Optimal Iterative Pricing over Social Networks

Hessameddin Akhlaghpour * Mohammad Ghodsi *
Nima Haghpanah † Hamid Mahini * Vahab S. Mirrokni ‡
Afshin Nikzad *

Abstract

We study the optimal pricing for revenue maximization over social networks in the presence of positive network externalities. In our model, the value of a digital good for a buyer is a function of the set of buyers who have already bought the item. In this setting, a decision to buy an item depends on its price and also on the set of other buyers that have already owned that item. The revenue maximization problem in the context of social networks has been studied by Hartline, Mirrokni, and Sundararajan [9], following the previous line of research on optimal viral marketing over social networks [11, 12, 13].

We consider the Bayesian setting in which there are some prior knowledge of the probability distribution on the valuations of buyers. In particular, we study two iterative pricing models in which a seller iteratively posts a new price for a digital good (visible to all buyers). In one model, re-pricing of the items are only allowed at a limited rate. For this case, we give a FPTAS for the optimal pricing strategy in the general case. In the second model, we allow very frequent re-pricing of the items. We show that the revenue maximization problem in this case is inapproximable even for simple deterministic valuation functions. In the light of this hardness result, we present constant and logarithmic approximation algorithms for a special case of this problem when the individual distributions are identical.

1 Introduction

Despite the rapid growth, online social networks have not yet generated significant revenue. Most efforts to design a comprehensive business model for monetizing such social networks [15, 16], are based on contextual display advertising [20]. An alternative way to monetize social networks is viral marketing, or advertising through word-of-mouth. This can be done by understanding the *externalities* among buyers in a social network. The increasing popularity of these networks has allowed companies to collect and use information about inter-relationships among users of social networks.

*Department of Computer Engineering, Sharif University of Technology, {akhlaghpour,mahini,nikzad}@ce.sharif.edu, ghodsi@sharif.edu

†Northwestern University, EECS Department, nima.haghpanah@eecs.northwestern.edu

‡Google Research NYC, 76 9th Ave, New York, NY 10011, mirrokni@google.com

In particular, by designing certain experiments, these companies can determine how users influence each others' activities.

Consider an item or a service for which one buyer's valuation is influenced by other buyers. In many settings, such influence among users are positive. That is, the purchase value of a buyer for a service increases as more people use this service. In this case, we say that buyers have *positive externalities* on each other. Such phenomena arise in various settings. For example, the value of a cell-phone service that offers extra discounts for calls among people using the same service, increases as more friends buy the same service. Such positive externality also appears for any high-quality service through positive reviews or the word-of-mouth advertising.

By taking into account the positive externalities, sellers can employ forward-looking pricing strategies that maximize their long-term expected revenue. For this purpose, there is a clear trade-off between the revenue extracted from a buyer at the beginning, and the revenue from future sales. For example, the seller can give large discounts at the beginning to convince buyers to adopt the service. These buyers will, in turn, influence other buyers and the seller can extract more revenue from the rest of the population, later on. Other than being explored in research papers [9], this idea has been employed in various marketing strategies in practice, e.g., in selling TiVo digital video recorders [19].

In an earlier work, Hartline, Mirrokni, and Sundararajan [9] study the optimal marketing strategies in the presence of such positive externalities. They study optimal adaptive ordering and pricing by which the seller can maximize its expected revenue. However, in their study, they consider the marketing settings in which the seller can go to buyers one by one (or in groups) and offer a price to those specific buyers. Allowing such price discrimination makes the implementation of such strategies hard. Moreover, price discrimination, although useful for revenue maximization in some settings, may result in a negative reaction from buyers [14].

Preliminaries. Consider a case of selling multiple copies of a digital good (with no cost for producing a copy) to a set V of n buyers. In the presence of network externality, the *valuation* of buyer i for the good is a function of buyers who already own that item, $v_i : 2^V \to R$, i.e., $v_i(S)$ is the value of the digital good for buyer i, if set S of buyers already own that item. We say that users have *positive externality* on each other, if and only if $v_i(S) \leq v_i(T)$ for each two subsets $S \subseteq T \subseteq V$. In general, we assume that the seller is not aware of the exact value of the valuation functions, but she knows the distribution $f_{i,S}$ with an accumulative distribution $F_{i,S}$ for each random variable $v_i(S)$, for all $S \in V$ and any buyer i. Also, we assume that each buyer is interested only in a single copy of the item. The seller is allowed to post different prices at different time steps and buyer i buys the item in a step t if $v_i(S_t) - p_t \geq 0$, where S_t is the set of buyers who own the item in step t, and p_t is the price of the item in that step. Note that $v_i(\emptyset)$ does not need to be zero; in fact $v_i(\emptyset)$ is the value of the item for a user before any other buyer owns the item and influence him.

We study optimal iterative pricing strategies without price discrimination during k *time steps*. In particular, we assume an *iterative posted price* setting in which we post

a *public price* p_i at each step i for $1 \leq i \leq k$. The price p_i at each step i is visible to all buyers, and each buyer might decide to buy the item based on her valuation for the item and the price of the item in that time step. An important modeling decision to be made in a pricing problem is whether to model buyers as *forward-looking* (strategic) or *myopic* (impatient). For the most of this paper, we consider myopic or impatient buyers who buy an item at the first time in which the offered price is less than their valuations. We discuss this issue along with forward-looking buyers in more details in Appendix A after stating our results for myopic buyers.

In order to formally define the problem, we should also define each time step. A time step can be long enough in which the influence among users can propagate completely, and we can not modify the price when there is a buyer who is interested to buy the item at the current price. On the other extreme, we can consider settings in which the price of the item changes fast enough that we do not allow the influence amongst buyers to propagate in the same time step. In this setting, as we change the price per time step, we assume the influence among buyers will be effective on the next time step (and not on the same time step). In the following, we define these two problems formally.

Definition 1. The Basic(k) **Problem:** *In the* Basic(k) *problem, our goal is to find a sequence $p_1, \ldots, p_k$ of k prices in k consecutive time steps or days. A buyer decides to buy the item during a time step as soon as her valuation is more than or equal to the price offered in that time step. In contrast to the* Rapid(k) *problem, the buyer's decision in a time step immediately affects the valuations of other buyers in the same time step. More precisely, a time step is assumed to end when no more buyers are willing to buy the item at the price at this time step.*

Note that in the Basic(k) problem, the price sequence will be decreasing. If the price posted at any time step is greater than the previous price, no buyer would purchase the product at that time step.

Definition 2. The Rapid(k) **Problem:** *Given a number k, the* Rapid(k) *problem is to design a pricing policy for k consecutive* days *or* time steps*. In this problem, a pricing policy is to set a public price p_i at the start of time step (or day) i for each $1 \leq i \leq k$. At the start of each time step, after the public price p_i is announced, each buyer decides whether to buy the item or not, based on the price offered on that time step* [1] *and her valuation. In the* Rapid(k) *problem, the decision of a buyer during a time step is not affected by the action of other buyers in the same time step* [2].

One insight about the Rapid(k) model is that buyers react slowly to the new price and the seller can change the price before the news spreads through the network. On the other hand, in the Basic(1) model, buyers immediately become aware of the new

[1] In discussions for both Basic(k) and Rapid(k) problems, we use the terms time step and day interchangeably. The definition should be clear in the context

[2] Note that in the Rapid(k) problem, a pricing policy is adaptive in that the price p_i at time step i may depend on the actions of buyers in the previous time steps.

state of the network (the information spreads fast), and therefore respond to the new state of the world before the seller is capable of changing prices. For more insight about the above two models, see the example in Appendix B.

A common assumption studied in the context of network externalities is the assumption of *submodular influence functions*. This assumption has been explored and justified by several previous work in this framework [6, 9, 11, 13]. In the context of revenue maximization over social networks, Hartline et. al. [9] state this assumption as follows: suppose that at some time step, S is the set of buyers who have bought the item. We use the notion of *optimal (myopic) revenue* of a buyer for S, which is $R_i(S) = \max_p p \cdot (1 - F_{i,S}(p))$. Following Hartline et.al [9], we consider the optimal revenue function as the *influence function*, and assume that the optimal revenue functions (or influence functions) are submodular, which means that for any two subsets $S \subset T$, and any element $j \notin S$, $R_i(S \cup \{j\}) - R_i(S) \geq R_i(T \cup \{j\}) - R_i(T)$. In other words, submodularity corresponds to a diminishing return property of the optimal revenue function which has been observed in the social network context [6, 11, 13].

Definition 3. *We say that all buyers have* identical initial distributions *if there exists a distribution F_0 so that the valuation of a player given that the influence set is equal to S is the sum of two independent random variables, one from F_0, and another one from $F_{i,S}$, with $F_{i,\emptyset} = 0$.*

Definition 4. *A probability distribution f with accumulative distribution F satisfies the monotone hazard rate condition if the function $h(p) = f(p)/(1 - F(p))$ is monotone non-decreasing. Several natural distributions like uniform distributions and exponential distributions satisfy this condition.*

Our Contributions. We first show that the deterministic Basic(k) problem is polynomial-time solvable. Moreover, for the Bayesian Basic(k) problem, we present a fully polynomial-time approximation scheme. We study the structure of the optimal solution by performing experiments on randomly generated preferential attachment networks (appendix E). In particular, we observe that using a small number of price changes, the seller can achieve almost the maximum achievable revenue by many price changes. In addition, this property seems to be closely related to the role of externalities. In particular, the density of the random graph, and therefore the role of network externalities increases, fewer number of price changes are required to achieve almost optimal revenue.

Next we show that in contrast to the Basic(k) problem, the Rapid(k) problem is intractable. For the Rapid(k) problem, we show a strong hardness result: we show that the Rapid(k) problem is not approximable within any reasonable approximation factor even in the deterministic case unless P=NP. This hardness result holds even if the influence functions are submodular and the probability distributions satisfy the monotone hazard rate condition. In light of this hardness result, we give an approximation algorithm using a minor and natural assumption. We show that the Rapid(k) problem for buyers with submodular influence functions and probability distributions with the monotone hazard rate condition, and *identical initial distributions* admits logarithmic

approximation if k is a constant and a constant-factor approximation if $k \geq n^{\frac{1}{c}}$ for any constant c. We propose several future research directions in appendix C.

Related work. Optimal pricing mechanisms in the presence of network externalities have been considered in the economics literature [3, 4, 7, 10]. Carbal, Salant, and Woroch [4] consider an optimal pricing problem with network externalities when the buyers are strategic, and study the properties of equilibrium prices. In their model, buyers tend to buy the product as soon as possible because of a discount factor, which reduces the desirability of late purchases. Previous work in economic literature such as [8, 5, 17] had shown that without network externalities, the equilibrium prices decrease over time. On the other hand, Carbal et.al. show that in a social network the seller might decide to start with low *introductory* prices to attract a *critical mass* of players, when the players are *large* (i.e, the network effect is significant). They observe that this pattern (of increasing prices) also happens when there is uncertainty about customer valuations, no matter how strong the network effect is.

Optimal viral marketing over social networks have been studied extensively in the computer science literature [12]. For example, Kempe, Kleinberg and Tardos [11] study the following algorithmic question (posed by Domingos and Richardson [6]): How can we identify a set of k influential nodes in a social network to influence such that after convincing this set to use this service, the subsequent adoption of the service is maximized? Most of these models are inspired by the dynamics of adoption of ideas or technologies in social networks and only explore influence maximization in the spread of a *free* good or service over a social network [6, 11, 13]. As a result, they do not consider the effect of pricing in adopting such services. On the other hand, the pricing (as studied in this paper) could be an important factor on the probability of adopting a service, and as a result in the optimal strategies for revenue maximization.

2 The Basic(k) Problem

We define $B^1(S,p) := \{i|v_i(S) \geq p\} \cup S$. Assume a time step where at the beginning, we set the global price p, and the set S of players already own the item. So $B^1(S,p)$ specifies the set of buyers who immediately want to buy (or already own) the item. As $B^1(S,p)$ will own the item before the time step ends, we can recursively define $B^k(S,p) = B^1(B^{k-1}(S,p),p)$ and use induction to reason that $B^k(S,p)$ will own the item in this time step. Let $B(S,p) = B^{\hat{k}}(S,p)$, where $\hat{k} = \max\{k|B^k(S,p) - B^{k-1}(S,p) \neq \emptyset\}$, knowing that all buyers in $B(S,p)$ will own the item before the time step ends. One can easily argue that the set $B(S,p)$ does not depend on the order of users who choose to buy the item.

Solving Deterministic Basic(1)**:** First, we state the following lemma, which can be proved by induction.

Lemma 1. *For any a and b such that $a < b$, $B(\emptyset,b) \subseteq B(\emptyset,a)$.*

In the Basic(1) problem, the goal is to find a price p_1 such that $p_1 \cdot |B(\emptyset,p_1)|$ is maximized. Let $\beta_i := sup\{p|i \in B(\emptyset,p)\}$ and $\beta := \{\beta_i|1 \leq i \leq n\}$. WLOG we

assume that $\beta_1 > \beta_2 \ldots > \beta_n$. Using Lemma 1, player i will buy the item if and only if the price is set to be less than or equal to β_i.

Lemma 2. *The optimal price p_1 is in the set β.*

Now we provide an algorithm to find p_1 by finding all elements of the set β and considering the profit $\beta_i \cdot B(\emptyset, \beta_i)$ of each of them, to find the best result. Throughout the algorithm, we will store a set S of buyers who have bought the item and a global price g. In the beginning $S = \emptyset$ and $g = \infty$. The algorithm consists of $|\beta|$ steps. At the i-th step, we set the price equal to the maximum valuation of remaining players, considering the influence set to be S. We then update the state of the network until it stabilizes, and moves to the next step. Our main claim is as follows. At the end of the i-th step, the set who own the item is $B(\emptyset, \beta_i)$, and the maximum valuation of any remaining player is equal to β_{i+1}.

Generalization to Deterministic Basic(k)**:** We attempt to solve the Basic(k) problem by executing the Basic(1) algorithm consisting of m steps and by using a dynamic algorithm. We are looking for an optimal sequence $(p_1, p_2, \ldots, p_k)$ in order to maximize $\sum_{i=1}^{k} |B(\emptyset, p_i) - B(\emptyset, p_{i-1})| \cdot p_i$.

We claim that an optimal sequence exists such that for every i, $p_i = \beta_j$ for some $1 \leq j \leq |\beta|$. This can be shown by a proof similar to that of lemma 2. Thus the problem Basic(k) can be solved by considering the subproblem $A[k', m]$ where we must choose a non-increasing sequence π of k' prices from the set $\{\beta_1, \beta_2, \ldots \beta_m\}$, to maximize the profit, and setting the price at the last day to β_m. This subproblem can be solved using the following dynamic program:

$$A[k', m] = \max_{1 \leq t < m} A[k' - 1, t] + |B(\emptyset, \beta_m) - B(\emptyset, \beta_t)| \cdot \beta_m$$

FPTAS for the Bayesian setting: For the Bayesian (or probabilistic) Basic(k) problem, we run a similar dynamic program, but the main difficulty for this problem is that the space of prices is continuous, and we do not have the same set of candidate prices as we have for the deterministic case. To overcome this issue, we employ a natural idea of discritizing the space of prices. Then we estimate the expected revenue by a sampling technique (see appendix D).

3 The Rapid(k) Problem

3.1 Identical Initial distributions

As we will see in section 3.2, the Rapid(k) problem is hard to approximate even with submodular influence functions and probability distributions satisfying the monotone hazard rate condition. So we consider the Rapid(k) problem with submodular influence functions and probability distributions satisfying the monotone hazard rate condition, and buyers have identical initial distributions. For this problem, we present an approximation algorithm whose approximation factor is logarithmic for a constant k and its approximation factor is constant for $k \geq n^{\frac{1}{c}}$ for any constant $c > 0$. The algorithm is as follows:

1. Compute a price p_0 which maximizes $p(1-F_0(p))$ (the myopic price of F_0), and let R_0 be this maximum value. Also compute a price $p_{1/2}$ such that $F_0(p_{1/2}) = 0.5$. With probability $\frac{1}{2}$, let $c = 1$, otherwise $c = 2$.
2. If $c = 1$, set the price to the optimal myopic price of F_0 (i.e, p_0) on the first time step and terminate the algorithm after the first time step.
3. if $c = 2$, do the following:
   (a) Post the price $p_{1/2}$ on the first time step.
   (b) Let $\overline{S}$ be the set of buyers that do not buy in the first day, and let their optimal revenues be $R_1(V - \overline{S}) \geq R_2(V - \overline{S}) \geq \ldots \geq R_{|\overline{S}|}(V - \overline{S})$.
   (c) Let p_j be the price which achieves $R_j(V - \overline{S})$, and Pr_j be the probability with which j accepts p_j for any $1 \leq j \leq |\overline{S}|$. Thus we have $R_j(V - \overline{S}) = p_j Pr_j$.
   (d) Let $d_1 < d_2 < \ldots < d_{k-1}$ be the indices returned by lemma 7 as an approximation of the area under the curve $R(V - \overline{S})$.
   (e) Sort the prices $\frac{p_{d_j}}{e}$ for $1 \leq j \leq k-1$, and offer them in non-increasing order in days 2 to k.

To analyze the expected revenue of the algorithm, we need the following lemmas:

Lemma 3. *Let S be the set formed by sampling each element from a set V independently with probability at least p. Also let f be a submodular set function defined over V, i.e., $f : 2^V \rightarrow R$. Then we have $E[f(S)] \geq pf(V)$ [9].*

Lemma 4. *If the valuation of a buyer is derived from a distribution satisfying the monotone hazard rate condition, she will accept the optimal myopic price with probability at least $1/e$ [9].*

Lemma 5. *Suppose that f, defined over $[a, b]$, is a probability distribution satisfying the monotone hazard rate condition, with expected value μ and myopic revenue $R = \max_p p(1 - F(p))$. Then we have $R(1 + e) \geq \mu$.*

Lemma 6. *Let i be the index maximizing ia_i in the set $\{a_1, a_2, \ldots, a_m\}$. Then we have $ia_i \geq \sum_{j=1}^{m} a_j/(\lceil \log(m+1) \rceil)$.*

Lemma 7. *For a set $\{a_1 \geq a_2 \geq \ldots \geq a_n\}$, let $D = \{d_1 \leq d_2 \leq \ldots \leq d_k\}$ be the set of indices maximizing $S(D) = \sum_{j=1}^{k} (d_j - d_{j-1}) a_{d_j}$ (assuming $d_0 = 0$), over all sequences of size k. Then we have $S(D) \in \Theta(\frac{\sum_i a_i}{\log_k n})$.*

Proof idea: We present an algorithm that iteratively selects rectangles, such that after the m-th step the total area covered by the rectangles is at least $m/\log n$ using $4^m - 1$ rectangles. At the start of the m-th step, the uncovered area is partitioned into 4^{m-1} independent parts. In addition, the length of the lower edge of each of these parts is e_p which is at most $n/(2^{m-1})$. The algorithm solves each of these parts independently as follows. We use 3 rectangles for each part in each step. First, using lemma 6 we

know that we can use a single rectangle to cover at least $1/\log e_p$ of the total area of part. Then, we cover the two resulting uncovered parts by two rectangles, which each equally divide the lower edge of the corresponding part. The complete proof appears in the appendix F.2. ■

Theorem 1. *The expected revenue of the above pricing strategy A as described above is at least $\frac{1}{8e^2(e+1)\log_k n}$ of the optimal revenue.*

Proof: For simplicity assume that we are allowed to set $k+1$ prices. In case $c = 1$, we set the optimal myopic price of all players and therefore achieve the expected revenue of nR_0. If $c = 2$, consider the second day of the algorithm. By lemma 4, we know that each remaining buyer accepts her optimal myopic price with probability at least $1/e$, so for every j we have $Pr_j \geq 1/e \geq Pr_i/e$. In addition, we know that for each $j \leq i$, $R_j(V - \overline{S}) \geq R_i(V - \overline{S}) \geq p_i/e$. We also know that $R_j(V - \overline{S}) \leq p_j$. As a result, $p_j \geq p_i/e$, for each $j \leq i$. Therefore, if we offer the player $j \leq i$ the price p_i/e, she will accept it with probability at least Pr_i/e (she would have accepted p_j with probability at least $Pr_j \geq Pr_i/e$; offering a lower price of p_i/e will only increase the probability of acceptance).

For now suppose that we are able to partition players to k different groups, and offer each group a distinct price. Ignore the additional influence that players can have on each other. In that case, we can find a set $d_1 < d_2 < \ldots < d_k$ maximizing $\sum_{j=1}^{k}(d_j - d_{j-1}) \cdot R_{d_j}(V - \overline{S})$. Assume that D_i is the set of players y with $d_{i-1} < y \leq d_i$. As we argued above, if we offer each of these players the price p_{d_i}/e, she will accept it with probability at least Pr_{d_i}/e. So the expected value of each of the players in D_i when offered p_{d_i}/e is at least $Pr_{d_i}/e \cdot p_{d_i}/e = R_{d_i}(V - \overline{S})/e^2$. The total expected revenue in this case will be $\sum_{j=1}^{k}(d_i - d_{i-1}) \cdot R_{d_j}(V - \overline{S})/e^2$, which, using lemma 7 is at least $\sum_i R_i(V - \overline{S})/(e^2 \log_k n)$. An important observation is that, if the expected revenue of a player when she is offered a price p is R, her expected revenue will not decrease when she is offered a non-increasing price sequence P which contains p . As a result, we can sort the prices that are offered to different groups, and offer them to *all players* in non-increasing order.

Finally, using Lemma 3, and since every player buys at the first day independently with probability 1/2, we conclude that any buyer i that remains at the second day observe an expected influence of $R_i(V)/2$ from all other buyers.

As a result, the expected revenue of our algorithm is $nR_0/2$ (from setting p_0 with probability 1/2 in the first day) plus $\sum_i R_i(V) \cdot (1/8) \cdot (1/(e^2 \log_k n))$. Since we set $p_{1/2}$ with probability 1/2, a player does not buy at first day with probability 1/2, and we achieve $1/(e^2 \log_k n)$ of the value of remaining players in the second day. We also know that the expected revenue that can be extracted from any player i is at most $E(F_0) + E(F_{i,V})$. Thus, using lemma 5, we conclude that the approximation factor of the algorithm is $8e^2(e+1)\log_k n$. ■

3.2 Hardness

In this section, we prove the hardness of the Rapid(k) problem even in the deterministic case with additive (modular) valuation functions. Specifically, we consider the following special case of the problem: (i) $k = n$; (ii) The valuations of the buyers are deterministic, i.e., $f_{i,S}$ is an impulse function, and its value is nonzero only at $v_i(S)$; and finally (iii) The influence functions are additive; $\forall i, j, S$ such that $i \neq j$ and $i, j \notin S$ we have $v_i(S \cup \{j\}) = v_i(S) + v_i(\{j\})$, also each two buyers $i \neq j$, $v_i(\{j\}) \in \{0, 1\}$, and each buyer has a non-negative initial value, i.e, $v_i(\emptyset) \geq 0$.

We use a reduction from the *independent set* problem; We show that using any $\frac{1}{n^{1-\epsilon}}$-approximation algorithm for the specified subproblem of Rapid(k), any instance of the *independent set* problem can be solved in polynomial time.

For the special case of additive influence functions, it is convenient to use a graph to represent the influence among buyers, i.e., a directed graph G^* has node set $V(G^*)$ equal to the set of all buyers, and $(j, i) \in E(G^*)$ if and only if $v_i(\{j\}) = 1$. Now we show how to construct an instance of Rapid(k), the graph G^*, from an instance $G = (V, E)$ of the the independent set problem. Our goal is to determine whether there exists an independent set of vertices, larger than a given K. Let $N = |V|$. The set of vertices of G^* is formed from the union of five sets, denoted by A, F, C, X, and Y. In the first set A, there are two vertices d_i and a_i for each vertex i in G (see figure 1). As we will see later, selling the item to d_i corresponds to selecting i as a member of the independent set. The *activator* of vertex i, a_i, is used to activate (will be made clear shortly) the next vertex, d_{i+1}. The initial values of d_1 and a_1 are $K - 1 + \epsilon$ and $K - 1 + 2\epsilon$, respectively. For $i > 1$, the initial values of d_i and a_i are $K - 2 + (2i - 1)\epsilon$ and $K - 2 + 2i\epsilon$, respectively. There are two edges from a_i to d_{i+1}, and a_{i+1}, for each $i < n$. We observe that initially, the first couple have the highest valuations. We can consider selling the item to a_i, d_i couples in order. On day i, we sell the item to a_i and we can choose to sell or not to sell to d_i. But we do not sell the item to any other buyer. Then next day (day $i + 1$), with the influence of the a_i buyer on the next couple, which we referred to as *activation*, the $(i + 1)$-th couple will have the highest values. This allows us to sell to both of them, or only the activator, without having any other buyer buy the item. Thus we can use the set A to show our selection of vertices for the independent set as follows. Start from d_1 and a_1 and visit vertices in order. When visiting the i-th couple, if we want vertex i of G to be in the independent set, we set the price to $K - 1 + (2i - 1)\epsilon$ (causing both d_i and a_i to buy). Otherwise, set the price to $K - 1 + 2i\epsilon$. In both cases the activator will buy, and makes the next couple have the highest values. From now on, by *selecting* $\{i_1, i_2, \ldots, i_l\}$ from G or *selecting* $\{d_{i_1}, d_{i_2}, \ldots, d_{i_l}\}$ from G^*, we mean selling to $\{d_{i_1}, d_{i_2}, \ldots, d_{i_l}\}$.

The set F is used to represent the edges in E. There is a vertex f_e in F for each edge e in E. The initial values of all these vertices are $K - 2$. Let $e = (i, j)$ be an edge in E. There is one edge from d_i, and one from d_j to f_e. In this way, if two endpoints of an edge e are selected to be in the independent set, the value of the corresponding buyer f_e increases to K. Otherwise, it would be either $K - 1$, or $K - 2$. We are going to build the rest of the graph in a way that if the value of any vertex in F increases

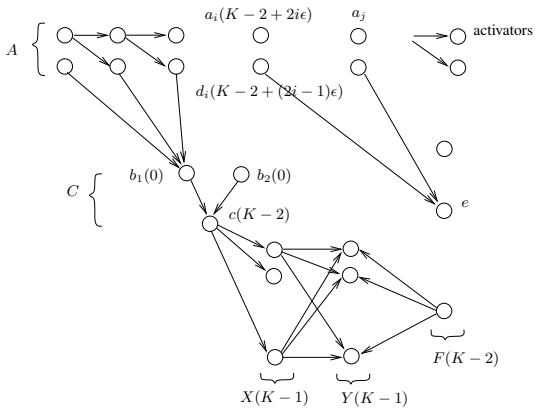

Figure 1: The reduction of subsection 3.2. The number in the parentheses next to the name of a vertex (set of vertices) is the initial valuation of that vertex (vertices). In this instance, the edge e is adjacent to vertices i and j (in graph G of independent set problem).

to K, we lose a big value. This will prevent the optimal algorithm from selecting two adjacent vertices.

The set C consists of only three vertices, the two counters b_1 and b_2, and another vertex c. The initial values of the counters are zero, and the initial value of c is $K-2$. There are two edges from each d_i going to b_1 and b_2. Also there is an edge from b_1 and b_2 to c (note that b_1 and b_2 are completely identical). The selection of any vertex d_i will increase the value of the counters by one. So we can use the counters to keep track of the number of vertices selected from the set A.

Finally there are two important sets of $X = \{x_1, x_2, \ldots, x_L\}$ and $Y = \{y_1, y_2, \ldots, y_L\}$, where L is a large number to be determined later. The initial value of all the vertices in X and Y is $K-1$. There is one edge from c to each vertex in X. There is also an edge from each vertex in F, to each vertex in Y. Finally we add an edge from each vertex in X to each vertex in Y. These L^2 edges are so important that if we do not take advantage of them when possible, we will be unable to approximate the revenue by the $\frac{1}{n^{1-\epsilon}}$ factor; we must sell the item for price at least L to all buyers in Y, if possible. To do so, we have no choice but to select at least K independent vertices from the G. First observe that all the vertices in X are identical, and so are all the vertices in Y. So the prices with which any buyer in X buys the item, is equal to the price with which any other buyer in X buys the item. The same argument is true for buyers in Y. In addition, the initial values of all the vertices in X and Y are equal. In order to take advantage of the L^2 edges, we have to find a way to increase the value of the buyers in X to some value more than the value of the buyers in Y. To do this, we should activate the only incoming edges of X without activating any of the incoming edges of Y. This is only possible by selling to c, and not selling to any of the vertices in F. We are going to see that the only possible scenario is to increase the value of b_1 and b_2 to K by selecting K vertices from V, and making sure that the selected vertices form an independent set.

Theorem 2. *The* Rapid(k) *problem with additive influence functions can not be approximated within any multiplicative factor unless P=NP (see appendix F.3).*

References

[1] N. Bansal, N. Chen, N. Cherniavsky, A. Rudra, B. Schieber, and M. Sviridenko. Dynamic pricing for impatient bidders. In *SODA*, pages 726–735, 2007.

[2] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509, 1999.

[3] B. Bensaid and J.-P. Lesne. Dynamic monopoly pricing with network externalities. *International Journal of Industrial Organization*, 14(6):837–855, October 1996.

[4] L. Cabral, D. Salant, and G. Woroch. Monopoly pricing with network externalities. Industrial Organization 9411003, EconWPA, Nov. 1994.

[5] R. H. Coase. Durability and monopoly. *Journal of Law & Economics*, 15(1):143–49, April 1972.

[6] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD '01*, pages 57–66, New York, NY, USA, 2001. ACM.

[7] J. Farrell and G. Saloner. Standardization, compatibility, and innovation. *RAND Journal of Economics*, 16(1):70–83, Spring 1985.

[8] O. Hart and J. Tirole. Contract renegotiation and coasian dynamics. *Review of Economics Studies*, 55:509–540, 1988.

[9] J. Hartline, V. S. Mirrokni, and M. Sundararajan. Optimal marketing strategies over social networks. In *WWW*, pages 189–198, 2008.

[10] M. L. Katz and C. Shapiro. Network externalities, competition, and compatibility. *American Economic Review*, 75(3):424–40, June 1985.

[11] D. Kempe, J. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *KDD '03*, pages 137–146, New York, NY, USA, 2003. ACM.

[12] J. Kleinberg. *Cascading behavior in networks: algorithmic and economic issues*. Cambridge University Press, 2007.

[13] E. Mossel and S. Roch. On the submodularity of influence in social networks. In *STOC '07*, pages 128–134, New York, NY, USA, 2007. ACM.

[14] R. L. Oliver and M. Shor. Digital redemption of coupons: Satisfying and dissatisfying eects of promotion codes. *Journal of Product and Brand Management*, 12:121–134, 2003.

[15] E. Oswald. http://www.betanews.com/article/Google_Buy_MySpace_Ads_for_900m/1155050350.

[16] K. Q. Seeyle. www.nytimes.com/2006/08/23/technology/23soft.html.

[17] J. Thpot. A direct proof of the coase conjecture. *Journal of Mathematical Economics*, 29(1):57 – 66, 1998.

[18] G. van Ryzin and Q. Liu. Strategic capacity rationing to induce early purchases. *Management Science*, 54:1115–1131, 2008.

[19] R. Walker. http://www.slate.com/id/1006264/.

[20] T. Weber. http://news.bbc.co.uk/1/hi/business/6305957.stm?lsf.

A Myopic vs. Strategic Buyers

An important modeling decision to be made in a pricing problem is whether to model buyers as *forward-looking* (strategic) or *myopic* (impatient). Forward-looking players will choose their strategies based on the prices that are going to be set in the future. As a result, they might decide not to buy an item although the offered price is less than their valuations. On the other hand, myopic players are supposed to buy the item in the first day in which the offered price is less than their valuations. Buyers might be impatient for several reasons. This happens when the good can be consumed (food, beverage, ...), when the customers make impulse purchases ([18]), or when the item is offered in limited amount (so that it might not be available in the future). Also, players might be myopic when the effect of the discount factor is stronger than the possible decrement in prices in the future (that is why we buy electronic devices although we are aware of the discounted prices in the future). Optimal pricing for myopic or impatient buyers have been also studied from algorithmic point of view in the computer science literature. For example, Bansal et. al. [1] study dynamic pricing problems for impatient buyers, and design approximation algorithms for this revenue maximization for this problem.

In this paper, we focused on the myopic or impatient buyers, and presented all our algorithms and models in this model. We also ignored the role of discount factors in the valuation of players. It can be shown that some of our results can be extended to take into account forward-looking behaviors and also discount factors, but we do not discuss them in this paper.

B Example

Example. For more insight about the $\mathsf{Basic}(\mathsf{k})$ and $\mathsf{Rapid}(\mathsf{k})$ models and their differences, we study the following example: Consider n buyers numbered 1 to n. For buyer i ($1 \leq i \leq n$), $v_i(\emptyset)$ is $\epsilon \cdot (i-1)$. Any purchase by any buyer $i \neq 1$ increases buyer 1's valuation by L. The valuations of the rest of the buyers does not change (See Figure 2).

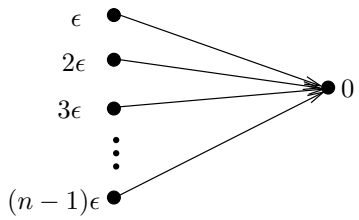

Figure 2: Each node represents a buyer. The numbers written next to each buyer i is $v_i(\emptyset)$. The arrows represent influences of L. L (and ϵ) are arbitrarily large (and small) numbers respectively.

Consider the Rapid(n) problem on this example. One may think the Rapid(n) problem can be easily solved by posting the highest price that anyone is willing to buy the product for it. Using this naive approach for this example, in the second time step, the seller would sell the product for price L to buyer 1. But if she decides to post a public price ϵ on the first time step, she can sell the product for price $(n-1)L$ to player 1 afterwards. For the Basic(n) problem, no matter what the seller does, she can not sell the product to buyer 1 for more than $1 + (n-1)\epsilon$. This example shows that the seller can get much more revenue in the Rapid(n) problem compared to the Basic(n) problem.

C Concluding Remarks

In this paper, we introduce new models for studying the optimal pricing and marketing problems over social networks. We study two specific models and show a major difference between the complexity of the optimal pricing in these settings. This paper leaves many problems for future study. (i) We presented preliminary results for strategic buyers, but many problems remain open in this setting. Studying optimal pricing strategies for strategic or patient buyers is an interesting problem. In fact, one can model the pricing problem for the seller and the optimal strategy for buyers as a game among buyers and the seller, and study equilibria of such a game. (ii) We present insights from simulation results. Proving these insights in general settings is an interesting problem. (iii) Our results for the Basic(k) problem are for the non-adaptive pricing strategy in which the sequence of prices are decided at the beginning. It would be interesting to study adaptive pricing strategies in which the seller decides about the price at each step by looking at the history of buyers' re-actions to the previous price sequence. (iv) We studied a monopolistic setting in which a seller does not compete with other sellers. It would be nice to study this problem non-monpolistic settings in which other sellers may provide similar items over time, and the seller should compete with other sellers to attract parts of the market. (v) An important research direction for marketing over social networks is to combine the pricing algorithms with learning

algorithms to learn the externalities of buyers on each other in the context of social networks. Such a model should combine an exploration and explotation phases in a careful manner.

Finally, studying optimal advertising strategies over social networks is closely related to our marketing problems and is an interesting subject for future research.

D FPTAS for Basic(k)

Here, we extend the dynamic programming approach and present an FPTAS for the Basic(k) problem.

We assume a minimum price $p_{min} = 1$ for the item, and design pricing algorithms that optimize the revenue using a minimum price of 1\$. Let $ROUND$ be the running time of simulating one trial of the buying process during one time step (given probability distributions $f(i, S)$ for each user i and each subset S of users). We first observe a simple FPTAS for Basic(1), and then use it to solve Basic(k). In the Basic(1) problem, our goal is to find a price p that maximizes the expected revenue of the seller when she sets the price to p. Let C_p and X_p be random variables for the revenue and number of buyers who buy the item when we set the price to p. Our goal is to find a price p that maximizes $E[C_p] = pE[X_p]$. Note that we can estimate $E[X_p]$ using a standard sampling method, i.e., by sampling from the random process using a price p for a polynomial number of trials and and taking the average of the number of buyers who bought the item in each trial. Since $0 \leq X_p \leq |V|$, using Chernoff-Hoeffding concentration inequality, we can easily show that $E[X_p]$ can be computed within an error factor of ϵ with high probability.

Using the sampling method, we estimate $E[C_p]$ with high probability for any given p. Assuming values $p_{min} = 1$ and a maximum price p_{max} such that $p_{min} \leq p_{OPT} \leq p_{max}$, the idea is to check some specific price values between p_{min} and p_{max} and choose one of them with respect to their estimated revenue. The algorithm is as follows:

1. Let $i_{max} = \log_{1+\epsilon}^{p_{max}/p_{min}} + 1$.
2. Define values $p_i = (1 + \epsilon)^i p_{min}$ for any integer i, $0 \leq i < i_{max}$, and compute $E[C_{p_i}]$ for each p_i.
3. Return p_i with the maximum calculated $E[C_{p_i}]$.

Theorem 3. *For every $m \geq 3$ and $t \geq 1$, there exists an polynomial time algorithm which finds a price p such that $E[C_p] \geq E[C_{p_{OPT}}] \frac{1-\epsilon}{(1+\epsilon)^2}$ with high probability.*

Now, we are ready to design an algorithm for solving Basic(k) problem. In this problem, we have k time steps and we need to set a price p_i in time step i. Let $X_{p_1,p_2,...,p_t}$ be the number of buyers who buy the item at time step t when the price is p_i at time step $i \leq t$. Thus, the total revenue is

$$\text{revenue} = p_1 X_{p_1} + p_2 X_{p_1,p_2} + ... + p_k X_{p_1,p_2,...,p_k}.$$

In Basic(k), our goal is to maximize the expected revenue, which is: $E[\text{revenue}] = p_1 E[X_{p_1}] + p_2 E[X_{p_1,p_2}] + ... + p_k E[X_{p_1,p_2,...,p_k}]$.

Using standard sampling technique and Chernoff-Hoeffding bound, we can estimate $E[X_{p_1,p_2,...,p_t}]$ within a small error with high probability (since $0 \leq X_{p_1,p_2,...,p_t} \leq n$).

Given any set of valuations, the set of all buyers who have bought the item after time step t, is the same as the set of buyers who have bought the item when we have only one time step in which we set the price to be p_t. Thus, $X_{p_1} + X_{p_1,p_2} + ... + X_{p_1,p_2,...,p_t} = X_{p_t}$ which implies $X_{p_1,p_2,...,p_t} = X_{p_t} - X_{p_{t-1}}$. Thus, $E[X_{p_1,p_2,...,p_t}] = E[X_{p_t}] - E[X_{p_{t-1}}]$.

First, we present a simple dynamic program with a polynomial running time in p_{max}. Then we modify it and design an FPTAS for the problem. As a warm-up example, let's assume that $E[X_p]$ can be computed precisely for every p, and we are allowed to offer only integer prices. We design a dynamic programming algorithm to solve the problem. Consider the state of the network when we set the price to p and wait until the network becomes stable. In this state, some subset of buyers has bought the item. We call this state of the network, *Net-Stable(p)*. We define $A[t, p]$ as the maximum expected revenue when we have t time steps and the state of the network is Net-Stable(p). In order to calculate $A[t, p]$ we can search for the price to be offered in the first time step. It could be any price below p. The recurrence relation for the dynamic program is as follows:

$$A[t, p] = \max_{0<p'<p} \{A[t-1, p'] + p'(E[X_{p'}] - E[X_p])\} \quad (1)$$

Note that in state Net-Stable($p_{max} + 1$), no buyer has bought the item and the network is in the initial state. Therefore our solution is stored in $A[k, p_{max} + 1]$. The above algorithm is based on some unrealistic assumption and its running time is polynomial in terms of to p_{max}.

A $\frac{1-\epsilon}{2(1+\epsilon)}$-approximation Algorithm. Let $i_{max} = \lceil \log(p_{max}) \rceil$. First we observe that if $k \geq i_{max} + 1$, one can easily design a $\frac{1}{2}$-approximation algorithm: Offer price $p_i = 2^{i_{max}-i}$ at time step i. Assume that in the optimum solution, we offer price p_i^o at time step i. Consider the set of buyers who has bought the item in Net-Stable(p) and call this set S_p. It is clear that $S_p \subseteq S_{p'}$ for $p \geq p'$. Let buyer x be a buyer who has bought the item at price $2^r \leq p_x < 2^{r+1}$ in the optimum solution. Thus buyer x will buy the item when the price is 2^r in our solution. Since $2^r \geq p_x/2$, the above simple algorithm is a $\frac{1}{2}$-approximation algorithm for the case $k \geq i_{max} + 1$. But how can we solve the problem for the case of $k \leq i_{max}$?

Assume that we are only allowed to set prices among values $1, 2, ..., 2^{i_{max}}$ at each time step. As discussed above, we can show that the optimum expected revenue in this case will be at least $\frac{1}{2}$ of the optimum expected revenue in the general case. Now, we propose an algorithm which solves the problem in the case that we are allowed to set prices to one of values $1, 2, ..., 2^{i_{max}}$ in each time step. Let $B[t, q]$ be the maximum expected revenue when we have t time steps and the state of the network is

Net-Stable(2^q). In this situation, we can set the price to $2^{q'}$ in the first time step for $q' < q$. Thus, we can calculate $B[t, q]$ as follows:

$$B[t, q] = \max_{0 \le q' < q} \{B[t-1, q'] + 2^{q'}(E[X_{2^{q'}}] - E[X_{2^q}])\}. \quad (2)$$

An issue with the above equation is that we do not have $E[X_{2^{q'}}] - E[X_{2^q}]$ precisely, but we can estimate it within a small error with high probability using standard sampling and Chernoff-Hoeffding bound. Let matrix B_s be the matrix corresponding to equation 2 when we use an estimate value for $E[X_{2^{q'}}] - E[X_{2^q}]$ instead of its exact value in this equation. Since the estimation value for $E[X_{2^{q'}}] - E[X_{2^q}]$ is within a small error of the exact value with high probability, using union bound, we can show that the following inequalities also hold with high probability:

$$(1-\epsilon)B[t, q] \le B_s[t, q] \le (1+\epsilon)B[t, q].$$

At the end, the expected revenue of the solution will be stored in $B_s[k, i_{max}+1]$. In order to compute the sequence of prices, we can store the index q' which maximizes $B_s[t-1, q'] + 2^{q'}(E[X_{2^{q'}}] - E[X_{2^q}])$ while computing $B_s[t, q]$. By storing these values, we can compute the price at time step $1 \le i \le t$ by following an appropriate sequence of $[t, q]$ pairs in the dynamic program. Since $B[q, t]$ is between of $(1-\epsilon)B_s[q, t]$ and $(1+\epsilon)B[q, t]$ with high probability, the above algorithm produces a sequence of prices $(2^{p_1}, 2^{p_2}, \ldots, 2^{p_k})$ whose expected revenue is within a $\frac{(1-\epsilon)}{2(1+\epsilon)}$ of the optimum.

Changing the algorithm to an FPTAS. We can easily change the constant-factor dynamic-programming-based algorithm discussed above and give an algorithm with approximation factor $\frac{1-\epsilon}{(1+\epsilon)^2}$. To do so, we should consider the prices of form $(1+\epsilon)^i$ instead of 2^i. The term $\log p_{max}$ will be replaced with $\log_{1+\epsilon}^{p_{max}}$ in the running time and the approximation factor is $\frac{1-\epsilon}{(1+\epsilon)^2}$.

E Experiments for Preferential Attachmant Networks

In addition to finding efficient algorithms, studying the structural properties of the problem and optimal solutions are desirable. Equipped with the FPTAS presented in the paper, we present experimental results studying the form of the optimal solution and its connection to the underlying social network. Throughout our simulation results, we obtain insights about properties of the optimal price sequence, e.g., the seller achieves a large proportion of the maximum achievable revenue using a small number of price changes.

We now describe our simulation framework. We first describe how we generate a random network using the *preferential attachment process* [2]. In this process, nodes of the social network are added one by one to the graph. Each new node is connected to a number of nodes in the existing graph, where the probability of connecting to an existing node is proportional to the degree of that node in the graph. The random graph has degree parameter, ℓ, which is a constant and is the number of edges attached to a new node. Formally, to generate a network of n nodes, we start the process with a complete graph of size ℓ, and then expand this graph to an n-node graph in $n - \ell$ steps

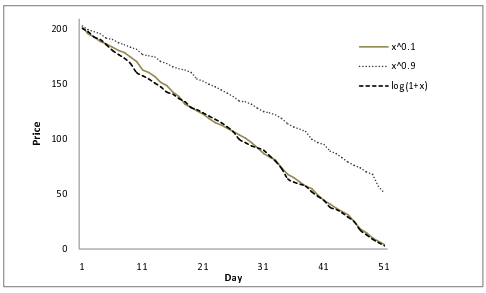

Figure 3: The optimal price sequence for uniform distribution.

by adding a new node at each step, and connecting this node to ℓ existing nodes at random, proportional to the degree of each existing node.

Given a social network with the neighborhood set N_i for any node i in the network, we consider the special case of valuation functions in which the value of the item for a node i only depends on the *number of neighbors* of node i already adopting the item, i.e, the valuation depends on $|N_i \cap S|$ where S is the set of players already adopting the item. Formally, we define the valuation of a player i to be :

$$v_i(S) = v_i^1 + v_{i,d}^2,$$

where v_i^1 and $v_{i,d}^2$ are random variables with distributions F^1 and F_d^2, and $d = |N_i \cap S|$. v_i^1 shows the initial (individual) valuation of the node i, and $v_{i,d}^2$ shows the externality that her neighborhood imposed on her. We consider uniform and Normal distributions. In case of uniform distributions, F^1 is uniform on $[0, 2m]$ and in case of Normal distribution, F^1 is defined to be the Normal distribution with mean m and standard deviation $m/2$, for some value m. To define F_d^2, we consider a function f as a function of d. More specifically, in case of uniform distributions, F_d^2 is uniform on $[0, 2f(\alpha d)]$ and in case of Normal distribution, F_d^2 is defined to be the Normal distribution with mean $f(\alpha d)$ and standard deviation $f(\alpha d)/2$, where α is a fixed scaling factor. In our simulations, we consider the case where $f(x) = x^c$ for some constant c, or $f(x) = \log(x + 1)$. In all examples, $\alpha = 20$, mean of distributions F^1 is 100, and the number of nodes in the network is 200.

Figure 3 shows the optimal price sequence for the case of uniform distributions. In this example, the number of days (or price changes) is 50. As can be observed in the figure, no matter what the function f is, the optimal price sequence decreases almost linearly for uniform distributions.

In Figure 4, we study the increase in the expected revenue as a function of the number of price changes. In other words, we modify the maximum number of price changes (or days), and observe the maximum revenue that can be achieved as the number of price changes increases. This figure shows that with a limited number of price changes (say between 10 and 15 days), the seller can extract the revenue that is achievable by many more price changes (like 100 days). In particular, we observe that no

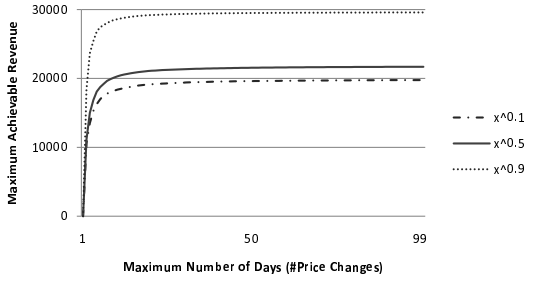

Figure 4: Maximum achievable revenue vs. the allowed number of price changes.

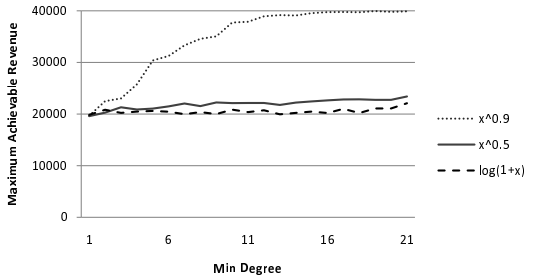

Figure 5: Maximum achievable revenue vs. the density of the network.

matter what the influence functions or the distributions are, the maximum achievable revenue increases very fast at the beginning as we increase the number of allowed price changes, and then it does not increase much. Moreover, one can observe that the more the influence is, the less number of price changes is required to achieve the maximum achievable revenue (e.g., for $f(x) = x^{0.9}$ the achievable revenue does not increase much after 8 days, as opposed to 16 days for $f(x) = x^{0.1}$).

Finally, we study the effect of the density of the social network and the amount of the influence by neighbors on the maximum achievable revenue. As seen in figure 5, as the externality effects increase, the revenue becomes more sensitive to the density of the graph.

F Missing Proofs

F.1 Missing Proofs of Section 2

Proof of Lemma 2: By increasing p to the smallest price in β which is larger than p, we would achieve a better revenue without losing any customers. For the extreme case where $p > \beta_1$, we can decrease p to β_1 to achieve a better revenue. ■

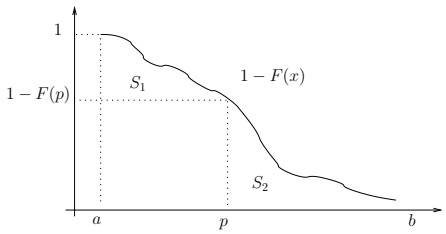

Figure 6: the graph of $1 - F(x)$. The expectation of f is the area under the graph, partitioned to S_1 and S_2, and p is the value maximizing $p(1 - F(p))$.

F.2 Missing Proofs of Section 3.1

Proof of Lemma 5: We know that $\mu = \int_a^b 1 - F(t)dt$, which is the area under the graph of figure 6. Also, R is the area of the largest rectangle under that graph. Let p be the price for which $p(1 - F(p))$ is maximized. The area under the graph is the sum of two parts, first the integral from a to p, named S_1, and then from p to b, named S_2. According to lemma 4, we know that $1 - F(p) \geq 1/e$. As a result, we can conclude that $S_1 \leq 1 \cdot p \leq e(1 - F(p))p = eR$. Also, p is the value maximizing $p(1-F(p))$, so we have $h(p) = 1/p$ (by setting $p(1-F(p))' = 1-F(p)-pf(p) = 0$). Also, since $h(p)$ is a non-decreasing function, we know that for any $p' \geq p$, we have $h(p') \geq h(p)$. Therefore, we conclude that $h(p') = \frac{f(p')}{1-F(p')} \geq h(p) = 1/p$, and thus $f(p') \geq (1 - F(p'))/p$. Integrating both sides from p to b we have $\int_p^b f(t)dt \geq (1/p)\int_p^b(1 - F(t))dt$. But $\int_p^b(1 - F(t))$ is equal to S_2, and $\int_p^b f(t)dt$ is equal to $1-F(p)$. Therefore we have $S_2 \leq p(1-F(p))$. So the area under the graph, $S_1 + S_2$, is at most $(1 + e)R$. Note that we do not have any conditions on a and b. ■

Proof of Lemma 6: Assume for each $1 \leq j \leq m$, $a_j < \frac{\sum_{j=1}^m a_j}{j \lceil \log(m+1) \rceil}$. By summing these inequalities we have: $\sum_{j=1}^m a_j < (\sum_{j=1}^m a_j)(\sum_{j=1}^m \frac{1}{j \lceil \log(m+1) \rceil}) \Rightarrow \lceil \log(m + 1) \rceil < \sum_{j=1}^m \frac{1}{j}$, a contradiction. ■

Proof of Lemma 7: To give some intuition on the problem, assume the function $f : [0, n] \rightarrow R$ such that for x between $i - 1$ and i, $f(x) = a_i$. Our problem is to fit k rectangles under the graph of f, such that the total covered area by the rectangles is maximized.

We present an algorithm that iteratively selects rectangles, such that after the m-th step the total area covered by the rectangles is at least $m/\log n$ using $4^m - 1$ rectangles. At the start of the m-th step, the uncovered area is partitioned into 4^{m-1} independent parts. In addition, the length of the lower edge of each of these parts is at most $n/(2^{m-1})$. The algorithm solves each of these parts independently as follows. For each part $p \leq 4^{m-1}$, we use s_p to denote the area of that part, and e_p to denote the length of the lower edge of that part (see figure 7). We use 3 rectangles for each part in each step. First, using lemma 6 we know that we can use a single rectangle to cover at least $1/\log e_p$ of the total area of part p. Then, we cover the two resulting uncovered

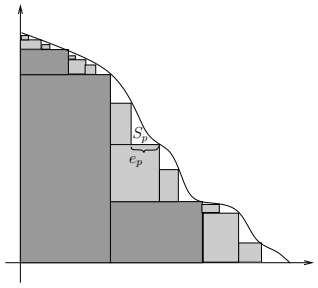

Figure 7: The darker rectangles are selected at the first step, and the lighter ones at the second step.

parts by two rectangles, which each equally divide the lower edge of the corresponding part. As a result, four new uncovered parts are created, each with a lower edge with length less than $e_p/2$, therefore satisfying the necessary conditions for the next step.

The area that we cover in each step of the algorithm is at least $\sum_p \frac{s_p}{\log e_p} \geq \frac{\sum_p s_p}{\log n-(m-1)}$ (since $e_p \leq \frac{n}{2^{m-1}}$). The fraction of the total covered area by the algorithm after step m, assuming that at each step we cover exactly $\frac{1}{\log n-(m-1)}$ of the remaining area, is $\sum_{i=1}^{m} \frac{1}{\log n-(i-1)} \cdot \frac{\log n-(i-1)}{\log n} = \frac{m-1}{\log n}$. And if at any step i we cover more than $\frac{1}{\log n-(i-1)}$ of the remaining area, the algorithm still covers at least $\frac{m-1}{\log n}$ of the entire area after m steps. As a result, after step m, we have used $4^m - 1 = k$ rectangles, and have covered $\frac{m-1}{\log n} \in \Theta(\frac{\log k}{\log n})$ of the entire area. ■

F.3 Missing Proofs of Section 3.2

In order to prove Theorem 2, we should prove following lemmas.

Lemma 8. *If an independent set of size K exists in G, the maximum revenue is at least L^2.*

Proof: We shall describe an algorithm that uses the independent set S to gain a revenue of at least L^2. The algorithm works as follows. For the first N days, at day i, if $i \notin S$ it sets the price to be an amount so that only a_i (and not d_i) would buy. Otherwise the price is set so that only d_i and a_i would buy. Stop this at the day D in which the the K-st buys. We can no longer sell to the players in A because the valuations of b_1 and b_2 are now K.

Knowing that all vertices in S are independent, there is no vertex in F with a valuation more that $K-1$; no two endpoints of any edge in G has been chosen. On the other hand, $|S| = K$, hence the value of b_1 and b_2 is equal to K. So on day $D+1$ we can sell the item to both of the b buyers, without selling it to any other buyer, by setting the price to be K. Then these two buyers would increase the value of buyer c up to K, again allowing us to sell only to c at the next day.

When we manage to sell to c without selling to any of the vertices in F, the valuations of vertices in X increases to K, while keeping the valuations of vertices in Y at $K-1$. We set the price to be K on day $D+3$, causing all the vertices in X to buy. This results in an large influence on each of the vertices in Y; the value of the those buyers would rise to $L+K-1$. If we set the price to that value on day $D+4$ (the last day), all of the L buyers $y_1 \dots y_L$ would buy the item, and we can gain a total revenue of more than L^2. ■

Lemma 9. *If there is no independent set of size K in G, it is impossible to sell the item to the buyers in X before the buyers in Y buy it.*

Proof: Let d be the day the buyers in X buy the item. Note that they all buy at the same day, if they do buy at all. At that day, none of the buyers in Y should have been influenced by any of the buyers in F, otherwise the value of the buyers in Y would have risen to K, which is equal to the upper bound for the value of any buyer in X.

To have the buyers in X buy the item before the buyers in Y do, we must sell the item to c before we sell to any y_i. This is because initially the value of any buyer in X is equal to the value of any buyer in Y and the only edge going into any buyer of X comes from c. It is obvious that before day d, we can not set a price equal to or less than $K-1$, or else the y buyers would have bought it. Therefore c must have paid more than $K-1$ for the item. The only edges entering c are from b_1 and b_2. So before day $d-1$ buyers b_1 and b_2 must have bought the item. Similarly, these two buyers have paid more than $K-1$ for the item. Since their value is always an integer, they should have paid at least K. This means that before day $d-2$, at least K of the d_i buyers have bought the item. If any of these d_i buyers represent two endpoints of any edge in G. In this case, the valuation of at least one of the players in F will increase to K, and she will buy no later than c, which will cause the players in Y to be influenced.

Therefore if the item can be sold to the buyers of X before the buyers of Y buy it, there exists a set of at least K vertices in G which are independent. ■

Theorem 4. *The special variant of the* Rapid(k) *problem defined in the section 3.2 can be approximated within a factor $\frac{1}{n^{1-\epsilon}}$ using a polynomial-time algorithm only if the independent set problem is solvable in polynomial time.*

Proof: Our goal is to set L to be so large, that the L^2 edges between X and Y becomes unavoidable. More specifically, if the optimal revenue is greater than or equal to L^2, we have no choice but to sell the item to all the Y buyers for price at least L; the revenue that is achievable from the rest of the buyers is negligible (less the $\frac{1}{n^{1-\epsilon}} \cdot L^2$). Although we should make sure that L is polynomial relative to $N = |V|$; otherwise our reduction algorithm would not be polynomial.

We first need to find a suitable value for L. Assuming the maximum revenue is at least L^2, our revenue must be at least $\frac{1}{n^{1-\epsilon}}L^2$. If we do not sell the item for price L to all y buyers, the maximum revenue would be $(n-L-2)K + L(N^2+N) + 2N$. The upper bound of the value of any buyer is K, except b_1 and b_2 and buyers in Y.

The counters have an upper bound of N, and the Y buyers have an upper bound of $|E|+K-1 \leq N^2+N$. We must specify L to be such that $\frac{1}{n^{1-\epsilon}}L^2 > (n-L-2)K + L(N^2+N) + 2N$ holds true. We know that $(n-L-2)K + L(N^2+N) + 2N < n(N^2+N) + 2N < 2nN^2$. Therefore it is enough to prove $\frac{1}{n^{1-\epsilon}}L^2 > 2nN^2$ which is equivalent to $L^2 > 2n^{2-\epsilon}N^2$. Now considering that $n = 2L + 2N + |E| + 3 \leq 2L + 2N + N^2 + 3$, if we assume $L \geq N^2 + 2N + 3$, we conclude $n \leq 3L$. Therefore it is enough to show that $L^2 > (2 \cdot 3^{2-\epsilon})L^{2-\epsilon}N^2$. Taking a logarithm from both sides and replacing L by N^α we get $\alpha > \frac{1}{\epsilon}(2 + \frac{\log(2 \cdot 3^{2-\epsilon})}{\log N})$.

So the lemma is correct if we set L to be the maximum of $(N^2 + 2N + 3)$ and $N^{1+\frac{1}{\epsilon}(2+\frac{\log(2 \cdot 3^{2-\epsilon})}{\log N})}$.

Now we prove the theorem. If such an approximation algorithm exists, we can solve the independent set problem by the described reduction. By running the algorithm on the constructed instance, and based on lemmas 8 and 9, the independent set problem has a positive answer iff the maximum revenue from the instance is at least L^2. ■

Having proved the hardness of approximation for the unweighted graphs in which $v_i(\{j\}) \in \{0, 1\}$, we now show that the problem cannot be approximated within any multiplicative factor when the edges of the corresponding graph are allowed to have arbitrary weights.

Proof of Theorem 2: We observed that the edges from set X to Y are the key to discriminate the instances of independent set problem; the only instances that have independent sets of size k can be transformed to instances of the Rapid(k) problem that can use the value of the edges from X to Y. We can use this idea to construct instances of the Rapid(k) problem that are hard to approximate. Simply replace sets X and Y by single nodes x and y. Let Δ be the weight of the edge going from x to y. Let the weight of all the other edges of G^* remain the same. From the previous lemmas, we know that we can have the revenue of at least Δ if there is an independent set of size k in G. Otherwise, the revenue is at most the value of other edges and nodes in the graph, which is at most $2N^2$. As a result, if Rapid(k) is approximable with a factor of α, we will be able to discriminate between the cases if we set $\Delta/\alpha > 2N^2$, and solve the independent set problem. ■