Designing Multicast Protocols for Non-Cooperative Networks

Weizhao Wang, Xiang-Yang Li, *Senior Member, IEEE,* Yu Wang, *Member, IEEE,* and Zheng Sun

***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 are *selfish*: they will try to maximize their own benefits instead of altruistically contributing to the network by following the prescribed protocols. Thus, new protocols should be designed for the *non-cooperative network* that is composed of selfish entities. In this paper, we specifically show how to design *truthful* multicast protocols for non-cooperative networks such that these selfish entities will follow the protocols out of their own interests. By assuming that every entity has a fixed cost for a specific multicast, we give a general framework to decide whether it is possible and how, if possible, to transform an existing multicast protocol to a truthful multicast protocol by designing a proper payment 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 running examples, we show how to design truthful multicast protocols for several multicast structures that are currently used in practice.**

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

I. Introduction

Since first introduced by Deering in [1] and the audiocast experiment by IETF, multicast has received more and more attention 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 the packet is only duplicated at the branching nodes. Numerous multicast protocols have been proposed, and most of them assumed that the network entities will relay the multicast packets as prescribed by the multicast protocol without any deviation. While this may be true for the case of LAN multicast in which all network entities belong to the same organization, it can not be taken for granted when the multicast datagrams are routed through different IP networks (called *autonomous systems* (ASs) in some places). Although multicast benefits the whole system by saving bandwidth and resource, it is dubious that multicast will also bring benefit to every individual AS who relays packets. Thus, it is more reasonable to assume that these ASs, probably owned by different organizations or users, are *selfish*: they aim to maximize their own benefits instead of faithfully conforming to the prescribed multicast protocols. A network composed of selfish ASs is generally known as a *non-cooperative network*. In this paper, we would like to use the terminology "agent" instead of AS because it reflects the selfish nature of the AS.

Nisan and Ronen [2] studied the unicast routing problem in non-cooperative networks and introduced the idea of *algorithmic mechanism design*. They proposed to give the agents some *proper* payments to ensure that every agent conforms to the prescribed protocol regardless of other agents' behavior, which is known as *truthful* or *strategyproof*. They designed the payment for unicast by using the VCG mechanism [3]–[5], which is considered as one of the most positive results in mechanism design. Unfortunately, the VCG mechanism has its own drawback. For multicast, if we want to apply the VCG mechanism, we have to find the minimum cost multicast tree, which is known to be NP-Hard for both link weighted networks [6], [7] and node weighted networks [8], [9]. If we insist on applying the VCG mechanism to a multicast topology that does not have the minimum cost, the VCG mechanism is no longer truthful [10]. Thus, some payment schemes other than the VCG mechanism should be designed for multicast. Recently, several non-VCG truthful payment schemes were proposed in [10] for several commonly used multicast trees. In this paper, instead of focusing on a specific multicast structure, we study whether it is possible to transform a multicast protocol using any given multicast structure to a truthful multicast protocol, and if possible, how to design such truthful multicast protocol.

Designing a truthful payment scheme is not the whole story for many practical applications. A natural question to be answered is who will be charged for the payments to the relay agents. A simple solution is that the organization to which the receivers belong pays [10]. However, this solution is not panacea. In many applications such as video streaming, each individual receiver often has to pay for receiving the data. How to charge the receivers for multicast transmission has been studied extensively in literatures [11]–[16]. 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 will reveal their true costs, and (3) the multicast tree is fixed as the union of the shortest paths from the source to receivers. In a sharp contrast, we take the selfish behavior of the relay agents into account in this paper. Thus, we model the network differently by assuming that

W. Wang and Z. Sun are with Google Inc., USA (e-mail: weizhao@google.com; sunzheng@gmail.com).

X.-Y. Li is with Illinois Institute of Technology, Chicago, IL, USA (e-mail: xli@cs.iit.edu). The research of X.-Y. Li is partially supported by National Basic Research Program of China (973 Program) under grant No. 2006CB30300, the National High Technology Research and Development Program of China (863 Program) under grant No. 2007AA01Z180, the RGC under Grant HKBU 2104/06E and CERG under Grant PolyU-5232/07E. Partial of the work was done when Li visited Microsoft Research Asia, and State Key Laboratory of Novel Software Technology, NanJing University.

Y. Wang is with the University of North Carolina at Charlotte, Charlotte, NC, USA (e-mail: yu.wang@uncc.edu). The work of Y. Wang was partially supported by the US National Science Foundation (NSF) under Grant No. CNS-0721666, and funds provided by Oak Ridge Associated Universities.

(1) the relay agents are selfish and rational, (2) the receivers always receive the data and pay what they "should" pay in a fair way, and (3) the multicast topology could be any structure, 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. We also show the hardness when both the receivers and the relay agents are selfish and rational, and each receiver has a privately known valuation.

The main contributions of this paper are two-fold. First, we present a general framework to decide whether it is possible, and how, if possible, to transform an existing multicast protocol to a truthful one. We then show how the payments to the relay agents are shared *fairly* among the receivers. As running examples, we show how to design truthful multicast protocols for some commonly used Inter-AS multicast protocols.

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 the truthful payment and how to compute it based on a given multicast structure. We show how to design truthful multicast protocols for the Inter-AS multicast protocol based on source-based tree in Section IV and shared-based tree in Section V. An alternative model for truthful multicast is discussed in Section VI. We conclude our paper in Section VII.

II. Technical Preliminaries

A. Algorithmic Mechanism Design

In a standard model of algorithmic mechanism design, there are n agents $\{1, 2, \cdots, n\}$. Each agent i has some *private* information t_i, called its *type*, *e.g.*, its cost to forward a packet in a network environment. All agents' types define a *profile* $t = (t_1, t_2, \cdots, t_n)$. Each agent i declares a valid type τ_i, which may be different from its actual type t_i, and all agents' strategies define a declared type vector $\tau = (\tau_1, \cdots, \tau_n)$. A mechanism $M = (\mathcal{O}, \mathcal{P})$ is composed of two parts: an allocation method $\mathcal{O}$ that maps a declared type vector τ to an output o, and a *payment* scheme $\mathcal{P}$ that decides the monetary payment $p_i = \mathcal{P}_i(\tau)$ 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* (also 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 τ_i to maximize its utility u_i. Let $\tau_{-i} = (\tau_1, \cdots, \tau_{i-1}, \tau_{i+1}, \cdots, \tau_n)$, *i.e.*, the strategies of all other agents except i and $\tau|^i a = (\tau_1, \tau_2, \cdots, \tau_{i-1}, a, \tau_{i+1}, \cdots, \tau_n)$. In this paper, we are only interested in a mechanism $M = (\mathcal{O}, \mathcal{P})$ that satisfies the following three conditions:

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

A mechanism is *truthful* if it satisfies both IR and IC. Thus, for every agent i, revealing its true type t_i maximizes its utility regardless of what other agents do.

VCG mechanism: A mechanism $M = (\mathcal{O}, \mathcal{P})$ belongs to the Vickrey-Clarke-Groves (VCG) mechanism family [3]–[5] if (1) there are fixed positive numbers β_i, $i \leq i \leq n$, such that the output $\mathcal{O}(t)$ maximizes the objective function $g(o, t) = \sum_i \beta_i \cdot w_i(t_i, o)$, and (2) the payment to the agent i is $\mathcal{P}_i(t) = \frac{1}{\beta_i} \sum_{j \neq i} \beta_j \cdot w_j(t_j, \mathcal{O}(t)) + h_i(t_{-i})$. Here $h_i()$ is an arbitrary function of t_{-i}, e.g., $h_i(t_{-i}) = -\frac{1}{\beta_i} \sum_{j \neq i} \beta_j \cdot w_j(t_j, \mathcal{O}(t_{-i}))$. A VCG mechanism is truthful [5].

B. Network Model and Problem Statement

In this paper, we focus on the Inter-AS multicast instead of the Intra-AS routing because Intra-ASs are usually cooperative instead of non-cooperative. Here, we model the Inter-AS network topology as a graph $G = (V, E, c)$, where $V = \{v_1, \cdots, v_n\}$ is the set of ASs, $E = \{e_1, e_2, \cdots, e_m\}$ is the set of links between ASs. Usually, in Inter-AS routing, each AS actually is an independent economic decision maker who could choose its strategy for financial advantage in routing decisions. We assume that each AS v_i is an individual agent and it has a *fixed* private cost c_i to transmit a unit size of data in multicast. Thus, every AS is called upon to declare its cost to the protocol. When the nodes are the selfish agents, we call this network a *node weighted network*. On the other hand, sometimes we need to treat the selfish agents as links in the network, *e.g.*, the multicast datagram is sent from one AS to another AS by using application layer tunneling through other ASs. If links are agents, the network is modeled as a *link weighted network*. Most of our general techniques in Section III and Section IV are not specific to one model, and thus can be applied to both models. Notice that all our results also apply to other network models, such as peer-to-peer networks (P2P) [25], [26].

Given a set of multicast group members, in this paper, the receivers are the ASs with some attached group members instead of the actual end hosts who are the multicast group members. For the convenience of our analysis, we assume that s is the source AS in one specific multicast and the size of the data is normalized to 1. We also assume that agents in the network will not *collude* to improve their profits together. In order to prevent monopoly, we assume that the network is bi-connected. Given a source node $s = q_0$ and a set of multicast receivers $R = \{q_1, q_2, \cdots, q_r\} \subset V$, we need to

1) construct a topology (a tree, a mesh, a ring, etc.) that spans the source and all receivers;
2) calculate a payment for each relay AS according to a *payment scheme* that is truthful;
3) charge each receiver according to a *payment sharing scheme* that is *fair*. We will formally define what is fair in subsection III-C.

Here the multicast protocol for the network with n ASs is a mechanism $M = (\mathcal{O}, \mathcal{P}, \xi)$ for the n selfish agents. The allocation method $\mathcal{O}$ is the method to construct the multicast topology and the output o is the constructed topology which includes all relay ASs who are selected to participate the multicast sessions. The payment scheme $\mathcal{P}$ decides the payment for each relay AS. The payment sharing scheme ξ is used for sharing the charges for each receiver.

One thing we should highlight here is that, instead of reinventing the wheel by designing some new multicast structures, we focus on how we can design a truthful payment scheme for a certain 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 fair way.

Given a structure $H \subseteq G$, we use $\mathbf{c}(H)$ to denote the total cost of all agents in H. If we change the cost of any AS 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 AS v_i from the network, we denote it as $c|^i \infty$. Hereafter, we use $\mathsf{LCP}(u, v, c)$ to denote the least cost path from node u to node v in a network $G = (V, E, c)$. For simplicity of notations, we will use only the cost vector c to denote the network $G = (V, E, c)$ if no confusion is caused. We let c_{-i} denote the costs of all ASs other than AS v_i. We summarize all notations and abbreviations used in this paper in Table I which is given in Appendix.

C. Related Work

Routing has been part of algorithmic mechanism design from the very beginning. Nisan and Ronen [17] provided a polynomial-time truthful mechanism for unicast routing in a centralized computational model. Each link e_i of the network is an agent and has a private cost t_i of sending a message. Their mechanism is essentially a VCG mechanism. The result in [17] is extended in [18] to deal with unicast problem for all pairs of agents. They assume that there is a traffic demand $T_{i,j}$ from an agent i to an agent j. They also gave a distributed method to compute the payment. Anderegg and Eidenbenz [19] recently proposed a similar routing protocol based on the VCG mechanism for wireless ad hoc networks. By assuming that each node is a selfish agent, Wang and Li [20] proposed an asymptotically optimum centralized method to compute the payment for unicast and showed that *no* truthful mechanism can prevent collusion among any pair of agents.

For multicast, Feigenbaum *et al.* [15] assumed that there is a universal tree T spanning all receivers and for every subset $Q \subseteq R$ of receivers, the tree $T(Q)$ spanning Q is merely the subtree of T that spans Q. They also assumed that the link costs are publicly known and each receiver q_i has a privately known valuation w_i on receiving the data. It will report a number w_i', which is the amount of money it is willing to pay to receive the data, and w_i' may be different from w_i. They studied how to select a subset $Q \subset R$ of receivers according to some criteria and proposed to use *Shapely value* and *marginal cost* to share the link cost of the multicast tree. Maximizing profit in multicast was studied in [21], [22] ([22] is based on cancellable auction [23]). Sharing the *cost* of the multicast structure among receivers to achieve some fairness was studied in [14], [16], [24], [27]–[29]. Wang *et al.* [10] studied how to design truthful multicast protocols for various multicast trees in wireless networks when the nodes or links are selfish.

III. Characterization of Truthful Multicast Routing

Several multicast topologies have been proposed and used in practice and more topologies are expected to appear in the near future. It will be difficult, if not impossible, to design a truthful multicast mechanism for each of these topologies individually. Thus, instead of studying some specific multicast topologies, we focus on designing a general framework to solve the problem whether there is, and how to design if it exits, a truthful mechanism for a given multicast topology. We also consider how to charge the receivers to cover the payments to the selfish 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 $\kappa_i(\mathcal{O}, c_{-i})$.
```

Intuitively, we may still want to use the VCG payment schemes for these multicast topologies. Notice that an allocation method 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 solution with non-optimal approximation usually leads to untruthful mechanisms [10]. Thus a mechanism other than VCG is needed when we cannot find the optimal solution or the objective is not to maximize the total valuation of the agents. This paper presents the *first general* framework to design truthful mechanisms for multicast in case we cannot find a structure with the minimum total cost.

A. Existence of 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. The following definition and theorem will present a sufficient and necessary condition for the existence of the truthful payment scheme.

Definition 1: A method $\mathcal{O}$ constructing a multicast topology satisfies the **monotone non-increasing property** (MNP) if for every agent i and fixed c_{-i}, the 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 cost $c_{i_1} < c_{i_2}$.

Obviously, the above condition is equivalent to the following condition: there exists a threshold value $\kappa_i(\mathcal{O}, c_{-i})$ such that if i is selected as a relay agent, then its cost is at most $\kappa_i(\mathcal{O}, c_{-i})$. For convenience, we use $\mathcal{O}_i(c) = 1$ (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 $\mathcal{O}$ constructing a multicast topology, there exists a payment $\mathcal{P}$ such that $M = (\mathcal{O}, \mathcal{P})$ is truthful if and only if $\mathcal{O}$ satisfies the MNP.
The detailed proof of this theorem is given in Appendix. In the proof, we first prove that if there exists a truthful payment $\mathcal{P}$ based on $\mathcal{O}$ then $\mathcal{O}$ satisfies the MNP. Then, by constructing the following payment scheme $\mathcal{P}$, we prove that if $\mathcal{O}$ satisfies MNP, there exists a truthful mechanism $M = (\mathcal{O}, \mathcal{P})$.

B. Rules to Find Truthful Payment Scheme

Given a multicast structure satisfying MNP, 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. Instead

Algorithm 2 A round-based multicast method

```
1: Set $r = 1$ and $c^{(1)} = c$ and $Q^{(1)} = R$ initially.
2: repeat
3:     Let $\mathcal{O}^r$ be a deterministic method that decides in round $r$ which agents will be selected.
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 an updating rule $\mathcal{U}^r$:
           $\mathcal{U}^r : \mathcal{O}^r \times [c^r, Q^{(r)}] \to [c^{(r+1)}, Q^{(r+1)}].$
5: until the desired property of the multicast topology is met
6: Return the union of the selected relay agents in all rounds.
```

of trying to propose a unified approach that can find the threshold value for all multicast topologies satisfying MNP, we present some useful techniques to find the threshold value under certain circumstances. Our general approach works as follows. First, given an allocation method $\mathcal{O}$ that constructs a multicast structure, we decompose it into several simpler allocation 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. In the following, we present several useful decomposition techniques.

1) Simple Combination: Given a multicast method $\mathcal{O}$, let $\kappa(\mathcal{O}, c)$ denote the 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 constructed 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 g allocation methods $\mathcal{O}^1, \cdots, \mathcal{O}^g$ each satisfying MNP, and the threshold value $\kappa(\mathcal{O}^i, c)$ for each $\mathcal{O}^i$, the method $\mathcal{O}(c) = \mathcal{O}^1(c) \bigvee \mathcal{O}^2(c) \bigvee \cdots \bigvee \mathcal{O}^g(c)$ satisfies MNP. Moreover, the threshold value for $\mathcal{O}$ is

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

The proof of Theorem 2 is straightforward and thus is omitted here. We will show how to use this simple combination technique in Section IV. Notice each individual method $\mathcal{O}^i$ may not construct a multicast tree at all.

2) Round-based Method: Many multicast topologies are constructed in a *round-based* manner: in each round some previously unselected agents are selected, and then the network and the receiver set are updated if necessary. In Algorithm 2, we give a general characterization of a round-based method that constructs a multicast topology.

To illustrate the general round-based method, in Algorithm 3 we review a round-based multicast tree construction method [7] that finds a tree whose cost is no more than 2 times that of a minimum cost Steiner tree (MCST) in a link weighted network. We denote the constructed multicast tree as LST, which stands for Link-weighted Steiner Tree.

Here, *no receiver remains in R* corresponds to the *desired property* of the general round-based method; $\mathsf{LCP}(s, q_i, d)$ in round r corresponds to $\mathcal{O}^r$; setting costs of links on $\mathsf{LCP}(s, q_i, d)$ to 0 and removing q_i from R is the *updating rule* $\mathcal{U}^r$. To study whether a general round-based method implies a truthful payment scheme we propose the following definition.

Algorithm 3 Link weighted multicast structure [7]

```
1: repeat
2:     Let $d$ be the vector of costs declared by all agents.
3:     Find one receiver in the receiver set $R$, say $q_i$, that is closest to the source $s$, i.e., $\mathsf{LCP}(s, q_i, d)$ has the lowest cost among the shortest paths from $s$ to all receivers. Connect $q_i$ to the source $s$ using $\mathsf{LCP}(s, q_i, d)$, i.e., all agents on this path are selected.
4:     Set the cost of every link on this path to 0. Remove $q_i$ from the receiver set $R$.
5: until no receiver remains in $R$
```

Definition 2: An updating rule $\mathcal{U}^r$ is said to be *crossing-independent* if for any agent i not selected in round r:

- $c_{-i}^{(r+1)}$ and $Q^{(r+1)}$ do not depend on $c_i^{(r)}$.
- For a fixed $c_{-i}^{(r)}$, if $d_i^{(r)} < c_i^{(r)}$ then $d_i^{(r+1)} < c_i^{(r+1)}$.

Theorem 3: A round-based multicast method $\mathcal{O}$ satisfies MNP if, for every round r, method $\mathcal{O}^r$ satisfies MNP and the updating rule $\mathcal{U}^r$ is crossing-independent.

Proof: For an agent i, fix the cost c_{-i} of all other agents. We prove that if i is selected when the cost vector is $a = \{c_{-i}, c_i\}$, then it is also selected when the cost vector is $b = \{c_{-i}, c'_i\}$ such that $c'_i < c_i$. Without loss of generality, we assume that i is selected in round r when the cost vector is a. Then when the cost vector is b, if agent i is selected before round r, our claim holds. Otherwise, in round r, $a_{-i}^{(r)} = b_{-i}^{(r)}$ and $a_i^{(r)} > b_i^{(r)}$ since agent i is not selected in the previous rounds. Notice that agent i is selected in round r when the cost vector is $a_i^{(r)}$. Thus, agent i is also selected in round r when the cost vector is $b_i^{(r)}$ since $\mathcal{O}^r$ satisfies MNP. ■

In Algorithm 4, we show how to find the threshold value for any selected agent k when the truthful payment scheme exists for a round-based multicast method.

We use the network in Figure 1 to illustrate how to find the threshold value for link v_3v_4 based on LST. In the first round, v_3v_4 cannot 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 that the updating rule of Algorithm 3 does not change the cost of an unselected agent, *i.e.*, it is crossing-independent and $f_i(x) = x$. Thus, the final threshold value is simply $\max\{\ell_1, \ell_2\} = 0.9$, which is the payment to link v_3v_4. Similarly, we can find all selected links' threshold values as shown by the numbers in the parenthesis in Figure 1(b).

C. Fair Payment Sharing Scheme

For a given set of receivers, after we calculate the payment $p_k(d)$ for every relay agent k based on declared costs d, we are ready to study how to share the payments fairly among

Algorithm 4 Computing payment for a selected agent k based on round-based method $\mathcal{O}$

```
1: Initially set the cost $c_k$ of $k$ to $\infty$ and $r = 1$.
2: repeat
3:   Find the threshold value for agent $k$ based on $\mathcal{O}^r$ under cost vector $c_{-k}^{(r)}$ and receiver set $Q^{(r)}$. Let $\ell_r = \kappa_k(\mathcal{O}^r, c_{-k}^r)$ be the threshold value found. Here we set $\ell_r = 0$ if agent $k$ cannot be selected in this round under any cost.
4:   Update the cost vector and receiver set to obtain the new cost vector $c^{(r+1)}$ and $Q^{(r+1)}$. Set $r = r + 1$.
5: until the desired property of the multicast topology is met
6: Fix $c_{-k}$ and assume $x$ is the payment for agent $k$. Let $f_i(x)$ be the cost for agent $k$ in round $i$ if the original cost is $c|^k x$. Then $x$ the largest value that satisfies the following inequalities: $f_i(x) \le \ell_i$ for $1 \le i \le r$. In other words, the payment to an agent $k$ is the largest possible value it could declare such that it is still selected in some round.
```

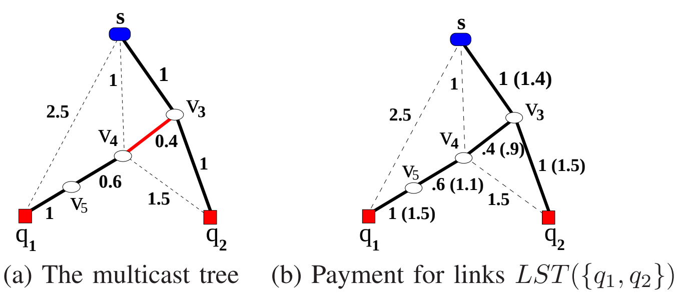


Fig. 1. Payment calculation based on LST found by Algorithm 3.

receivers. 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 [11], [12], [15], [27], with the assumption that the costs 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 use the ELSD to share the total payment among receivers, it usually is not fair.

Given a set of receivers R, let $\mathcal{P}(R, d) = \sum_k p_k(R, d)$ denote the total payment to all relay agents. For a sharing scheme ξ, let $\xi_i(R, d)$ denote the sharing (or called charge) of a receiver q_i. Let $\xi(R, d) = \sum_{q_i \in R} \xi_i(R, d)$ be the total payment collected from all receivers. We call a sharing scheme ξ *reasonable* or *fair* if it satisfies the following criteria.

1) **Budget Balance** (BB): The total payment to all agents should be shared by all receivers, *i.e.*, $\mathcal{P}(R, d) = \xi(R, d)$.
2) **Nonnegative Sharing** (NNS): Any receiver q_i's sharing should be positive, *i.e.*, $\xi_i(R, d) > 0$.
3) **Cross-Monotone** (CM): For any two receiver sets $R_1 \subseteq R_2$ containing q_i: $\xi_i(R_1, d) \le \xi_i(R_2, d)$. In other words, for a given network, receiver i's sharing does not increase when more receivers require service.
4) **No-Free-Rider** (NFR): The sharing $\xi_i(R, d)$ of a receiver $q_i \in R$ is at least $\frac{1}{|R|}$ of its unicast sharing $\xi_i(q_i, d)$. Thus, the sharing of any receiver will not be too small.

By assuming a universal multicast tree and publicly known link costs, Feigenbaum *et al.* [15] proved that ELSD cost sharing scheme is fair. Unfortunately, the ELSD scheme is not fair if it is used to share the payment.

Lemma 4: ELSD is not a fair sharing scheme for payment $\mathcal{P}$ defined based on tree LST.

Proof: We prove it by presenting a counter example using the network shown in Figure 1 (a). When consider only one receiver in LST, we have $\mathcal{P}(q_1, c) = 2.6$ and $\mathcal{P}(q_2, c) = 1.4 + 1.5 = 2.9$. See Figure 2 for illustration. For two receivers q_1, q_2, if we use ELSD to share payment, the sharing by q_1 is $\xi_1(\{q_1, q_2\}, c) = \frac{1.4}{2} + 0.9 + 1.1 + 1.5 = 4.2$ which is larger than its sharing $\xi_1(q_1, c) = 2.6$ when q_1 is the only receiver. Thus, ELSD violates the CM property. It implies that ELSD is not a fair sharing scheme for multicast topology LST. ■

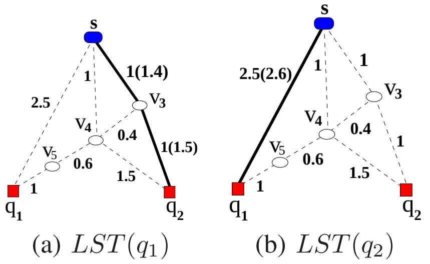


Fig. 2. $LST(q_1)$ and $LST(q_2)$ and their corresponding payment.

Furthermore, using the same example, we prove that:

Lemma 5: No payment sharing scheme satisfies both CM and BB for the truthful payment scheme based on LST.

Proof: For the sake of contradiction, we assume that a sharing scheme ξ' satisfies both CM and BB. From the property of BB, we have ${\xi'}_1(q_1, c) = 2.6$, ${\xi'}_1(q_2, c) = 2.9$ and ${\xi'}_1(\{q_1, q_2\}, c) + {\xi'}_2(\{q_1, q_2\}, c) = 6.4$. From CM, we have ${\xi'}_1(\{q_1, q_2\}, c) \le {\xi'}_1(q_1, c) = 2.6$ and ${\xi'}_2(\{q_1, q_2\}, c) \le {\xi'}_2(q_2, c) = 2.9$. Combining these two inequalities, we obtain $6.4 = {\xi'}_1(\{q_1, q_2\}, c) + {\xi'}_2(\{q_1, q_2\}, c) \le 2.9 + 2.6 = 5.5$, which is a contradiction. ■

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

IV. Truthful Multicast Using Source-Based Tree

In this section, we illustrate how to design a truthful multicast protocol with the support of Multiprotocol Extensions for BGP-4 [30]. We treat every AS i in the network as a node in the graph, and assume that it has a fix cost c_i to relay a unit size of datagram for a specific multicast regardless of its downstream links. This could be because that the multicast ASs adopt the Reverse Path Broadcasting (RPB) scheme or the cost of sending extra copies to other interfaces is negligible. Thus, the network is modeled as a node weighted graph. All our results presented hereafter also apply to the case when the network is modeled as a link weighted graph. We focus on the *source-based tree* in this section and discuss the *shared-based tree* in the next section.

A. Construct Multicast Tree

Before designing a truthful multicast protocol, we review some technical details of MBGP including the multicast tree construction method. Multiprotocol extension for BGP (MBGP) [30] is an extension to the existing Border Gate Way (BGP) protocol [31]. In BGP, every node v_i stores, for each other node v_j, the least cost path (the sequence of ASs traversed) from v_i to v_j. Let D be the diameter of the network, *i.e.*, the maximum number of ASs in an least cost path (LCP). An AS stores $O(n \cdot D)$ AS numbers. In BGP, to perform Inter-AS multicast routing, we use the BGP infrastructure that was in place for unicast routing. A multicast routing protocol, such as Protocol Independent Multicast (PIM) dense mode, uses the multicast BGP database to perform Reverse Path Forwarding (RPF) lookups for multicast-capable sources.

Thus, given a set of receivers R, the least cost path between the source s and each receiver $q_i \in R$ under the reported cost profile d is already in receiver q_i's unicast database. The union of all least cost paths between the source and the receivers is called the *least cost path tree*, denoted by $\mathsf{LCPT}(R, d)$. Every node that is the part of the multicast tree LCPT has a copy of the tree topology and all datagrams are routed along the tree.

B. Payment Scheme

It was shown in [10] that the direct application of VCG payment scheme on LCPT is not truthful. In other words, a node may have incentives to lie about its cost when VCG payment scheme is used. On the other hand, since LCPT is formed by the union of the least cost paths, by applying Theorem 2, we can show that LCPT satisfies MNP. Thus, there exists a truthful payment scheme and the truthful payment can be found according to Theorem 1. It works as follows.

For each receiver $q_i \in R$, we find the least cost path $\mathsf{LCP}(s, q_i, d)$ from the source s (say q_0) to q_i, and compute an intermediate payment $p_k^{i,0}(d)$ to every node v_k on $\mathsf{LCP}(q_0, q_i, d)$ using the VCG payment scheme for unicast $p_k^{i,0}(d) = d_k + \mathbf{c}(\mathsf{LCP}(q_0, q_i, d|^k\infty)) - \mathbf{c}(\mathsf{LCP}(q_0, q_i, d))$.

The final payment to a node $v_k \in \mathsf{LCPT}$ is

$$p_k(d) = \max_{q_i \in R} p_k^{i,0}(d) \qquad (1)$$

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

C. Distributed Payment Algorithm

Remember that MBGP is only an extension to the BGP which is used for unicast. Usually the unicast is a dominant activity in the Inter-AS routing instead of multicast. Thus, we assume that each AS already implements a truthful payment scheme based on VCG for unicast. In [18], Feigenbaum *et al.* proposed a distributed algorithm to compute the payment $p_k^{i,j}$ for every pair of nodes v_i, v_j and every node v_k on the least cost path $\mathsf{LCP}(v_i, v_j, d)$. Their approach is an extension to the existing BGP routing and converges to a stable state after D_{-k} rounds, where D_{-k} is the maximum possible diameter of graph G after removing any node k from the network. In their approach, at every node v_i, they only store the length of the path $\mathsf{LCP}(v_i, v_j, d)$ for every node v_j, which requires an extra $O(n)$ space. However, in our approach, we require that every node v_i stores all the payments $p_k^{i,j}$ for every possible source node v_j and every node v_k on path $\mathsf{LCP}(v_i, v_j, d)$. Our approach requires an extra space size of $O(\alpha \cdot D)$ for every AS, where α is the number of possible source node and D is the diameter of the network. Clearly, it avoids the recalculation of every $p_k^{i,j}$ when some nodes' costs are updated. The following algorithm summarize the distributed payment computing for multicast when $s = q_0$ is the source node.

Algorithm 5 Distributed payment computing

```
1: for every receiver $q_i$ do
2:     Prepare a control datagram composed of the payment $p_k^{i,0}$ for every node $v_k$ on path $\mathsf{LCP}(q_0, q_i, d)$.
3:     Sends datagram containing the payment information to its parent in the tree $\mathsf{LCPT}$.
4: Upon receiving a packet containing the payment from its child which is originated from receiver $q_i$, node $v_k$ extracts the payment $p_k^{i,0}$ and sends the datagram containing all remaining payment information to its parent if it exists.
5: When a node $v_k$ receives $p_k^{i,0}$ from every downstream receiver $q_i$, it computes the maximum of them as its final payment.
```

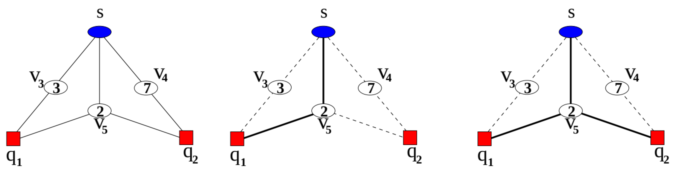
Now we discuss the overhead of our distributed multicast payment computation in terms of both communication messages and memory space used in the AS. It is not difficult to observe that every node receives at most r packets of size $O(D)$ where r is the number of the receivers and D is the diameter of the network. For every node v_i, it only needs to store for each multicast session S the final payment p_S, which is negligible. However, sometime in order to achieve a high efficiency, node v_k may cache every intermediate payment $p_k^{i,0}$. Even in this case, it only needs an extra $O(r)$ space which is much smaller than the space needed for one session of multicast in a cooperative network. Overall, the overhead needed to calculate the payment is small both in terms of space and network message.

D. Payment Sharing Among Receivers

In literature, the Shapely value [32] is one of the most commonly used sharing schemes to achieve BB and CM. If the total payment $\mathcal{P}(R, d)$ satisfies non-decreasing and submodular property, then the Shapely value minimizes the worst-case network welfare loss among all sharing schemes that achieve BB and CM. Here, a payment $\mathcal{P}$ is *submodular* if $\forall R_1 \subseteq Q$ and $R_2 \subseteq Q$, $\mathcal{P}(R_1, d) + \mathcal{P}(R_2, d) \geq \mathcal{P}(R_1 \cup R_2, d) + \mathcal{P}(R_1 \cap R_2, d)$. The *network welfare* is defined as the total valuation of all selected receivers minus the cost of the network providing service. If we apply Shapely value to multicast payment sharing, we obtain the following formula

$$\xi_i(R, d) = \sum_{T \subseteq R - q_i} \frac{|T|!(|R| - |T| - 1)!}{|R|!} (\mathcal{P}(T \cup \{q_i\}, d) - \mathcal{P}(T, d))$$

By assuming a fixed multicast tree and publicly known link costs, Feigenbaum *et al.* [15] proved that ELSD sharing scheme is the Shapely Value. Intuitively, one may want to use



(a) Network (b) $\mathsf{LCPT}(q_1, d)$ (c) $\mathsf{LCPT}(\{q_1, q_2\}, d)$

Fig. 3. ELSD sharing scheme is not fair for payment based on LCPT.

ELSD as the payment sharing scheme. Unfortunately, we will show by example that ELSD is not fair when coupled with LCPT. Consider a network shown by Figure 3(a). There are two receivers q_1, q_2. Tree $\mathsf{LCPT}(q_1, d)$ is shown in Figure 3(b). The total payment to nodes on $\mathsf{LCPT}(q_1, d)$ is 3. Consider $\mathsf{LCPT}(\{q_1, q_2\}, d)$ illustrated by Figure 3(c). The payment to only relay node v_5 is 7. If we apply ELSD to share this payment, the shared payment of receiver q_1 is $\frac{7}{2} = 3.5$ when the receiver set is $\{q_1, q_2\}$. Notice that the payment sharing by q_1 is only 3 when it is the only receiver. Thus, ELSD violates the CM property here. Therefore some fair sharing scheme other than ELSD should be designed. We can use Shapely value due to the following lemma.

Lemma 6: The total payment $\mathcal{P}(R, d)$ for tree LCPT, is nondecreasing and submodular with respect to receiver set R.

Please see Appendix for the proof of the lemma. Consequently, we obtain a sharing scheme satisfying CM and BB by applying Shapely value. However, for any receiver $q_i \in R$, there are $2^{|R|-1}$ subsets in $R - q_i$. Thus, simply applying Shapely value directly is computational intractable when the number of receivers is large. Therefore, we present another interpretation of the sharing scheme that can be computed efficiently. The basic idea 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 a node k is the maximum of payments p_k^i by all receivers. Since different receivers may have different values of payment to agent k, the final payment $\mathcal{P}_k$ should be shared *proportionally* to their values, not *equally* among them (as what we do for cost sharing). Figure 4 illustrates the payment sharing scheme that follows. For any node v_k, let $R(v_k)$ be the set of downstream receivers of v_k. Without loss of generality, we assume that $R(v_k) = \{q_{\sigma_1}, q_{\sigma_2}, \cdots, q_{\sigma_{|R(v_k)|}}\}$ such that $0 \le p_k^{\sigma_1} \le p_k^{\sigma_2} \le \cdots \le p_k^{\sigma_{|R(v_k)|}}$, *i.e.*, $p_k = p_k^{\sigma_{|R(v_k)|}}$. We then divide the payment p_k into $|R(v_k)|$ portions: $p_k^{\sigma_1}$, $p_k^{\sigma_2} - p_k^{\sigma_1}$, $\cdots$, $p_k^{\sigma_i} - p_k^{\sigma_{i-1}}$, $\cdots$, $p_k^{\sigma_{|R(v_k)|}} - p_k^{\sigma_{|R(v_k)|-1}}$. Each portion $p_k^{\sigma_i} - p_k^{\sigma_{i-1}}$ is then equally shared among the last $|R(v_k)| - i + 1$ receivers, which have the largest $|R(v_k)| - i + 1$ payments to v_k.

We first illustrate how to calculate the payment sharing by receiver q_1 using Algorithm 6 for a network represented by Figure 3. For node v_5, the two intermediate payments are $p_{v_5}^1 = 3$ and $p_{v_5}^2 = 7$. First, we obtain a rank of these receivers based on the intermediate payments of $\{q_1, q_2\}$. Then $p_{v_5}^1 = 3$ is equally split between q_1 and q_2 and $p_{v_5}^2 - p_{v_5}^1 = 4$ is charged to q_2 alone. Thus, receiver q_1 is charged $3/2 = 1.5$ and receiver q_2 is charged $1.5 + 4 = 5.5$ in $\mathsf{LCPT}(\{q_1, q_2\}, d)$.

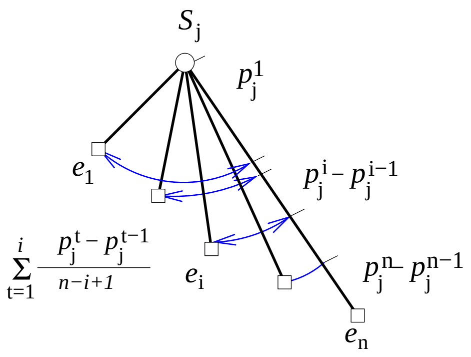


Fig. 4. Share the payment to service providers among receivers fairly.

Algorithm 6 Fair payment sharing scheme for LCPT.

```
1: for each node $v_k \in \mathsf{LCPT}(R, d)$ do
2:     Let $R(v_k)$ be the set of downstream receivers of $v_k$, i.e., $p_k(d) = \max_{q_i \in R(v_k)} p_k^i(d) = \max_{q_i \in R} p_k^i(d)$.
3:     Sort the receivers in $R(v_k)$ according to $p_k^i(d)$ in an ascending order. If two or more receivers have the same value, the receiver with smaller ID ranks first. Let $\sigma = \{\sigma_0, \sigma_1, \cdots, \sigma_{|R(v_k)|}\}$ be the ranking. Here, we add a dummy payment $p_k^{\sigma_0}(d) = 0$ to ranking $\sigma$.
4:     For a receiver not in $R(v_k)$, its sharing of the payment $p_k(d)$ of node $v_k$ is 0.
5:     For a receiver $q_{\sigma_a} \in R(v_k)$, its sharing of the payment $p_k(d)$ to node $v_k$ is:
           $$f_{\sigma_a}^k(R, d) = \sum_{x=1}^{a} \frac{p_k^{\sigma_x}(d) - p_k^{\sigma_{x-1}}(d)}{|R(v_k)| - x + 1} \quad (2)$$
       In other word, for two receivers $q_{\sigma_x}, q_{\sigma_{x+1}}$ who are consecutive in ranking $\sigma$, the difference $p_k^{\sigma_{x+1}}(d) - p_k^{\sigma_x}(d)$ is shared by all receivers who rank after $q_{\sigma_{x-1}}$.
6: The total charge for receiver $q_i$ in $\mathsf{LCPT}$ is
       $$\xi_i(R, d) = \sum_{v_k \in \mathsf{LCPT}(R,d)} f_i^k(R, d) \quad (3)$$
```

Here, q_1's sharing is smaller than the sharing 3 when q_1 is the only receiver. This shows that the payment sharing scheme described by Algorithm 6 is fair for this specific network. The following theorem shows that our sharing scheme is indeed the Shapely value.

Theorem 7: Our sharing scheme defined by Algorithm 6 is the Shapely value.

Refer to Appendix for the proof of the theorem. Recall that when applying Shapely value to a payment satisfying submodular and non-increasing property, the resulting sharing scheme satisfies BB, CM, NNS and NFR. Thus, we have the following theorem directly.

Theorem 8: The sharing scheme in Algorithm 6 for LCPT satisfies NNS, CM, NFR and BB.

E. Distributed Computing of Payment-Sharing

In practice, we may need to implement a distributed payment sharing scheme. In the following, we present a distributed

Algorithm 7 Distributed payment sharing scheme

```
1: Initially, the source node $s$ sends all its children in $\mathsf{LCPT}$ a $r$-dimensional vector $\vartheta = 0$ for all receivers.
2: Every node $v_k$ in $\mathsf{LCPT}(R, d)$, upon receiving a sharing vector $\widetilde{\vartheta}$ from its parent, updates the charge for each of its downstream receivers $q_i$ as $\vartheta_k[i] = \widetilde{\vartheta}[i] + f_k^i(R(v_k))$. Here, $f_k^i(R(v_k))$ is calculated according to Algorithm 6.
3: if node $v_k$ has at least one downstream receiver then
4:     for every children node $v_j$, it constructs a charge vector $\vartheta_j = (\vartheta[i_1], \vartheta[i_2], \cdots, \vartheta[i_{|R(v_j)|}])$ Here, the charge $\vartheta[i_t]$, $1 \leq t \leq |R(v_j)|$, is for receiver $q_{i_t}$ who is a downstream receiver of node $v_j$. It then sends vector $\vartheta_j$ to node $v_j$.
5: Every receiver $q_i$ will finally receive a charge $\vartheta[i]$ which is equal to $\xi_i(R, d)$ defined in Equation (3).
```

algorithm that implements our payment sharing scheme. It requires at most $O(r)$ space for each agent and with $O(r \cdot h)$ total messages, where h is the height of LCPT.

In our distributed algorithm, for any node $v_k \in \mathsf{LCPT}(R, d)$, we not only need its final payment $p_k(d)$, but also need the intermediate payment $p_k^j(d)$ for every downstream receiver q_j. We assume that this is already available through our distributed payment computing scheme (see Algorithm 5). In our distributed charge scheme, at every node v_k, we use $\vartheta_k[i]$ to store the sum of the charge of v_k's upstream nodes to the receiver q_i. Our distributed payment sharing scheme is implemented in a top-down fashion from the source to all receivers. It is easy to show that Algorithm 7 indeed computes the payment sharing of each receiver correctly.

V. Truthful Multicast Using Shared-Based Tree

In section IV, we discussed how to design a truthful multicast protocol using MBGP based on a source-based tree LCPT. However, in practice, Inter-AS multicast usually uses a shared-based tree (SBT) instead due to the following reasons:

1) Multicast routing protocols (such as MOSPF, DVMRP and PIM-DM) using a source-based tree are suitable for LAN networks while multicast routing protocols (such as PIM-SM and CBT) using a shared-based tree are more suitable for networks composed of different ASs;
2) The shared-based tree is more scalable than the source-based tree for applications in which every group member could act as a source.

Furthermore, we can show that the size of extra space needed to support the multicast payment calculation could be reduced significantly. Here, we use the PIM-SM as the routing protocol and the AS should also support MBGP in order to conduct multicast.

We first review the multicast tree construction method by the PIM-SM multicast protocol. For a specific multicast group, the PIM-SM protocol specifies a Rendezvous Point (RP) and the RP maintains a RP-tree, which is usually a least cost path tree that spans all the group members. When any group member wants to send data to the group, it first encapsulates each data packet in a *Register* message and sends it by unicast to the RP for that group. The RP decapsulates the register messages and forwards the enclosed data packet to downstream group members on the shared RP-tree. Upon receiving data packet from its upstream AS, each intermediate AS further forwards data packets to its downstream ASs. Thus, we can treat the multicast based on a shared-based tree as two separate activities: a unicast from the source to RP, and a multicast with RP as the virtual source node.

Algorithm 8 Truthful payment scheme for SBT

```
1: Assume that $s = q_0$ is the RP for a multicast group; and $q_i$ is the source node for a specific multicast session.
2: Let $d$ be the cost vector declared by all relay nodes.
3: Set the receiver set $Q$ as $R \backslash q_i$.
4: Compute the payment $p_k^Q(d)$ for every node $v_k$ on the tree $\mathsf{LCPT}(Q, d)$ rooted at RP $s$ and spanning all receivers $Q$. Set $p_k^Q(d) = 0$ for other nodes $v_k$.
5: Calculate the payment $p_k^{i,0}(d)$ for every node $v_k$ on path $\mathsf{LCP}(q_i, q_0, d)$. Set $p_k^{i,0}(d) = 0$ otherwise.
6: for each node $v_k$ do
7:     $p_k(d) = p_k^Q(d) + p_k^{i,0}(d)$.
```

Algorithm 9 Fair payment sharing scheme for SBT

```
1: Set the receiver set $Q = R \backslash q_i$.
2: Share the payment incurred by unicast between $q_i$ and RP equally among all receivers $Q$. The payment shared by receiver $q_k$ is denoted as $\xi_k^{uni}(Q, d)$.
3: Share the payment of multicast with source $s = q_0$ and receiver set $Q$ among all receivers according to Algorithm 6. The payment shared by receiver $q_k$ is denoted as $\xi_k^{mul}(Q, d)$.
4: The final payment shared by the receive $q_k$ is $\xi_k(Q, d) = \xi_k^{uni}(Q, d) + \xi_k^{mul}(Q, d)$ when $q_i$ is the source.
```

We then discuss how to compute the payment to each relay agent and share these payments among receivers. Let $p_k^R(d)$ denote the payment to a relay node $v_k \in \mathsf{LCPT}(R, d)$ according to our truthful payment scheme (see formula (1)). Algorithm 8 presents our truthful payment scheme for multicast based on a shared-based tree.

Theorem 9: The payment scheme defined by Algorithm 8 is truthful.

The proof of Theorem 9 is straightforward and thus is omitted. A distributed payment computing protocol similar to Algorithm 5 can be easily designed and thus is omitted here. We then discuss how to share the payments among receivers in Algorithm 9.

Theorem 10: The payment sharing scheme defined in Algorithm 9 is fair, *i.e.*, it satisfies NNS, CM, NFR and BB.

Both the proof of the correctness of the above method and distributing payment-sharing computing are similar to the source-based tree case, thus are omitted here. Here, we do not consider the source q_i as a receiver, which implies that q_i does not share any payment. If q_i should also be treated as a receiver and share the payment in certain circumstances, we just need to modify the receiver set $Q = R$ instead of $Q = R \backslash q_i$ in Line 1 of Algorithm 9.

Algorithm 10 Payment sharing for selfish receivers R

```
1: $Q \leftarrow R$.
2: repeat
3:     Prune out the branches of the original LCPT that do not have receivers in $Q$.
4:     For each receiver $q_i \in Q$, compute the payment sharing $\xi_i(Q, d)$ based on the declared costs of all relay agents.
5:     $\forall q_i \in Q$, the receiver $q_i$ is removed from $Q$ if $\xi_i(Q, d) > \zeta_i$, i.e., $Q \leftarrow Q - \{q_i\}$ if $\xi_i(Q, d) > \zeta_i$.
6: until no receiver is removed in this round
7: All remaining receivers $Q \subseteq R$ will receive the data and pay a sharing $\xi_i(Q, d) \leq \zeta_i$.
```

VI. Sharing Payment Among Selfish Receivers

So far, each receiver q_i is assumed to pay its fair sharing $\xi_i(R, d)$ computed by our payment sharing Algorithm 6. In practice, each individual receiver may have a maximum valuation indicating how much it is willing to pay to receive the information from the source. A receiver will choose to receive the information if and only if the charge is at most its valuation. Furthermore, a receiver could also be *selfish* and *rational*: it will always maximize its profit by manipulating its reported valuation, should it be possible. This makes the multicast design even harder when both the relay agents and the receivers could be selfish. It is well-known that a cross-monotone *cost sharing scheme* implies a truthful mechanism for selfish receivers [27]. Thus, when each receiver q_i is willing to pay at most ζ_i for the data, we may design a payment-sharing mechanism as follows.

However, we found out that a selected relay agent may have incentives to lie about its relay cost under payment scheme defined in Algorithm 6. Next, we show that a relay agent could change the payment sharing of its downstream receivers by reporting a higher or lower cost.

Figure 5 illustrates such an example of reporting a lower cost. Here the private valuations of receivers q_1 and q_2 are 12 and 17 respectively. The true costs of links are $c(sv_3) = 5$, $c(sv_4) = 3$, $c(v_3q_1) = 5$, $c(v_4q_2) = 5$, and $c(q_1q_2) = 3$. For the sake of simplicity, we assume that all links (except link v_4q_2) report their costs truthfully in the remaining discussion. Notice when link v_4q_2 truthfully reports its cost, the multicast tree consists of links sv_3, v_3q_1, sv_4 and v_4q_2, as shown by Figure 5 (b). In addition, the payments to selected links are $p_{sv_4} = c(sv_3) + c(v_3q_1) + c(q_1q_2) - c(v_4q_2) = 8$, $p_{v_4q_2} = 10$, $p_{sv_3} = 6$, $p_{v_3q_1} = 6$, and the payments to all other links are 0. Consider two receivers q_1 and q_2: the payment sharing by receiver q_1 is $p_{sv_3} + p_{v_3q_1} = 12$, which is not larger than its valuation 12; the payment sharing by q_2 is $p_{sv_4} + p_{v_4q_2} = 18$, which is larger than its valuation 17. Consequently, the receiver q_2 will *not* join the multicast (illustrated by Figure 5 (c)). In other words, link v_4q_2 gets payment 0.

Let's see what happens if link v_4q_2 lies its cost down to $3 < c(v_4q_2)$ (illustrated by Figure 5 (d)). Figure 5 (e) shows the multicast tree constructed in this scenario. Notice that when link v_4q_2 reported its cost as 3, the payments to selected links are $p_{sv_4} = 10$, $p_{v_4q_2} = 10$, $p_{q_1q_2} = 4$, and the payments to all other links are 0. It is easy to show that the payment sharings by receivers q_1 and q_2 are 7 and 16 respectively. Then both q_1 and q_2 will join the multicast now. Thus, the link v_4q_2 gets a payment 10 when it lies its cost down to 3. This example shows that a relay agent could lie down its cost to improve its utility. We can devise an example in which a relay agent can lie up its cost to improve its utility.

VII. Conclusion

In this paper we discuss how to design truthful payment schemes and payment sharing mechanisms that stimulate cooperation for multicast in a non-cooperative network. It is well-known that the traditional protocols designed for conforming agents cannot prevent the selfish agents from manipulating their reported costs to increase their benefits. Instead of redesigning the wheel, 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, if possible, to transform an existing multicast protocol to a truthful multicast protocol. We then showed how the payments to all the relay agents could be shared *fairly* among all receivers so that it encourages collaboration among receivers. As running examples, we showed how to design a truthful multicast protocol when the least cost path tree or the shared-based tree is used for multicast. We also discussed in detail how to implement this scheme on each selfish node in a distributed manner. As all truthful mechanisms, the proposed scheme pays each relay agent more than its declared cost to prevent it from lying. This paper is the first step to exploring the general network protocol design when relay agents are non-cooperative. There are many interesting and important issues that have not been touched (such as how to prevent collusion among agents, how to model multicast game as repeated games, and how to use Nash equilibrium to design multicast protocol instead of truthful algorithmic mechanism) and thus are left for further study. In addition, how to efficiently implement the proposed multicast mechanisms in real ASs need to be further investigated.

References

[1] Steve E. Deering, *Multicast Routing in a Datagram Internetwork*, Ph.D. thesis, Stanford University, Dec 1991.

[2] Noam Nisan, "Algorithms for selfish agents," in *Proceedings of the 16th Symposium on Theoretical Aspects of Computer Science, LNCS*, vol. 1563, pp. 1–15, 1999.

[3] W. Vickrey, "Counterspeculation, auctions and competitive sealed tenders," *Journal of Finance*, vol. 16, no. 1, pp. 8–37, 1961.

[4] E. H. Clarke, "Multipart pricing of public goods," *Public Choice*, vol. 11, no. 1, pp. 17–33, 1971.

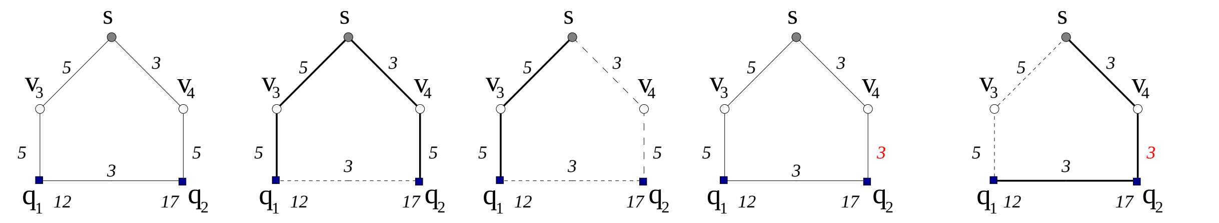
[5] T. Groves, "Incentives in teams," *Econometrica*, vol. 41, no. 4, pp. 617–631, 1973.

[6] Gabriel Robins and Alexander Zelikovsky, "Improved steiner tree approximation in graphs," in *Proc. of ACM SODA*, 2000, pp. 770–779.

[7] H. Takahashi and A. Matsuyama, "An approximate solution for the steiner problem in graphs," *Math .Jap.*, vol. 24, no. 6, pp. 573–577, 1980.

[8] P. Klein and R. Ravi, "A nearly best-possible approximation algorithm for node-weighted steiner trees," *Journal of Algorithms*, vol. 19, no. 1, pp. 104–115, 1995.

[9] S. Guha and S. Khuller, "Improved methods for approximating node weighted steiner trees and connected dominating sets," *Information and Computation*, vol 150, no. 1, pp. 57–74, 1999.



(a) original network (b) LCPT (c) pruned LCPT (d) link v_4q_2 lies (e) pruned LCPT after lie

Fig. 5. A relay agent could lie down its cost to improve its utility using Algorithm 6.

[10] Weizhao Wang, Xiang-Yang Li, and Yu Wang, "Truthful multicast in selfish wireless networks," in *Proc. of ACM MobiCom*, pp. 402–413, 2004.

[11] Shai Herzog, Scott Shenker, and Deborah Estrin, "Sharing the cost of multicast trees: an axiomatic analysis," in *ACM SigComm*. 1995, pp. 315–327.

[12] Ron Cocchi, Scott Shenker, Deborah Estrin, and Lixia Zhang, "Pricing in computer networks: motivation, formulation, and example," *IEEE/ACM Trans. Netw.*, vol. 1, no. 6, pp. 614–627, 1993.

[13] Micah Adler and Dan Rubenstein, "Pricing multicasting in more practical network models," in *ACM SODA*. 2002, pp. 981–990.

[14] Joan Feigenbaum, Arvind Krishnamurthy, Rahul Sami, and Scott Shenker, "Hardness results for multicast cost sharing," *Theor. Comput. Sci.*, vol. 304, no. 1-3, pp. 215–236, 2003.

[15] Joan Feigenbaum, Christos H. Papadimitriou, and Scott Shenker, "Sharing the cost of multicast transmissions," *Journal of Computer and System Sciences*, vol. 63, no. 1, pp. 21–41, 2001.

[16] Joan Feigenbaum, Arvind Krishnamurthy, Rahul Sami, and Scott Shenker, "Approximation and collusion in multicast cost sharing (abstract)," in *Proc. of ACM Economic Conference*, pp. 253–255, 2001.

[17] Noam Nisan and Amir Ronen, "Algorithmic mechanism design," in *Proc. of ACM STOC*, 1999, pp. 129–140.

[18] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker, "A BGP-based mechanism for lowest-cost routing," in *ACM PODC*, 2002, pp. 173–182.

[19] Luzi Anderegg and Stephan Eidenbenz, "Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents," in *ACM MobiCom*. 2003, pp. 245–259.

[20] Weizhao Wang and Xiang-Yang Li, "Low-Cost Routing in Selfish and Rational Wireless Ad Hoc Networks," *IEEE Trans. on Mobile Computing* vol. 5, no. 5, pp. 596–607, 2006.

[21] Kamal Jain and Vijay V. Vazirani, "Applications of approximation algorithms to cooperative games," in *ACM STOC*, 2001, pp. 364–372.

[22] Shuchi Chawla, David Kitchin, Uday Rajan, R. Ravi, and Amitabh Sinha, "Profit maximizing mechanisms for the extended multicasting games," Tech. Rep. CMU-CS-02-164, Carnegie Mellon University, July 2002.

[23] J. D. Hartline A. Fiat, A. V. Goldberg and A. R. Karlin, "Competitive generalized auctions," in *ACM STOC*. 2002, pp. 72–81.

[24] Lavy Libman and Ariel Orda, "Atomic resource sharing in noncooperative networks," *Telecommunication Systems*, vol. 17, no. 4, pp. 385–409, 2001.

[25] Yunhao Liu, Li Xiao, and Lionel M Ni, "Building a Scalable Bipartite P2P Overlay Network," IEEE TPDS, Vol. 18, No. 9, 2007, Pages 1296-1306.

[26] Yunhao Liu, Li Xiao , Xiaomei Liu, Lionel M Ni, and Xiaodong Zhang, "Location Awareness in Unstructured Peer-to-Peer Systems," IEEE TPDS, Vol. 16, No. 2, 2005, Pages 163-174.

[27] Herve Moulin and Scoot Shenker, "Strategyproof sharing of submodular costs: Budget balance versus efficiency," *Economic Theory*, vol. 18, no. 3, pp. 511–533, 2001.

[28] S. Shenker, D. Clark, E. Estrin, and S. Herzog, "Pricing in computer networks: Reshaping the research agenda," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 2, pp. 19–43, 1996.

[29] Shai Herzog, Scoot Shenker, and Deborah Estrin, "Sharing the cost of multicast trees: An axiomatic analysis," *IEEE/ACM Transactions on Networks*, vol. 5, no. 6, pp. 847–860, 1997.

[30] Juniper Networks, *Multiprotocol Extensions for BGP-4*, RFC 2858.

[31] Cisco Systems, *A Border Gateway Protocol 4 (BGP-4)*, RFC 1771.

[32] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games*, pp. 31–40. Princeton University Press, 1953.

APPENDIX

Theorem 1 *Given a method $\mathcal{O}$ constructing a multicast topology, there exists a payment $\mathcal{P}$ such that $M = (\mathcal{O}, \mathcal{P})$ is truthful if and only if $\mathcal{O}$ satisfies the MNP.*

Proof: We first prove that if there exists a truthful payment $\mathcal{P}$ based on $\mathcal{O}$ then $\mathcal{O}$ satisfies the MNP. For the sake of contradiction, we assume that there is a truthful payment scheme $\mathcal{P}$ and $\mathcal{O}$ that does not satisfy MNP. From the definition of MNP, there exists an agent i and two cost vectors $c|^i c_{i_1}$ and $c|^i c_{i_2}$ with $c_{i_1} < c_{i_2}$ such that $\mathcal{O}_i(c|^i c_{i_2}) = 1$ and $\mathcal{O}_i(c|^i c_{i_1}) = 0$. Let $\mathcal{P}_i(c|^i c_{i_1}) = p_i^0$ and $\mathcal{P}_i(c|^i c_{i_2}) = p_i^1$.

Consider a network with a cost vector $c|^i c_{i_1}$, the utility for the 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 $\mathcal{P}$ is truthful, we have $p_i^0 \geq p_i^1 - c_{i_1}$.

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^0. Since payment scheme $\mathcal{P}$ is truthful, $p_i^0 \leq p_i^1 - c_{i_2}$.

Thus, we have $p_i^1 - c_{i_2} \geq p_i^0 \geq p_i^1 - c_{i_1}$, which implies that $c_{i_1} \geq c_{i_2}$. It is a contradiction to $c_{i_1} < c_{i_2}$.

We then prove that if $\mathcal{O}$ satisfies MNP, there exists a truthful mechanism $M = (\mathcal{O}, \mathcal{P})$. We prove it by constructing the payment scheme $\mathcal{P}$ shown in Algorithm 1 .

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

Case 1: Agent i lies its cost upward to $\overline{c_i}$ or downward to $\underline{c_i}$, but it does not change the output whether agent i is selected or not. Notice that, for a fixed c_{-i}, when the output of agent i does not change, its payment is the same. Thus, agent i's utility remains the same, implying 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 $\overline{c_i}$ such that it is not selected. From the property of MNP, 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 $\overline{c_i}$, 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 $\underline{c_i}$ 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. ■

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* truthful payment scheme.

Lemma 6 *The total payment $\mathcal{P}(R,d)$ to tree LCPT is nondecreasing and submodular with respect to receiver set R.*

Proof: By the definition of LCPT, obviously if $R \subset R' \subseteq Q$, then $\mathsf{LCPT}(d,R) \subseteq \mathsf{LCPT}(d,R')$. Remember the final payment to a relay agent v_k based on receiver set R is

$$p_k(R,d) = \max_{q_i \in R} p_k^i(d)$$

Observe that $p_k^i(d)$ is not affected by the receiver set R. Thus, for any relay node v_k, if $R \subset R' \subseteq Q$ then $p_k(R,d) \le p_k(R',d)$. Thus, the total payment to agents on tree $\mathsf{LCPT}(R,d)$ is nondecreasing.

We then prove that the total payment $\mathcal{P}(R,d)$ is a submodular function of set R, *i.e.*, $\forall R_1 \subseteq Q$ and $R_2 \subseteq Q$, $\mathcal{P}(R_1,d)+\mathcal{P}(R_2,d) \ge \mathcal{P}(R_1\cup R_2,d)+\mathcal{P}(R_1\cap R_2,d)$. Since $\mathcal{P}(R,d)=\sum_{v_k\in R} p_k(R,d)$, it is sufficient to prove that, $\forall k$,

$$p_k(R_1,d)+p_k(R_2,d) \ge p_k(R_1\cup R_2,d)+p_k(R_1\cap R_2,d).$$

We prove this by studying two cases whether the agent v_k is on $\mathsf{LCPT}(R_1\cap R_2,d)$ or not.

Case 1: Agent v_k is not on $\mathsf{LCPT}(R_1\cap R_2,d)$. Without loss of generality, assume that v_k is on $\mathsf{LCPT}(R_1 \setminus R_2,d)$. Then $p_k(R_2,d)=p_k(R_1\cap R_2,d)=p_k(R_2\setminus R_1,d)=0$. Consequently, $p_k(R_1\cup R_2,d) = \max_{q_i\in R_1\cup R_2} p_k^i(d) = \max_{q_i\in R_1} p_k^i(d) + \max_{q_i\in R_2\setminus R_1} p_k^i(d) = \max_{q_i\in R_1} p_k^i(d)$. Therefore, in this case we have

$$p_k(R_1,d)+p_k(R_2,d)=p_k(R_1\cap R_2,d)+p_k(R_1\cup xR_2,d).$$

Case 2: Agent v_k is on $\mathsf{LCPT}(R_1\cap R_2,d)$. Without loss of generality, assume $p_k(R_1,d)\le p_k(R_2,d)$. Thus,

$$\begin{aligned}
p_k(R_1\cup R_2,d) &= \max_{q_i\in R_1\cup R_2} p_k^i(d)\\
&= \max\{\max_{q_i\in R_2} p_k^i(d), \max_{q_i\in R_1\setminus R_2} p_k^i(d)\}\\
&\le \max\{\max_{q_i\in R_2} p_k^i(d), \max_{q_i\in R_1} p_k^i(d)\}\\
&= \max_{q_i\in R_2} p_k^i(d) = p_k(R_2,d)
\end{aligned}$$

On the other hand, we have $p_k(R_2,d)\le p_k(R_1\cup R_2,d)$. Thus, $p_k(R_2,d)=p_k(R_1\cup R_2,d)$. The fact $R_1\cap R_2\subseteq R_1$ implies $p_k(R_1\cap R_2,d)\le p_k(R_2,d)$. Thus, $p_k(R_1,d)+p_k(R_2,d)\ge p_k(R_1\cap R_2,d)+p_k(R_1\cup R_2,d)$. ■

Theorem 7: *Our payment sharing scheme defined in Algorithm 6 is the Shapely value.*

Proof: Remember Shapely value for multicast is

$$f_i(R) = \sum_{T\subseteq R\setminus q_i} \frac{|T|!(|R|-|T|-1)!}{|R|!}[\mathcal{P}(T\cup q_i,d)-\mathcal{P}(T,d)] \quad (4)$$

In other words, the Shapely value of the receiver q_i is $f_i(R)$ given a set of receivers R. Notice that an agent v_k will contribute to $\mathcal{P}(T\cup q_i,d)-\mathcal{P}(T,d)$ if and only if

1) Agent v_k is an upstream agent of receiver q_i.
2) $p_k^T(d) < p_k^i(d)$, where $p_k^T(d)=\max_{q_j\in T} p_k^j(d)$.

For fixed T, agent v_k satisfying above two criteria will add non-negative value $p_k^i(d)-p_k^T(d)$ to $\mathcal{P}(T\cup q_i,d)-\mathcal{P}(T,d)$. Let $T_{=x}$ be a receiver set with the highest rank in σ that is exactly x. Similarly, we use $T_{<x}$ to denote a receiver set with the highest rank in σ that is less than x. Let $g_k^i(R)$ be payment to agent v_k that is shared by receiver q_i. Assume that q_i is ranked a in the ranking σ when sorting the payment to agent v_k in a increasing order. Then

$$\begin{aligned}
g_k^i(R) &= \sum_{T_{<a}\subseteq R\setminus q_i} \frac{|T_{<a}|!(|R|-T_{<a}-1)!}{|R|!}\cdot p_k^i(d)\\
&\quad -\sum_{x=0}^{a-1}\sum_{T_{=x}\subseteq R-q_i} \frac{|T_{=x}|!(|R|-|T_{=x}|-1)!}{|R|!}\cdot p_k^{\sigma_x}(d).
\end{aligned}$$

Let γ be the number of receivers who are not the downstream receivers of v_k. Simplifying the first part of the equation, we get

$$\begin{aligned}
&\sum_{T_{<a}\subseteq R-q_i} \frac{|T_{<a}|!(|R|-T_{<a}-1)!}{|R|!}\cdot p_k^i(d)\\
&= p_k^i(d)\cdot \sum_{x=0}^{\gamma+a-1} \frac{x!(|R|-x-1)!}{|R|!}\cdot \binom{a+\gamma-1}{x}\\
&= \frac{p_k^i(d)}{|R|-a-\gamma+1} = \frac{p_k^i(d)}{|R(v_k)|-a+1}.
\end{aligned}$$

Simplifying the second part of the equation, we get

$$\begin{aligned}
&\sum_{x=0}^{a-1}\sum_{T_{=x}\subseteq R-q_i}\left(\frac{|T_{=x}|!(|R|-|T_{=x}|-1)!}{|R|!}\cdot p_k^{\sigma_x}(d)\right)\\
&= \sum_{x=0}^{a-1}\left(p_k^{\sigma_x}(d)\cdot \sum_{y=0}^{x+\gamma-1}\frac{(y+1)!(|R|-y-2)!}{|R|!}\cdot\binom{x+\gamma-1}{y}\right)\\
&= \sum_{x=1}^{a-1}\frac{p_k^{\sigma_x}(d)}{(|R|-x-\gamma+1)\cdot(|R|-x-\gamma)}\\
&= \sum_{x=1}^{a-1}\frac{p_k^{\sigma_x}(d)}{(|R(v_k)|-x+1)\cdot(|R(v_k)|-x)}\\
&= \sum_{x=1}^{a-1} p_k^{\sigma_x}(d)\cdot\left(\frac{1}{(|R(v_k)|-x)}-\frac{1}{(|R(v_k)|-x+1)}\right)\\
&= \frac{p_k^{\sigma_{a-1}}(d)}{(|R(v_k)|-a+1)} - \sum_{x=1}^{a-1}\frac{p_k^{\sigma_x}(d)-p_k^{\sigma_{x-1}}(d)}{(|R(v_k)|-x+1)}.
\end{aligned}$$

Combining the above two equations, then $g_k^i(R)$ equals to

$$\begin{aligned}
&\frac{p_k^i(d)}{|R(v_k)|-a+1} - [\frac{p_k^{\sigma_{a-1}}(d)}{(|R(v_k)|-a+1)}\\
&-\sum_{x=1}^{a-1}\frac{p_k^{\sigma_x}(d)-p_k^{\sigma_{x-1}}(d)}{(|R(v_k)|-x+1)}] = \sum_{x=1}^{a}\frac{p_k^{\sigma_x}(d)-p_k^{\sigma_{x-1}}(d)}{(|R(v_k)|-x+1)}
\end{aligned}$$

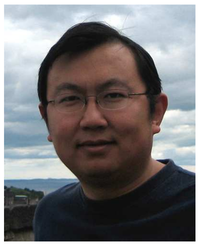
It shows that the sharing $f_k^i(R)$ computed in Algorithm 6 equals the sharing defined by the Shapely value. ■

Weizhao Wang received his BS and MS degree in computer science from Shanghai Jiaotong University, China, in 1999 and 2002 respectively, and Ph.D degree in computer science from Illinois Institute of Technology in 2006. He is currently with Google Inc. His current research interests include wireless networks, game theory, algorithm design, and next generation Internet.

TABLE I
NOTATIONS AND ABBREVIATIONS USED IN THIS PAPER.

V, E, c, c_i	The set of ASs, the set of links between ASs, the cost vector of ASs, and the cost of AS v_i
$G = (V, E, c)$	The network (a node-weighted graph)
$R = \{q_1, q_2, \cdots, q_r\}$, $s = q_0$	The set of multicast receivers and the source
$G' = (V, E, c\|^i c'_i)$ or $c\|^i c'_i)$	A new network if we change the cost of AS v_i to c'_i
$\mathsf{LCP}(u, v, c)$	The the least cost path from node u to node v in a network G
c_{-i}	The costs of all ASs other than AS v_i
$\mathcal{O}$, $\mathcal{P}$, ξ	The method of constructing a multicast topology, the payment scheme, the sharing scheme
$\kappa_i(\mathcal{O}, c_{-i})$	The threshold value such that if v_i is selected as a relay agent, then its cost is at most $\kappa_i(\mathcal{O}, c_{-i})$
$\mathcal{O}_i(c) = 1$ (or 0)	Agent v_i is selected (or not selected) to the multicast topology when the cost vector is c
d, d_i	The vector of declared cost of ASs, the declared cost of AS v_i
$p_i(d)$ or $p_i(R, d)$	The payment of agent v_i based on declared cost d
$\mathcal{P}(R, d) = \sum_i p_i(R, d)$	The total payment to all relay agents
$\xi_i(R, d)$	the sharing (or called charge) of a receiver q_i
$\xi(R, d) = \sum_{q_i \in R} \xi_i(R, d)$	The total payment collected from all receivers
$c^{(r)}$, $Q^{(r)}$, $c_i^{(r)}$, $d_i^{(r)}$, $\mathcal{U}^r$, $\mathcal{O}^r$	The cost vector, receiver set, cost of v_i, declared cost of v_i, updating rule, and selection method in round r
$\ell_r = \kappa_k(\mathcal{O}^r, c^r_{-k})$	The threshold value found in round r
$f_r(x)$	The cost for agent v_k in round r if the original cost is $c\|^k x$
$p_k^{i,j}(d)$ or $p_k^{i,j}$	The payment to node v_k for unicast from node v_i to node v_j
$p_k^{i,0}(d)$ or $p_k^{i,0}$, $p_k^i(d)$ or p_k^i	The payment to node v_k for traffic from the source q_0 to receiver q_i
$f_i^k(R, d)$ or $f_i^k(R)$	The sharing of the payment $p_k(d)$ to node v_k for a receiver q_i
LCPT, LST, SBT	Least cost path tree, link-weighted Steiner tree, share-based tree
MNP	Monotone Non-increasing Property
ELSD	Equal Link Split Downstream
BB, NNS, CM, NFR	Budget Balance, Nonnegative Sharing, Cross-Monotone, No-Free-Rider

Xiang-Yang Li (M'99, SM'08) has been an Associate Professor (since 2006) and Assistant Professor (from 2000 to 2006) of Computer Science at the Illinois Institute of Technology. He is a visiting professor of Microsoft Research Asia for one year from May, 2007. He also holds visiting professorship or adjunct-professorship at the following universities in China: TianJing University, WuHan University, and NanJing University. He received MS (2000) and PhD (2001) degree at Department of Computer Science from University of Illinois at Urbana-Champaign. He received the Bachelor degree at Department of Computer Science and Bachelor degree at Department of Business Management from Tsinghua University, China, both in 1995. His research interests span the wireless ad hoc networks, game theory, computational geometry, and cryptography and network security. He has published more than 120 papers in peer-reviewed journals and conferences. He served various positions (various chairs and TPC members) at numerous international conferences. He served as a co-chair of ACM FOWANC 2008 workshop, a co-chair of AAIM 2007 conference, a TPC co-chair of WTASA 2007, and TPC members of ACM MobiCom, ACM MobiHoc, IEEE INFOCOM, IEEE ICDCS, and so on. He has been invited to serve on the panel or review research proposals such as National Science Foundation (US), National Science Foundation of China, and RGC HongKong. He is an editor of Ad Hoc & Sensor Wireless Networks: An International Journal. He has served as a guest editor of IEEE JSAC and ACM MONET. He is a Senior Member of the IEEE, member of IEEE Communication Society, ACM, ACM Sigmobile.



Yu Wang received the PhD degree in computer science from Illinois Institute of Technology in 2004, the BEng degree and the MEng degree in computer science from Tsinghua University, China, in 1998 and 2000. He has been an assistant professor of computer science at the University of North Carolina at Charlotte since 2004. His current research interests include wireless networks, ad hoc and sensor networks, mobile computing, and algorithm design. He has published more than 60 papers in peer-reviewed journals and conferences. He has served as program chair, publicity chair, and program committee member for several international conferences. He is the program co-chair of the first ACM International Workshop on Foundations of Wireless Ad Hoc and Sensor Networking and Computing (FOWANC 2008), and was the program co-chair of the 26th IEEE International Performance Computing and Communications Conference (IEEE IPCCC 2007). Dr. Wang is an editorial board member of the International Journal of Ad Hoc and Ubiquitous Computing, and an associate editor of the International Journal of Mobile Communications, Networks, and Computing. Dr. Wang is a recipient of Ralph E. Powe Junior Faculty Enhancement Awards from Oak Ridge Associated Universities. He is a member of the ACM, IEEE, and IEEE Communications Society.

Zheng Sun (M'04) received the B.S. degree in computer science from Fudan University, Shanghai, China, in 1995, and the M.S. and Ph.D. degrees in computer science from Duke University, Durham, NC, in 1998 and 2003, respectively. From 1999 to 2002, he was with Microsoft Corporation and Perfect Commerce as a Software Developer. In 2003, he was with the Hong Kong Baptist University, Hong Kong, where he was an Assistant Professor of computer science. He has been with Google Inc., Mountain View, CA, since 2005. His research interests include robotic motion planning, computational geometry, and data mining and its applications in e-commerce.