Strategyproof Mechanisms for Content Delivery via Layered Multicast

Ajay Gopinathan, Zongpeng Li

Strategyproof Mechanisms for Content Delivery via Layered Multicast

Ajay Gopinathan and Zongpeng Li

Department of Computer Science, University of Calgary
{ajay.gopinathan, zongpeng}@ucalgary.ca

Abstract. Layered multicast exploits the heterogeneity of user capacities, making it ideal for delivering content such as media streams over the Internet. In order to maximize either its revenue or the total utility of users, content providers employing layered multicast need to carefully choose a routing, layer allocation and pricing scheme. We study algorithms and mechanisms for achieving either goal from a theoretical perspective. When the goal is maximizing social welfare, we prove that the problem is NP-hard, and provide a simple 3-approximation algorithm. We next tailor a payment scheme based on the idea of critical bids to derive a truthful mechanism that achieves a constant fraction of the optimal social welfare. When the goal is *revenue maximization*, we first design an algorithm that computes the revenue-maximizing layer pricing scheme, assuming truthful valuation reports. This algorithm, coupled with a new revenue extraction procedure for layered multicast, is used to design a randomized, strategyproof auction that elicits truthful reports. Employing discrete martingales to model the auction, we show that a constant fraction of the optimal revenue can be guaranteed with high probability. Finally, we study the efficacy of our algorithms via simulations.

1 Introduction

Data dissemination applications such as media streaming, file downloading and video conferencing represent an increasingly significant fraction of today's Internet traffic[1]. A natural protocol for one-to-many content delivery is *multicast* [2, 3], which reduces redundant transmissions and utilizes network bandwidth efficiently. The Internet is an inherently heterogeneous 'network of networks' with diverse connection types, and therefore disparate download capabilities among its users. As a result, single-rate multicast lacks the required flexibility to cater simultaneously to the needs of every user. *Layered multicast* [4–6] offers an attractive solution by encoding the media content into layers with varying sizes: a base layer provides basic playback quality, and improvement is possible through further reception of a flexible number of enhancement layers. Users are then able to enjoy playback quality commensurate with their download capacities [4–6].

The Internet contains private entities driven by selfish and often commercial interests. Content providers are thus compelled to include social and economic considerations when computing an appropriate multicast scheme. Previous studies on layered multicast have focused on optimizing metrics such as throughput

[5, 6], and do not consider the potential for *strategic behaviour* by users. Users subscribing to media content via layered multicast have different *valuations* for receiving the service[7, 8], which is naturally proportional to the number of layers received. However, such valuation is private to the user itself, who may strategically misreport its value for potential economic benefit, *e.g.* in the hopes of receiving the multicast service with a lower payment. It is critical for content providers to take such strategic behaviour into consideration when routing and pricing the layers. An altruistic content provider aims to maximize *social welfare*, *i.e.*, the sum of the user valuations for layers received. A commercially motivated content provider wishes to maximize its own *revenue* instead. The challenge for the content provider in either case is to determine simultaneously a *layer allocation, routing* and *pricing* scheme that approaches the stated goal, while ensuring users have no incentive to misreport their valuations.

In this paper, we provide a theoretical study on the classical economic problems of maximizing social welfare and revenue in the layered multicast setting. We adopt a *network information flow* approach to model layered multicast, and design algorithms and mechanisms for maximizing either social welfare or revenue, all the while treating multicast receivers as *strategic agents*. We focus on the common *cumulative layering* scheme, where decoding layer k requires the successful reception of layers 1 through k. Our solutions are both *efficiently computable* and provably *strategyproof*. A mechanism is strategyproof if users have no incentive to lie about their true valuations of the multicast service to the content provider.

We first show that the social welfare maximization problem for layered multicast is NP-Hard. Nevertheless, we design a 3-approximation algorithm for the problem. Our algorithm exploits network coding [9] to find a layer allocation that simultaneously maximizes social welfare and guarantees a feasible routing scheme. We next focus on the problem of eliciting truthful valuation reports for maximizing social welfare. It turns out that the direct application of the well known VCG mechanism [10–12] is not strategyproof. We design a payment scheme built on the technique of finding the minimum *critical bid* capable of changing the outcome of the approximation algorithm, and proceed to prove that the resulting mechanism using this payment scheme is truthful.

We next focus on designing mechanisms that maximize revenue for content providers. The first challenge here is computing an appropriate layer pricing scheme for maximizing revenue. We show that a greedy algorithm that computes the best price for layer k, under the assumption that this same price will be used for all layers $k' \geq k$, is indeed optimal. The next challenge is to design a mechanism to elicit user valuations truthfully. We focus on the more realistic *prior-free* setting [13], when no information on the distribution of user valuations is known *a priori*. We first prepare a new *revenue extraction* procedure tailored to the layered multicast setting. This procedure, coupled with the revenue-maximization pricing algorithm, are used as ingredients in a strategyproof, randomized auction first proposed by Goldberg *et al.* [13]. We model the auction using discrete martingales [14], and show that one can achieve a

δ-fraction of the optimal revenue with probability at least $1 - 2\exp(-\frac{(1-2\delta)^2}{2\alpha})$, where α is the ratio of the maximum contribution by any agent to the optimal revenue.

Due to space constraints, we omit a number of technical results and their corresponding proofs from this paper. The interested reader is urged to examine the full paper [15] for further details. The rest of this paper is organized as follows – we discuss related work in Sec. 2, and introduce our model and notations in Sec. 3. Sec. 4 and Sec. 5 study social welfare maximization, Sec. 6 studies revenue maximization. Simulations results are presented in Sec. 7 before we conclude the paper.

2 Previous Research

Traditionally, computing an optimal multicast routing scheme is modeled as optimal Steiner tree packing, a well-known NP-Hard problem [2, 3]. The seminal work of Ahlswede *et al.* [9] on *network coding* dramatically changed the landscape of multicast algorithm design. Exploiting the fact that information can be both replicated and encoded, network coding leads to efficient polynomial time algorithms for optimal multicast routing[16, 17].

The field of mechanism design originates from economic theory, where the goal is to implement a desired social choice in the presence of agents that behave strategically [8]. The seminal work of Vickrey in auction design [10] and Clarke's pivot payment rule [11] pioneered research on strategyproof mechanism design, which culminated in the general VCG mechanism as presented by Groves [12]. The VCG mechanism is the best known strategyproof method for maximizing social welfare, but in general fails to be truthful when the social welfare maximization problem is NP-Hard and approximate solutions are used [7].

Revenue maximizing mechanisms are known as optimal auctions in the parlance of economic theory [8]. Classic literature in such auction mechanism design assume that user valuations are drawn from a probability distribution known to the auctioneer [18]. The seminal work of Myerson [18] showed that applying the VCG [10–12] mechanism using *virtual valuations* of users yields a mechanism that is revenue-maximizing. From a computer science perspective however, the *prior-free* setting where such distribution information is not assumed is more realistic yet also more challenging. The work by Goldberg *et al.*[13] show that deterministic, revenue-maximizing auctions that are symmetric (i.e., independent of any ordering on agents) and strategyproof do not exist. Consequently, they design a randomized, strategyproof auction that guarantees a constant fraction of the optimal revenue. Borgs *et al.* [19] further consider budget-constrained agents, and show that when budgets are private information, the VCG scheme is not truthful; they design a randomized revenue-maximizing auction instead.

3 Preliminaries

We model the network as a directed graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each edge $\vec{uv} \in \mathcal{E}$ has a finite capacity $C(\vec{uv})$. A distinguished source node $s \in V$ provides a multicast streaming service. The media is encoded into K layers, where the size of layer $k \in [1..K]$ is l_k. We will focus on the most commonly used layering technique – the *cumulative layering* scheme [20, 21], where decoding layer k requires the use of layers 1 through $k-1$ as well.

Let $\mathcal{T}$ denote the set of users or *agents* in the network potentially interested in this multicast service. We will take the *network flow* approach to modeling the multicast routing scheme. We employ network coding within each data layer, to enable polynomial time computability of the optimal flow for a given layer. Our model for computing multicast flows is based on a classic result on multicast network coding, which states that *a multicast rate of d is feasible if and only if it is a feasible unicast rate to each receiver* [9]. This allows us to view the optimal flow within a network coded layer as the union of *conceptual unicast flows* [17, 16] to each receiver, where these flows do not compete for bandwidth.

For an agent $i \in \mathcal{T}$, let $t_i \in V$ be the corresponding node in the network. We assume that an agent i is willing to pay $v_i \in \mathbb{Z}_+$ monetary units for receiving a multicast layer, where without loss of generality, v_i is assumed to be of type integer. The value v_i is private information known only to i. While we plan to consider more complex valuation functions in the future, the present work nonetheless provides valuable insight on the design of strategyproof mechanisms for layered multicast. An agent i may be charged a price p^k for receiving a layer k. Let $\mathbf{x}$ be a 2D matrix with each entry x_i^k indicating whether agent i is allocated layer k or not. We will also use $\mathbf{x}_i$ when referring to the layer allocation for agent i, while $\mathbf{x}^k$ denotes the allocation of layer k for all agents. If an agent receives up to k' layers and is charged p^k for every layer received, then its overall *utility function* is given as $u_i = v_i k' - \sum_{k=1}^{k'} p^k$. Each agent is assumed to be *selfish* and *rational*, and behaves strategically with the aim of maximizing its utility function. The utility of the content provider on the other hand is simply the sum of all payments received.

A mechanism is essentially a protocol that implements some desired social choice function. We will focus on *revelation mechanisms* [8], where the only strategy available to each agent in the network is to declare its valuation for receiving a data layer. In this case, the mechanism reduces to the well known auction problem, and hence we will also refer to the value declared by agent i as the bid b_i. Let $\mathbf{b}$ denote the bid vector of all agents, and denote by $\mathbf{b}_{-i}$ the bid vector of all agents but i. For a given bid vector, the utility of agent i when bidding b_i will be denoted as $u_i(b_i, \mathbf{b}_{-i})$. Our goal is to design mechanisms that either maximize *social welfare* or *revenue*. The social welfare of a mechanism is simply the sum of the utilities of all agents in the system, including the mechanism designer (this case, the content provider), which is equivalent to the sum of all user valuations since payments and revenue cancel each other. A revenue-maximizing mechanism or auction maximizes the payments collected from all agents in the system.

The seminal work of Myerson [18] paved the way for strategyproof revenue-maximizing auctions, under the assumption that agent valuations were drawn from distributions known to the auctioneer. In contrast, recent work in the computer science community [13, 19, 8] consider revenue-maximizing auctions when this information is unavailable. We will focus on this latter, *prior-free* setting as well in this paper.

A mechanism is said to be *strategyproof* if the dominant strategy of every agent is to bid its true valuation, regardless of the bids submitted by other agents, *i.e.*, $u_i(v_i, \mathbf{b}_{-i}) \geq u_i(b_i, \mathbf{b}_{-i}), \forall b_i \neq v_i, \forall \mathbf{b}_{-i}$. The following characterization of truthfulness, due to Myerson [18], will be particularly useful:

Lemma 1 ***[Myerson, 1981]*** *Let $x_i(b_i)$ be the allocation function for bidder i with bid b_i. A mechanism is strategyproof if and only if the following hold for a fixed $\mathbf{b}_{-i}$:*

- *$x_i(b_i)$ is monotonically non-decreasing in b_i*
- *Bidder i bidding b_i is charged $b_i x_i(b_i) - \int_0^{b_i} x_i(z)dz$*

Observe that the payment function is completely determined by the allocation function, and vice versa for a fixed $\mathbf{b}_{-i}$. This implies two equivalent methods of viewing a truthful mechanism: (i) there exists a *critical bid* b_i' that depends only on $\mathbf{b}_{-i}$, such that if i bids at least b_i', then i is allocated the item, or (ii), the payment of every agent should not depend on its bid b_i. We will use the first point of view to design strategyproof mechanisms that achieve efficiency in Sec. 5, while the second perspective will be useful when we design truthful revenue maximization mechanisms in Sec. 6.

4 The Social Welfare Maximization problem

In the *social welfare maximization* (SWM) problem, we seek a feasible multicast routing and layer allocation scheme that maximizes the sum of the utilities for all agents and the utility of the service provider. We can formulate SWM as the following linear integer program (IP):

$$\text{Maximize} \sum_k \sum_i v_i x_i^k \tag{1}$$

Subject To:

$$\sum_{v \in N(u)} [f_i^k(\overrightarrow{uv}) - f_i^k(\overrightarrow{vu})] = 0 \qquad \forall k, \forall i, \forall u$$

$$f_i^k(\overrightarrow{uv}) \leq f^k(\overrightarrow{uv}) \qquad \forall k, \forall i, \forall \overrightarrow{uv}$$

$$\sum_k f^k(\overrightarrow{uv}) \leq C(\overrightarrow{uv}) \qquad \forall \overrightarrow{uv}$$

$$x_i^{k+1} \leq x_i^k \leq \frac{f_i^k(\overrightarrow{t_i s})}{l^k} \qquad \forall k = 1..K-1, \forall i$$

$$f_i^k(\overrightarrow{uv}), f^k(\overrightarrow{uv}) \geq 0; x_i^k \in \{0, 1\} \quad \forall k, \forall i, \forall \overrightarrow{uv}$$

Since the payment by the agents cancels the content provider's utility, both terms can be ignored in the objective function. We use the flow vector f to denote the multicast flow. The variable $f_i^k(\overrightarrow{uv})$ indicates the *conceptual flow* [17, 16] to agent t_i carrying layer k, on edge $\overrightarrow{uv}$. Similarly, the variable $f^k(\overrightarrow{uv})$ indicates

Algorithm 1: Approximation algorithm for `SWM`

Input: Set of agents $\mathcal{T}$, network $\mathcal{G}$
Output: A 3-approximate multicast routing and layer allocation scheme, $\mathbf{x}$

Initialize $x_i^k := 0, s(i) := 0 \quad \forall i, \forall k$;
feasible := True ;
while *feasible* **do**
  Compute max-flow values m_i for each agent i ;
  $k_i := \max_{k'} | \sum_{k=s(i)}^{k'} l_k \leq m_i \quad \forall i$;
  if $k_i = 0$ *for all* i **then**
    feasible:= False ;
  else
    foreach $k \in 1 \dots K$ **do**
      $\mathcal{W}(k) := \emptyset$;
      foreach $t_i \in \mathcal{T}$ **do**
        if $k_i \leq k$ **then**
          $\mathcal{W}(k) := \mathcal{W}(k) \cup i$;
    foreach $k \in 1 \dots K$ **do**
      $S(k) := \sum_{i \in \mathcal{W}(k)} v_i k$;
    $W(k') := \max_k S(k)$;
    foreach $i \in W(k')$ **do**
      $x_i^k := 1 \quad \forall k = 1 \dots k'$; $s(i) := k' + 1$;
    Solve LP degradation of Eq. (1) on $\mathcal{G}$ with $\mathbf{x}$;
    Set $\mathcal{G}$:=Residual network of $\mathcal{G}$;

the actual flow on edge $\overrightarrow{uv}$. For succinctness, the above formulation assumes a virtual, uncapacitied directed edge from each $t_i \in T$ to s. In the first constraint, $N(u)$ denotes the set of u's neighbours in $\mathcal{G}$. This constraint ensures the conceptual unicast flow is conserved at all nodes. The second constraint captures the notion that the true flow on an edge for a given layer k is the maximum of all conceptual flows on that edge carrying layer k. The third requirement ensures that capacity constraints are respected on all edges. The final constraint models: (i) the cumulative layering scheme requirement, a node is able to play layer k only if layers 1 through $k-1$ has been obtained as well, and (ii) a layer can be decoded only if all its l_k bits have been received. However, even disregarding the routing dimension, the decision version of `SWM` is NP-Hard. The proof for this hardness result can be found in the full version of this paper [15].

We are thus motivated to design an approximation scheme for `SWM`. Algorithm 1 shows our approximation algorithm for the `SWM` problem. We first compute the individual max flows for each agent, to decide the maximum number of layers it can receive. We then create k sets, where the set $\mathcal{W}(k)$ contains all agents that can receive up to *at least* layer k. If the set $\mathcal{W}(k')$ yields the maximum social welfare, we set all agents in $\mathcal{W}(k')$ to receive up to k' cumulative layers. We are guaranteed that such a multicast scheme is feasible due to the classic result on multicast feasibility with network coding stated earlier. The next theorem shows that Algorithm 1 achieves a constant approximation ratio:

Theorem 2 *Alg. 1 is a 3-approximation algorithm for* `SWM`.

A proof of the above theorem can be found in the full version of this paper [15].

5 Strategyproof social welfare maximization

In this section, we design a strategyproof mechanism for content providers who wish to seek a layer allocation scheme that maximizes social welfare. Our goal is to ensure that the mechanism is both efficiently computable and strategyproof. A common approach here is to directly apply the celebrated Vickrey-Clarke-Groves (VCG) mechanism [10–12] in conjunction with Algorithm 1. We have shown that the latter is efficiently computable with at most a constant factor loss in the social welfare, while the former is a well known strategyproof mechanism. However, applying the VCG mechanism while using suboptimal algorithms can harm truthfulness [7], and we demonstrate that this is also the case with Algorithm 1. Please see the full paper for details [15].

Next, we design a payment scheme that makes Algorithm 1 strategyproof. The key insight is to ensure that each agent is made to pay a *critical bid* for every outcome selected during each iteration of Algorithm 1. Recall that the outcome of every feasible iteration of Algorithm 1 is a chosen set $\mathcal{W}(k)$, where all agents in this set are allocated up to layer k. If the absence of an agent i causes some other outcome $\mathcal{W}(k')$ to be selected by the algorithm, then this agent should made to pay the *minimum* bid required to ensure $\mathcal{W}(k)$ is selected ahead of $\mathcal{W}(k')$. Let $p_i(r)$ be the payment charged to agent i in iteration r of Algorithm 1. The payment $p_i(r)$ can be computed using the procedure shown in Algorithm 2, which is called during every iteration of Algorithm 1, for every agent i. The total payment of each agent is then the sum of all payments incurred over all feasible iterations of Algorithm 1, $p_i = \sum_r p_i(r)$. It can be shown that this payment scheme yields a truthful mechanism.

Theorem 3 *Using the payment scheme in Algorithm 2, bidding truthfully is a dominant strategy for all agents when Algorithm 1 is used to solve* `SWM`.

The proof for this theorem can be found in the full version of this paper [15].

6 Strategyproof Revenue Maximization

It is known that mechanisms that achieve optimal social welfare have poor revenue generating properties [8]. In this section, we will design strategyproof mechanisms for optimizing the revenue for the content provider. To attain this goal, we need to first find a pricing scheme that maximizes the content provider's revenue, assuming truthful valuation reports are given. We will focus on the case when the content provider chooses a fixed price per layer, in the interest of fairness. Once we know how to compute the optimal layer prices, the next challenge is to design a mechanism that is strategyproof. We will focus on the prior-free setting, when no Bayesian information on the valuations of the agents are known. Such an assumption is both realistic, and without loss of generality. It implies that any guarantee our mechanisms make apply for all distributions of user valuations.

Algorithm 2: Strategyproof Payment Scheme for Algorithm 1

Input: A chosen allocation $\mathcal{W}(k)$ in iteration r of Algorithm 1, set of agents $\mathcal{T}$, an agent i

Output: Payment charged to agent i for iteration r, $p_i(r)$

$\mathcal{T}' := \mathcal{T} \setminus i$;
Use Algorithm 1 to find allocation $\mathcal{W}(k')$ in round r for $\mathcal{T}'$;
if $k' < k$ **then**
  $\mathcal{H} := \mathcal{W}(k') \setminus \mathcal{W}(k)$;
  $p_i(r) := \sum_{j \in \mathcal{H}} v_j k'$;
else if $k' > k$ **then**
  $\mathcal{H} := \mathcal{W}(k) \cap \mathcal{W}(k')$;
  $p_i(r) := \sum_{j \in \mathcal{H}} v_j (k' - k)$;
else
  $p_i(r) := 0$;

6.1 Revenue-Maximizing Layer Prices

Let us first consider how to compute the optimal layer prices when agent valuations have been disclosed truthfully. First, observe that when there is only a single layer ($K = 1$), the optimal revenue maximizing layer price p is given as $p = \arg\max_{v_i} |\{v_l | v_l x_l \geq v_i\}|$. That is, the optimal price is given as the valuation v_i that maximizes the size of the set of agents whose valuations are *at least* v_i. One may be tempted to use this equation to find the best prices beginning from layer 1 through K. However, doing so ignores the cumulative layering requirement, and results in a revenue that has an unbounded gap from the optimal. For example, consider three agents with $v_1 = 3$ and $v_2 = v_3 = 1$, and $\mathbf{x}_1 = (1, 0)$, $\mathbf{x}_2 = \mathbf{x}_3 = (1, 1)$. Then the previously mentioned greedy scheme charges $p^1 = 3$ for layer 1, resulting in overall revenue of 3. The optimal pricing scheme is $p^1 = p^2 = 1$, resulting in a revenue of 5. It is easy to make this example arbitrarily bad. Instead, one needs to recursively optimize the layer prices greedily beginning at each layer, while considering the *potential revenue* from higher layers. This is precisely the idea behind Algorithm 3. The algorithm takes as input an algorithm A for computing the social welfare maximizing layer allocation. Hence, A can either be Algorithm 1, or an integer program solver used in conjunction with Eq. (1). Algorithm 3 computes the best price for layer k, under the assumption that this price will be used for all layers k and above. Once this price is set, we fix the allocation for layer k by assigning it only to agents whose valuation is at least the current price. This may result in some agents being "deallocated" layer k, which in turn leads to excess capacity in the network. Hence, we re-solve the welfare maximizing layer allocation for layer $k+1$. Algorithm 3 is optimal, the proof of which can be found in the full version of the paper [15].

Theorem 4 *Algorithm 3 computes revenue-maximizing prices with respect to the social welfare computed by A.*

Algorithm 3: Computing Revenue-Maximizing Layer Prices

Input: Set of agents $\mathcal{T}$, algorithm A for solving SWM, network $\mathcal{G}$
Output: Optimal layer price vector $\mathbf{p}$, revenue R

Initialize $k := 1$, $R := 0$;
while $k \leq K$ **do**
  Fixing $\mathbf{x}^1 \dots \mathbf{x}^{k-1}$, use A to compute allocation $\mathbf{x}^k \dots \mathbf{x}^K$ on $\mathcal{G}$;
  Let $v_i^l := v_i x_i^l \quad \forall i, \forall l = k \dots K$;
  $p^k := \arg\max_{v_j^k} |\{v_i^l | v_i^l \geq v_j^k \wedge l \geq k\}|$;
  $\forall i$ such that $v_i^k < p^k$, fix $x_i^k := 0$;
  Set $R := R + |\{v_i^k | v_i^k \geq p^k\}|$ and let $k := k + 1$;

Algorithm 4: ProfitExtract(R, $\mathcal{S}$, $\mathbf{w}$)

Input: Target revenue R, bidder set S with valuation $\mathbf{w}$
Output: Winner vector $\mathbf{y}$

Set price $p := \frac{R}{|S|}$
For each bidder $i \in S$ with $w_i < p$, remove i from S
If $S = \emptyset$, return failure
If no bidder i is removed in Step 2, set $y_i := 1$ for all $i \in S$. Return price p and winner vector $\mathbf{y}$
Otherwise repeat from step 1

6.2 A Randomized Revenue-Maximizing Mechanism

We have shown that if user valuations are known, then the optimal prices can be computed using Algorithm 3. The next challenge is to elicit user valuations truthfully. In designing a strategyproof revenue-maximizing mechanism, the key obstacle in the prior-free setting is the lack of information on the distribution of user valuations. We require that the mechanism can somehow "guess" an optimal price to be charged to an agent i. We know from Lemma 1 that if the "guessing" process does *not* use i's valuation, then the dominant strategy for i is to declare its valuation truthfully. This intuition naturally suggests a *sampling* based approach. We can pick a random sampling of agents, whose valuations can be used to compute a good price. This price is then offered to the other agents not in the sample. To ensure truthfulness, we ignore the potential revenue from the sampled agents. This technique was originally used by Goldberg *et al.* in a random sampling auction [13] for digital goods, and we apply it as a platform to build our own revenue-maximizing auction for layered multicast.

We begin by introducing some key ingredients that will be required in the final mechanism. The first is a *profit extraction algorithm* [13], shown in Algorithm 4. The algorithm takes as input a target revenue R, and a set of agents with the 1-dimensional valuation vector $\mathbf{w}$ for some item. Algorithm 4 essentially runs the Moulin-Shenker *tattonement* process, which attempts to whittle down the initial set of agents in S until the remaining agents can adequately afford to share equally the target revenue R. Note that R, and therefore the price p offered to each agent, is independent of i's valuation w_i, and the mechanism is therefore truthful. Let O be the optimal single price revenue that can be made

Algorithm 5: Profit Extraction with Layers

Input: Set of agents $\mathcal{S}$, allocation $\mathbf{x}$, target revenue R
Output: Layer price vector $\mathbf{p}$, winner vector $\mathbf{z}$

Initialize $r(k) := 0 \quad \forall k = 1 \dots K$;
Set $k := 0$;
while $R > 0$ **do**
  $k := k \mod K + 1$; $r(k) := r(k) + 1$; $R := R - 1$;
Set $k := K$;
while $k > 0$ **do**
  Set $next := 1$ and let $w_k := v_i x_i^k$;
  while *True* **do**
    Run ProfitExtract($r(k)$, $\mathcal{S}$, $\mathbf{w}$);
    if *ProfitExtract returns failure* **then**
      $r(k) := r(k) - 1$;
      if $r(k) = 0$ or $k = 1$ **then** break ;
      $r(next) := r(next) + 1$;
      $next := next \mod (k-1) + 1$;
    else
      Let $\mathbf{y}$ and p be the winner vector and price respectively as returned by ProfitExtract ;
      Set $z_i^k := y_i \quad \forall i \in \mathcal{S}$, and let $p[k] := p$;
      break ;
  $k := k - 1$;

Algorithm 6: Competitive Auction

Input: Set of agents, $\mathcal{S}$
Output: Total revenue R

Randomly assign bidders in $\mathcal{S}$ to one of two sets, $\mathcal{A}$ or $\mathcal{B}$
Use Algorithm 3 to compute optimal price and revenue for sets A and B
Let R_A and R_B denote the revenue from sets $\mathcal{A}$ and $\mathcal{B}$ respectively
Run Algorithm 5 on $\mathcal{A}$ (resp. $\mathcal{B}$) using target revenue R_B (resp. R_A)
Return revenue gained from running Algorithm 5 successfully

from S. Then observe that the mechanism is able to successfully find a set of agents to share the revenue R if and only if $R \leq O$.

We wish to generalize the design to a truthful profit extraction mechanism when valuations are multi-valued instead of single-valued, as is the case in Algorithm 4. Further, due to the cumulative layering scheme, we must ensure that if an agent is priced out of layer k, no revenue can be extracted from it for all layers $k' > k$. Our profit extraction scheme for agents with valuations for layers is shown in Algorithm 5. The algorithm again takes a target revenue R, and attempts to share them among agents in the set S. The allocation vector $\mathbf{x}$ is provided by a social welfare maximizing algorithm, and ensures that agents are only considered for layers which they can receive in the network. The algorithm utilizes the fact that the revenue extracted from any layer k cannot exceed the revenue from any layer $k' < k$. This follows from (a) agents have the same valuations for all layers, and (b) cumulative layering implies that the number of

agents who can receive layer k does not exceed the number of agents that can receive layer $k' < k$. As a result, Algorithm 5 begins by distributing R among all layers, while favouring lower layers. Beginning at layer $k = K$, it attempts to extract the target revenue for each layer $r(k)$ using Algorithm 4 with the valid bids in layer k. If this fails, it then reduces and redistributes $r(k)$ by 1 — since valuations are integral, the revenue available from each layer is integral. Once again, since the prices offered at each layer to each agent is independent of its bid, Algorithm 5 is truthful. Further, if the revenue-maximizing pricing scheme for S generates a revenue of O and $R \leq O$, Algorithm 5 will find a successful allocation of prices.

We have now developed all the necessary machinery to design a strategyproof, revenue-maximizing mechanism. The full mechanism is shown in Algorithm 6. The algorithm randomly splits agents into two sets, A and B, and computes the optimal revenue R_A and R_B from each set. It then attempts to extract R_A from agents in B and vice versa. Since the target revenue for each set in Algorithm 5 is independent of the bids of agents in that set, this algorithm is strategyproof. We are guaranteed a revenue of $R = \min(R_A, R_B)$. It remains now to analyze how R compares to the optimal revenue when all valuations are known exactly, which is *at most* $R_A + R_B$.

Let $p^*(k)$ be the optimal layer price for layer k, and OPT be the optimal revenue when all agent valuations are known. In the optimal solution, let $w(k)$ be the number of agents who contribute to OPT in each layer $1 \dots k$. Therefore, we get $OPT = \sum_k w(k)\Big(\sum_{l=1}^{k} p^*(l)\Big)$. After the random splitting process, let $w_A(k)$ and $w_B(k)$ be the number of agents who end up in sets A and B respectively, who contribute to OPT up to k layers. Denote by $M = \sum_{k=1}^{K} p^*(k)$ Then M represents the highest possible contribution to OPT by any one agent. The parameter $\alpha = \frac{M}{OPT}$, is a measure of the ratio of the maximum contribution of any one agent to the optimal revenue. We will show that the performance of our mechanism hinges on α.

Theorem 5 *For $0 < \delta < 0.5$, Algorithm 6 achieves a revenue of δOPT with probability at least $1 - 2\exp\Big(-\frac{(1-2\delta)^2}{2\alpha}\Big)$*

Proof. First, observe that $R_A \geq \sum_k w_A(k)\Big(\sum_{l=1}^{k} p^*(l)\Big)$, while a similar revenue guarantee holds for B. Since the revenue of Algorithm 6 is $\min(R_A, R_B)$, to achieve a revenue of at least δOPT, then we must have $|R_A - R_B| \leq (1-2\delta)OPT$. Now, let X_i be the random variable which takes value 1 if agent i ends up in the set A during the random splitting process, and -1 if it ends up in B. Let t_i be the revenue contributed by i to OPT, and so we have $t_i \leq M$, for all i. Impose some arbitrary ordering on the set of agents, and define the random variable $Y_j = \sum_{i \leq j} X_i t_i$, with $Y_0 = 0$. We then have that in expectation:

$$E[Y_{j+1}|Y_1 \dots Y_j] = Y_j + E[X_{j+1}t_{j+1}] = Y_j + \frac{1}{2}t_{j+1} + \frac{1}{2}(-t_{j+1}) = Y_j$$

That is, the random variable Y_j forms a *martingale sequence* with respect to itself. If there are a total of n agents, then $|Y_n - Y_0| = |Y_n| = |R_A - R_B|$.

Further, we have $|Y_{j+1} - Y_j| \leq M$, for all $j \leq n$. Hence we can bound the probability of the event $|R_A - R_B| \geq (1-2\delta)OPT$ using Azuma's inequality [14] as the following:

$$Pr(|Y_n| \geq (1-2\delta)OPT) \leq 2\exp\Big(-\frac{(1-2\delta)^2 OPT^2}{2\sum_{j=1}^{n} M^2}\Big) = 2\exp\Big(-\frac{(1-2\delta)^2 OPT^2}{2nM^2}\Big)$$
$$\leq 2\exp\Big(-\frac{(1-2\delta)^2 OPT}{2M}\Big) = 2\exp\Big(-\frac{(1-2\delta)^2}{2\alpha}\Big)$$

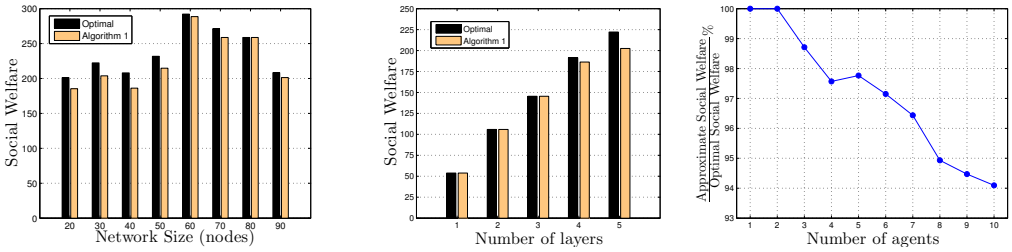
The first inequality is due to Azuma's concentration result on martingales with bounded differences [14], while the second comes about since $nM \leq OPT$.

We need to ensure strategyproofness is maintained at every stage of our mechanism. Hence, we use a strategyproof pricing scheme in conjunction with the algorithm A in Algorithm 3, and let $\mathbf{q}$ be the payment vector. We then run Algorithm 6 as described, and let $\mathbf{r}$ be the vector of revenue obtained from each agent. For each agent i in the winning set (either A or B), we charge $\max(q_i, r_i)$. It is easy to see that the entire mechanism is both strategyproof, and revenue-maximizing.

7 Simulations

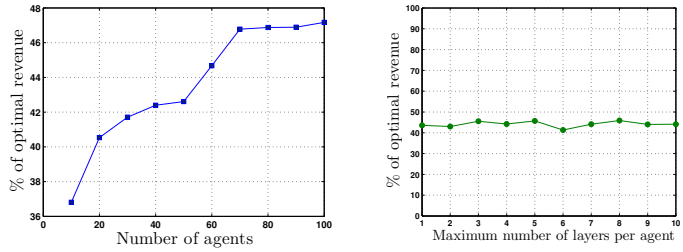
We now present results from simulations performed to determine the effectiveness of our techniques in practical scenarios. We first studied the performance of Algorithm 1. We used BRITE [22] to generate random network topologies. Link capacities were distributed randomly to model the heterogeneous nature of the network. The multicast group was chosen randomly as well. For each topology, layer sizes were generated randomly between 1 and 5. Fig. 1a shows the social welfare achieved by Algorithm 1 and the optimal solution of Eq. (1) as computed by an integer linear program solver, for varying network sizes with 10 agents and 5 layers. In all cases, Algorithm 1 achieves at least 90% of the optimal social welfare. We next examined the effect of the number of layers on the performance of Algorithm 1. As can be seen in Fig. 1b, the greedy technique of Algorithm 1 starts to suffer when the number of layers increases beyond 3. The same effect is also observed in Fig. 1c, where Algorithm 1's approach becomes less effective when the number of agents increase. Nevertheless, in all cases observed, we obtain at least 90% of the optimal social welfare.

We next implemented Algorithm 6, together with pricing scheme of Algorithm 3. We then performed simulations on randomly generated sets of agents, with randomly chosen allocations and valuations. Fig. 2a, shows the effect of the number of agents on the revenue gained by Algorithm 6. The revenue obtained by our mechanism improves significantly as the number of agents increase. This is due to the random splitting process. As the number of agents increase, the amount of revenue available from each set is more likely to be balanced. In Fig. 2b, we simulated the auction for different values of the maximum layer size an agent may receive. Here, we find that the auction performs well in all cases, and is not affected by the number of layers. We note that the auction performs well regardless, and tends to obtain at least 40% of the optimal revenue.



(a) (b) (c)

Fig. 1. The performance of Algorithm 1 is compared to the optimal social welfare for varying network sizes in (a), number of layers in (b) and number of multicast receivers in (c).



(a) (b)

Fig. 2. The revenue obtained by Algorithm 6 is compared to the optimal revenue for varying number of agents in (a) and number of layers in (b).

8 Conclusion

In summary, we provide tools for maximizing social welfare as well as revenue for content providers using layered multicast to disseminate information over the Internet. The algorithms and mechanisms developed are both efficiently computable, and strategyproof. We provide a constant factor approximation algorithm for maximizing social welfare in layered multicast, which we augment with a tailored payment scheme for strategyproofness. We also show how to compute optimal prices for maximizing revenue, and design a randomized strategyproof mechanism with provable revenue guarantees. Simulation results confirm the effectiveness of the proposed solutions in practical settings.

References

1. J. Kurose, K. Ross, and K. Ross, *Computer networking: a top-down approach featuring the Internet*. Addison-Wesley Reading, MA, 2003.
2. K. Jain, M. Mahdian, and M. R. Salavatipour, "Packing Steiner Trees," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2003.

3. M. Thimm, "On The Approximability Of The Steiner Tree Problem," in *Proceedings of the 26th International Symposium on Mathematical Foundations of Computer Science*, 2001.
4. S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proceedings of ACM SIGCOMM*, 1996.
5. J. Zhao, F. Yang, Q. Zhang, Z. Zhang, and F. Zhang, "Lion: Layered overlay multicast with network coding," *IEEE Transactions on Multimedia*, vol. 8, p. 1021, 2006.
6. A. Gopinathan and Z. Li, "Optimal layered multicast," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 7, no. 2, 2011.
7. N. Nisan and A. Ronen, "Algorithmic mechanism design (extended abstract)," in *Proceedings of the ACM Symposium on Theory of computing (STOC)*, 1999.
8. N. Nisan, T. Roughgarden, E. Tardos, and V. V. (Eds.), *Algorithmic Game Theory*. Cambridge University Press, 2007.
9. R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.
10. W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *Journal of finance*, pp. 8–37, 1961.
11. E. Clarke, "Multipart pricing of public goods," *Public choice*, vol. 11, no. 1, pp. 17–33, 1971.
12. T. Groves, "Incentives in teams," *Econometrica: Journal of the Econometric Society*, pp. 617–631, 1973.
13. A. Goldberg, J. Hartline, A. Karlin, M. Saks, and A. Wright, "Competitive auctions," *Games and Economic Behavior*, vol. 55, no. 2, pp. 242–269, 2006.
14. K. Azuma, "Weighted sums of certain dependent random variables," *Tohoku Mathematical Journal*, vol. 19, no. 3, pp. 357–367, 1967.
15. A. Gopinathan and Z. Li, "Strategyproof mechanisms for layered multicast," Available from http://ajay.gopinathan.net.
16. Z. Li, B. Li, D. Jiang, and L. C. Lau, "On Achieving Optimal Throughput with Network Coding," in *Proceedings of IEEE INFOCOM*, 2005.
17. Z. Li and B. Li, "Efficient and Distributed Computation of Maximum Multicast Rates," in *Proceedings of IEEE INFOCOM*, 2005.
18. R. Myerson, "Optimal auction design," *Mathematics of operations research*, pp. 58–73, 1981.
19. C. Borgs, J. Chayes, N. Immorlica, M. Mahdian, and A. Saberi, "Multi-unit auctions with budget-constrained bidders," in *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2005.
20. ISO/IEC, "Generic coding of moving pictures and association audio information," *ISO/IEC*, pp. 13 818–2, 1995.
21. ITU, "Video coding for low bit rate communication," *ITU-T Recommendation H.263*, 1998.
22. "Boston University Representative Internet Topology gEnerator," http://www.cs.bu.edu/brite/.