 \le t \le r)$, there exists at most one branch $L_l^{-v_k}(v_j)$ such that they have terminals in common. If such branch exists, define upper bound uc_t for $L_t^{v_k}(v_j)$ equals $\omega(L_l^{-v_k}(v_j)) - \omega(L_t^{v_k}(v_j))$; else set $uc_t = \infty$. Linear scan uc_t for $t = 1$ to $r - 1$, if $uc_t \ge uc_{t+1}$, then remove $L_{t+1}(v_j)$ from sequence $L^{v_k}(v_j)$.*
5. *Initialize $t = 1$, $l = 1$, lower bound $lb = 0$ and upper bound $ub = 0$. Then apply the following algorithm:*
While $L_t^{v_k}(v_j) \in L^{v_k}(v_j)$ and $t \le r$ do:
   - *(a) while $\omega(L_l^{-v_k}(v_j)) \ge \frac{\Omega_l(L^{-v_k}(v_j)) + \omega(L_t^{v_k}(v_j)) + lb}{l}$ and $l < r$:*
     - *i. Set $ub = l \times \omega(L_l^{-v_k}(v_j)) - \Omega_l(L^{-v_k}(v_j)) - \omega(L_t^{v_k}(v_j))$.*
     - *ii. If $ub \ge uc_t$ break;*
     - *iii. Set $y = \frac{\Omega_{l-1}(L^{-v_k}(v_j)) - \omega(L_t^{v_k}(v_j)) + x}{l}$ for $x \in [lb, ub)$.*
     - *iv. Set $lb = ub$.*
     - *v. Set $l = l + 1$.*
   - *(b) Set $y = \frac{\Omega_{l-1}(L^{-v_k}(v_j)) - \omega(L_t^{v_k}(v_j)) + x}{l}$ for $x \in [lb, uc_t)$.*
   - *(c) Set $lb = uc_t$ and $t = t + 1$.*
6. *The ratio-cost function is a collection of line segments in $x - y$ coordinate, find the value x_k such that $\mathcal{R}(x_k) = \omega(S^{-v_k}(v_j))$. Replace the function for $x \ge x_k$ as $y = \omega(S^{-v_k}(v_k))$.*

Given a real value x, the corresponding cost for terminal v_k is denoted by $\mathcal{R}_{v_j}^{-1}(x)$. Finally, we give the algorithm to find value $B_k(d_{-k})$.

Algorithm 8. ***Algorithm to find*** $B_k(d_{-k})$

```
1. Remove terminal $v_k$ and find the multicast tree by using spider structure.
2. For every round $i$ in the first step, we have a graph called $G_i$ and a selected spider with ratio $\rho_i^{-v_k}$. Adding node $v_k$ and all its incident edges to $G_i$ get graph $G_i'$.
3. Find the function $y = \mathcal{R}_{v_j}^{-1}(x)$ for every terminal $v_j$ in graph $G_i'$ using algorithm 7.
4. Calculate $B_k^r(d_{-k}) = \max_{v_j \in V(G_i')}\{\mathcal{R}_{v_j}^{-1}(\rho_i^{-v_k})\}$.
5. $B_k(d_{-k}) = \max_{1 \le i \le r} B_k^i(d_{-k})$
```

The correctness of the algorithm is omitted due to space limit, please refer to the full version of this paper for details.

4.2.4 Computational Complexity

If we use Algorithm 5 to find $NST(d)$, every round we need time $O(rn\log n + rm)$, where r is the number of receivers. Notice there at most r rounds, so the overall time complexity is $O(r^2\log n + r^2m)$. For every node $v_k \in NST(d)$, if we apply Algorithm 6 to calculate the payment, it is not very difficult to get time complexity $O(rn\log n + rm)$ for each round. Thus, it takes time $O(r^2n\log n + r^2m)$ to find the payment for a single node $v_k \in NST(d)$. In the worst case, there could be up to $O(n)$ terminals in $NST(d)$, so overall time complexity is $O(r^2n^2\log n + r^2nm)$, which is quite expensive. Finding a more efficient way to reduce the time complexity will be one of our future works.

5. EXPERIMENTAL STUDIES

Remember that the payment of our structure is always larger than or equals the structure's actually cost. For a structure H, let $c(H)$ be its cost and $p_s(H)$ be the payment of scheme s based on this structure. We define the overpayment ratio of the payment scheme s based on structure H as

$$OR_s(H) = \frac{p_s(H)}{c(H)}. \qquad (6)$$

When it is clear from the context, we often simplify the notation as $OR(H)$.

Actually, there are some other definitions about overpayment ratio in the literature. In [2], the authors propose to compare the payment $p(H)$ with the cost of the new structure obtained from the graph $G - H$, i.e., removing H from the original graph G. Here, we only focus our attention on the overpayment ratio defined in (6).

We conducted extensive simulations to study the overpayment ratio of various schemes proposed in this paper. In our experiments, we will compare the performance of different structures proposed according to three different metrics: actual cost, total payment and overpayment ratio. Notice that, it is meaningless to compare the performance of structures for link weighted network with these structures for node weighted networks. Therefore, we consider LCPS(link weighted version), PMST and LST as one group for link weighted networks and LCPS(node weighted version), VMST and NST as another group for node weighted networks. Figure 6 and Figure 7 show the different multicast structures when the original graph is a unit disk graph (UDG). Here, the grey nodes are receivers.

5.1 Fixed Transmission Range and Fixed Number of Receivers

In the first experiment, we randomly generate n terminals uniformly in a $2000ft \times 2000ft$ region. The transmission range *range* of each terminal is set to $400ft$. For a link weighted graph, we assume the power needed to deliver a packet on e_i is $c_i(\frac{|e_i|}{100})^\kappa$, where κ is a value between 2 and 5. In our experiment $\kappa = 2.5$ and c_i is randomly drawn from the uniform distribution between 1 and 10. For a node weighted network, the weight of a node i is $c_i * 3^\kappa$ where c_i is randomly selected from a power level between 1 and 10. We vary the number of terminals in this region from 100 to 320, and fix the number of sender to 1 and receivers to 15. For a specific number of terminals, we generate 100 different networks, and compare the performance of different structures according to six different metrics: average cost(AC), maximum cost(MC), average payment(AP) and maximum payment(MP), average overpayment ratio(AOR) and maximum overpayment ratio(MOR).

For link weighted network, as shown in the upper figures of Figure 8, all structures' cost and payment decrease dramatically as the number of terminals increase. The PMST structure is the maximum for both cost, payment and overpayment ratio. But one advantage is that PMST is a shared based tree, which means it can be shared by all receivers/senders on the tree. LCPS is the most commonly used structure for source based tree, and it does win over the other two structures regarding AOR and MOR in our experiment. But in practice, people tend to care more about the actual cost (the so called "social efficiency") and the total payment. From this aspect, LST is the best candidate. Similar to LCPS, LST only needs information of LCP between terminals which can be obtained from the routing table for unicast. Thus, LST can also be implemented in a distrusted way but with more computational cost compared to LCPS.

For node weighted network, as shown in the lower figures of Figure 8, all structures' cost and payment also decrease as the number of terminals increase. Notice for VMST structure, we assume all receivers(senders) will relay message for free, so in order to compare the performance of these structure in a fair way, we set all receivers' private cost to 0 for both LCPS and NST structure. Unlike in link weighted network, the cost and payment of VMST and NST are much lower than the cost and payment of LCPS although the previous two are shared based tree. Like we expected, due to the lowe cost of the VMST and NST structures, the max overpayment ratio of these two structures are very unsteady and much high than the max overpayment ratio of LCPS.

5.2 Random Transmission Range and Fixed Number of Receivers

In our second experiment, we vary the transmission range of each wireless node from $100ft$ to $500ft$.

For link weighted network, the cost c_i of a link e_i is $(c_1 + c_2(\frac{|e_i|}{100})^\kappa)/10$, where c_1 takes value from 300 to 500 and c_2 takes value from 10 to 50. For node weighted network, the cost c_i of a terminal v_i is $(c_1 + c_2(\frac{r_i}{100})^\kappa)/10$, where c_1 takes

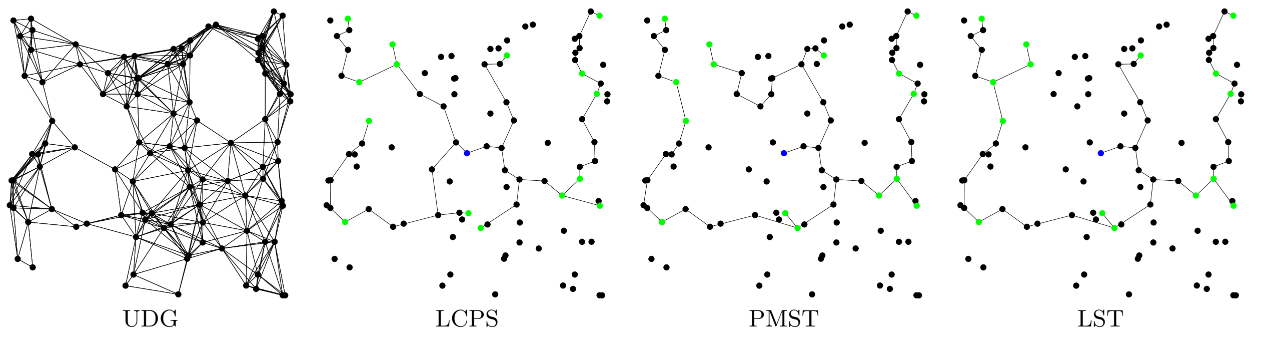


Figure 6: Multicast Structures for Link Weighted Network

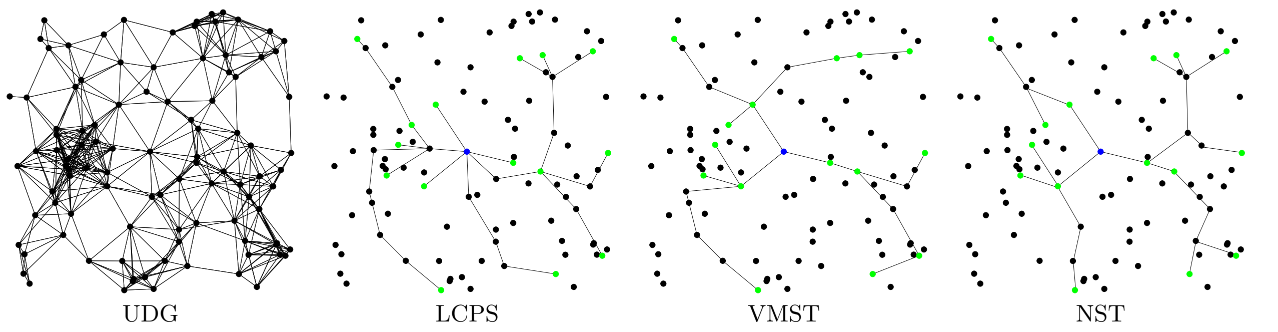


Figure 7: Multicast Structures for Node Weighted Network

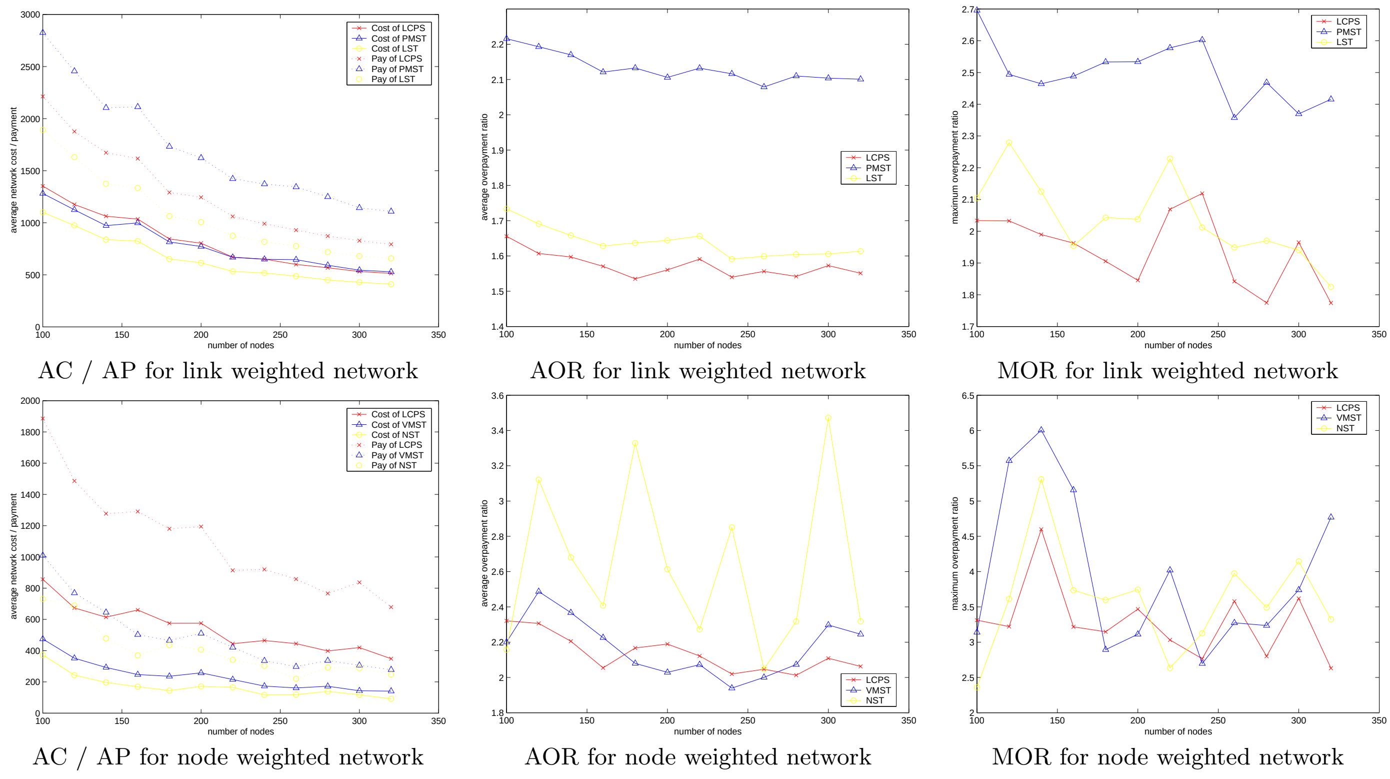
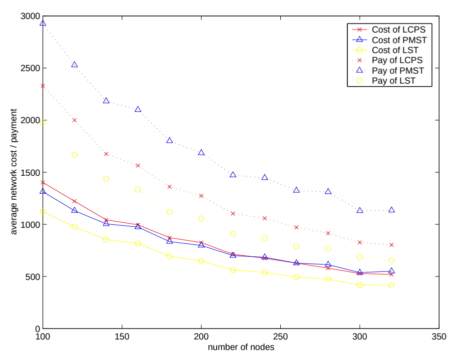
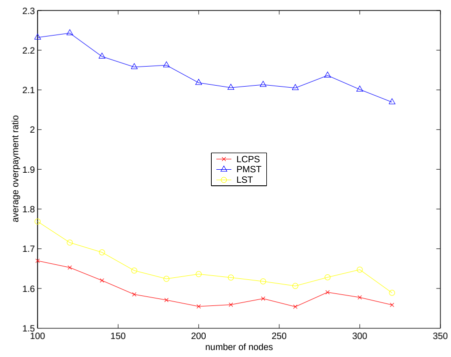


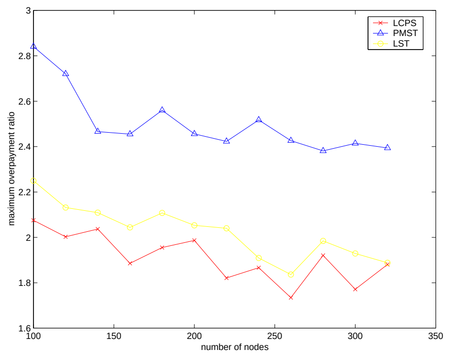
Figure 8: Results when the number of nodes in the networks are different (from 100 **to** 320**) for link weighted structures and node weighted structures . Here, we fix the transmission range to** $400ft$**.**



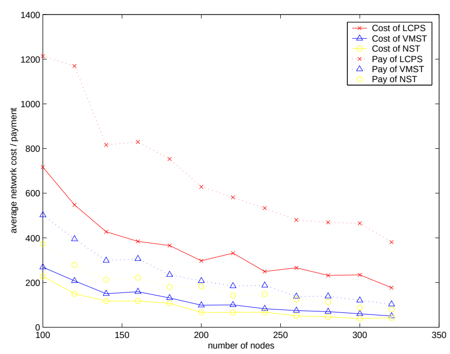
AC / AP for link weighted network



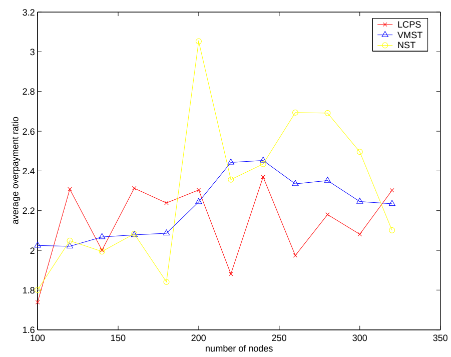
AOR for link weighted network



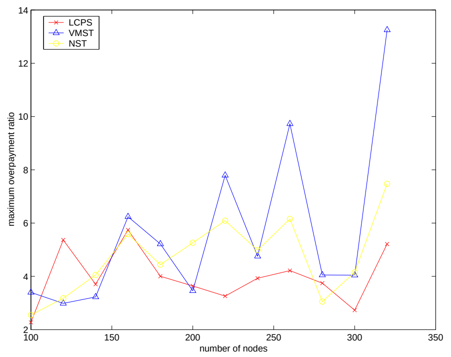
MOR for link weighted network



AC / AP for node weighted network



AOR for node weighted network



MOR for node weighted network

Figure 9: Results when the number of nodes in the networks are different (from 100 to 320) for link weighted structures and node weighted structures. Here, we randomly set the transmission range from $100ft$ to $500ft$.

value from 300 to 500, c_2 takes value from 10 to 50 and r_i is v_i's transmission range. The ranges of c_1 and c_2 we used here reflects the actual power cost in one second of a node to send data at $2Mbps$ rate.

Similar to the fixed transmission experiment, we vary the number of terminals in the region from 100 to 320, and fixed number of sender to 1 and receivers to 15. For a specific number of terminals, we generate 100 different networks, and compare the average cost, maximum cost, average payment and maximum payment, average overpayment ratio and maximum payment ratio.

Figure 9 shows the similar result for both link weighted network and node weighted network as the fixed transmission range experiments.

5.3 Random Transmission Range and Variable Number of Receivers

For structure H, we define cost density $CD(H) = \frac{c(H)}{r}$ and payment density $PD(H) = \frac{p(H)}{r}$, where r is the number of terminals in structure H.

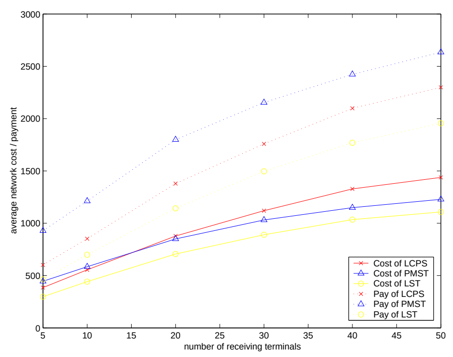
In this experiment, we study the relationship between average cost(AC), average payment(AP), average overpayment ratio(AOR), average cost density(CD), average payment density(PD) and the number of the terminals. We use the same power cost model in the previous experiment and the number of nodes in the region is set to 200. We vary the number of receivers from 5, 10, 20, $\cdots$ to 50.

Figure 10 shows that when the number of the receivers increases, under most circumstance, the overall payment and cost increase while the cost and payment for every terminal decrease. One exception is for node weighted network. Notice in node weighted network, we set all terminals' cost to 0, so it is naturally to expected that when the number of terminals larger than some threshold, even the total cost and payment will decrease. This experiment shows that more terminals in a multicast group can incur a lower cost and payment per terminal, which is one of the attractive properties of multicast.

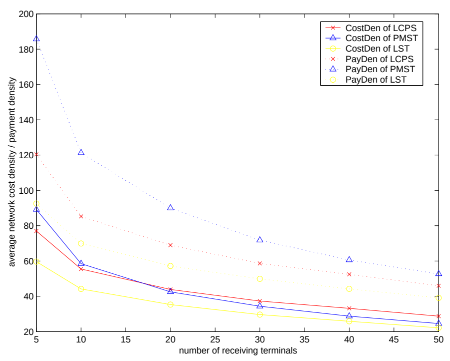
6. CONCLUSION

In this paper, we studied how to conduct efficient multicast in *selfish* wireless networks by assuming that each wireless terminal or communication link will incur a cost when it has to transit some data, and the cost is privately known to the wireless terminal or communication link. For each of the widely used structures for multicast, we designed a strategyproof multicast mechanism such that each agent has to truthfully report its cost to maximize its profit. The structures studied in this paper are least cost path star, pruning minimum spanning tree, virtual minimum spanning tree and the edge(node) weighted Steiner tree. Extensive simulations were conducted to study the practical performances of the proposed protocols.

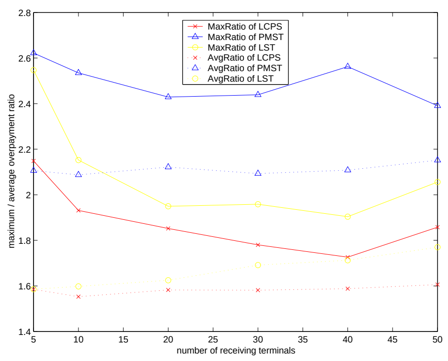
There are several unsolved challenges we left as future works. First, we would like to design algorithms that can compute these payments in asymptotically optimum time complexities. Second, in this paper, we only studied the tress-based structures for multicast. Practically, mesh-based structures maybe more needed for wireless networks to improve the fault tolerance of the multicast. We would like



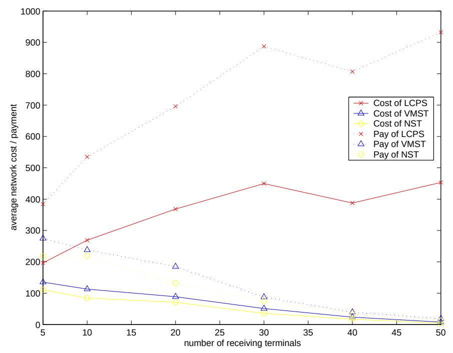
AC / AP for link weighted network



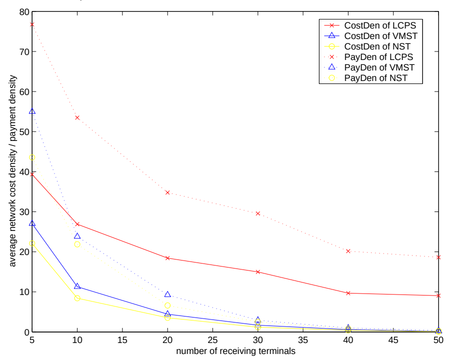
ACD / APD for link weighted network



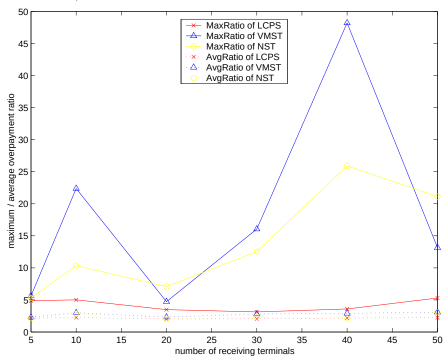
AOR / MOR for link weighted network



AC / AP for node weighted network



ACD / APD for node weighted network



AOR / MOR for node weighted network

Figure 10: Results when the number of receivers in the networks are different (from 10 to 50) for both link weighted and node weighted structures. Again, we randomly set the transmission range from $100ft$ to $500ft$.

to know whether we can design a strategyproof multicast mechanism for some mesh-based structures used for multicast. Third, all of our tree construction and payment calculation are performed in a centralized way, we would like to study how to design some distributed algorithm for it.

7. REFERENCES

[1] Anderegg, L., and Eidenbenz, S. Ad hoc-vcg: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *Proc. of the 9th ACM Mobicom* (2003), ACM Press, pp. 245–259.

[2] Archer, A., and l. Tardos. Frugal path mechanisms. In *ACM SODA* (2002), pp. 991–998.

[3] Blazevic, L., Buttyan, L., Capkun, S., Giordano, S., Hubaux, J. P., and Boudec, J. Y. L. Self-organization in mobile ad-hoc networks: the approach of terminodes. *IEEE Communications Magazine 39*, 6 (June 2001).

[4] Buttyan, L., and Hubaux, J. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications 5*, 8 (October 2003).

[5] Buttyan, L., and Hubaux, J. P. Enforcing service availability in mobile ad-hoc WANs. In *Proc. of the 1st ACM MobiHoc* (2000), pp. 87–96.

[6] Clarke, E. H. Multipart pricing of public goods. *Public Choice* (1971), 17–33.

[7] Feigenbaum, J., Papadimitriou, C., Sami, R., and Shenker, S. A BGP-based mechanism for lowest-cost routing. In *Proc. of ACM PODC* (2002), pp. 173–182.

[8] Feigenbaum, J., Papadimitriou, C. H., and Shenker, S. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences 63*, 1 (2001), 21–41.

[9] Green, J., and Laffont, J. J. Characterization of satisfactory mechanisms for the revelation of preferences for public goods. *Econometrica* (1977), 427–438.

[10] Groves, T. Incentives in teams. *Econometrica* (1973), 617–631.

[11] Guha, S., and Khuller, S. Improved methods for approximating node weighted steiner trees and connected dominating sets. In *Foundations of Software Technology and Theoretical Computer Science* (1998), pp. 54–65.

[12] Jakobsson, M., Hubaux, J.-P., and Buttyan, L. A micro-payment scheme encouraging collaboration in multi-hop cellular networks. In *Proceedings of Financial Cryptography* (2003).

[13] Klein, P., and Ravi, R. A nearly best-possible approximation algorithm for node-weighted steiner trees. Tech. Rep. CS-92-54, 1992.

[14] Marti, S., Giuli, T. J., Lai, K., and Baker, M. Mitigating routing misbe-havior in mobile ad hoc networks. In *Proc. of MobiCom* (2000).

[15] Nisan, N., and Ronen, A. Algorithmic mechanism design. In *Proc. 31st STOC* (1999), pp. 129–140.

[16] Robins, G., and Zelikovsky, A. Improved steiner tree approximation in graphs. In *Proc. ACM SODA* (2000).

[17] Srinivasan, V., Nuggehalli, P., Chiasserini, C. F., and Rao, R. R. Energy efficiency of ad hoc wireless networks with selfish users. In *European Wireless Conference* (2002).

[18] Srinivasan, V., Nuggehalli, P., Chiasserini, C. F., and Rao, R. R. Cooperation in wireless ad hoc wireless networks. In *IEEE Infocom* (2003).

[19] Takahashi, H., and Matsuyama, A. An approximate solution for the steiner problem in graphs. *Math .Jap. 24* (1980), 573–577.

[20] Vickrey, W. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance* (1961), 8–37.

[21] Wang, W., and Li, X.-Y. Truthful low-cost unicast in selfish wireless networks. In *4th WMAN of IPDPS04* (2004).