Revenue Maximization with Nonexcludable Goods

MOHAMMADHOSSEIN BATENI, Google Research
NIMA HAGHPANAH, Massachusetts Institute of Technology
BALASUBRAMANIAN SIVAN, Microsoft Research
MORTEZA ZADIMOGHADDAM, Google Research

We study the design of revenue-maximizing mechanisms for selling nonexcludable public goods. In particular, we study revenue-maximizing mechanisms in Bayesian settings for facility location problems on graphs where no agent can be excluded from using a facility that has been constructed. We show that the pointwise optimization problem involved in implementing the revenue optimal mechanism, namely, optimizing over arbitrary profiles of virtual values, is hard to approximate within a factor of $\Omega(n^{2-\epsilon})$ (assuming $P \neq NP$) even in star graphs. Furthermore, we show that optimizing the expected revenue is APX-hard. However, in a relevant special case, rooted version with identical distributions, we construct polynomial time truthful mechanisms that approximate the optimal expected revenue within a constant factor. We also study the effect of partially mitigating nonexcludability by collecting tolls for using the facilities. We show that such "posted-price" mechanisms obtain significantly higher revenue and often approach the optimal revenue obtainable with full excludability.

Categories and Subject Descriptors: F.2.0 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms: Algorithms, Economics, Theory

Additional Key Words and Phrases: Nonexcludability, revenue maximization, pricing

ACM Reference Format:
MohhamadHossein Bateni, Nima Haghpanah, Balasubramanian Sivan, and Morteza Zadimoghaddam. 2015. Revenue maximization with nonexcludable goods. ACM Trans. Econom. Comput. 3, 4, Article 18 (July 2015), 16 pages.
DOI: http://dx.doi.org/10.1145/2790131

1. INTRODUCTION

How should a seller maximize his revenue while selling nonexcludable services? As a representative example, consider the following facility location problem: firms in an industrial town want to connect their respective warehouses to their outlets with good road links. But the seller (in this case the construction company that lays the roads) cannot enforce exclusivity on the roads it lays: once a firm buys a connection between its warehouse and outlet, the seller cannot exclude others from using those road links. We study the following two questions in this work. Which potential sites should the seller choose for laying roads to maximize his own revenue in such nonexcludable settings? If the seller could enforce partial excludability by collecting tolls for roads, where only those firms that pay the toll for a road could use it, how much more revenue could he earn (collecting tolls only enforces partial excludability because every firm that pays the toll is allowed to use the road)?

Part of the work was done while N. Haghpanah and M. Zadimoghaddam were visiting Google Research, New York.
Authors' addresses: M. Bateni, Google Research, New York; email: bateni@google.com; N. Haghpanah (corresponding author), CSAIL and Sloan School of Management; email: nima@csail.mit.edu; B. Sivan, Microsoft Research; email: bsivan@microsoft.com; M. Zadimoghaddam, Google Research, New York; email: zadim@google.com.

2167-8375/2015/07-ART18
DOI: http://dx.doi.org/10.1145/2790131

Mechanism Design for Connectivity. We model the problem with a graph where every unordered pair of vertices denotes an agent and every edge denotes the *possibility* of a road to be laid between two vertices. We assume that each agent/firm has a nonnegative private value for getting her pair of vertices connected, drawn independently from a known distribution. We aim for the design of the optimal mechanism which, given the value reports of all the agents, selects a set of edges to build roads on and computes payments to be made by agents so that the expected revenue is maximized (expectation over the distribution of private values) in equilibrium. We invoke Myerson's [1981] characterization which states that the expected revenue of any mechanism in a Bayesian Nash Equilibrium (BNE) is equal to the expected *virtual surplus* of the agents served, that is, the sum of virtual values[1] of those agents whose warehouse and outlet are connected in the solution.[2] This reduces the problem of computing the expected revenue maximizing mechanism to a *pointwise* optimization problem: given a profile of values of all agents, compute the set of edges to build roads on so that the sum of the virtual values of the agents served is maximized.

The pointwise optimization problem to maximize virtual values is an interesting graph-theoretic problem, which to our knowledge has not been studied before. Given a graph with a weight on every pair of vertices (the weights represent virtual values, and thus, could be negative), design an algorithm that selects a subset of edges that maximizes the sum of the weights of every vertex pair whose endpoints are connected by the edges selected. If the weights were all positive, clearly the optimal choice for the seller would be to pick all the edges in the graph. However, weights could be negative. In particular, while aiming to connect some subset of agents (i.e., vertex pairs), another subset of agents with negative weights automatically get connected, and these negative weighted pairs cannot be excluded due to nonexcludability. Deciding on the the set of edges to select for maximizing the sum of weights of the vertex pairs whose endpoints are connected is the nontrivial underlying graph theoretic problem.

The Rooted and Unrestricted Versions. We study two versions of the graph theoretic problem just described. In the unrestricted version, every (unordered) pair of vertices in the graph is an agent. In the rooted version (which is a special case of the unrestricted version), a single vertex is designated as root, and only those unordered pairs having root as one of their vertices are agents. This corresponds to the root being a central location where all warehouses are located, and other nodes being outlet locations.

1.1. Results

The Unrestricted Version (Section 3). The natural place to start is to solve the pointwise optimization problem in graphs presented. Clearly, any α-approximate monotone[3] solution to the pointwise optimization problem implies an α-approximate truthful mechanism to the problem of expected revenue maximization. However, we show that except for very special cases, pointwise optimization is not possible unless $P = NP$. In particular, we show that the pointwise problem is NP-hard to approximate to within a factor $\Omega(n^{2-\epsilon})$, even on star graphs. For the special case where the underlying graph is

[1]The virtual value of an agent is a function of her value and the distribution from which her value is drawn; it can be negative. When the virtual value function is not monotone, Myerson [1981] applies a fix by considering the ironed virtual value function (see Section 2 for more details), and all our results go through with virtual values replaced by ironed virtual values.

[2]Note that Myerson's mechanism remains revenue optimal in all single-parameter settings regardless of whether or not the good is excludable.

[3]In single-parameter settings, a mechanism's allocation is monotone if fixing the values of other agents and agent i alone reporting a higher value results in agent i getting served with no smaller probability. Monotone allocation is necessary and sufficient for truthfulness, that is, they alone lend themselves to truthful payments.

a path, we give a dynamic program to solve the pointwise optimization. Thus, for paths, we get the expected optimal revenue with nonexcludability for any product distribution over agents' values.

The Rooted Version (Section 4). Our results for the rooted version are threefold.

(1) First, we give a simple polynomial time algorithm for pointwise optimization in trees—this contrasts with our result that for the unrestricted version the pointwise problem is hard to approximate beyond a factor of $\Omega(n^{2-\epsilon})$, even for star graphs.
(2) Second, as our main result, we give a polynomial-time algorithm for optimization in expectation (over the distribution of values, and hence virtual values) for arbitrary graphs, when the agents' valuations are drawn independently and identically distributed.
(3) Third, we establish APX-hardness of optimization in expectation for arbitrary graphs, when the valuations of all agents are independent but not necessarily identical.

Our main result, which is the second point, is that we design an algorithm that guarantees a constant factor approximation to the optimal virtual surplus in expectation. To this end, we extensively use the key (yet simple) property that even though virtual values can be negative, the expected virtual value of an agent is nonnegative. This suggests the following high-level approach to the problem. Partition agents into a constant number of sets. Pick a target set at random, and run an algorithm that is a constant factor approximation for the agents of that set (ignoring other agents). The nonnegativity of the expected virtual value implies that the contribution from nontargeted sets is nonnegative. Since each set is targeted with constant probability, this implies a constant factor approximation.

We reduce the problem to an abstract edge-weighted problem (in which *edges*, rather than vertices, are agents who derive value from being connected to the root), and then use the partitioning approach previously mentioned to solve the problem. In particular, we present an algorithm that partitions the edges of any graph into two sets that, loosely speaking, correspond to edges in well-connected parts of the graph and edges in sparse cuts of the graph. We show that once we contract the edges in a set (which corresponds to ignoring their value in the high-level approach), the remaining graph has a nice structure that allows for constant approximations: in a well-connected graph, the negative effect of nonexcludability is minimal, since we can select paths to avoid low-valued edges, and a sparse graph can be solved optimally via a bottom-up approach.

Partial Excludability via Pricing (Section 5). If the seller were allowed to set prices on edges (i.e., collect tolls), could he get close to the optimal revenue when excludability is allowed? Typically the optimal revenue with full excludability is much higher than the optimal revenue with nonexcludability. In fact, we show that the optimum revenue, if full excludability were allowed, could be a factor of $\sqrt{n}$ larger than the optimum revenue with nonexcludability. We therefore explore the performance of collecting tolls, as a simple mechanism with partial excludability. In this problem, the seller first has to decide which edges to pick to lay roads on and decide how to price roads to maximize revenue. We construct a pricing scheme so that when the value distributions are independent and identically distributed, the seller obtains (for the rooted version) the optimal revenue possible when full excludability is allowed. Further, if the distributions are independent but not identical and satisfy a technical MHR condition[4],

[4]Roughly speaking, this means that the tail of the distribution is no heavier than the exponential distribution. Many natural classes of distributions like the uniform, exponential, Gaussian (Normal) distributions satisfy the MHR condition. See Section 2 for a formal definition.

we show how to price edges to obtain a $\frac{1}{\log n}$ fraction of the optimal revenue possible when full excludability is allowed (again for the rooted version). This is a big jump in revenue, and an implication of these results is that mitigating externalities via tolls is much more remunerative than aiming for the optimal solution with nonexcludability.

1.2. Related Work

Reducing mechanism design to algorithm design is one of the main focuses of algorithmic game theory. Myerson [1981] reduces revenue maximization in single parameter settings to virtual surplus maximization subject to the global constraint of monotonicity. Hartline and Lucier [2010] show that the monotonicity requirement can be, in general, removed for welfare and revenue maximization problems. This in effect reduces revenue maximization, which is inherently Bayesian, to the pointwise problem of virtual surplus maximization. Cai et al. [2013a, 2013b] show that this reduction is possible in general multiparameter settings.

Approximating the expected version of a problem that is hard to approximate in the worst case, via exploiting the fact that the expected virtual value for any distribution is nonnegative, is a relatively new idea. To our knowledge, it has been used only in Haghpanah et al. [2013]. Common between this work and Haghpanah et al. [2013] is the observation that by applying the standard Bayesian-to-pointwise reduction, we are reducing the problem to one that is "too hard" to approximate. Instead, we can solve the problem by reducing it to the average-case problem of maximizing virtual surplus given the fact that virtual values are positive in expectation. Given this observation, Haghpanah et al. [2013] showed that we can approximate revenue within a constant factor with a simple influence-and-exploit approach. In contrast, in our setting we heavily use the structural properties of the problem, for example, the decomposition techniques in Section 4.3. This could motivate studying other combinatorial problems in the positive-in-expectation settings, that could otherwise be inapproximable in a worst-case analysis.

Auctions with externality (a notion related to nonexcludability where an agent's utility does not just depend on the services he received but also on the outcomes for the other agents) have been studied in multiple flavors before. Settings with positive externality include increased value for having a telephone or a music player or a new technology if more of your neighbors have them [Rohlfs 1974; Hartline et al. 2008]. A prime and well-studied example for a setting with negative externality is the sale of contiguous ad slots to two competing businesses [Aggarwal et al. 2008; Athey and Ellison 2011; Giotis and Karlin 2008; Gomes et al. 2009; Jeziorski and Segal 2009; Kempe and Mahdian 2008]. Equilibria which are surprising at first sight, like no allocation equilibria, can result in large revenue for the auctioneer in settings with negative externality [Deng and Pekec 2011]. Posted pricings have often been a mechanism of choice for settings with externalities [AhmadiPourAnari et al. 2013; Akhlaghpour et al. 2010; Candogan et al. 2010; Hartline et al. 2008]. The main difference between these and the posted prices studied in our work (apart from the presence of nonexcludability in our settings) is that these works allow agent-specific prices, whereas our setting is more constrained: we place prices on edges that are common for all agents.

2. MODEL AND NOTATION

Unrestricted Version. We consider a universe of n potential sites, located on the vertices of a graph, $G = (V, E)$, with m undirected edges. An undirected edge $(i, j) \in E$ means that a link/road connecting sites i and j is allowed (but need not necessarily be constructed). An agent is an unordered pair of sites (i, j), interested in having some path constructed between i and j. Thus, there could be up to $\binom{n}{2}$ agents in a mechanism.

An instance $\mathcal{I} = (G, A, F)$ of the problem consists of an undirected graph G, a set A of agents (represented by a set of pairs of vertices), and a distribution F_i associated with agent i (distributions are explained below). Sometimes we use i to denote a single site and sometimes to denote an agent, who is actually a pair of vertices. The context will make it clear whether i is a single site or a pair of sites.

The Rooted Version. The rooted version is a special case of the setting just described. A special vertex r is designated as the root. Only vertex pairs of the form (i, r) are agents, that is, the root r is one of the end points of the path desired for every agent.

Mechanisms. An outcome $o \in \Omega = \{0, 1\}^m$ is the set of edges constructed. Agent i has a valuation function $v_i : \Omega \to \mathbf{R}^+ \cup \{0\}$, which maps outcomes to nonnegative real numbers. We study mechanisms in the single-parameter setting, where the function $v_i(\cdot)$ takes only two values: $v_i \geq 0$ and zero. An agent i has a nonnegative value v_i if and only if the set of edges selected contains a path between the two sites she represents, and in this case, we say that agent i was served. Let $\mathcal{S}$ denote the set of all feasible sets of agents, that is, the set of all sets of agents that can be simultaneously served. Note that this set system $\mathcal{S}$ is not downward closed: $S \in \mathcal{S}$ does not necessarily mean that $S' \in \mathcal{S}$ for all $S' \subset S$. Clearly, the non-downward closedness stems from the inability to exclude agents from using the roads.

We study mechanisms for this problem in a Bayesian setting, that is, for every i, the single parameter v_i is assumed to be drawn independently from a publicly known distribution function F_i. Thus $F = F_1 \times F_2 \times \ldots F_n$ denotes the product distribution from which the vector of types $\mathbf{v}$ is drawn. The mechanisms in this article assume the availability of any expectation defined with respect to F.

A direct revelation mechanism or auction solicits sealed bids $(b_1, b_2, \ldots, b_n)$ from all the agents and determines the outcome $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and payments $\mathbf{p} = (p_1, p_2, \ldots, p_n)$. Each agent i is risk neutral and has a linear utility $u_i(\mathbf{b}) = v_i \cdot x_i(\mathbf{b}) - p_i(\mathbf{b})$.

Let $(\mathbf{x}(\cdot), \mathbf{p}(\cdot))$ denote a mechanism. When agent i is bidding in the auction, she knows only her own value v_i. A mechanism $(\mathbf{x}(\cdot), \mathbf{p}(\cdot))$ is incentive compatible (IC) if $v_i x_i(v_i, v_{-i}) - p_i(v_i, v_{-i}) \geq v_i x_i(z, v_{-i}) - p_i(z, v_{-i})$ for all i, z. A necessary and sufficient condition on $\mathbf{x}(\cdot)$ for IC payments to exist is monotonicity: for all $i, v_i, v_i{}' \geq v_i, v_{-i}$, we have $x_i(v_i, v_{-i}) \leq x_i(v_i{}', v_{-i})$. Since we focus on the class of IC mechanisms, we have $\mathbf{b} = \mathbf{v}$.

Optimal Auctions. To solve for optimal auctions, Myerson [1981] defines *virtual valuations* for agents as $\phi_i(v_i) = v_i - \frac{1-F_i(v_i)}{f_i(v_i)}$ and proves that the expected payment of an agent, $\mathbf{E}_{v_i}[p_i(v_i)]$, in any truthful mechanism, is equal to her expected virtual value $\mathbf{E}_{v_i}[\phi_i(v_i)x_i(v_i)]$.[5] The distribution F_i is said to be *regular* if the virtual valuation function is monotone. For regular distributions, maximizing virtual values pointwise results in an incentive-compatible allocation rule and therefore the corresponding revenue-optimal auction serves that feasible set of agents who maximize virtual value. For irregular distributions, where maximizing virtual values may result in non-IC allocation rules, Myerson applies a fix by describing a general *ironing* technique. The ironing procedure converts any virtual valuation function $\phi_i(\cdot)$ to an *ironed virtual value function* $\bar{\phi}_i(\cdot)$ such that maximizing $\bar{\phi}_i(\cdot)$ pointwise results in an IC allocation rule.

THEOREM 2.1 [MYERSON 1981]. *The revenue optimal auction in single-parameter Bayesian settings serves the set S of agents, where $S = \text{argmax}_{S \in \mathcal{S}} \sum_{i \in S} \bar{\phi}_i(v_i)$. Further, for all values v_i for which $\bar{\phi}_i(v_i)$ remains the same, agent i's allocation remains the same.*

[5]In fact the equality holds even for a specific $\mathbf{v}_{-i}$, that is, $\mathbf{E}_{v_i}[p_i(\mathbf{v})] = \mathbf{E}_{v_i}[\phi_i(v_i)x_i(\mathbf{v})]$.

As a corollary, when $\mathcal{S}$ contains all possible 2^n sets, the revenue optimal auction puts a price of $\bar{\phi}_i^{-1}(0)$ for agent i and makes a take-it-or-leave-it offer to each of them.

Monotone Hazard Rate. A class of distributions that always satisfy the regularity condition previously described are the ones which satisfy the monotone hazard rate (MHR) condition. A distribution with cdf F satisfies the MHR condition if $\frac{f(x)}{1-F(x)}$ is nondecreasing in x. The MHR condition holds for many natural classes of distributions like the uniform, exponential, and normal distributions.

Nonnegative Virtual Valuations. An important property of virtual valuations (and ironed too) is that when the support of the distribution is nonnegative (which is true in our case since values are nonnegative), the virtual valuation in expectation over an agent's distribution is always nonnegative, even though certain realization of values can be mapped to negative virtual values. For example, if the distribution of values is uniformly distributed on the interval $[0, v]$ for some v, the mapped virtual value will be uniformly distributed on the interval $[-v, v]$, with expectation 0. As another example, it is impossible for virtual value to be deterministically mapped to a certain negative number, since if an agent's value is deterministic, virtual value will be equal to the value, which is by assumption non-negative. This property crucially helps us in providing approximations in expectation, as the hard instances of the problem of virtual value optimization are the ones that include negative virtual values. The fact that virtual values are nonnegative in expectation implies that, roughly speaking, hard instances of the problem are not very likely.

3. THE UNRESTRICTED VERSION

In this section, we obtain the following results for the unrestricted version of our problem.

(1) A simple polynomial-time dynamic program that solves the pointwise problem in paths optimally.
(2) A hardness result showing that the pointwise problem is hard to approximate within a factor of $\Omega(n^{2-\epsilon})$ for any $\epsilon > 0$ (assuming $P \neq NP$), even in stars, that is, we cannot generalize the result for paths even to stars.
(3) While the pointwise optimization problem involved in implementing the revenue optimal mechanism is hard to approximate within $\Omega(n^{2-\epsilon})$ for any $\epsilon > 0$, this still doesn't rule out the possibility of designing a PTAS for optimizing expected revenue. This is because while an arbitrary profile of virtual values can contain both negative and positive numbers, virtual values are always nonnegative in expectation. However, even with nonnegativity in expectation, we show that a PTAS is still not possible. We establish the APX-hardness of this problem in arbitrary graphs. The reduction resembles the one given by Haghpanah et al. [2013]. In fact, our hardness result holds even for the rooted version, and hence the proof is presented in Section 4.4 along with other results for the rooted version.

3.1. Pointwise Optimization in Paths

Consider a path with vertices 1 to n. For $1 \leq i \leq j \leq n$, let $S(i, j)$ be the sum of virtual values of the set of all agents (i.e., vertex pairs) whose both endpoints are in the interval $[i, j]$. We set $S(i, i) = 0$ for all i—assuming that there is no trivial agent whose vertex pairs are the same. Define $OPT(i)$ to be the optimal solution when we are restricted to choose only among edges connecting vertices 1 through i. The unrestricted optimal solution OPT is therefore $OPT(n)$. We set $OPT(0) = 0$. The following recursive formula can be used in the dynamic program to solve for OPT.

$$\forall i \leq n, OPT(i) = \max_{0 \leq k < i} OPT(k) + S(k+1, i).$$

The idea is that the solution to $OPT(i)$ can be decomposed into two parts: a (potentially empty) maximal contiguous set of vertices selected from some $k+1$ to i; and a subset of vertices from 1 to k. That is, all vertices $k+1, \ldots, i$ are connected, and $S(k+1, i)$ is by definition their contribution to the objective. Since the edge connecting k to $k+1$ is not included, the set of edges chosen among the first k vertices must be equal to $OPT(k)$.

3.2. $O(n^{2-\epsilon})$ Hardness of Pointwise Optimization in Stars

Consider a star graph with vertices $\{0, 1, \ldots, n\}$ and edges $(i, 0)$ for all i, and agents being (i, j) for $i, j \neq 0$. We show that there exists an assignment of virtual values to agents for which the virtual value maximizing subset cannot be found. To demonstrate this, it is enough to show that there is some set of $\binom{n}{2}$ virtual values (positive or negative), one for each agent, for which finding the optimal set of agents is inapproximable.

THEOREM 3.1. *Given an arbitrary set of virtual values, the problem of finding the virtual value maximizing set of agents in a star graph is $O(n^{2-\epsilon})$-hard to approximate, unless $P = NP$.*

PROOF. The reduction is from the $O(n^{1-\epsilon})$-hardness of approximation of independent set [Håstad 1996]. Given an arbitrary graph $G = (V, E)$, assign virtual values on the star graph G' with n spokes and a center 0 as follows. For every edge $(i, j) \in E$, let the virtual value of agent (i, j) in the star graph be $-n^2$. For every pair $(i, j) \notin E$, let the virtual value of agent (i, j) in the star graph be 1. It is easy to see that the optimal solution in the star graph will pick an independent set of vertices from G and connect them to the center. Thus if the independent set is of size k, the value of the optimal solution is $\binom{k}{2}$. We know that independent set is inapproximable to a factor of $O(n^{1-\epsilon})$ in general graphs. Thus, the value of the optimal set in star graphs is inapproximable to a factor of $O(n^{2-\epsilon})$. $\square$

4. THE ROOTED VERSION FOR I.I.D. AGENTS

Given the hardness results in Section 3 even for undirected graphs, we consider an important special case of our problem in this section: there is a designated root node in an undirected graph, and each nonroot vertex is an agent wishing to connect to the root. We start by studying the pointwise version of this problem and prove that it is NP-hard to achieve any constant approximation algorithm in general graphs. We then show how to solve it optimally on trees. Finally, we present our main result for general graphs with the assumption that agents values are i.i.d. We note that in all these settings, an agent derives value only if there is a path constructed from his node to the root node. As before, construction of an edge e can happen only if $e \in G$.

4.1. Hardness of Pointwise Optimization in General Graphs

We prove that it is NP-hard to achieve any constant pointwise approximation in the rooted version of the problem for general graphs. We use a reduction from the Set Buying problem introduced in Feige et al. [2013] which is a prize collecting version of the set cover problem.

Definition 4.1 (*Set Buying (SB)*). The input to SB consists of a set $\mathcal{J}$ of n items, where each item $j \in \mathcal{J}$ has revenue $u_j \geq 0$, and a family $\mathcal{F}$ of m sets of items, where each set $S_i \in \mathcal{F}$ has cost $c_i \geq 0$. The value of a subfamily $\mathcal{W} \subseteq \mathcal{F}$ of sets is

$V(\mathcal{W}) = \sum_{j \in \cup_{S_i \in \mathcal{W}} S_i} u_j - \sum_{S_j \in \mathcal{W}} c_j$. The goal is to find a collection $\mathcal{W}$ with the maximum value $V(\mathcal{W})$.

Feige et al. [2013] prove that for any constant $0 < \alpha < 1$, there exists an instance I_α of the set buying problem (by setting $u_j = 1$ for each item j, and $c_i = \alpha|S_i|$ for each set S_i) which cannot be approximated to within a factor better than better than $f(\alpha) = \frac{1-\alpha-\alpha\ln(\frac{1}{\alpha})}{1-\alpha}$ unless $P = NP$. We show an approximation preserving reduction from this instance I_α of the set buying problem to a rooted instance of the revenue maximization problem. Since $f(\alpha)$ approaches zero as α approaches 1, the pointwise optimization of rooted instances of the revenue maximization problem do not admit any constant approximation unless $P = NP$. To prove the following theorem, we construct an approximation preserving reduction from set buying to revenue maximization problem.

THEOREM 4.2. *Given an arbitrary set of virtual values, the problem of finding the virtual value maximizing set of agents for the rooted version does not admit any constant approximation, unless $P = NP$.*

PROOF. We show an approximation preserving reduction from an arbitrary instance of the Set Buying problem to our problem as follows. For each item $j \in \mathcal{J}$, add a vertex and set its virtual value to 1. For each set $S_i \in \mathcal{F}$, add a vertex and set its virtual value to $-\alpha|S_i|$. For each set S_i and each item j, put an edge between them if $j \in S_i$. Add a root r and put an edge between r and each of the sets S_i. It is clear that any solution to the revenue maximization problem connects a subfamily $\mathcal{W}$ to the root and also all the items in sets of $\mathcal{W}$. The revenue of this solution is equal to value $V(\mathcal{W})$, which is the objective of the set buying problem. Therefore revenue maximization in this instance is equivalent to finding the subfamily $\mathcal{W}^*$ with maximum value. Since this set buying instance I_α does not admit any approximation factor better than $f(\alpha)$, by letting α approach 1 arbitrarily, we exclude the possibility of any constant approximation algorithm for revenue maximization problem, unless $P = NP$. □

4.2. Pointwise Optimization in Trees

In contrast to the unrestricted version where we show that the pointwise problem in hard to approximate within a factor of $\Omega(n^{2-\epsilon})$ (assuming $P \neq NP$) even in stars, we show that for the rooted version, we can solve the pointwise problem in polynomial time in trees. As a subroutine for optimization in trees, we first show how to perform pointwise optimization in paths (much easier and faster than the pointwise optimization for paths in the general version).

Pointwise Optimization in Paths. Consider an undirected path of length $n+1$ with one end of the path, namely, node 0, as the root. If the root is not one of the end vertices, such a path can be broken into two at the root, and each path will have a root at one of its ends. (These problems can be solved independently.) Thus, without loss of generality, root is at the end of the path. Once the agents submit their values, the auction has to decide on the set of edges to pick. In the path case, this task reduces to picking the best among paths of the form $0 - i$ for $i = 1 \ldots n$. For every i, the mechanism computes the sum of the virtual values of agents from 1 to i and picks the i with the maximum virtual value.

Pointwise Optimization in Trees. Consider an undirected tree with a designated root. The revenue optimal mechanism in trees is constructed using the optimal mechanism in paths as a subroutine. The mechanism proceeds from the bottom up. It chooses the bottom-most node, say b, in the tree which has branches below. By definition, each such branch must be a path. The mechanism computes the optimal solution and the corresponding optimal virtual value for each path and adds them up and assigns them to b. All the descendants of b are now deleted, with the virtual value of b now adjusted to the new value. This procedure is now repeated in the new tree till we exhaust all nodes. By induction on the depth of the tree, it is easy to establish the revenue optimality of this mechanism.

4.3. Main Result: Optimization in Expectation in Arbitrary Graphs

Next we present a constant-factor approximation algorithm for optimizing the expected revenue for the rooted, node-weighted version of the problem in arbitrary graphs (the virtual value of an agent is the weight of the nonroot node of that agent). We first explain how the problem can be solved on edge-weighted graphs, that is, agents are edges, and they get a value when they are connected to the root. This is later used as a subroutine to tackle the vertex-weighted problem.

Edge-Connectivity for the Rooted Version. Given a connected undirected graph, we show how to partition its edges into two parts such that (a) contracting the former edge set yields a 3-edge connected subgraph[6]; (b) contracting the latter edge set results in a *roulette* subgraph—a special series-parallel graph to be defined next. We then demonstrate that it is possible to solve the problem (approximately) on each of the two subgraphs, and finally argue that this suffices to obtain a constant-factor approximation for the general rooted edge-weighted case.

To define roulette graphs, we need the following notation. For a set S of edges in a graph G, subgraph G/S is obtained from G after contracting all edges S one at a time, where contracting an edge simply refers to removing the edge and merging its endpoint vertices. We call an edge in a connected graph a bridge if removing it makes the graph disconnected.

A *2-edge connected roulette graph* is defined recursively as follows.

—*Base*. A simple cycle is a 2-edge connected roulette.
—*Recursion*. A graph G is a 2-edge connected roulette if there exists a subset of vertices V such that the subgraph induced by V is a 2-edge connected roulette, and the subgraph obtained by contracting all the edges inside V is also a 2-edge connected roulette.

In other words, a cycle of roulette graphs is itself a 2-edge connected roulette (see Figure 1). For example, we can construct a 2-edge connected roulette by replacing vertices of a simple cycle with roulettes (with one or two vertices of the inner roulette taking the place of an original vertex of the cycle).

A general graph G (not necessarily 2-edge connected) is a roulette graph if the graph G' derived from G by contracting all its bridge edges is a 2-edge connected roulette graph. In other words, we can connect two 2-edge connected roulette graphs by a bridge edge, and the result will be a roulette graph (see Figure 2).

Now we can present the main structural lemma that reduces our general problem into two tractable subproblems.

LEMMA 4.3. *Given a graph $G(V, E)$, there exists a polynomial-time algorithm that partitions the edge set E into two sets S_1 and S_2 such that graphs $G_1 = G/S_1$ and $G_2 = G/S_2$ are, respectively, 3-edge connected and roulette.*

[6]A k-edge connected graph is a graph that remains connected whenever fewer than k edges are removed. Equivalently, any cut in such a graph has size at least k.

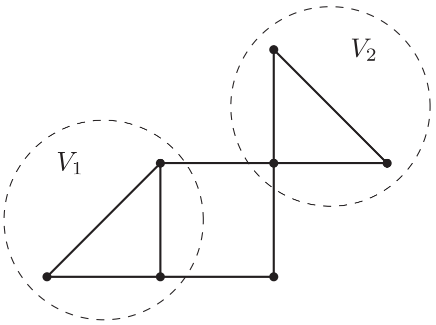

Fig. 1. This graph is a 2-edge connected roulette since it is a cycle of 2-edge connected roulette graphs. Contracting subgraphs induced by V_1 and V_2, which are simple cycles themselves (and therefore 2-edge connected roulettes), results in a simple cycle.

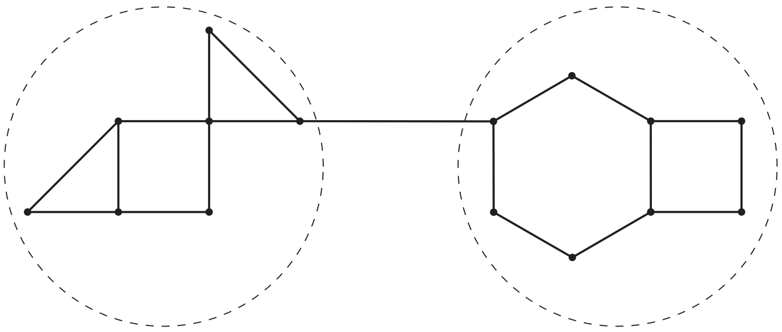

Fig. 2. Connecting two 2-edge connected roulette graphs by a bridge edge yields a roulette graph.

Proof. Define S_B to be the set of all bridge edges in G. Let S_1 be the union of S_B and all edges that belong to a cut of size 2 in graph G/S_B. We note that all cuts of size 2 in G/S_B can be found in polynomial time, for example, using a naïve brute-force search. Since $G_1 = G/S_1$ is obtained by a series of edge contractions, every cut in G_1 represents a cut in G as well. Therefore, since all edges of S_1 are contracted in G_1, no cut of size at most 2 is present in G_1, hence G_1 is 3-edge connected.

On the other hand, every edge in G_2 is either in S_B (the set of bridge edges in both G and G_2) or belongs to a cut of size 2 in G/S_B, hence in G_2/S_B (recall that G_2 is the graph resulting from contracting all the edges but S_1). To prove G_2 is a roulette graph, it suffices to prove that G_2/S_B is a 2-edge connected roulette graph. We claim that if each of $\{e_1, e_2\}$ and $\{e_1, e_3\}$ is a cut in G_2/S_B, then so is $\{e_2, e_3\}$. Therefore, the edges of G_2/S_B form an equivalence class. To see this equivalence relation, it suffices to focus on the following alternative definition of edge cuts of size 2. In a 2-edge connected graph, two edges e and e' form a cut of size 2 if and only if the set of cycles in the graph that contain e is the same as the set of cycles in the graph that contain e'. Based on this new definition, the two sets of cycles containing e_2 and e_3, respectively, are both equal to the set of cycles containing e_1, and therefore they are equal to each other as well, which means that e_2 and e_3 form a cut of size 2.

Therefore, the graph looks like a cycle of this equivalence class (the equivalence class of e_1), where some vertices are replaced by another subgraph; the same argument applies to each of these smaller structures, giving rise to the inductive definition of 2-edge connected roulette graphs. We should note that two edges from two of these smaller structures do not belong to the same equivalence class (they cannot form a cut of size 2), and therefore each of the other equivalence classes belong to one smaller structure and is not split between different structures. Thus we can inductively claim each smaller structure is a 2-edge connected roulette graph. □

The following decomposition lemma serves as the starting point for our 3-approximation algorithm of the 3-edge connected graph G_1.

LEMMA 4.4. *There exists a polynomial-time algorithm that finds three spanning trees T_1, T_2, and T_3 in a 3-edge connected graph G such that every edge of G is missing in at least one of these spanning trees.*

PROOF. We replace each edge of G by two parallel edges to get the 6-edge connected graph G^2 with the same vertex set. Catlin et al. [2009] show, among other things, that any $2k$-edge connected graph has k edge-disjoint spanning trees, while Roskind and Tarjan [1985] show how to find k edge-disjoint spanning trees in a graph (if they exist) in quadratic time. Therefore, we can find three edge-disjoint spanning trees T_1, T_2, and T_3 in G^2. Edge-disjointness guarantees that each edge of G can belong to at most two of these spanning trees. □

On the other hand, the problem can be solved optimally for roulette graphs. The intuition behind the algorithm is that we can recursively solve the problem on cycles via dynamic-programming and then remove them by contracting them.

LEMMA 4.5. *There exists a polynomial-time algorithm that finds the optimal solution for roulette graphs.*

PROOF. Let us say we have a *cycle decomposition* of our roulette graph, which describes the recursive structure of its 2-edge connected components. Each cycle representing one equivalence class is called an *essential* cycle of the graph. Instead of solving the rooted problem, we consider a slightly more general problem where up to two vertices s, t on an essential cycle are specified, and these vertices should both appear in the connected subgraph of the output. At the beginning we are going to have only one vertex $s = t$, which is the root vertex.

Let us ignore the bridges at this point and assume the graph is 2-edge connected. Focus on the essential cycle and imagine that all the recursive structures are contracted for now. In the resulting graph, the optimal solution looks like a path or it is the entire cycle. There are polynomially many cases to consider, and we will output the best solution among them. In each case, we have a subproblem for each contracted piece if we decide that the optimal solution passes through or ends at the contracted vertex. Since the base cases of the induction (i.e., vertices or cycles) are easily solvable, we can argue inductively that our slightly more general problem can be solved using a dynamic program.

For each bridge connected to the essential cycle, we compute the best solution for the rooted problem on the other side of the bridge (whose root is the endpoint of the bridge) and add that to the main solution if its weight plus the weight of the bridge turns out positive. □

We conclude this section by putting together these ideas to obtain a 4-approximation algorithm for the rooted edge-weighted problem.

LEMMA 4.6. *There exists a polynomial-time monotone algorithm that achieves an approximation factor of 4 for any graph G.*

PROOF. If the graph is not connected, we can focus on the connected component that contains the root and disregard the rest of the connected components. Using Lemma 4.3, we partition the edges of G into two parts S_1 and S_2. We also use Lemma 4.4 to find three spanning trees T_1, T_2, and T_3 in graph G_1.

We consider four candidate solutions. One solution is the union of S_1 and the edges in tree T_1. Since T_1 is a spanning tree in $G_1 = G/S_1$, the union of T_1 and S_1 is a connected spanning subgraph of G. Serving this connected spanning subgraph allows us to choose any subset of the remaining edges to serve. Among the remaining edges (edges not in $S_1 \cup T_1$), we serve those with positive realized virtual values. In a similar fashion, we construct two other candidate solutions based on T_2 and T_3. The fourth and last candidate solution is to contract set S_2 of edges and solve the problem optimally in the roulette graph G_2 using Lemma 4.5. The optimal solution we find in G_2 is a connected subgraph of G_2, however, it might not be a connected subgraph of G that includes the root. Nonetheless, it is always possible to serve a subset of edges $S_2' \subseteq S_2$ to make the whole solution not only a connected subgraph of G but also one that includes the root vertex as well. Our fourth candidate solution consists of the edges in S_2' and the optimal solution for G_2.

We now show that for every instance (i.e., a graph with a designated root and a distribution), the sum of the revenues of these four solutions is at least the optimum revenue. Thus picking one at random will guarantee a 4-approximation. For $1 \leq i \leq 3$, if we stick to solution i all the time, our gain from edges in S_1 and T_i is nonnegative (i.e., the sum of their expected virtual values), and in addition we get all edges with positive virtual value outside $S_1 \cup T_i$. Since each edge of S_2 is missing in at least one $S_1 \cup T_i$, the total value we get from the first three solutions is at least the projection of the optimal solution on set S_2 of edges. On the other hand, in the fourth solution, our expected revenue from edges added from S_2 is nonnegative (once again since the expectation of the virtual values are positive), and our expected revenue from S_1 is at least the amount that the optimal solution gains from them. Therefore, these four solutions together achieve no less than the optimal revenue. Thus, one candidate solution has expected revenue at least a quarter of the optimum.

The algorithm is monotone because it first decides which of the four candidate solutions to use independently of the realized values. Further each candidate solution is monotone. The first three solutions are monotone because the edge selection criteria (for the edges they consider) is just non-negative virtual value. The fourth solution is monotone because it is an optimal algorithm for the edges it focuses on. □

Vertex-Connectivity for the Rooted Version. In this section, we provide a constant-factor monotone approximation algorithm for the vertex-connectivity problem using the algorithm for the edge-connectivity version of the problem just described. Let V_2 be the set of degree-2 vertices in graph G. Similar to the edge-connectivity approach, we describe two algorithms (one using a reduction to the edge-connectivity problem) that achieve constant-factor approximations to the problems, where the values of $V \setminus V_2$ and V_2 are replaced by zero, respectively. Again, since the expected value of each vertex is nonnegative, this implies a constant approximation for the vertex-connectivity problem.

First consider the instance in which the values of vertices in $V \setminus V_2$ are replaced by zero. Notice that this results in an instance in which any vertex with nonzero value has degree 2. Construct another instance in which each vertex of degree 2 is replaced by an edge with the same value. We can solve this instance using our edge-connectivity algorithm.

Next consider the instance in which the values of vertices in V_2 are replaced by zero. We can then remove any such vertex and connect its two neighbors directly, and therefore assume that the graph does not contain any degree-2 vertex. The following lemma shows that this graph has a spanning tree where at least $\frac{1}{7}$ of its vertices are leaves. The algorithm uses the *internal* nodes (i.e., nonleaves) of this tree to connect the leaves that are positive, to the root. Since the vertex values are drawn i.i.d., this gives a 7-approximation to the problem.

Putting these two algorithms together, we obtain a constant-factor approximation algorithm for the vertex-connectivity rooted revenue maximization in expectation. In particular, a balancing argument puts a bound of 11 on its approximation ratio.

Monotonicity of the resulting algorithm follows from the monotonicity of the algorithms used for the two subcases. When we use the algorithm for the edge-connected version, the monotonicity of the edge-connected version implies the same here. For the other case, note that a leaf is picked whenever its virtual value is positive thus resulting in a monotone allocation.

LEMMA 4.7. *Given a graph with no degree-2 vertices, a spanning tree can be constructed in polynomial time, where at least $\frac{1}{7}$ of the vertices are leaves.*

PROOF. Start from an arbitrary spanning tree T of G. Let T_2 be the set of vertices of degree 2 in T, and let $\hat{T}_2 \subseteq T_2$ be those vertices in T_2 both whose neighbors, too, are in T_2. Modify T as long as any of the following two rules apply.

(1) If there exists a vertex $v \in \hat{T}_2$ that has an edge in G to an internal vertex u of T, update T by adding the edge (v, u) to T, and removing the edge incident to v in the unique cycle formed after adding (v, u). Since both neighbors of v in T had degree 2, this process generates a new leaf without removing any of the old leaves.
(2) If two vertices $v, u \in \hat{T}_2$ have edges in G to the same leaf l of T, add edges from v and u to that leaf, and remove two edges from T as follows. The addition of edge (u, l) to T produces a cycle that passes through exactly one of the two neighbors of u; call it u'. Note that u' has degree 2 in T by definition of $\hat{T}_2$; let u, u'' be its neighbors. Remove the edge (u', u'') from T, removing the cycle u and maintaining the connectivity of T. We carry out a similar operation, mutatis mutandis, for v. The result will be a tree T on the same set of vertices with one more leaf (increasing the degree of l but turning two other internal vertices into leaves).

The process terminates in a linear number of iterations since the number of leaves increases in each step. We end up with a tree T for which neither of the rules applies. Let T_1 and $T_{\geq 3}$, respectively, denote the subsets of vertices of degrees one and at least three in T. We argue next that $|T_1|$ is at least a constant fraction of $|T_2| + |T_{\geq 3}|$. As no vertex of G has degree two, any vertex in $\hat{T}_2$ is bound to have degree at least three in G, hence an edge not in T. This edge cannot be to an internal vertex of T because Rule (1) no longer applies to T. Rule (2), on the other hand, implies that these leaves are distinct for different vertices of $\hat{T}_2$. Therefore, we have

$$|T_1| \geq |\hat{T}_2|. \tag{1}$$

As trees have average degree less than two, we know

$$|T_1| > |T_{\geq 3}|. \tag{2}$$

To bound $|T_2| - |\hat{T}_2|$, if this quantity is not zero, orient T from an arbitrary vertex in $T_2 \setminus \hat{T}_2$ towards the leaves. Assign each vertex $v \in T_2 \setminus \hat{T}_2$ to its closest descendant in the oriented tree that is in $T_1 \cup T_{\geq 3}$. Such an assignment is always possible since no vertex in the former group is a leaf of the (oriented) tree. Each vertex in the latter group is assigned to at most twice; otherwise, there should be a path of vertices of degree two with more than two vertices in $T_2 \setminus \hat{T}_2$—a contradiction. As a result, we get

$$|T_2| - |\hat{T}_2| \leq 2(|T_1| + |T_{\geq 3}|) \leq 4|T_1|, \tag{3}$$

where the last inequality is due to Eq. (2).

Summing up Eqs. (1), (2), and (3) with $|T_1| \geq |T_1|$, we obtain $7|T_1| \geq |T_1| + |T_2| + |T_{\geq 3}|$ as desired. □

4.4. APX-Hardness in Expectation in Arbitrary Graphs

In this section, we prove the APX-hardness of the rooted version in arbitrary graphs when the valuations of agents are independent but not necessarily identical.

Definition 4.8. The prize-collecting set cover problem (PCSCP) consists of a collection of sets $S_1, S_2, \dots, S_n$ over a universe U. For a collection C of sets, let $Q_C = \cup_{i \in C} S_i$. The goal is to find a collection C^* that maximizes $\alpha|Q_{C^*}| + n - |C^*|$ for some given $\alpha > 0$.

We first show that there is an approximation-preserving reduction from PCSCP to our problem, and then invoke the result from Haghpanah et al. [2013] that establishes the APX-hardness of PCSCP, and that its approximation ratio is at least $\frac{530}{529} = 1.002$.

LEMMA 4.9. *There is an approximation preserving reduction from PCSCP to the revenue maximization problem in arbitrary graphs.*

PROOF. Given an instance of the PCSCP, where the sets are denoted $S_1, S_2, \dots S_n$ and the elements are denoted $e_1, e_2, \dots e_m$, we construct an instance of our problem as follows.

Vertices. We start with a root vertex r. For every element e_i, we construct one vertex with the same name e_i. For every set S_i, we construct one vertex denoted by S_i as well.

Edges. For each i, set vertex S_i is connected by an edge to root r and all $e_j \in S_i$.

Agents. The agents (r, S_i) are called set-agents, and (r, e_i) are called element-agents.

Distribution. The value distribution of element agents is deterministic α. For set agents, the value is drawn from the distribution Bernoulli$(L-1, 1/L)$ (i.e., the value is equal to $L-1$ with probability $1/L$, and zero otherwise), where $L \gg mn\alpha$. Thus, the virtual value for these agents is -1 w.p. $1-1/L$ and $L-1$ w.p. $1/L$.

The optimal revenue in our problem, as $L \to \infty$, can be analyzed in two cases.

(1) If at least one set-agent has positive virtual valuation (which happens w.p. approximately $n/L \to 0$), the solution chooses all the edges incident on those set agents (with positive virtual valuation) to obtain expected revenue n. The expected revenue from the remaining agents (set-agents with negative virtual valuation and element-agents) is at most $\alpha mn/L \ll 1$. Therefore, the optimal solution has contribution n from this event as $L \to \infty$, and this solution is trivial to compute.
(2) If no set has positive virtual valuation (which happens w.p. $1-n/L \to 1$), the value of the solution is precisely $\alpha|Q_{C^*}| - |C^*|$. This is because once a set-agent is chosen, clearly all the edge-agents contained by this set must be chosen, since they all have deterministic positive virtual value α. We should also note that you can obtain the α virtual value of an element agents only if you connect it to the root using one of the set agents it belongs to, and for each set you pick, you have -1 virtual value in this case.

Therefore the value of the optimal solution is $\alpha|Q_{C^*}| + n - C^*$. □

5. POSTED-PRICING RESULTS

As mentioned in the introduction, one goal of this work is to compare two kinds of mechanisms: (a) natural mechanisms for nonexcludable public goods, such as a direct revelation mechanism, that have to deal with nonexcludabilities, and (b) posted price mechanisms which mitigate nonexcludability to a certain extent by ensuring that only those who pay for an edge can use that edge. We show that for the rooted version with i.i.d. agents, a posted price mechanism obtains the optimal revenue possible when full excludability is allowed (i.e., even partially mitigating nonexcludability via tolls gets us the benefit of full excludability). When the agents values are independent but not necessarily identical, we design a pricing scheme that obtains a $O(\frac{1}{\log n})$ fraction of the optimal revenue possible when full excludability is allowed.

5.1. Rooted Version with I.I.D. Agents

Consider a directed graph G with a designated root r. Without any externality constraints, Myerson's mechanism's revenue is $\sum_{i \neq r} \phi^{-1}(0)(1 - F(\phi^{-1}(0))$, where $1 - F(\phi^{-1}(0))$ is the probability that agent i's value is above $\phi^{-1}(0)$. The posted price mechanism places a price of $\phi^{-1}(0)$ on every edge incident on the root (directed either way). Every agent who can reach the root must use one such edge and thus pays the price if his value is higher than his price. This gives a revenue of precisely $\sum_{i \neq r} \phi^{-1}(0)(1 - F(\phi^{-1}(0))$, which is the optimal revenue possible.

On the other hand, if we use a direct revelation mechanism, the optimal revenue achievable, even for paths in i.i.d. settings, is a factor of $\sqrt{n}$ away from the optimal revenue obtainable with full excludability. Consider the simple example of a path $(0, 1, 2, \ldots, n)$ on $n + 1$ vertices with vertex 0 as the root. The value of each nonroot vertex is 1 with probability 1/2, and 0 otherwise. Thus, the virtual value for every vertex is 1 with probability 1/2 and -1 with probability 1/2. If we are allowed to select vertices regardless of nonexcludability, we will select each vertex when its virtual value is 1, achieving the expected value of $n/2$. On the other hand, the optimal revenue with nonexcludability is obtained by selecting the subpath (that is connected to the root) with maximum sum of virtual values contained in it. The expectation of this number can be seen to be at most the expected maximum position reached by a symmetric random walk in one dimension after n steps.[7] This is well known to be $O(\sqrt{n})$.

5.2. Rooted Version with Non-I.I.D. Agents

When the distributions are independent but not necessarily identical, we design a pricing scheme which obtains a $O(\log n)$ approximation when the distributions satisfy the MHR condition (i.e., the hazard function $\frac{f(x)}{1-F(x)}$ is monotonically nondecreasing). When the agents are non-i.i.d., the monopoly price of all agents are not necessarily the same, so the previous mechanism clearly fails. We need two facts to give a $O(\log n)$ approximation.

(1) When a distribution F follows MHR property, we have $F(\phi^{-1}(0)) \leq 1 - 1/e$. This follows from

$$F(x) = 1 - e^{-\int_0^x h(t)\,dt} \leq 1 - e^{-\int_0^x h(x)\,dt} = 1 - e^{-xh(x)},$$

where the inequality follows from the monotonicity of $h(x)$. Since $xh(x) = 1$ at $x = \phi^{-1}(0)$, we have that $F(\phi^{-1}(0)) \leq 1 - 1/e$.

(2) Given n numbers $v_1, v_2, \ldots, v_n$, we have $\max_i iv_i \geq \frac{1}{O(\log n)} \sum_i v_i$. Let $t = \text{argmax}_i\, iv_i$. So for all i, $tv_t \geq iv_i$, and thus $v_i \leq tv_t/i$ for all i. Summing over this inequality for all i, we get $\sum_i v_i \leq tv_t(O(\log n))$.

Without loss of generality, let $\phi_1^{-1}(0) \geq \phi_2^{-1}(0) \geq \cdots \geq \phi_n^{-1}(0)$. Let $t = \text{argmax}_i\, i\phi_i^{-1}(0)$. Now suppose we post a price of $\phi_t^{-1}(0)$. All agents with monopoly price above $\phi_t^{-1}(0)$

[7]After i steps, the location of the random walk is equal to the number of positive movements, minus the number of negative movements. This is exactly equal to the sum of the numbers on edges on the subpath from root to the ith vertex.

will pay the toll with probability at least $1/e$ (by point 1 above). Thus we get a revenue of $\frac{1}{e}\max_i i\phi_i^{-1}(0) \geq \frac{1}{e\log n}\sum_i \phi_i^{-1}(0)$ (by point 2 above). Thus we get a $O(\log n)$ approximation to the optimal revenue with full excludability.

REFERENCES

G. Aggarwal, J. Feldman, S. Muthukrishnan, and M. Pál. 2008. Sponsored search auctions with markovian users. In *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE'08)*. 621–628.

N. AhmadiPourAnari, S. Ehsani, M. Ghodsi, N. Haghpanah, N. Immorlica, H. Mahini, and V. S. Mirrokni. 2013. Equilibrium pricing with positive externalities. *Theor. Comput. Sci.* 476, 1–15.

H. Akhlaghpour, M. Ghodsi, N. Haghpanah, V. S. Mirrokni, H. Mahini, and A. Nikzad. 2010. Optimal iterative pricing over social networks (extended abstract). In *Proceedings of the 6th International Workshop on Internet and Network Economics (WINE'10)*. 415–423.

S. Athey and G. Ellison. 2011. Position auctions with consumer search. *Quart. J. Econ.* 126, 3, 1213–1270.

Y. Cai, C. Daskalakis, and S. M. Weinberg. 2013a. Reducing revenue to welfare maximization: Approximation algorithms and other generalizations. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'13)*. 578–595.

Y. Cai, C. Daskalakis, and S. M. Weinberg. 2013b. Understanding incentives: Mechanism design becomes algorithm design. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS'13)*. 618–627.

O. Candogan, K. Bimpikis, and A. E. Ozdaglar. 2010. Optimal pricing in the presence of local network effects. In *Proceedings of the 6th International Workshop on Internet and Network Economics (WINE'10)*. 118–132.

P. A. Catlin, H.-J. Lai, and Y. Shao. 2009. Edge-connectivity and edge-disjoint spanning trees. *Disc. Math.* 309, 5, 1033–1040.

C. Deng and S. Pekec. 2011. Money for nothing: Exploiting negative externalities. In *Proceedings of the 12th ACM Conference on Electronic Commerce (EC'11)*. 361–370.

U. Feige, N. Immorlica, V. S. Mirrokni, and H. Nazerzadeh. 2013. PASS approximation: A framework for analyzing and designing heuristics. *Algorithmica* 66, 2, 450–478.

I. Giotis and A. R. Karlin. 2008. On the equilibria and efficiency of the GSP mechanism in keyword auctions with externalities. In *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE'08)*. 629–638.

R. Gomes, N. Immorlica, and E. Markakis. 2009. Externalities in keyword auctions: An empirical and theoretical assessment. In *Proceedings of the 5th International Workshop on Internet and Network Economics (WINE'09)*. 172–183.

N. Haghpanah, N. Immorlica, V. S. Mirrokni, and K. Munagala. 2013. Optimal auctions with positive network externalities. *ACM Trans. Econ. Comput.* 1, 2.

J. D. Hartline and B. Lucier. 2010. Bayesian algorithmic mechanism design. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC'10)*. 301–310.

J. D. Hartline, V. S. Mirrokni, and M. Sundararajan. 2008. Optimal marketing strategies over social networks. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. 189–198.

J. Håstad. 1996. Clique is hard to approximate within $\mathrm{n}^{1\text{-epsilon}}$. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS'96)*. 627–636.

P. Jeziorski and I. Segal. 2009. What makes them click: Empirical analysis of consumer demand for search advertising. *SSRN eLibrary* 33.

D. Kempe and M. Mahdian. 2008. A cascade model for externalities in sponsored search. In *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE'08)*. 585–596.

R. B. Myerson. 1981. Optimal auction design. *Math. Oper. Res.* 6, 1, 58–73.

J. Rohlfs. 1974. A theory of interdependent demand for a communications service. *Bell J. Econ. Manag. Sci.* 5, 1, 16–37.

J. Roskind and R. E. Tarjan. 1985. A note on finding minimum-cost edge-disjoint spanning trees. *Math. Oper. Res.* 10, 4, 701–708.

Received April 2014; revised September 2014; accepted February 2015