Abstract—Conventionally, most network protocols assume that the network entities who participate in the network activities will always behave as instructed. However, in practice, most network entities will try to maximize their own benefits instead of altruistically contribute to the network by following the prescribed protocols, which is known as selfish. Thus, new protocols should be designed for the non-cooperative network which is composed of selfish entities. In this paper, we specifically show how to design strategyproof multicast protocols for non-cooperative networks such that these selfish entities will follow the protocols out of their own interests. By assuming that a group of receivers is willing to pay to receive the multicast service, we specifically give a general framework to decide whether it is possible, and how if possible to transform an existing multicast protocol to a strategyproof multicast protocol. We then show how the payments to those relay entities are shared fairly among all receivers so that it encourages collaboration among receivers. As a running example, we show how to design the strategyproof multicast protocol for the currently used core-based multicast structure. We also conduct extensive simulations to study the relations between payment and cost of the multicast structure.

Index Terms—Control theory, combinatorics, economics, non-cooperative, multicast, payment, sharing.

I. INTRODUCTION

Multicast has received considerable attentions over the past few years due to its resource sharing capability. In multicast, there is a topology, either a tree or a mesh, that connects the source to a set of receivers, and packet is only duplicated at the branching nodes. Numerous multicast protocols have been proposed, and most of them assumed that the network entities, either links or nodes, will relay the multicast packets as prescribed by the multicast protocol without any deviation. However, the Internet, which is composed of different heterogeneous and autonomous systems (AS), raises a doubt about this common belief. Although multicast benefits the whole system by saving bandwidth and resource, it is dubious that multicast will also bring benefits to every individual node or link who relays the packet. Thus, it is more reasonable to assume that these ASs, probably owned by some organizations and private users, are selfish: aim to maximize their own benefits instead of faithfully conform the prescribed multicast protocol. A network composed of selfish ASs is generally known as a non-cooperative network.

WeiZhao Wang† Xiang-Yang Li† Zheng Sun‡ Yu Wang‡

Nisan and Ronen [1] studied the unicast routing problem in non-cooperative networks and introduced the idea of algorithmic mechanism design: they proposed to give the ASs some proper payments to ensure that every AS conforms to the prescribed protocol regardless of all other ASs’ behavior, which is known as strategy-proof or truthful. They designed the payment for unicast by using the VCG mechanism[2], [3], [4], which is considered as one of the most positive results in algorithm mechanism design. Unfortunately, VCG mechanism has its own drawback. For multicast, if we want to apply VCG mechanism, the multicast tree should have the least cost among all trees spanning the receivers. However, finding the minimum cost multicast tree is known to be NP-Hard for both edge weighted networks [14], [15] and node weighted networks [16], [17]. If we insist on applying the VCG mechanism to a multicast topology that does not have the minimal cost, VCG mechanism may fail [18]. Thus, some payment schemes other than VCG mechanism should be designed for multicast. Recently, in [18], the authors proposed several non-VCG strategy-proof payment schemes for several commonly used multicast trees. In this paper, instead of focusing on some specific multicast structures, we study whether it is possible to transform a multicast protocol based on any given multicast topology to a strategyproof multicast protocol, and if possible, how to design the strategyproof protocol.

Designing a truthful payment scheme is not the whole story for many practical applications. A natural question has to be answered is who will afford the payments. A simple solution is that the organization to which the receivers belong pays [18]. However, this solution is not panacea. In many applications such as video streaming, often the individual receivers have to pay the relay agents to receive the data. How to charge the receiver for multicast transmission has been studied extensively in literatures [19], [20], [21], [22], [23], [24]. In most of their models, they assumed that 1) every receiver has a valuation for receiving the data and the receiver is selfish, 2) all relay agents are cooperative and reveal their true cost, and 3) the multicast tree is formed by the union of the shortest paths from the source to receivers. In the sharp contrast, in this paper, we also take the selfish behavior of the relay nodes (or links) into account. Thus, we model the network differently by assuming that 1) the relay agents are selfish, 2) the receivers always receive the data and pay what they “should” pay, and 3) the multicast topology could be any structure specified by some existing multicast protocols, including trees and meshes. To the best of our knowledge, this is the first paper to consider multicast pricing when the relay agents are non-cooperative. Notice that there is a possible work...
left for future exploration: what happens if both the receivers and relay agents are selfish and each receiver has a valuation and would receive the data if and only if its valuation is greater than what it needs to pay according to a strategy-proof multicast protocol.

One thing we should point out is that algorithmic mechanism design is not the only way to achieve strategyproofness. There are lots of literatures which use Nash equilibrium, a state at which no agent can improve its utility by unilaterally deviating from its current strategy when other agents keep their strategies. Since Nash equilibrium has a weak requirement for the strategies used by the agents, it often can achieve a wider variety of outcomes.

The main contributions of this paper are two-folded. First, we present a general framework about whether it is possible, and how if possible, to transform an existing multicast protocol to a strategyproof one. We then show how the payments to the relay agents are shared fairly among the receivers. As a running example, we show how to design a strategyproof multicast protocol, and how the payments are shared among receivers when the least cost path tree is used for multicast. We also conduct extensive simulations to study the relations between payment and cost of the multicast structure. Our simulations show that by only overpaying a small amount to the relay nodes (or links), each relay node (or link) will declare its true cost to maximize its profit.

The rest of the paper is organized as follows. We introduce some preliminaries, related works, our communication model, and the problems to be solved in Section II. In Section III, we discuss the existence of truthful payment and how to find it if a given multicast structure is used. We show how to design a truthful multicast protocol based on a specific routing topology in Section IV. Several other important issues are discussed in Section V. The performance study of our proposed truthful core-based multicast protocol is presented in Section VI. We conclude our paper in Section VII.

II. TECHNICAL PRELIMINARIES

A. Algorithmic Mechanism Design

In a standard model of algorithm mechanism design, there are \( n \) agents \( \{1, 2, \ldots, n\} \). Each agent \( i \in \{1, \ldots, n\} \) has some private information \( t_i \), called its type, e.g. its cost to forward a packet in a network environment. All agents’ type defines a profile \( t = (t_1, t_2, \ldots, t_n) \). Each agent \( i \) declares a valid type \( \tau_i \) which may be different from its actual type \( t_i \) and all agents’ strategy defines a declared type vector \( \tau = (\tau_1, \ldots, \tau_n) \). A mechanism \( M = (O, P) \) is composed of two parts: an output function \( O \) that maps a declared type vector \( \tau \) to an output \( o \) and a payment function \( P \) that decides the monetary payment \( p_i = P_i(\tau) \) for every agent \( i \). Each agent \( i \) has a valuation function \( w_i(t_i, o) \) that expresses its preference over different outcomes. Agent \( i \)’s utility or called profit is \( u_i(t_i, o) = w_i(t_i, o) + p_i \). An agent \( i \) is said to be rational if it always chooses its strategy \( \tau_i \) to maximize its utility \( u_i \).

Let \( \tau_{-i} = (\tau_1, \ldots, \tau_{i-1}, \tau_{i+1}, \ldots, \tau_n) \), i.e., the strategies of all other agents except \( i \) and \( \tau^t_i = (\tau_1, \tau_2, \ldots, \tau_{i-1}, t_i, \tau_{i+1}, \ldots, \tau_n) \). A mechanism is strategy-proof if for every agent \( i \), revealing its true type \( t_i \) will maximize its utility regardless of what other agents do. In this paper, we are only interested in mechanisms \( M = (O, P) \) that satisfy the following three conditions:

1) Incentive Compatibility (IC): For every agent \( i \), \( w_i(t_i, O(\tau^t_i)) + p_i(\tau^t_i) \geq w_i(t_i, O(\tau)) + p_i(\tau) \quad \forall \tau \).
2) Individual Rationality (IR): It is also called Voluntary Participation. Every participating agent must have a non-negative utility, i.e., \( w_i(t_i, O(\tau^t_i)) + p_i(\tau^t_i) \geq 0 \).
3) Polynomial Time Computability (PC): \( O \) and \( P \) are computed in polynomial time.

VCG MECHANISM: Arguably the most important positive result in mechanism design is what is usually called the generalized Vickrey-Clarke-Groves (VCG) mechanism [2], [3], [4]. A direct revelation mechanism \( M = (O(t), \mathcal{P}(t)) \) belongs to the VCG family if (1) the output \( O(t) \) computed based on the type vector \( t \) maximizes the objective function \( g(o, t) = \sum w_i(t_i, o) \) and (2) the payment to agent \( i \) is \( \mathcal{P}_i(t) = \sum_{j \neq i} w_j(t_j, O(t)) + h_i(t_i - 1) \). Here \( h_i() \) is an arbitrary function of \( t_i - 1 \). It is proved in [4] that a VCG mechanism is truthful, i.e., satisfying the IC property. Under mild assumptions, VCG mechanisms are the only truthful implementations [5] for utilitarian problems, i.e., \( g(o, t) = \sum w_i(t_i, o) \).

B. Network Model and Problem Statement

Consider any communication network \( G = (V, E, c) \), where \( V = \{v_1, \ldots, v_n\} \) is the set of communication terminals, \( E = \{e_1, e_2, \ldots, e_m\} \) are the set of links. Every agent \( i \) in the network has a private cost \( c_i \) to transmit a unit size of data. Here agents could be either terminals or links whoever could behave selfishly. If agents are terminals then \( G \) is node weighted; if agents are links then \( G \) is link weighted. Given a set of terminals \( Q = \{q_1, q_2, \ldots, q_r\} \subset V \) who are willing to receive the data, we will design a multicast protocol that

1) constructs a topology (a tree, a mesh, a ring, etc) that spans these receivers;
2) calculates a payment for each relay agent according to a payment scheme that is strategy-proof;
3) charges each receiver according to a pricing scheme that is reasonable. We will formally define what is reasonable in subsection III-C.

For the convenience of our analysis, we assume that \( s = 0 \) is the source node in one specific multicast and the size of the data is normalized to 1. We also assume throughout this paper that agents in the network will not collude to improve their profits together. In order to prevent the monopoly, we assume the network is bi-connected.

One thing we should highlight here is that instead of reinventing the wheels by designing some new multicast structures, we focus on how we can design a truthful payment scheme for the existing multicast protocols to ensure that they work correctly even in non-cooperative networks. Based on the truthful payment scheme we designed, we further study how we charge the receivers in a reasonable way.

Given a network \( H \subseteq G \), we use \( \omega(H) \) to denote the sum of the costs 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 $c_i \cdot \infty$.  Denote $G \setminus e_i$ as the network without link $e_i$, and denote $G \setminus v_i$ as the network without node $v_i$ and all its incident links. For the simplicity of notations, we will use only the cost vector $c$ to denote the network $G = (V, E, c)$ if no confusion is caused.

C. Related Work

Routing has been part of the algorithmic mechanism-design from the very beginning. Nisan and Ronen [6] provided a polynomial-time strategyproof mechanism for unicast routing in a centralized computational model. Each link $e$ of the network is an agent and has a private cost $t^e$ of sending a message. Their mechanism is essentially a VCG mechanism. The result in [6] is extended in [25] to deal with unicast problem for all pairs of terminals. They assume there is a traffic demand $T_{i,j}$ from a node $i$ to a node $j$. They also gave a distributed method to compute the payment. Anderegg and Eidenbenz [26] recently proposed a similar routing protocol for wireless ad hoc networks based on VCG mechanism again. In [29], Wang and Li proposed an asymptotically optimum for wireless ad hoc networks based on VCG mechanism again. Eidenbenz [26] recently proposed a similar routing protocol a distributed method to compute the payment. Anderegg and Eidenbenz [26] recently proposed a similar routing protocol for wireless ad hoc networks based on VCG mechanism again.

Intuitively, we may still want to use the VCG payment schemes for these multicast topologies. Notice that an output function of a VCG mechanism is required to maximize the total valuations of agents. This makes the mechanism computationally intractable in many cases, e.g., multicast. Notice that replacing the optimal algorithm with non-optimal approximation usually leads to untruthful mechanisms [18]. Thus a mechanism other than VCG is needed when we cannot find the optimal solution or the objective is not to maximize the total valuations of all agents. This paper presents the first general framework to design strategyproof mechanism for multicast in which we cannot find the structure with minimum total cost.

A. Existence of the Truthful Payment Mechanism

Before we design some truthful payment scheme for a given multicast topology, we should decide whether such payment scheme exists or not. Following definition and theorem will present a sufficient and necessary condition for the existence of the truthful payment scheme.

Definition 1: A method $O$ computing a multicast topology satisfies the monotone property (MP) if for every agent $i$ and fixed $c_{-i}$, following condition is satisfied: If agent $i$ is selected as a relay agent with cost $c_{i_2}$, then it is also selected with a smaller cost $c_{i_1}$.

Obviously, the above condition is equivalent to the following condition: There exists a threshold value $\kappa_i(O, c_{-i})$ such that if $i$ is selected as a relay agent, then its cost is at most $\kappa_i(O, c_{-i})$. For the convenience of our presentation, we use $O_i(c) = 1$ (respectively 0) to denote that agent $i$ is selected (respectively not selected) to the multicast topology when the cost vector is $c$.

Theorem 1: Given a method $O$ computing a multicast topology, there exists a payment $P$ such that $M = (O, P)$ is strategyproof iff $O$ satisfies monotone property.

Proof: We first prove if there exists a truthful payment based on $O$ then $O$ satisfies the monotone property. We prove it by contradiction by assuming there is a truthful payment scheme $P$ based on $O$ that does not satisfy MP. From the definition of MP, there exists an agent $i$ and two cost vectors $c_i|c_{i_1}$ and $c_i|c_{i_2}$, where $c_{i_1} < c_{i_2}$ such that $O_i(c_i|c_{i_1}) = 1$ and $O_i(c_i|c_{i_2}) = 0$. Let $p_i(c_i|c_{i_1}) = p_i^0$ and $p_i(c_i|c_{i_2}) = p_i^1$.

Consider a network with a cost vector $c_i|c_{i_2}$, the utility for agent $i$ when it reveals its true cost is $u_i(c_{i_1}) = p_i^0$. When agent $i$ lies its cost to $c_{i_2}$, its utility becomes $p_i^1 - c_{i_1}$. Since payment scheme $P$ is truthful, we have $p_i^0 > p_i^1 - c_{i_1}$.

Similarly we consider another network with a cost vector $c_i|c_{i_2}$. Agent $i$’s utility is $p_i^1 - c_{i_2}$ when it reveals its true cost. Similarly, if it lies its cost to $c_{i_1}$, its utility is $p_i^1$. Since payment scheme $P$ is truthful, we have $p_i^0 > p_i^1 - c_{i_2}$.

Thus, we have $p_i^1 - c_{i_2} > p_i^0 > p_i^1 - c_{i_1}$. This inequality implies that $c_{i_1} > c_{i_2}$, which is a contradiction.

We then prove that if $O$ satisfies the monotone property then there exists a truthful payment based on $O$. We prove it by constructing the following payment scheme $P$ for a given a network $G = (V, E, c)$.

III. Characterization of Truthful Multicast Routing

Several multicast topologies have been proposed and used in practice and it is expected that more topologies will be proposed in the near future. It will be difficult if not impossible to design a strategyproof multicast mechanism for each of these topologies individually. Thus, instead of studying some specific multicast topologies, we present a general framework to decide whether there is, and how to design if it exists, a strategyproof mechanism for any given multicast topology. We also consider how to charge the receivers to cover the total payments to the relay agents.
Algorithm 1 Payment Scheme $\mathcal{P}$
1: For any agent $i$ not selected to relay, its payment is 0.
2: For any agent $i$ selected to relay, its payment is $\kappa_i(\mathcal{O}, c_{-i})$.

From the definition of MP, the IR property is obvious. Thus we only need to prove that the payment scheme $\mathcal{P}$ satisfies IR. We prove it by cases.

Case 1: Agent $i$ lies its cost upward to $\sigma_i$ or downward to $c_i$. but it does not change the output whether agent $i$ is selected or not. Notice for fixed $c_{-i}$, when the output of agent $i$ does not change, its payment is the same. Thus, agent $i$’s utility keeps the same which in turn implies that agent $i$ does not have incentive to lie in this case.

Case 2: Agent $i$ is selected when it reveals its actual cost $c_i$, and it lies its cost upward to $\sigma_i$ such that it is not selected. From the property of MP, we know $c_i \leq \kappa_i(\mathcal{O}, c_{-i})$. This ensures that agent $i$ gets non-negative utility when it reveals its actual cost $c_i$. When $i$ lies its cost to $\sigma_i$, it gets zero payment and zero utility. Therefore, agent $i$ won’t lie in this case.

Case 3: Agent $i$ is not selected when it reveals its actual cost $c_i$, and it lies its cost downward to $\sigma_i$ such that it is selected. Similarly, we have $c_i \geq \kappa_i(\mathcal{O}, c_{-i})$, which implies that agent $i$ gets a non-positive utility. Comparing with the zero utility when agent $i$ reveals its true cost, agent $i$ also has no incentive to lie in this case.

This finishes our proof.

Actually, if we require that relay agents who are not selected should receive zero payment, our payment scheme illustrated by Algorithm 1 is the only strategy-proof payment scheme. The proof is omitted here due to space limit.

B. Rules to Find the Truthful Payment Scheme

Given a multicast structure satisfying MP, it seems quite simple to find a truthful payment scheme by applying Algorithm 1. However, sometimes the process to find the threshold value in Algorithm 1 is far more complicated. As to our knowledge, our approach presented later is the first ever effort to find the threshold value efficiently. Instead of trying to give a unified approach that can find the threshold value for all multicast topologies satisfying MP, we present some useful techniques to find threshold value under certain circumstances.

Our general approach works as follows. First, given an output method $\mathcal{O}$ that computes a multicast structure, we decompose it into several simpler output methods. We then find the threshold value for each of the decomposed methods. Finally, we calculate the original threshold value by combining the threshold values for those decomposed methods.

1) Simple Combination: Given a multicast method $\mathcal{O}$, let $\kappa(\mathcal{O}, c)$ denote a $n$-tuple vector

$$
(\kappa_1(\mathcal{O}, c_{-1}), \kappa_2(\mathcal{O}, c_{-2}), \cdots, \kappa_n(\mathcal{O}, c_{-n})).
$$

Here, $\kappa_i(\mathcal{O}, c_{-i})$ is the threshold value for agent $i$ when the multicast topology is computed by $\mathcal{O}$ and the costs $c_{-i}$ of all other agents are fixed. We then present a simple but useful technique to find the threshold value.

Theorem 2: Given $n$ multicast methods $\mathcal{O}^1, \cdots, \mathcal{O}^n$ satisfying monotone property, and $\kappa(\mathcal{O}^i, c)$ is the threshold values for $\mathcal{O}^i$ where $1 \leq i \leq n$. Then the output method $\mathcal{O}(c) = \mathcal{O}^1(c) \cup \mathcal{O}^2(c) \cup \cdots \cup \mathcal{O}^n(c)$ also satisfies monotone property. Moreover, the threshold value for $\mathcal{O}$ is

$$
\kappa(\mathcal{O}, c) = \max_{1 \leq i \leq n} \{\kappa(\mathcal{O}^i, c)\}.
$$

The proof of this theorem is quite simple and is omitted here. We will show how to use this simple combination technique in Section IV.

2) Round-based Method: Many multicast topologies are constructed in a round-based manner: for each round they select some unselected agents, update the problem and the cost profile if necessary. Following is a general characterization of a round-based method that constructs a multicast topology.

Algorithm 2 A Round-Based Multicast Method
1: Set $r = 1$ and $c^{(1)} = c$ and $Q^{(1)} = Q$ initially.
2: repeat
3: Let $\mathcal{O}^r$ be a deterministic method that decides in round $r$ whether agent $i$ is selected or not.
4: Update the network cost vector and receiver set, i.e., we obtain a new network cost vector $c^{(r+1)}$ and receiver set $Q^{(r+1)}$ according to a update rule $U^r$:
5: \[ U^r : Q^r \times [c^r, Q^r] \rightarrow [c^{(r+1)}, Q^{(r+1)}]. \]
6: until the desired property of the multicast topology is met
7: Return the union of the relay agents in all rounds as the final output. Here, every agent can be selected at most once.

To help the understanding of general round-based method, we present a multicast topology that is constructed in such way. The example we used is the polynomial time method in [15] that finds a multicast topology whose cost is no more than 2 times of the minimum cost Steiner tree (MCST) in a link weight network. For the completeness of our presentation, we review their method here.

Algorithm 3 Link Weighted Multicast Structure [15]
1: repeat
2: Find one receiver in the receiver set $Q$, say $q_i$, that is closest to the source $s$, i.e., the LCP$(s, q_i, d)$ has the least cost among the shortest paths from $s$ to all receivers.
3: Connect $q_i$ to the source $s$ using the least cost path between them. Update the cost of all edges on this path as 0. Remove $q_i$ from the receiver set $Q$.
4: until no receiver remains

Here no receiver remains corresponds to the desired properties of general round-based method: LCP$(s, q_i, d)$ in round $r$ corresponds to $\mathcal{O}^r$; updating cost of edges on LCP$(s, q_i, d)$ to 0 and removing $q_i$ from $Q$ is the update rule $U^r$.

Figure 1 shows how to apply Algorithm 3. Initially, the receiver set is $Q = Q^1 = \{q_1, q_2\}$ and the link costs are shown in Figure 1. The first round selects the nearest receiver $q_2$ from
s, and its corresponding path $sv_3q_2$ is selected. Remove $q_1$ from $Q^1$ and set cost of link $sv_3$ and $v_3q_2$ to 0. The network in the end of first round is shown in Figure 1 (b). In the second round, the receiver set is $Q^2 = \{q_1\}$, and the least cost path from $s$ to $q_1$ is $sv_3v_4v_5q_1$ instead of the least cost path $sq_1$ in original network. The final multicast tree, shown as solid lines in Figure 1, is the union of the two paths.

**Fig. 1. Illustration of Algorithm (3).** Here, $s$ is the source node.

**Definition 2:** An updating rule $U^r$ is said to be crossing-independent if for any unselected agent $i$:
- $c_{i-1}^{(r+1)}$ and $Q^{(r+1)}$ do not depend on $c_i^{(r)}$.
- Fixed $c_i^{(r)}$, if $d_i^{(r)} \leq c_i^{(r)}$ then $d_i^{(r+1)} \leq c_i^{(r+1)}$.

**Theorem 3:** A round-based multicast method $O$ satisfies MP if, for every round $r$, method $O^r$ satisfies MP and the updating function $U^r$ is crossing-independent.

**Proof:** For an agent $i$, fix the original cost $c_{i-1}$ of all other agents. We prove that if $i$ is selected when the original cost vector is $a = \{c_{i-1}, c_i\}$, then it is also selected when the original cost vector is $b = \{c_{i-1}, c_i'\}$ such that $c'_i < c_i$. Without loss of generality assume that $i$ is selected in round $r$ under cost vector $a$. Then under cost vector $b$, if agent $i$ is selected before round $r$, our claim holds. Otherwise, in round $r$, $a_{i-1} = b_{i-1}^{(r)}$ and $a_i^{(r)} > b_i^{(r)}$ since agent $i$ is not selected in the previous rounds. Notice $i$ is selected in round $r$ under cost vector $a_i^{(r)}$, thus $i$ is also selected in round $r$ under cost vector $b_i^{(r)}$ from the monotone property of the method $O^r$. This finishes the proof.

Theorem 3 presents a sufficient condition for the existence of truthful payment scheme for a round-based multicast method. Following, we show how to find the threshold value for any selected agent $k$.

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

**Algorithm 4 Computing payment for selected agent $k$ based on round-based multicast method $O$**

1: Initially set the cost of $k$ to $\infty$ and $r = 1$
2: repeat
3: Find the threshold value for agent $k$ based on $O^r$ under cost vector $c_{-k}^{(r)}$ and receiver set $Q^{(r)}$. Let $\ell_r = s_{\kappa_k}(O^r, c_{-k}^{(r)})$ be the threshold value found. Here we set $\ell_r = 0$ if agent $k$ cannot be selected in this round for any cost.
4: Update cost vector and receiver set to obtain the new cost vector $c_{-k}^{(r+1)}$ and $Q^{(r+1)}$. Set $r = r + 1$.
5: until a valid output is found
6: Fix $c_{-k}$ and assume $x$ is the payment for agent $k$. Let $x'$ be the cost for agent $k$ in round $r$ if the original cost is $c_k^r x$. Then $x$ the minimum value that satisfies the following inequations: $x' \geq \ell_i$ for $1 \leq i \leq r$.

**C. Reasonable Charging Scheme**

For a given set of receivers, after we calculate the payment $p_k(d)$ for every relay agent $k$ based on a declared cost vector $d$, it is natural to ask who will pay these payments. Two possible payment models have been proposed in the literature.

1) **Outside bank or Group payment model:** an outside bank or an organization to which the receivers belong will pay all these relay agents.
2) **Payment sharing model:** each receiver $i$ should pay a reasonable sharing $S_i$ of the total payment. We will address what reasonable means later.

For outside bank model, the only thing we should care is how to find the truthful payment scheme for the given multicast topology, which has been addressed in the previous subsection. In practice, it is often the case that the receivers have to share the payments among themselves. Thus, we will study how to share the payments fairly. Notice that the payment sharing is different from the traditional cost sharing. How to share the multicast cost among the receivers has been studied previously in [27], [20], [23], [19], in which the cost of relay agents are public and the multicast topology is a fixed tree. Most of the literatures used the Equal Link Split Downstream (ELSD) pricing scheme to charge receivers: the cost of a link is shared equally among all its downstream receivers. As we will show later, if we simply use the ELSD as our charging scheme to share the payment, it usually is not reasonable in common sense.

We use the same network in Figure 1 to illustrate how to find the threshold value for edge $v_3v_4$ based on the multicast tree found by Algorithm 3. In the first round, $v_3v_4$ can not be selected, thus $\ell_1 = 0$. In second round, it is easy to observe that when $v_3v_4$’s cost is smaller than 0.9, the path $v_3v_4v_5q_1$ is selected and when $v_3v_4$’s cost is greater than 0.9, path $sq_1$ is selected. Thus, the threshold value for $v_3v_4$ in this round is $\ell_2 = 0.9$. Notice the updating by Algorithm 3 does not change the cost of an unselected agent, thus the final threshold value is simply the maximum of $\ell_1$ and $\ell_2$, which is 0.9. In other words, we have to pay link $v_3v_4$ 0.9. Similarly, we can find all selected edge’s threshold value as shown in Figure 2 (b): the numbers in the parenthesis are the threshold values.
Given a set of receivers $R$, let $P(R, d) = \sum_{k} p_k(R, d)$ denote the total payments to all relay agents. For a charging scheme $S$, let $S_i(R, d)$ denote the charge (or called sharing) to receiver $i$. We call a charging scheme $S$ reasonable or fair if it satisfies the following criteria.

1) **Nonnegative Sharing (NNS):** Any receiver $q_i$'s sharing should not be negative. In other words, we don’t pay the receiver to receive.

2) **Cross-Monotone (CM):** For any two receiver sets $R_1 \subseteq R_2$ containing $q_i$: $S_i(R_1, d) \leq S_i(R_2, d)$. In other words, for a given network, receiver $i$’s sharing does not increase when more receivers require service.

3) **No-Free-Rider (NFR):** The sharing $S_i(R, d)$ of a receiver $q_i \in R$ is never less than $\frac{1}{\pi d}$ of its unique sharing $S_i(q_i, d)$. This guarantees that the sharing of any receiver will not be too small.

4) **Budget Balance (BB):** The payment to all relay agents should be shared by all receivers, i.e., $P(R, d) = \sum_{q_i \in R} S_i(R, d)$.

Notice the definition of reasonable can be changed due to different requirements. For example, a common criterion for multicast charging scheme is to maximize network welfare: select a subset of receivers such that the network welfare is maximized. Here, the network welfare is defined as the total valuations of all selected receivers minus the cost of the network providing service. Since in our model we do not consider receiver’s valuation, we will only focus on budget balance instead of maximizing the network welfare.

In literature, the Shapely value [28] is one of the most commonly used charging schemes to achieve BB and CM. By assuming a universal multicast tree and the publicly known link costs, Feigenbaum et al. [23] proved that ELSD charging scheme is a Shapely Value. Unfortunately, the ELSD charging scheme is not always fair if we want to share the payment.

**Lemma 4:** For tree LST, ELSD sharing is not fair.

**Proof:** As a running example, we will use the multicast tree, denoted by LST, found by Algorithm 3 to show that ELSD is not fair. We still use the same network shown in Figure 1 (a). Let $Q = q_1, q_2$ be receivers. The multicast tree $LST(Q)$ is shown in Figure 1 (c). Tree $LST(q_1)$ and $LST(q_2)$ are shown in Figure 3 (a) and (b) respectively.

![Fig. 3. LST(q1) and LST(q2) and their corresponding payment(3).](image)

We now show that ELSD is not fair in this situation. Figure 3 (a) and (b) illustrate the payment $P_1(q_1) = 2.5$ and $P_2(q_2) = 2.9$. If we use ELSD as our charging scheme, the sharing by $q_1$ is $S_1(q_1 \cup q_2, c) = \frac{1.6}{2} + 0.9 + 1.1 + 1.5 = 4.2$ which is larger than its sharing $S_1(q_1, c) = 2.6$ when $q_1$ is the only receiver. Thus, it violates the property CM. It implies that ELSD is not a fair charging scheme for multicast topology LST.

Furthermore, using the same example, we show by contradiction that there is no charging scheme satisfying both CM and BB.

**Lemma 5:** For multicast topology LST, there is no charging scheme that satisfies both CM and BB for a truthful payment scheme.

**Proof:** For the sake of contradiction, we assume that a charging scheme $S'$ satisfies both CM and BB. From the property of BB, we have $S'_1(q_1, c) = 2.6, S'_1(q_2, c) = 2.9$ and $S'_1(q_1 \cup q_2, c) + S'_2(q_1 \cup q_2, c) = 6.4$. From CM, we have $S'_1(q_1 \cup q_2, c) \leq S'_1(q_1, c) = 2.6$ and $S'_2(q_1 \cup q_2, c) \leq S'_2(q_2, c) = 2.9$. Combining these two inequalities, we obtain $6.4 = S'_1(q_1 \cup q_2, c) + S'_2(q_1 \cap q_2, c) \leq 2.9 + 2.6 = 5.5$, which is a contradiction.

Thus, given an arbitrary multicast topology and its corresponding truthful payment scheme, a fair charging scheme may not exist at all. It is attractive and important to find the necessary and sufficient condition for the existence of a fair charging scheme for a given multicast topology.

**IV. CASE STUDY: CORE-BASED MULTICAST**

In this section, we illustrate how to design a truthful multicast protocol for the currently used core-based multicast which uses the least cost path tree (LCPT) as its topology. Here, we assume that the network is modelled as a link weighted graph. All our results presented in this section also apply to the case when the network is modelled as a node weighted graph.

Given a set of receiver $R$, we first compute the least cost path, denoted by $LCP(s, q_i, d)$, between the source $s$ and every receiver $q_i \in Q$ under the reported cost profile $d$. The union of all least cost paths between the source and the receivers is called least cost path tree, denoted by $LCP(R, d)$.

**A. Payment Scheme**

Intuitively, we may use the VCG payment scheme in conjunction with the LCPT tree structure as follows. The payment $p_k(d)$ to each link $e_k$ that is not in LCPT is 0 and the payment to each link $e_k$ on LCPT is $p_k(d) = \omega(LCP(R, d)) - \omega(LCP(R, d)) + d_k$. In other words, the payment is its declared cost plus the difference between the cost of the least cost path tree without using $e_k$ and the cost of the least cost path tree.

We show by example that the above payment scheme is not strategyproof. In other words, if we simply apply VCG scheme on LCPT, a link may have incentives to lie about its cost.

The payment to link $e_4 = sv_3$ is 0 and its utility is also 0 if it reports its cost truthfully. The total payment to link $e_4$ when $e_4$ lies its cost down to 4 is $\omega(LCP(R, e_4)) - \omega(LCP(R, d)) + d_k$. In other words, the payment is its declared cost plus the difference between the cost of the least cost path tree without using $e_k$ and the cost of the least cost path tree.
the truthful payment can be found according to Theorem 2 as follows.

\[ \omega(\text{LCPT}(R, c|4d_4)) + d_4 = 20 - 8 + 4 = 16 \text{ and the utility of link } sv_4 \text{ becomes } u_4(c|4d_4) = 16 - 8 = 8, \text{ which is larger than } u_4(c) = 0. \text{ Thus link } e_4 \text{ has incentive to lie, which implies that VCG mechanism is not truthful.}

With the failure of the VCG mechanism, we may doubt whether there exists a truthful payment scheme based on LCPT. Remember LCPT is formed by union of least cost paths. By applying Theorem 2, we conclude that LCPT satisfies MP. Thus, there exists a truthful payment scheme and the truthful payment can be found according to Theorem 2 as following.

For each receiver \( q_i \in R \), we find the least cost path from the source \( s \) to \( q_i \) and compute an intermediate payment \( p_{q_i}^k(d) \) to link \( q_i \) on \( \text{LCPT}(s, q_i, d) \) using the VCG payment scheme for unicast

\[ p_{q_i}^k(d) = d_k + |\text{LCPT}(s, q_i, d)| - |\text{LCPT}(s, q_i, d)|. \]

Here \(|\text{LCPT}(s, q_i, d)|\) denotes the total cost of the least cost path \( \text{LCPT}(s, q_i, d) \). The final payment to link \( e_k \in \text{LCPT} \) is

\[ p_k(d) = \max_{q_i \in Q} p_{q_i}^k(d) \quad (1) \]

The payment to a link is zero if it is not on LCPT.

Let us illustrate the above payment scheme for LCPT by a running example in Figure 4. If link \( sv_4 \) reports its cost 8 truthfully, then it gets payment 0 since it is not in the LCPT. If link \( sv_4 \) reports a cost 4, it is now in the LCPT, as shown in Figure 3 (c). Its payment then becomes \( \max(p_{sv_4}^1, p_{sv_4}^2, p_{sv_4}^3) \), where \( p_{sv_4}^1 = |\text{LCPT}(s, q_1, d)| - |\text{LCPT}(s, q_1, d)| + 4 = 7 - 5 + 4 = 6 \). Similarly, \( p_{sv_4}^2 = 6 \) and \( p_{sv_4}^3 = 7 \). Then the utility of link \( sv_4 \) becomes \( \max(p_{sv_4}^1, p_{sv_4}^2, p_{sv_4}^3) - 8 = 7 - 8 = -1 \), which is less than what it gets by reporting its truth cost.

**B. Distributed Payment Algorithm**

Remember that LCPT is based on the union of the least cost paths from the source to all receivers. For unicast, Feigenbaum et al. [25] gave a distributed method such that each node \( i \) can compute a number \( p_{ij}^k > 0 \), which is the payment to node \( k \) for carrying the transit traffic from node \( i \) to node \( j \) if node \( k \) is on \( \text{LCP}(i, j, d) \), where \( d \) is the maximum of diameters of graph \( G \) removing a node \( k \), over all \( k \). We then briefly discuss how to compute the payment for multicast using LCPT. Our distributed algorithm uses the algorithm in [25] as the first phase and is shown as follows.

**Algorithm 5 Distributed payment computing**

1. Apply the distributed algorithm in [25] to compute the payment \( p_{si}^k \). After this step, every receiver \( q_i \) will compute the payment \( p_{qi}^k \) to each upstream edge \( e_k \) on the least cost path between \( s \) and \( q_i \).
2. Every receiver \( q_i \) sends the payment information it computed to its parent.
3. Upon receiving a packet containing the payment from its child which originated from receiver \( q_i \), link \( e_k \) only keeps payment \( p_{qi}^k \) and sends all remaining payment information to its parent if exists.
4. When link \( e_k \) receives \( p_{qi}^k \) from all its downstream receivers \( q_i \), it computes the maximum of them as its own final payment.

**C. Payment Sharing**

Intuitively, we may want to use ELSD as the charging scheme. Unfortunately, we will show by example that ELSD is not fair when coupled with LCPT. Consider the network shown by Figure 5 (a). There are two receivers \( q_1, q_2 \). Path \( \text{LCPT}(q_1, d) \) is shown in Figure 5 (b) and the payment to links is shown beside the link cost in parenthesis. The total payment to links on \( \text{LCPT}(q_1, d) \) is \( 2 + 2 = 4 \). If we consider \( \text{LCPT}(q_1 \cup q_2, d) \), the payment to links is shown in Figure 5 (c). If we apply ELSD to share payment, the payment to link \( sv_4 \) (which is 6) is split equally between \( q_1 \) and \( q_2 \). Thus, the shared payment of receiver \( q_1 \) is \( 3 + 2 = 5 \) when the receiver set is \( \{q_1, q_2\} \), while its payment is only 4 when \( q_1 \) is the only receiver. Thus, ELSD sharing method violates the CM property here, i.e., ELSD is not a fair charging scheme for LCPT. Therefore we should find some reasonable charging scheme other than ELSD. In this paper, we give one fair payment sharing method. The basic idea behind our method is that a receiver should only pay a proportion of the payment that is due to its existence.

Roughly speaking, our payment sharing scheme works as follows. Notice that a final payment to an agent \( j \) is the maximum of payments \( p_{ij}^k \) by all receivers. Since different receivers may have different value of payment to agent \( j \), the final payment \( P_j \) should be shared proportionally to their values, not equally among them as cost-sharing. Figure IV-C illustrates the payment sharing scheme that follows. Without loss of generality, assume that \( 0 \leq p_{1j} \leq p_{2j} \leq \cdots \leq p_{nj} \), i.e., \( p_j = p_{nj} \). We then divide the payment \( p_j \) into \( n \) portions: \( p_{j1}, p_{j2} - p_{j1}, \cdots, p_{j2} - p_{j1}, \cdots, p_{jn} - p_{j1} \). Each portion \( p_{j1} - p_{j1}^{n-1} \)...

![Image](image_url)
The receiver scheme satisfies NNS since

Fair charging scheme for LCPT.

is then equally shared among the last \( n - i + 1 \) elements, which have the largest \( n - i + 1 \) payments to \( S_j \).

Algorithm 6 Fair charging scheme for LCPT.

1. for edge \( e_k \in LCPT(R,d) \) do
2. Let \( R(e_k) \) be the set of downstream receivers of \( e_k \), i.e., \( p_k(d) = \max_{x \in R(e_k)} p_k^r(x) = \max_{x \in R} p_k^r(x) \).
3. Sort the receivers in \( R(e_k) \) according to \( p_k^r(d) \) in an ascending order. If two or more receivers have the same value, the receiver with smaller ID ranks first. Let \( \sigma = \{ \sigma_0, \sigma_1, \ldots, \sigma_{|R(e_k)|} \} \) be the ranking. Here, we add a dummy payment \( p_k^r(d) = 0 \) to ranking \( \sigma \).
4. For receivers not in \( R(e_k) \), its sharing of the payment \( p_k(d) \) of link \( e_k \) is 0.
5. For a receiver \( q_{\sigma_a} \in R(e_k) \), its sharing of the payment \( p_k(d) \) to link \( e_k \) is:

\[
S(a, R, d) = \frac{\sum_{k=1}^{n} p_k^r(d) - p_k^{\sigma_a-1}(d)}{|R(e_k)| - x + 1} \quad (2)
\]

In other word, for two receivers \( q_{\sigma_a}, q_{\sigma_a+1} \) who are consecutive in ranking \( \sigma \), the difference \( p_{k}^{\sigma_a+1}(d) - p_{k}^{\sigma_a}(d) \) is shared by all receivers who rank after \( q_{\sigma_a+1} \).
6. end for
7. The total charge for receiver \( q_i \) in LCPT is:

\[
S_i(R, d) = \sum_{e_j \in LCPT(R,d)} f_j^i(R, d) \quad (3)
\]

We first illustrate how to charge the receiver \( q_1 \) using Algorithm 6 for a network represented by Figure 5. For link \( SV_4 \), the two intermediate payments are \( p_{SV_4}^1 = 2 \) and \( p_{SV_4}^2 = 6 \). First, we obtain a rank of these receivers based on the intermediate payment \( \{q_1, q_2\} \). Then \( p_{SV_4}^1 = 2 \) is equally split between \( q_1 \) and \( q_2 \) and \( p_{SV_4}^2 - p_{SV_4}^1 = 4 \) is charged to \( q_2 \) alone. Thus, receiver \( q_1 \) is charged \( 2+1=3 \) totally in \( LCPT(q_1 \cup q_2, d) \), which is smaller than the price 4 when \( q_1 \) is the only receiver. This shows that charging scheme described by Algorithm 6 is reasonable for this specific network. Following theorem shows that it is reasonable for LCPT generally.

Theorem 6: The charging scheme defined in Algorithm 6 for LCPT satisfies NNS, CM, NFR and BB.

Proof: A link is called an upstream link of a receiver \( q_0 \) if it is on the unique simple path between the source and the receiver \( q_0 \) in the multicast. Obviously, our charging scheme satisfies NNS since \( p_k^r(x) - p_k^{\sigma_a-1}(d) \geq 0 \) for any two receivers \( q_{\sigma_a} \) and \( q_{\sigma_a+1} \). Remember for a receiver \( q_{\sigma_a} \in R(e_k) \), its sharing of the payment to its upstream link \( e_k \) is:

\[
S_{\sigma_a}(R, d) = \sum_{k=1}^{n} f_k^i(R, d) = \sum_{a=1}^{\sigma_a} p_k^r(d) - p_k^{\sigma_a-1}(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|}
\]

Thus, the total charge to receiver \( q_{\sigma_a} \) is:

\[
S_{\sigma_a}(R, d) = \sum_{k=1}^{n} f_k^i(R, d) = \sum_{a=1}^{\sigma_a} p_k^r(d) - p_k^{\sigma_a-1}(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|}
\]

It implies that the charging scheme 6 satisfies NFR.

Summing \( f_k^i(R, d) \) for all \( a \) from 1 to \( |R(e_k)| \), we obtain:

\[
\sum_{a=1}^{\sigma_a} f_k^i(R, d) = \sum_{a=1}^{\sigma_a} p_k^r(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|} \geq \sum_{a=1}^{\sigma_a} p_k^r(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|} = \sum_{a=1}^{\sigma_a} p_k^r(d) - p_k^{\sigma_a-1}(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|}
\]

Thus, we obtain:

\[
S(R, d) = \sum_{q_i \in R} S_i(R, d) = \sum_{q_i \in R} \sum_{e_j \in LCPT(R,d)} f_j^i(R, d) = \sum_{e_j \in LCPT(R,d)} f_j^i(R, d) = \sum_{e_j \in LCPT(R,d)} p_j(d) = \mathcal{P}(R, d)
\]

This proves that our charging scheme (6) satisfies BB.

We then show that our scheme does satisfy CM. Notice a necessary and sufficient condition for CM is that for any \( R \subset Q \) and \( q_j \in Q - R \) we have \( S_i(R,e_j) \geq \mathcal{S}(R \cup q_j, d) \) for every \( q_i \in R \). To prove this, it is sufficient to prove that \( f_k^i(R) \geq f_k^i(R \cup q_j) \). Assume \( q_i \) is ranked a in ranking \( \sigma \) when the receiver set is \( R \). We prove it by discussing all possible cases:

Case 1: \( p_k^i(d) \geq p_k^j(d) \). Let \( \sigma' \) be the new ranking for receiver set \( R \cup q_j \), then \( q_j \) ranked after \( q_i \) in \( \sigma' \). Thus \( \sigma_x = \sigma_x' \) for \( 1 \leq x \leq a \). In other words, all receivers ranked before or at \( a \) in ranking \( \sigma \) still has the same rank in \( \sigma' \). Therefore:

\[
S_i(R \cup q_j) = \sum_{x=1}^{a} p_k^i(d) - p_k^x(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|}
\]

\[
= \sum_{x=1}^{a} p_k^i(d) - p_k^{x-1}(d) \frac{|R(e_k)| - x + 2}{|R(e_k)|}
\]

\[
\leq \sum_{x=1}^{a} p_k^i(d) - p_k^{x-1}(d) \frac{|R(e_k)| - x + 1}{|R(e_k)|}
\]

Case 2: \( p_k^i(d) < p_k^j(d) \). In this case, \( q_j \) is ranked before \( q_i \) in \( \sigma' \) and \( q_j \) ranked \( a + 1 \) in \( \sigma' \). Without loss of generality, we assume \( q_j \) ranked \( b \) in ranking \( \sigma' \). Thus, we have \( \sigma_x = \sigma_x' \) for
$x < b$ and $\sigma_x = \sigma'_{x+1}$ for $x > b$. Therefore,

$$f'_k(R \cup q_i) = \sum_{x=1}^{a+1} \frac{p'_{k,x}(d)}{|R(e_k)| + 1 - x + 1}$$

$$= \frac{b}{x+1} \sum_{x=1}^{b} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2} + \sum_{x=b+1}^{a+1} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2}$$

For the first part of the equality we have

$$\frac{b}{x+1} \sum_{x=1}^{b} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2}$$

$$= \frac{b}{x+1} \sum_{x=1}^{b} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2} + \sum_{x=b+1}^{a+1} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2}$$

For the second part of the equality we have

$$\sum_{x=b+1}^{a+1} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)|}$$

$$= \sum_{x=b+1}^{a+1} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2}$$

Combining the above two, we obtain

$$f'_k(R \cup q_i)$$

$$= \frac{b}{x+1} \sum_{x=1}^{b} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2} + \sum_{x=b+1}^{a+1} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2}$$

$$\leq \frac{b}{x+1} \sum_{x=1}^{b} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2} + \sum_{x=b+1}^{a+1} \frac{p'_{k,x}(d) - p'_{k,x-1}(d)}{|R(e_k)| - x + 2}$$

This immediately implies that our charging scheme satisfies CM. This finishes the proof of Theorem 6.

D. Distributed Charge Calculation

Notice, if we implement the payment sharing scheme in a centralized way, for every link, it needs to store up to $|Q| = r$ intermediate payments. Thus the total space needed is $O(nr)$. In practice, it may be more desirable to implement a distributed payment sharing scheme. In the following, we present a distributed algorithm that implements our payment sharing scheme that requires at most $O(r)$ space for each link and total messages at most $O(r \cdot h)$, where $h$ is the height of the tree.

In our distributed algorithm, for any link $e_k$ in $LCPT(d)$, we not only need its final payment $p_k(d)$, but also need the intermediate payment $p'_k(d)$ for every downstream receiver $q_j$. We assume that this is already available through our distributed payment computing method. In our distributed charge scheme, at every link $e_k$ we use $MD_k[i]$ to store the payment it and all its upstream agents will receive from the receiver $q_i$. Our distributed charging scheme is implemented in a top-down fashion from the source to all receivers.

Algorithm 7 Distributed charging scheme

1: Initially, the source node $s$ sends all its children in the multicast tree a vector $MD = 0$ for all receivers.
2: Every link $e_k$ in $LCPT(d)$, upon receiving a charging vector $MD$ from its parent, updates the charge for each of its downstream receivers $q_i$ as $MD_k[i] = MD[i] + f'_k(R(e_k))$. Here, $f'_k(R(e_k))$ is calculated according to Algorithm 6.
3: If link $e_k$ has more than one downstream receivers, it constructs a new charge vector

$$MD_j = \{MD[i_1], MD[i_2], \ldots, MD[i_R(e_j)]\}$$

for every downstream adjacent link $e_j$. Here, the charge $MD[i_t]$ ($1 \leq t \leq |R(e_j)|$) is for receiver $q_{i_t}$ who is a downstream receiver of link $e_j$. Then send vector $MD_j$ to link $e_j$.

If link $e_k$ has only one downstream receiver $q_i$ then $e_k$ simply sends the modified charge $MD_k$ to its downstream link.
4: Every receiver $q_i$ will finally receive a charge which is equal to equation (3).

V. OTHER ISSUES AND OPEN QUESTIONS

As we mentioned early, this paper is the first step to explore the general network protocol design when relay agents are non-cooperative. There are many interesting and important issues that have been untouched and left for further study. We just list a few here.

**Collusion:** Throughout this paper, we assume all agents will not collude together to manipulate the protocol. It is interesting to study what will happen when agents will collude and how to find truthful mechanisms that are resistant to collusion. Our conjecture is that no truthful multicast protocol that can prevent the collusion from an initial work proved in [29] for unicast.

**Distributed Computing:** One thing we should notice is that these agents running the distributed algorithms are indeed non-cooperative. How to ensure they implement the correct distributed algorithm we designed also is an important question we have to consider.

**Receiver Valuation:** So far, we assume that the receivers will pay the fair amount of sharing of payment to receive data using multicast. In practice, each receiver often has a valuation to indicate how much it is willing to pay to receive the information. Receiver will choose to receive the information if and only if the charge is at most its valuation. Furthermore, receivers could also be non-cooperative and selfish: each receiver will always maximize its profit by manipulating its
reported valuation. This makes the multicast design even harder and it is a very promising and interesting future research direction. It is well-known that a cross-monotone cost sharing scheme implies a group-strategyproof mechanism [27]. Unfortunately, we can show that the simple application of a cross-monotone payment-sharing mechanism does not imply a group-strategyproof mechanism at all. The selfish relay agents could lie up or downward its cost to improve its utility.

VI. PERFORMANCE STUDY

We conduct extensive simulations to study the performance of strategy-proof multicast routing based on LCPT. Remember that the payment of LCPT is at least the actual cost of LCPT. For a LCPT $T$, let $c(T)$ be its cost and $P(T)$ be the total payment to all relay agents. We define the overpayment ratio (OR) of $T$ as

$$OR(T) = \frac{P(T)}{c(T)}. \quad (4)$$

In the worst case, the ratio $OR(T)$ could be as large as $O(n)$ for a network of $n$ nodes [30], even for the unicast special case. Notice there are some other definitions about overpayment ratio in the literature. In [30], the authors proposed to compare the total payment $P(T)$ with the cost of the new LCPT obtained from the graph $G \setminus T$, i.e., removing $T$ from the original graph $G$.

In addition to the overpayment ratio, we propose another metric to measure the performance of the strategy-proof multicast based on LCPT. Remember that the payments to relay agents are shared among receivers. Thus, for each receiver, it is more interested in how much extra it should pay to guarantee the truthfulness of the links. Given the LCPT $T$ for a set of receivers $R$, let $m_i(R, T)$ be the price that receiver $q_i$ is charged to receive the information if the links are cooperative. Notice that $S_i(R, T)$ is the amount that receiver $q_i$ is charged to receive the data if the links are non-cooperative. We define the Price-Cost-Ratio (PCR) as

$$PCR(q_i, T) = \frac{S_i(R, T)}{m_i(R, T)}. \quad (5)$$

In our experiment, we generate random networks with $n$ nodes, where $n$ is a parameter. In order to ensure the network is bi-connected, the average node degree should be greater than $\log n$ with high probability. First, for every node $u$, we randomly draw a number from $[\alpha \log n, 5\alpha \log n]$ as its degree $d_u$, where $\alpha \geq 1$ is a parameter. A random graph satisfying this degree requirement is then generated. The length of each edge is then uniformly drawn from distribution $[20, 100]$. By choosing different parameters, we study what aspects of the network affect the OR and PCR. To compute the probability distribution, we generate $10^4$ different networks and compute the number of instances that fall in some specific intervals. For other simulations, given all fixed parameters, we generate $10^3$ different network instances and computes the performances accordingly.

A. Effect of Network Size

In this simulation, we fix the parameter $\alpha$ to $\frac{10}{3 \log n}$, which means that node’ degrees are drawn from a uniform distribution $[\frac{10}{3}, \frac{50}{3}]$ with average 20. We also fix the size of receiver set $R$ to 15. We measure the performances of our strategy-proof multicast protocol based on the following four metrics: Average Overpayment Ratio (AOR), Maximum Overpayment Ratio (MOR), Average Price-Cost-Ratio (APCR) and Maximum Price-Cost-Ratio (MPCR). Figure 7 (a) and (b) plot the distribution of the average overpayment ratio and the average PCR when the number of nodes are 100 and 250. Observe that the probability distributions of AOR (also APCR) for different network size are similar. Figure 7 (c) shows that the AOR, MOR and APCR do not change when the number of network nodes grows from 100 to 500. On the other hand, MPCR fluctuates and is much larger than the other three metrics. Thus, we conclude that the number of nodes do not affect the overpayment ratio and price-cost-ratio in random network.

B. Effect of Network Density

Since the difference in the network size do not affect the performances of our strategy-proof protocol, we then study other effects by fixing the network size (100 in the results reported here). We specifically study the effect of the network density by changing the node degree parameter $\alpha$. Figure 8 (a) and (b) show the distributions of AOR and APCR respectively when the node degrees are drawn from two uniform distributions $[\log 100, 5 \log 100]$ and $[2 \log 100, 10 \log 100]$. Figure 8 (c) shows that the AOR, MOR and APCR change when the network density changes. It is interesting to observe that both AOR and APCR first decrease when the network density (i.e., the average node degree) increases from 10 to 32, and then increase slightly when the network density increases from 30 to 42. They both become steady when the network density is greater than 42. It is interesting to analyze this phenomenon theoretically.

C. Performance Comparison with Unicast

In this simulation, we compare the average cost and payment per receiver in multicast based on LCPT with those of unicast. We randomly generate $n$ terminals where $n$ varies from 100 to 500. The degree of each node is randomly drawn from the uniform distribution $[\log n, 5 \log n]$. For a specific network, we average the cost and payment for all receivers.

Figure 9 (a) plots the cost and payment for multicast and unicast per receiver when the number of receiver is 15, while Figure 9 (b) shows the results when 10% of nodes are receivers. Observe that the average cost and payment per receiver for multicast based on LCPT is smaller than the average cost and payment per receiver for unicast respectively. Furthermore, under most of the cases, the payment per receiver for LCPT payment is even smaller than the cost per receiver for unicast. This ensures us that multicast not only saves the total resources, but also benefits the individual receiver even in selfish networks. We then vary the network size among 100, 200, 300, 400, 500 and the number of receivers from 1 to
VII. CONCLUSION AND FUTURE WORKS

In this paper we give a strategyproof payment and charging mechanism that stimulates cooperation for multicast in a selfish network. We assumed that a group of receivers is willing to pay to receive the data. Each possible relay agent has a privately known cost of providing the relay service. In a multicast scheme, each selfish relay agent $k$ first is asked to declare a cost for relaying data for other nodes. In return, it will get a payment based on the reported costs of all relay agents that can provide the service. The objective of every individual relay agent is then to maximize its profit. A multicast protocol is said to be strategyproof if no speculation and counter speculation happens, i.e., every relay agent will maximize its profit when it truthfully reports its cost.

It is well-known that the traditional protocols designed for conforming agents cannot prevent the selfish agents from manipulating its cost to its benefit. Instead of redesigning the wheels, it is preferred to enhance an existing multicast protocol to deal with selfish agents. In this paper, we specifically gave a general rule to decide whether it is possible, and how to if

30. Figure 9 (c) shows the unicast cost (the red surface) and the LCPT based multicast payment (the blue surface).

From the results of previous three simulations, we observe that AOR and APCR are both quite small for a random network, and even the MOR is smaller than 1.7 generally. Thus, we conclude that the theoretical worst case almost surely will not happen in a random network.
possible transform an existing multicast protocol to a strategyproof multicast protocol. We then showed how the payments to all the relay agents are shared fairly among all receivers so that it encourages collaboration among receivers. As a running example, we showed how to design a strategyproof multicast protocol when the least cost path tree is used for multicast. We also discussed in detail how to implement this scheme on each selfish node in a distributed manner. Extensive simulations have been conducted to study the relations between payment and cost of the multicast structure. As all strategyproof mechanisms, the proposed scheme pays each relay agent more than its declared cost to prevent it from lying. Our extensive simulations showed that the overpayment is small when the cost of each agent is a random value between some range.

As we mentioned early, this paper is the first step to explore the general network protocol design when relay agents are non-cooperative. There are many interesting and important issues that have been untouched and left for further study, such as collusion, distributed computing of payments and charging.

REFERENCES