Maximizing Social Welfare in a Competitive Diffusion Model

Prithu Banerjee
University of British Columbia
Vancouver, Canada
prithu@cs.ubc.ca

Wei Chen
Microsoft Research
Beijing, China
weic@microsoft.com

Laks V.S. Lakshmanan
University of British Columbia
Vancouver, Canada
laks@cs.ubc.ca

ABSTRACT

Influence maximization (IM) has garnered a lot of attention in the literature owing to applications such as viral marketing and infection containment. It aims to select a small number of seed users to adopt an item such that adoption propagates to a large number of users in the network. Competitive IM focuses on the propagation of competing items in the network. Existing works on competitive IM have several limitations. (1) They fail to incorporate economic incentives in users' decision making in item adoptions. (2) Majority of the works aim to maximize the adoption of one particular item, and ignore the collective role that different items play. (3) They focus mostly on one aspect of competition – pure competition. To address these concerns we study competitive IM under a utility-driven propagation model called UIC, and study social welfare maximization. The problem in general is not only NP-hard but also NP-hard to approximate within any constant factor. We, therefore, devise instant dependent efficient approximation algorithms for the general case as well as a $(1 - 1/e - \epsilon)$-approximation algorithm for a restricted setting. Our algorithms outperform different baselines on competitive IM, both in terms of solution quality and running time on large real networks under both synthetic and real utility configurations.

PVLDB Reference Format:
Prithu Banerjee, Wei Chen, and Laks V.S. Lakshmanan. Maximizing Social Welfare in a Competitive Diffusion Model. PVLDB, 14(4): 613 - 625, 2021.
doi:10.14778/3436905.3436920

1 INTRODUCTION

Influence maximization (IM) on social and information networks is a well-studied problem that has gained a lot of traction since it was introduced by Kempe et al. [28]. Given a network, modeled as a probabilistic graph where users are represented by nodes and their connections by edges, the problem is to identify a small set of k seed nodes, such that by starting a campaign from those nodes, the expected number of users who will be influenced by the campaign, termed influence spread, is maximized. Here, the expectation is w.r.t. an underlying stochastic diffusion model that governs how the influence propagates from one node to another. The "item" being promoted by the campaign may be a product, a digital good, an innovative idea, or an opinion.

Proceedings of the VLDB Endowment, Vol. 14, No. 4 ISSN 2150-8097.
doi:10.14778/3436905.3436920

Existing works on IM typically focus on two types of diffusion models – *single item* diffusion and diffusion of *multiple items under pure competition*. The two classic diffusion models, Independent Cascade (IC) and Linear Threshold (LT), were proposed in [28]. Advances on these lines of research have led to better scalable approximation algorithms and heuristics [18, 43, 44]. Most studies on multiple-item diffusion focus on two items in pure competition [20, 35, 41, 45], that is, every node would only adopt at most one item, never both. The typical objective is to select seeds for the second item (the follower item) to maximize its number of adoptions, or minimize the spread of the first item [20].

There are a number of key issues on multiple item diffusion that are not satisfactorily addressed in most prior studies. First, most propagation models are purely stochastic, in which if a node v is influenced by a neighboring node u on certain item, it will either deterministically or probabilistically adopt the item, without any consideration of the utility of that item for the node. This fails to incorporate economic incentives into the user adoption behavior. Second, most studies focus on *pure* competition, where each node adopts at most one item, and ignore the possibility of nodes adopting multiple items. For instance, when items are involved in a partial competition, their combined utility may still be more than the individual utility, although it may be less than the sum of their utilities. Third, most studies on competition focus on the objective of maximizing the influence of one item given other items, or minimizing the influence of existing items, and do not consider maximizing the overall welfare caused by all item adoptions.

The study by Banerjee et al. [6] is unique in addressing the above issues. It proposes the utility-based independent cascade model UIC, in which: (a) each item has a utility determined by its value, price and a noise term, and each node selects the best item or itemset that offers the highest utility among all items that the node becomes aware of thanks to its neighbors' influence; and (b) the utility-based adoption naturally models the adoption of multiple items, in a framework that allows arbitrary interactions between items, based on chosen value functions. Banerjee et al. [6] study the maximization of expected social welfare, defined as the the total sum of the utilities of items adopted by all network nodes, in expectation. However, their study is confined to the *complementary* item scenario, where item utilities increase when bundled together.

In this paper, we complement the study in [6] by considering the *social welfare maximization* problem in the UIC model when items are purely or partially *competitive*. Partial (pure) competition means adopting an item makes a user less likely (resp., impossible) to adopt another item. To motivate the problem, we note that for a social network platform owner (also called the *host*), one natural objective might be to optimize the advertising revenue, as studied by Chalermsook et al. [15], or a proxy thereof, such as expected number of item adoptions. On the other hand, one of the key assets

of a network host is the loyalty and engagement of its user base, on which the host relies for its revenue from advertising and other means. Thus, while launching campaigns, it is equally natural for the host to take into account users' satisfaction by making users aware of itemsets that increase their utility. Social Welfare, being the sum of utilities of itemsets adopted by users, is directly in line with this objective.

As a real application, consider a music streaming platform such as the Last.fm. Benson et al. [7] using their discrete choice model showed existence of competition across different genres of songs in the Last.fm dataset. In a platform such as Last.fm, the platform owner (i.e., host) completely controls the promotion of songs and the host would like to keep making engaging recommendations to the users. Even when there are multiple competing songs from different genres, the host should recommend based on users' preferences, i.e., the users' utility. A similar idea extends to different competing products that an e-retailer like Amazon sells directly. Those products are already procured by the e-retailer and it has full control over how it wants to sell them. Once again, in this setting, keeping users' satisfaction from adopting these products high helps maintain a loyal and engaged user base. Thus maximizing the overall social welfare is in line with the goal of the platform. While [6] studies this problem for complementary items, social welfare maximization under competing products is open. Moreover, under pure competition, the bundling algorithm of [6] would lead to nodes adopting at most one of several competing items, leading to poor social welfare.

Compared to [6], we also consider a more flexible setting where the allocation of some items has been fixed (e.g., the items had the seeds selected by the host earlier) and the host is only allocating seeds for the remaining items. Once again, the objective is still to maximize the total social welfare of all users in the network. We call this the CWelMax problem (for Competitive Welfare Maximization). As it turns out, CWelMax under UIC is *significantly more difficult than the welfare maximization problem in the complementary setting studied in [6]*. We show that when treating the allocation as a set of item-node pairs, the welfare objective function is neither monotone nor submodular. Moreover, with a non-trivial reduction, we prove that CWelMax is in general NP-hard to approximate to within any constant factor. In contrast a constant approximation was possible in the setting considered in [6].

Despite all these difficulties, we design several algorithms that either provide an instance-dependent approximation guarantee in the general case, or better (constant) approximation guarantee in some special cases. In particular, we first design algorithm SeqGRD which provides a $\frac{u_{\min}}{u_{\max}}(1-\frac{1}{e}-\epsilon)$-approximation guarantee for the general CWelMax setting, where $u_{\min}$ is the minimum expected utility among all individual items, $u_{\max}$ is the expected maximum utility among all item bundles, and $\epsilon > 0$ is any small positive number. Next, when the fixed itemset is empty, we complement SeqGRD with MaxGRD, which guarantees $\frac{1}{m}(1-\frac{1}{e}-\epsilon)$-approximation, where m is the total number of items. Thus, when SeqGRD and MaxGRD work together, we can guarantee $\max(\frac{u_{\min}}{u_{\max}}, \frac{1}{m})(1-\frac{1}{e}-\epsilon)$-approximation when there are no prior allocated items. We can see that when the utility difference among items is not high or the number of items is small, the above algorithms can achieve a reasonable approximation performance. Finally, in the special case where we have a unique superior item with utility better than all other items, all other items have had their allocations fixed, and items exhibit pure competition, we design an efficient algorithm that achieves $(1-\frac{1}{e}-\epsilon)$-approximation.

We extensively test our algorithms against state-of-the-art IM algorithms under seven different utility configurations including both real and synthetic ones, which capture different aspects of competition. Our results on real networks show that our algorithms produce social welfare up to five times higher than the baselines. Furthermore, they easily scale to large networks with millions of nodes and billions of edges. We also empirically test the effect of social welfare maximization on adoption count and show that whereas the overall adoption count remains the same, social welfare is maximized by reducing adoption of just the inferior items. To summarize, our major contributions are as follows:

- We are the first to study the competitive social welfare maximization problem CWelMax under the utility-based UIC model (§3).
- We show that social welfare is neither monotone, submodular, nor supermodular; furthermore, it is NP-hard to approximate CWelMax within any constant factor, in general (§4).
- We provide several algorithms that either solve the CWelMax in the general setting with a utility-dependent approximation guarantee, or have better (constant) approximation guarantees in special cases (§5).
- We conducted an extensive experimental evaluation over several real social networks comparing our algorithms with existing algorithms. Our results show that our algorithms significantly dominate existing algorithms and validate that our algorithms both deliver good quality and scale to large networks (§6).

Background and related work are discussed in §2. We conclude the paper and discuss future work in §7. Proofs compressed or omitted for lack of space can be found in [5].

2 BACKGROUND & RELATED WORK

One Item IM: A directed graph $G = (V, E, p)$ represents a social network with users V and a set of connections (edges) E. The function $p : E \to [0, 1]$ specifies influence probabilities between users. Independent cascade (IC) model is a commonly used discrete time diffusion model [20, 28]. Given a seed set $S \subset V$, at time $t = 0$, only the seed nodes in S are active. For $t > 0$, if a node u becomes active at $t-1$, then it makes one attempt to activate its every inactive out-neighbor v, with success probability $p_{uv} := p(u, v)$. The diffusion stops when no more nodes can become active.

For a seed set $S \subset V$, we use $\sigma(S)$ to denote the *influence spread* of S, i.e., the expected number of active nodes at the end of diffusion from S. For a seed budget k and a diffusion model, *influence maximization* (IM) problem is to find a seed set $S \subset V$ with $|S| \leq k$ that the influence spread $\sigma(S)$ under the model is maximized [28].

A set function $f : 2^V \to \mathbb{R}$ is *monotone* if $f(S) \leq f(T)$ whenever $S \subseteq T \subseteq V$; *submodular* if for any $S \subseteq T \subseteq V$ and any $x \in V \setminus T$, $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$; f is *supermodular* if $-f$ is submodular; and f is *modular* if it is both submodular and supermodular.

Under the IC model, IM is intractable [18, 19, 28]. However $\sigma(\cdot)$ is *monotone* and *submodular*. Hence, using Monte Carlo simulation for estimating the spread, a simple greedy algorithm delivers a $(1-1/e-\epsilon)$-approximation to the optimal solution, for any $\epsilon > 0$ [26–28]. The concept of reverse reachable (RR) sets proposed by Borgs et al. [11], has led to a family of scalable state-of-the-art approximation algorithms such as IMM and SSA for IM [16, 25, 31, 39, 44].

Multiple item competitive IM: More recently, IM has been studied involving independent items [21], and competing items [8, 12, 24, 35, 45]. In [34] authors study the problem under pure competition, whereas [23] aims to maximize balanced exposure in the network for two competing ideas, and [37] addresses fairness in the adoption of competing items. These works, however, are restricted to specific types of competition. The Com-IC model proposed by Lu et al. [36] can model any arbitrary degree of interaction between a pair of items. Their main study is thus restricted to the diffusion of two items. Li et al. [33] look into different facets of items (e.g., topics of documents) to compute influence. However, unlike our work, they do not consider item utility in adoption decisions made by users. Furthermore, their objective function is based on traditional (expected) number of item adoptions. Our objective is to maximize the social welfare that none of these papers have studied. A more comprehensive survey on competitive influence models, can be found in [20, 32].

Social welfare maximization: Utility driven adoptions have been studied in economics [2, 10, 38, 40]. Given items and users, and the utility functions of users for various subsets of items, the problem is to find an allocation of items to users such that the sum of utilities of users, is maximized. Since the problem is intractable, approximation algorithms have been developed [22, 27, 29]. Benson et al.[7] propose a discrete choice model to learn the utilities of itemsets from the users' adoption logs. Learning utility is complementary to our work, and is used in our experiments. Moreover none of these works consider a social network and the effect of recursive propagation on item adoptions by its users.

Host's perspective in the context of IM have been studied. Aslay et al. [4] directly maximize the revenue earned by a network host, whereas [3] minimizes the regret of seed selection. These works do not consider the overall social welfare. Utility based adoption decisions of users are also not part of their formalism. Welfare maximization on social networks has been studied in a few recent papers[9, 42]. Bhattacharya et al. [9] consider item allocations to nodes for welfare maximization in a network with network externalities. Their model does not consider the effect of recursive propagation nor competition. In addition, they do not consider budget constraints. In contrast, our focus is on competition, with budget constraints on every item.

Banerjee et al. [6] studied welfare maximization under viral marketing using the UIC propagation model that we also use. However, their work focuses strictly on complementary items, with supermodular value functions. As a result, their objective is monotone, and further satisfies a nice "reachability" property (details in §4), which paved the way for efficient approximation. However, such complementary-only setting ignores many real world scenarios where competing items are present, as highlighted in the introduction. We instead focus on *competing* items. Consequently, the objective becomes not only non-monotone, non-submdular, and non-supermodular, but unlike in [6], is inapproximable within any constant factor. In spite of this, we develop utility dependent approximation algorithms as well as a constant approximation algorithm for special cases.

In summary, to our knowledge, *our study is the first to address social welfare maximization in a network with influence propagation, competing items, and budget constraints, where item adoption is driven by utility.*

3 UIC MODEL UNDER COMPETITION

In this section, we first briefly review the *utility driven independent cascade* model (UIC for short) proposed in [6]. Then we describe the competitive setting of UIC studied in this paper and formally state the new problem we address.

Review of UIC Model: UIC integrates utility driven adoption decision of nodes, with item propagation. Every node has two sets of items – *desire set* and *adoption set*. Desire set is the set of items that the node has been informed about (and thus potentially desires), via propagation or seeding. Adoption set, is the subset of the desire set that has the highest utility, and is adopted by the user. The utility of an itemset $I \subseteq \mathbf{I}$ is derived as $\mathcal{U}(I) = \mathcal{V}(I) - \mathcal{P}(I) + \mathcal{N}(I)$, where $\mathcal{V}(\cdot)$ denotes users' latent valuation for an itemset, $\mathcal{P}(\cdot)$ denotes the price that user needs to pay, and $\mathcal{N}(\cdot)$ is a random noise term that denotes our uncertainty in users' valuation.

Budget vector $\vec{b} = (b_1, ..., b_{|\mathbf{I}|})$ represents the budgets associated with the items, i.e., the number of seed nodes that can be allocated with that item. An *allocation* is a relation $\boldsymbol{\mathcal{S}} \subset V \times \mathbf{I}$ such that $\forall i \in \mathbf{I} : |\{(v, i) \mid v \in V\}| \leq b_i$. $S_i^{\boldsymbol{\mathcal{S}}} := \{v \mid (v, i) \in \boldsymbol{\mathcal{S}}\}$ denotes the *seed nodes* of $\boldsymbol{\mathcal{S}}$ for item i and $S^{\boldsymbol{\mathcal{S}}} := \bigcup_{i \in \mathbf{I}} S_i^{\boldsymbol{\mathcal{S}}}$. When the allocation $\boldsymbol{\mathcal{S}}$ is clear from the context, we write S (resp., S_i) to denote $S^{\boldsymbol{\mathcal{S}}}$ (resp., $S_i^{\boldsymbol{\mathcal{S}}}$).

Before a diffusion begins, the noise terms of all items are sampled, and they are used until the end of that diffusion. The diffusion proceeds in discrete time steps, starting from $t = 1$. $\mathcal{R}^{\boldsymbol{\mathcal{S}}}(v, t)$ and $\mathcal{A}^{\boldsymbol{\mathcal{S}}}(v, t)$ denote the desire and adoption sets of node v at time t. At $t = 1$, the seed nodes have their desire sets initialized according to the allocation $\boldsymbol{\mathcal{S}}$ as, $\mathcal{R}^{\boldsymbol{\mathcal{S}}}(v, 1) = \{i \mid (v, i) \in \boldsymbol{\mathcal{S}}\}, \forall v \in S^{\boldsymbol{\mathcal{S}}}$. These seed nodes then adopt the subset of items from the desire set that maximizes the utility. The propagation then unfolds recursively for $t \geq 2$ in the following way. Once a node u' adopts an item i at time $t-1$, it influences its out-neighbor u with probability $p_{u'u}$, and if it succeeds, then i is added to the desire set of u at time t. Subsequently u adopts the subset of items from the desire set of u that maximizes the utility. Adoption is progressive, i.e., once a node adopts an item, it cannot unadopt it later. Thus $\mathcal{A}^{\boldsymbol{\mathcal{S}}}(u, t) = \arg\max_{T \subseteq \mathcal{R}^{\boldsymbol{\mathcal{S}}}(u,t)} \{\mathcal{U}(T) \mid T \supseteq \mathcal{A}^{\boldsymbol{\mathcal{S}}}(u, t-1) \ \wedge \ \mathcal{U}(T) \geq 0\}$. The propagation converges when there is no new adoption in the network. For more details readers are referred to [6].

Social welfare maximization relative to a fixed seed set: Let $G = (V, E, p)$ be a social network, $\mathbf{I}$ the universe of items under consideration. We consider a utility-based objective called *social welfare*, which is the sum of all users' utilities of itemsets adopted by them after the propagation converges. Formally, $\mathbb{E}[\mathcal{U}(\mathcal{A}^{\boldsymbol{\mathcal{S}}}(u))]$ is the expected utility that a user u attains for a seed allocation $\boldsymbol{\mathcal{S}}$ after the propagation ends. The *expected social welfare* for $\boldsymbol{\mathcal{S}}$, is

$\rho(\boldsymbol{\mathcal{S}}) = \sum_{u \in V} \mathbb{E}[\mathcal{U}(\mathcal{A}^{\boldsymbol{\mathcal{S}}}(u))]$, where the expectation is over both the randomness of propagation and noise terms $\mathcal{N}(.)$

In a social network, a campaign may often be launched on top of other existing campaigns, where the seeds for some items $I_1 \subset \mathbf{I}$ may already be fixed. Let $\boldsymbol{\mathcal{S}}^P$ be this fixed allocation for items in I_1. Then $I_2 = \mathbf{I} \setminus I_1$ is the set of items for which the seeds are to be selected. We define the problem of maximizing expected social welfare, on top of a fixed seed allocation as follows.

Welfare maximization under competition: Motivated by economics theory [13], to model competition, we assume that $\mathcal{V}$ is submodular, i.e., as I contains more items, the marginal value of an item with respect to an itemset $I \subset \mathbf{I}$ decreases. We also assume $\mathcal{V}$ is monotone and that $\mathcal{V}(\emptyset) = 0$, since it is a natural property for valuations. For $i \in \mathbf{I}$, $\mathcal{N}(i) \sim \mathcal{D}_i$ denotes the noise term associated with item i, where the noise may be drawn from any distribution $\mathcal{D}_i$ having a zero mean. Every item has an independent noise distribution. For a set of items $I \subseteq \mathbf{I}$, we assume the noise and price to be additive. Since noise is drawn from a zero mean distribution, $\mathbb{E}[\mathcal{U}(I)] = \mathcal{V}(I) - \mathcal{P}(I)$. Below, we refer to $\mathcal{V}, \mathcal{P}, \{\mathcal{D}_i\}_{i \in \mathbf{I}}$, as the model parameters and denote them collectively as Param.

We now illustrate using a toy example how our framework models competition.

Example 1. *Suppose a user wants to buy a phone and desires (because of influence) an iPhone (ip) and an Android Phone (ap). Since the user does not own a phone yet, she enjoys no valuation, which is captured by $\mathcal{V}(\emptyset) = 0$. However after she adopts one phone, then although the overall value increases by adopting a second phone ($\mathcal{V}(.)$ is monotone), the marginal value gain decreases ($\mathcal{V}(.)$ is submodular). Formally $\mathcal{V}(\{ip, ap\}) = \mathcal{V}(\{ip\}) + \mathcal{V}(\{ap\} \mid \{ip\})$. Since value is monotone and submodular, $0 \leq \mathcal{V}(\{ap\} \mid \{ip\}) \leq \mathcal{V}(\{ap\})$, hence $\mathcal{V}(\{ip\}) \leq \mathcal{V}(\{ip, ap\}) \leq \mathcal{V}(\{ip\}) + \mathcal{V}(\{ap\})$. Finally price determines the exact adoption set of a user. The itemset that offers the highest utility is adopted by the user. If the second phone has a low price (i.e., $\mathcal{P}(\{ap\}) \leq \mathcal{V}(\{ap\} \mid \{ip\})$), then the user may still adopt both the phones (partial competition). Otherwise (i.e., $\mathcal{P}(\{ap\}) > \mathcal{V}(\{ap\} \mid \{ip\})$), a user who has already adopted ip will not adopt ap.*

Problem 1 (CWelMax). *Given $G = (V, E, p)$, the set of model parameters* Param*, an existing fixed allocation $\boldsymbol{\mathcal{S}}^P$, and budget vector $\vec{b}$, find a seed allocation $\boldsymbol{\mathcal{S}}^*$ for items I_2, such that $\forall i \in I_2, |S_i^{\boldsymbol{\mathcal{S}}^*}| \leq b_i$ and $\boldsymbol{\mathcal{S}}^* = \arg\max_{\boldsymbol{\mathcal{S}}} \rho(\boldsymbol{\mathcal{S}} \cup \boldsymbol{\mathcal{S}}^P)$.*

Note that this problem subsumes the typical "fresh campaigns" setting as a special case where $I_1 = \emptyset$ (and hence $\boldsymbol{\mathcal{S}}^P = \emptyset$).

An equivalent possible world model: In [6], the authors proposed an equivalent possible world interpretation of the diffusion under UIC, which we will find useful. We briefly review this below. Let $\langle G, \text{Param} \rangle$ be an instance of CWelMax, where $G = (V, E, p)$. A *possible world* $w = (w_1, w_2)$, consists an *edge possible world* (edge world) w_1, and a *noise possible world* (noise world) w_2: w_1 is a deterministic graph sampled from the distribution associated with G, where each edge $(u, v) \in E$ is sampled in with an independent probability of p_{uv}; and w_2 is a sample of noise terms for items in $\mathbf{I}$, drawn from noise distributions in Param. Note that propagation and adoption in w is fully deterministic. In a possible world w, $\mathcal{N}_w(i)$ is the noise for item i and $\mathcal{U}_w(I)$ is the (deterministic) utility of itemset I. The *social welfare* of an allocation $\boldsymbol{\mathcal{S}}$ in w is $\rho_w(\boldsymbol{\mathcal{S}}) := \sum_{v \in V} \mathcal{U}(\mathcal{A}_w^{\boldsymbol{\mathcal{S}}}(v))$, where $\mathcal{A}_w^{\boldsymbol{\mathcal{S}}}(v)$ is the adoption set of v at the end of the propagation in world w. The *expected social welfare* of an allocation $\boldsymbol{\mathcal{S}}$ is $\rho(\boldsymbol{\mathcal{S}}) := \mathbb{E}_w[\rho_w(\boldsymbol{\mathcal{S}})] = \mathbb{E}_{w_1}[\mathbb{E}_{w_2}[\rho_w(\boldsymbol{\mathcal{S}})]] = \mathbb{E}_{w_2}[\mathbb{E}_{w_1}[\rho_w(\boldsymbol{\mathcal{S}})]]$.

4 PROPERTIES OF UIC

It is easy to see that CWelMax is NP-hard.

Proposition 1. *CWelMax in the UIC model is NP-hard.*

Sketch. Classic IM is a special case of CWelMax. □

Given the hardness, we examine whether social welfare satisfies monotonicity, submodularity or supermodularity.

Item blocking. Under the *complementary setting* in [6] leveraged the reachability property: if a node v adopts an item i in any possible world w, then all the other nodes that are reachable from v in w will also adopt i. This property does *not* hold under the competitive setting. In fact, adoption of one particular item can block the propagation of another item, making social welfare non-monotone and non-submodular.

Theorem 4.1. *Expected social welfare is not monotone, and neither submodular nor supermodular, with respect to sets of node-item allocation pairs.*

Proof. We show a counterexample for each of the three properties. Consider a simple network with two nodes u and v, and a directed edge (u, v) with probability 1. Assume that there is no noise, i.e., noise is 0. There are three items in propagation whose utility configuration is shown in Table 1.

Item	$\mathcal{V}$	$\mathcal{P}$	$\mathcal{U}$
$\emptyset$	0	0	0
i_1	5	1	4
i_2	7	4	3
i_3	5	1	4
i_1, i_2	7	5	2
i_1, i_3	7	2	5
i_2, i_3	7	5	2
i_1, i_2, i_3	7	6	1

Table 1: Utility configuration used in Theorem 4.1

Monotonicity. Consider two allocations $\boldsymbol{\mathcal{S}}^1 = \{(u, i_1)\}$ and $\boldsymbol{\mathcal{S}}^2 = \{(u, i_1), (v, i_2)\}$. Clearly $\boldsymbol{\mathcal{S}}^1 \subset \boldsymbol{\mathcal{S}}^2$. Under $\boldsymbol{\mathcal{S}}^1$, both u and v adopt i_1, thus $\rho(\boldsymbol{\mathcal{S}}^1) = 8$. However under $\boldsymbol{\mathcal{S}}^2$, u adopts i_1 but v adopts i_2. Thus $\rho(\boldsymbol{\mathcal{S}}^2) = 7 < \rho(\boldsymbol{\mathcal{S}}^1)$.

Submodularity. Consider $\boldsymbol{\mathcal{S}}^1 = \{(v, i_2)\}$, $\boldsymbol{\mathcal{S}}^2 = \{(v, i_2), (v, i_3)\}$ and (u, i_1). Clearly $\boldsymbol{\mathcal{S}}^1 \subset \boldsymbol{\mathcal{S}}^2$ and $(u, i_1) \notin \boldsymbol{\mathcal{S}}^2$. Under $\boldsymbol{\mathcal{S}}^1$, only v adopts i_2. Under $\boldsymbol{\mathcal{S}}^1 \cup \{(u, i_1)\}$, u adopts i_1 and v adopts i_2. So $\rho(\boldsymbol{\mathcal{S}}^1 \cup \{(u, i_1)\}) - \rho(\boldsymbol{\mathcal{S}}^1) = 4$. Under $\boldsymbol{\mathcal{S}}^2$, v adopts i_3. Under $\boldsymbol{\mathcal{S}}^2 \cup \{(u, i_1)\}$, u adopts i_1 and v adopts i_1 and i_3. So $\rho(\boldsymbol{\mathcal{S}}^2 \cup \{(u, i_1)\}) - \rho(\boldsymbol{\mathcal{S}}^2) = 5 > \rho(\boldsymbol{\mathcal{S}}^1 \cup \{(u, i_1)\}) - \rho(\boldsymbol{\mathcal{S}}^1)$.

Supermodularity. Consider $\boldsymbol{\mathcal{S}}^1 = \emptyset$, $\boldsymbol{\mathcal{S}}^2 = \{(v, i_2)\}$ and (u, i_1). Clearly $\boldsymbol{\mathcal{S}}^1 \subset \boldsymbol{\mathcal{S}}^2$ and $(u, i_1) \notin \boldsymbol{\mathcal{S}}^2$. Under $\boldsymbol{\mathcal{S}}^1$, there is no adoption by any node. Under $\boldsymbol{\mathcal{S}}^1 \cup \{(u, i_1)\}$, u and v both adopt i_1. So $\rho(\boldsymbol{\mathcal{S}}^1 \cup \{(u, i_1)\}) - \rho(\boldsymbol{\mathcal{S}}^1) = 8$. Under $\boldsymbol{\mathcal{S}}^2$, v adopts i_2. Under $\boldsymbol{\mathcal{S}}^2 \cup \{(u, i_1)\}$, u adopts i_1 and v adopts i_2 . So $\rho(\boldsymbol{\mathcal{S}}^2 \cup \{(u, i_1)\}) - \rho(\boldsymbol{\mathcal{S}}^2) = 4 < \rho(\boldsymbol{\mathcal{S}}^1 \cup \{(u, i_1)\}) - \rho(\boldsymbol{\mathcal{S}}^1)$. □

The absence of these properties makes CWelMax really hard to approximate. In fact, there is no *PTIME* approximation algorithm within any constant, for the general version of CWelMax.

Theorem 4.2. *No PTIME algorithm can approximate CWelMax within any constant factor c, $0 < c \leq 1$, unless P = NP.*

Proof. We prove the theorem by a gap introducing reduction from SET COVER. Suppose there is a PTIME c-approximation algorithm $\mathcal{A}$ for CWelMax, for some $0 < c \leq 1$. Given an instance $\mathcal{I} = (\mathcal{F}, X)$ of SET COVER, where $\mathcal{F} = \{S_1, ..., S_r\}$ is a collection of subsets over a set of ground elements $X = \{g_1, ..., g_n\}$, and a number k ($k < r < n$), the question is whether there exist k subsets from $\mathcal{F}$ that cover all the ground elements, i.e., whether $\exists \mathcal{C} \subset \mathcal{F} : |\mathcal{C}| = k$ and $\cup_{S \in \mathcal{C}}^{k} S = X$. We can transform $\mathcal{I}$ in polynomial time to an instance $\mathcal{J}$ of CWelMax.

As an overview, our reduction shows that for a YES-instance of SET COVER, the optimal expected welfare in the corresponding CWelMax instance is high and for a NO-instance, it is low. More precisely, let x_y^* (resp., x_n^*) be the optimal welfare on the transformed instance $\mathcal{J}$ whenever the given instance $\mathcal{I}$ is a YES-instance (resp., NO-instance). Our reduction ensures that $x_n^* < cx_y^*$. In this case, running $\mathcal{A}$ on $\mathcal{J}$ will clearly allow us to decide if $\mathcal{I}$ is a YES-instance or not, which is impossible unless P = NP. The complete proof requires several non-trivial gadgets which are omitted here for the space constraint. It can be found in our full report [5] □

5 APPROXIMATION ALGORITHMS

Since the CWelMax problem cannot be approximated within any constant factor in general, in this section we propose several approximation algorithms that either produce a non-constant approximation guarantee dependent on the problem instance or a constant approximation guarantee for a special case of CWelMax. We first define some important notions.

Truncated utility. For accounting the social welfare of an allocation, we develop the notion of *truncated utility* of an item. Recall that when the noise of an item makes its utility negative, no node adopts the item. Hence what contributes to the final expected social welfare is the set of non-negative contributions to utility. We call this the *truncated utility*, denoted $\mathcal{U}^+(I) := max(0, \mathcal{U}(I))$. Thus for a (node, item) allocation pair (v, i), its expected social welfare (when there are no other allocations) is $\rho(v, i) = \mathbb{E}[\mathcal{U}^+(i)]\sigma(\{v\})$, where $\sigma(\{v\})$ is the influence spread of $\{v\}$.

Minimum and maximum utility bundle. We define $u_{\min} = \min_{i \in \mathbf{I}} \mathbb{E}[\mathcal{U}^+(i)]$ as the minimum expected truncated utility of any item in $\mathbf{I}$, and $u_{\max} = \mathbb{E}[\max_{I \subseteq \mathbf{I}} \mathcal{U}^+(I)]$ as the expected maximum truncated utility of any item *bundle* in $\mathbf{I}$. Note that the definitions of $u_{\min}$ and $u_{\max}$ are not symmetric: (a) $u_{\min}$ takes the minimum of an expectation, while $u_{\max}$ takes the expectation of a maximum; and (b) $u_{\min}$ takes minimum on single items while $u_{\max}$ takes maximum among all bundles. The reason of this asymmetry will be clear in our analysis.

Superior and inferior item. A given itemset $\mathbf{I}$ is said to have a *superior* item i_m, if the least possible utility of i_m is strictly higher than the highest possible utility of any item in $\mathbf{I} \setminus \{i_m\}$. Notice the definition of superior item entails that the noise distribution should be bounded in some way. We discuss a practical way to bound the noise in our experiments (§6). Given a superior item, all the other items of the itemset are called *inferior* items.

In what follows, we present three different algorithms with progressively better theoretical guarantees, under progressively stronger assumptions. As a preview, our first algorithm SeqGRD provides a $\frac{u_{\min}}{u_{\max}}(1-\frac{1}{e})$-approximation in the most general case. Our second algorithm, MaxGRD, assumes no prior allocations, i.e., $\boldsymbol{\mathcal{S}}^P = \emptyset$. Under this assumption, it provides a $\frac{1}{m}(1-\frac{1}{e})$-approximation, where m is the number of items. By simply returning the better of the two allocations produced by SeqGRD and MaxGRD, the bound is improved to $max\{\frac{u_{\min}}{u_{\max}}, \frac{1}{m}\}(1-\frac{1}{e})$, when $\boldsymbol{\mathcal{S}}^P = \emptyset$. Our final algorithm SupGRD assumes that there exists a superior item in the itemset, the allocations for all inferior items are fixed, and that items exhibit pure competition. Under these assumptions, it provides a $(1-\frac{1}{e})$-approximation.

5.1 SeqGrd Algorithm

The pseudocode of SeqGRD is shown in Algorithm 1.

Algorithm 1 SeqGRD($G, \epsilon, \ell, \boldsymbol{\mathcal{S}}^P, I_2, \vec{b}$)

1: $S^P \leftarrow$ Seed nodes of the allocation $\boldsymbol{\mathcal{S}}^P$
2: $S^{Seq} \leftarrow \mathsf{PRIMA}^+(G, \epsilon, \ell, S^P, \vec{b}, \sum_{i \in I_2} b_i)$
3: $\boldsymbol{\mathcal{S}}^{Seq} \leftarrow \emptyset$
4: Sort I_2 in decreasing order of the expected truncated utility
5: $Added \leftarrow \emptyset$
6: **for** $i \in I_2$ **do**
7: $\quad S_i^{Seq} \leftarrow$ top b_i nodes from S^{Seq}
8: $\quad$ **if** $\rho(\boldsymbol{\mathcal{S}}^{Seq} \mid \boldsymbol{\mathcal{S}}^P) < \rho((\boldsymbol{\mathcal{S}}^{Seq} \cup (S_i^{Seq} \times \{i\})) \mid \boldsymbol{\mathcal{S}}^P)$ **then**
9: $\quad\quad \boldsymbol{\mathcal{S}}^{Seq} \leftarrow \boldsymbol{\mathcal{S}}^{Seq} \cup (S_i^{Seq} \times \{i\})$
10: $\quad\quad$ Remove those b_i nodes from S^{Seq}
11: $\quad\quad Added \leftarrow Added \cup \{i\}$
12: $\quad$ **end if**
13: **end for**
14: **for** $i \in I_2 \setminus Added$ **do**
15: $\quad S_i^{Seq} \leftarrow$ top b_i nodes from S^{Seq}
16: $\quad \boldsymbol{\mathcal{S}}^{Seq} \leftarrow \boldsymbol{\mathcal{S}}^{Seq} \cup (S_i^{Seq} \times \{i\})$
17: $\quad$ Remove those b_i nodes from S^{Seq}
18: **end for**
19: Return $\boldsymbol{\mathcal{S}}^{Seq}$

Algorithm SeqGRD considers the general setting where a set of items have already been seeded and $\boldsymbol{\mathcal{S}}^P$ corresponds to this partial allocation. Let $S^P := \{v \mid (v, i) \in \boldsymbol{\mathcal{S}}^P\}$ be the seed set allocated in $\boldsymbol{\mathcal{S}}^P$ and let I_2 denote the remaining items which have yet to be allocated. The algorithm takes a graph G, to-be-allocated itemset I_2, item budget vector $\vec{b}$ for the items in I_2, accuracy parameter ϵ, tolerance parameter ℓ, the partial allocation $\boldsymbol{\mathcal{S}}^P$ as input. It first selects a seedset S^{Seq} of size $\overline{b}$, where $\overline{b} := \sum_{i \in I_2} b_i$ (We assume $I_1 \cap I_2 = \emptyset$). (line 2). To select the seeds it uses an algorithm, called PRIMA^+, which delivers a set of seeds that are approximately optimal w.r.t. the marginal gain $\sigma(S|S^P)$. We present the PRIMA^+ algorithm in §5.2.1 and establish its properties.

SeqGRD then sorts the items based on their truncated utility (line 4). Starting from the item i having the highest truncated utility, it tries to allocate the item to the top b_i nodes of S^{Seq}, S_i^{Seq}. If the allocation $S_i^{Seq} \times \{i\}$ yields a positive marginal welfare, it is added to the existing allocation and nodes of S_i^{Seq} are removed for future considerations (Lines 8-12). The items that are not allocated in this iteration are appended following an arbitrary order (lines 14-18) and allocated at the end.

Let $\Gamma_w(S)$ be the set of nodes reachable from a seed set S in the possible world w. Then we get the following lemma.

Lemma 1. *Let $\boldsymbol{\mathcal{S}}$ be an allocation, S be its seedset, let w be a random possible world. Then for any node $v \in V$, we have*

$$u_{\min} \leq \mathbb{E}_w \left[\mathcal{U}_w(\mathcal{A}_w^{\boldsymbol{\mathcal{S}}}(v)) \mid v \in \Gamma_w(S) \right] \leq u_{\max}.$$

Lemma 2. *Let $\boldsymbol{\mathcal{S}}$ be an allocation and S its corresponding seed nodes. Then $u_{\min} \cdot \sigma(S) \leq \rho(\boldsymbol{\mathcal{S}}) \leq u_{\max} \cdot \sigma(S)$.*

Using the lemmas, we get the following bound for SeqGRD.

Theorem 5.1. *Let $\boldsymbol{\mathcal{S}}^{Seq}$ be the allocation returned by the Algorithm* SeqGRD. *Given $\epsilon, \ell > 0$, we have $\rho(\boldsymbol{\mathcal{S}}^{Seq} \cup \boldsymbol{\mathcal{S}}^P) \geq \frac{u_{\min}}{u_{\max}}(1 - \frac{1}{e} - \epsilon)\rho(\boldsymbol{\mathcal{S}}^A \cup \boldsymbol{\mathcal{S}}^P)$ w.p. at least $1 - \frac{1}{|V|^\ell}$, where $\boldsymbol{\mathcal{S}}^A$ is any arbitrary allocation of items in I_2 respecting the budget constraint.*

We note that the property of PRIMA$^+$ that is exploited in the proof above is its ability to select seed nodes S such that they are approximately optimal w.r.t. the marginal gain over an existing seed set S^P. The prefix preserving on marginals property of PRIMA$^+$ is not needed in the above proof. However, our next algorithm MaxGRD relies on the prefix-preserving property.

SeqGRD-NM **Algorithm**

The proof of the approximation bound above does not rely on marginal check (Algorithm 1, line 8). We call the version of SeqGRD that does not perform marginal check SeqGRD-NM (No Marginal). Specifically, SeqGRD-NM simply sorts the items based on their truncated utility, allocates item i to the first b_i nodes of S^{Grd}, where S^{Grd} is selected using PRIMA$^+$, and removes those b_i nodes from S^{Grd}.

Computing marginals involves sampling, which takes significant time in large networks. On the other hand, the marginal check avoids the phenomenon of items with lower (truncated) utility blocking those with higher utility, to some extent. Thus even though SeqGRD-NM is faster than SeqGRD and has the same approximation guarantee, under certain utility configurations, the welfare produced by SeqGRD-NM can be worse than that of SeqGRD. We explore this in our experiments in §6. On the other hand, we still append all items in the end to exhaust the budget in SeqGRD (lines 14–18). To really discard a certain itemset, we need to exhaustively search through all itemset combinations, which is time-consuming. So we only do a simple marginal check in SeqGRD, and append all items at the end to ensure the theoretical guarantee.

5.2 MaxGrd Algorithm

Our next algorithm MaxGRD provides $\frac{1}{m}(1 - \frac{1}{e})$-approximation, when $\boldsymbol{\mathcal{S}}^P = \emptyset$, i.e., no prior allocation. The pseudocode is shown in Algorithm 2. Like SeqGRD, MaxGRD also selects its seedset S^{Max} using PRIMA$^+$, but the size of the seedset is different: $\overline{b} := \max_{i \in I_2} b_i$, i.e., the maximum budget of any unallocated item (line 1). Then for every item $i \in I_2$, it computes the expected marginal social welfare of the allocation $\rho((S_i^{Max} \times \{i\}) \mid \boldsymbol{\mathcal{S}}^P)$, where S_i^{Max} is the set of first b_i nodes of S^{Max}. It returns the allocation with the maximum welfare (line 3).

Notice that MaxGRD is applicable even when $S^p \neq \emptyset$, so we have provided the algorithm for this general case. However, it

Algorithm 2 MaxGRD($G, \epsilon, \ell, \boldsymbol{\mathcal{S}}^P, I_2, \vec{b}$)

1: $S^{Max} \leftarrow$ PRIMA$^+(G, \epsilon, \ell, S^P, \vec{b}, \max_{i \in I_2} b_i)$
2: $S_i^{Max} \leftarrow$ top b_i nodes of $S^{Max}, \forall i \in I_2$
3: $i_{max} \leftarrow \arg\max_{i \in I_2} \{\rho(S_i^{Max} \times \{i\} \mid \boldsymbol{\mathcal{S}}^P)\}$
4: Return $S_{i_{max}}^{Max} \times \{i_{max}\}$

enjoys an approximation bound only for the special case, when $S^p = \emptyset$. We prove the following lemma under this constraint, which is instrumental in the proof of the approximation bound. A key observation is that given a possible world w, the utility function $\mathcal{U}_w(\cdot)$ in that possible world is submodular. This follows from the fact that valuation is submodular and price and noise, being additive are both modular.

Lemma 3. *Let $\boldsymbol{\mathcal{S}} := \cup_{i=1}^m (S_i \times \{i\})$ be an arbitrary allocation, where S_i is the set of seed nodes of item i. Then $\rho(\cup_{i=1}^m (S_i \times \{i\})) \leq \sum_{i=1}^m \rho((S_i \times \{i\}))$.*

Sketch. Consider an arbitrary but fixed possible world w and an arbitrary item $i \in I_2$. Let v be any node that adopts i in w under the allocation $\cup_{i=1}^m (S_i \times \{i\})$. We can show that v must also adopt i in w when the allocation is only $(S_i \times \{i\})$. The lemma follows from this. □

Theorem 5.2. *Suppose that $\boldsymbol{\mathcal{S}}^P = \emptyset$. Let $\boldsymbol{\mathcal{S}}^{Max}$ be the allocation produced by* MaxGRD. *Given $\epsilon, \ell > 0$, we have $\rho(\boldsymbol{\mathcal{S}}^{Max}) \geq \frac{1}{m}(1 - \frac{1}{e} - \epsilon)\rho(\boldsymbol{\mathcal{S}}^A)$ w.p. at least $1 - \frac{1}{|V|^\ell}$, where $\boldsymbol{\mathcal{S}}^A$ is any arbitrary allocation.*

Can MaxGRD **produce better welfare than** SeqGRD**?** Hypothetically, there can be situations where MaxGRD can produce better welfare than SeqGRD. E.g., consider a network with nodes $\{u, v, w, x\}$ and edges $\{(u, v), (v, w), (x, w)\}$ where all edge probabilities are 1. There are two items i, j, with all noise terms being 0. The utilities are $\mathcal{U}(\{i\}) = 10, \mathcal{U}(\{j\}) = 1, \mathcal{U}(\{i, j\}) = 0$ and both items i and j have a budget of 1. Then SeqGRD will yield the allocation $\boldsymbol{\mathcal{S}}^{Seq} = \{(u, i), (x, j)\}$, resulting in a social welfare of $2 \times 10 + 1 \times 2 = 22$. On the other hand, MaxGRD will only allocate u to i, resulting in a social welfare of $3 \times 10 = 30$.

In our experiments, however, we find that situations where MaxGRD dominates SeqGRD are rare. We hypothesize that this is because in a large network, with a number of seeds that is a small fraction of the network size n, blocking caused by the allocation of seeds to additional items by SeqGRD is less likely to occur.

Note that the approximation guarantee of SeqGRD holds also when $\boldsymbol{\mathcal{S}}^p = \emptyset$. Thus running both SeqGRD and MaxGRD individually and returning the allocation with higher welfare would achieve a $max\{\frac{u_{\min}}{u_{\max}}, \frac{1}{m}\}(1 - \frac{1}{e})$-approximation, as a consequence of Theorems 5.1 and 5.2.

5.2.1 PRIMA$^+$. We now present our PRIMA$^+$ algorithm used by SeqGRD and MaxGRD to select seeds. First, we formally present the property of prefix preservation on marginals.

Definition 1. (Prefix Preservation on Marginals). *Given $G = (V, E, p)$, budget vector $\vec{b}$, the number of seeds to be selected $\overline{b}$ and a fixed seed set S^P, an influence maximization algorithm $\mathbb{A}$ is prefix-preserving on marginals w.r.t. $\vec{b}$ and S^P, if for any $\epsilon > 0$ and $\ell > 0$, $\mathbb{A}$ returns an ordered set S of size $\overline{b}$, such that w.p. at least $1 - \frac{1}{|V|^\ell}$, $\sigma(S \mid S^P) \geq (1 - \frac{1}{e} - \epsilon)\, OPT_{\overline{b}|S^P}$ and for every $b_i \in \vec{b}$, the first*

b_i nodes of S, denoted S_i, satisfies $\sigma(S_i \mid S^P) \geq (1-\frac{1}{e}-\epsilon)\, OPT_{b_i|S^P}$, where $OPT_{b|S^P}$ is the optimal marginal expected spread of b nodes on top the existing seeds S^P.

In [6], the authors proposed a seed selection algorithm called PRIMA that is prefix-preserving in spread, using the Reverse Reachable Sets (RR-sets), as proposed in IMM [44]. Here, we modify the standard RR-set construction slightly to account for the presence of existing seed set S^P: Given an existing allocation $\boldsymbol{\mathcal{S}}^P$, we construct a marginal RR-set as follows. Choose a root node $v \in V$ uniformly at random, add it to R_v and start a BFS from v. Whenever $u \in R_v$, sample each incoming edge (u', u) w.p. $p_{u'u}$ and add it to R_v. Stop when no new nodes are added to R_v; if at any stage R_v overlaps S^P, i.e., if $R_v \cap S^P \neq \emptyset$, then set $R_v := \emptyset$. That is, whenever a generated RR-set "hits" S^P, just set it to $\emptyset$.

PRIMA$^+$ achieves the property of prefix preservation on marginals and it runs in time $O((\overline{b} + \ell + \log_n |\vec{b}|)(n+m)\log\ n \cdot \epsilon^{-2})$, where $\overline{b} := max_{i \in I_2} b_i$, is the maximum budget of any item in I_2. The algorithm and the proof its correctness mainly follow that of PRIMA. Hence we omit the details here for space constraints, which can be found in our full report [5].

5.3 SupGrd Algorithm

Our third algorithm SupGRD provides a constant $(1-\frac{1}{e})$-approximation. The bound holds under more restrictive conditions as given below.

Conditions required for SupGRD **approximation bound.** (i) There exists a superior item (defined in §5) i_m in the item set: i.e., under any noise possible world $w_2, \mathcal{U}_{w_2}(i_m) > \mathcal{U}_{w_2}(i), \forall i \in \mathbf{I} \backslash \{i_m\}$. (ii) Seeds for all the inferior items are fixed: that is, $I_2 = \{i_m\}$ is the only item for which an allocation needs to be found; and (iii) There is pure competition between all items: every node can adopt at most one item. Under these conditions, the following two lemmas show that the social welfare is monotone and submodular.

Lemma 4. *Given $\boldsymbol{\mathcal{S}}^P$ and I_2, let $\boldsymbol{\mathcal{S}}_1$ and $\boldsymbol{\mathcal{S}}_2$ be two allocations over I_2 such that $\boldsymbol{\mathcal{S}}_1 \subseteq \boldsymbol{\mathcal{S}}_2$. Then $\rho(\boldsymbol{\mathcal{S}}_1 \cup \boldsymbol{\mathcal{S}}^P) \leq \rho(\boldsymbol{\mathcal{S}}_2 \cup \boldsymbol{\mathcal{S}}^P)$.*

Lemma 5. *Given $\boldsymbol{\mathcal{S}}^P$ and I_2, let $\boldsymbol{\mathcal{S}}_1$ and $\boldsymbol{\mathcal{S}}_2$ be two allocations over I_2 such that $\boldsymbol{\mathcal{S}}_1 \subseteq \boldsymbol{\mathcal{S}}_2$. Let $s = (u, i_m) \notin \boldsymbol{\mathcal{S}}_2$ be an allocation pair. Then $\rho(s \mid \boldsymbol{\mathcal{S}}_1 \cup \boldsymbol{\mathcal{S}}^P) \geq \rho(s \mid \boldsymbol{\mathcal{S}}_2 \cup \boldsymbol{\mathcal{S}}^P)$.*

Since social welfare is monotone and submodular, a standard greedy selection based on the marginal welfare will have $(1-\frac{1}{e})$-approximation. However since computing spread itself is #P-hard, computing the exact marginal is not feasible. In IM, sampling using RR-sets has been used to achieve state of the art performance. In what follows, by extending IMM [44], we adopt a martingale approach for seed selection in SupGRD. Given ϵ and ℓ, SupGRD returns a seed set that has a $(1-\frac{1}{e}-\epsilon)$-approximation w.p. at least $1-\frac{1}{n^\ell}$.

In the classical setting, RR-set samples are used to compute an unbiased estimation of the spread. In our case we need to estimate the *marginal welfare* using the RR-sets. Towards that we define a notion of weight for every RR-set. The weight of an RR-set R_v denotes the marginal gain in the expected social welfare achieved by activating the root v of the RR-set R_v. Thus it is the difference between the expected truncated utility of the item that the root v adopts under the existing partial allocation $\boldsymbol{\mathcal{S}}^P$ and that of i_m. To ensure that the root v indeed adopts i_m, the path from some seed of i_m to v should be no longer than that from any seed of $\boldsymbol{\mathcal{S}}^P$ to v. Thus, a weighted RR-set is constructed as follows.

Algorithm 3 $NodeSelection(\mathcal{R}, b')$

```
Initialize S_{b'} = ∅, i = 1
while i ≤ b' do
    Select v ∈ V \ S_{b'} which has the highest M_R(S_{b'} ∪ v) − M_R(S_{b'})
    S_{b'} ← S_{b'} ∪ {v}
    Remove R from R if v ∈ R
end while
return S_{b'} as the final seed set
```

Definition 2. *(Weighted Reverse Reachable Set). For a given fixed allocation $\boldsymbol{\mathcal{S}}^P$ and a node $v \in G$, a weighted RR-set of v, R_v is obtained by starting with $R_v = \{v\}$ and starting a BFS from v such that: for $u \in R_v$, sample each incoming edge (u', u) w.p. $p_{u',u}$ and add it to R_v; stop when either no new nodes are added or R_v overlaps S^P (so the distance from any node in R_v to v along the reversely generated edges is at most the distance from S^P to v). Then, the weight of R_v is $w(R_v) = \mathcal{U}^+(\{i_m\}) - max_{i \in I^s \mid s \in S^P \cap R(v)} \mathcal{U}^+(i)$, where I^s denotes the items allocated to node s in the allocation $\boldsymbol{\mathcal{S}}^P$.*

SupGRD samples RR-sets using an early termination as described in Definition 2. This construction ensures that if any member of a weighted RR-set is seeded with i_m, the root of the RR-set, v, will adopt i_m. In what follows, we first establish the connection between marginal social welfare and weighted RR-sets and then present efficient seed selection and RR-set sampling algorithms to maximize the marginal social welfare.

For a node set S, let $\mathbb{I}[.]$ be an indicator function denoting whether S covers the (weighted) RR set R, i.e., $\mathbb{I}(S \cap R \neq \emptyset) = 1$, if $S \cap R_v \neq \emptyset$, 0 otherwise. Also let $\mathcal{L}(G)$ denote the distribution of all the live edge graphs, then extending the result of Borg et al., we get the following lemma for weighted RR-sets.

Lemma 6. *For given seed sets S and S^P, we have $\mathbb{E}_{w_1 \sim G}[\rho_{w_1}(S \mid S^P)] = n \cdot \mathbb{E}_{v \sim V, w_1 \sim G}[\mathbb{I}(S \cap R_v \neq \emptyset) \cdot w(R_v)]$ where $n = |V|$ is the number of nodes in G.*

We now extend the RR-set based efficient approximation IM algorithm, IMM, for maximizing welfare. Similar to IMM, our algorithm SupGRD has two key phases, namely, *NodeSelection* and *Sampling*. The *NodeSelection* phase is similar to that of IMM, except we consider the weight of RR-sets while selecting seed nodes. For a node set S and a collection of weighted RR-sets $\mathcal{R}$, define $M_{\mathcal{R}}(S) := \sum_{R \in \mathcal{R}} \mathbb{I}[S \cap R \neq \emptyset] \cdot w(R)$. Let $b' = b_{i_m}$ be the budget of the superior item i_m. Given a set $\mathcal{R}$, *NodeSelection* selects b' seeds that maximizes $M_{\mathcal{R}}$ (Algorithm 3).

Next, the goal of the *Sampling* phase (Algorithm 4) is to generate $\mathcal{R}$ such that $|\mathcal{R}| \geq \lambda / OPT$, where OPT is the optimal welfare, and λ is defined as follows,

$$\lambda = 2n \cdot ((1-1/e) \cdot \alpha + \beta)^2 \cdot \epsilon^{-2}, \tag{1}$$

where, $\alpha = \sqrt{\ell \log n + \log 2}$ and
$\beta = \sqrt{(1-1/e) \cdot (\log \binom{n}{b'} + \ell \log\ n + \log 2)}$.

Since OPT is unknown, the *Sampling* first ensures that it finds a lower bound to OPT w.h.p. For that it deploys a statistical test using a binary search on the range of OPT. The maximum possible value of the welfare OPT is $UB = n \times u_{\max}$, when every node in

Algorithm 4 $Sampling(G, k, \epsilon, \ell)$

1: Initialize $\mathcal{R} = \emptyset$, $LB = 1$, $UB = |V| \times u_{\max}$, $i = 1$, $\epsilon' = \sqrt{2} \cdot \epsilon$, $\ell = \ell + log\ 2/log\ n$.
2: **for** $i = 1$ to $log_2 UB - 1$ **do**
3: $x = n/2^i$, $\theta_i = \lambda'/x$
4: **while** $|\mathcal{R}| \leq \theta_i$ **do**
5: Add a random RR set to $\mathcal{R}$
6: **end while**
7: $S_i = NodeSelection(\mathcal{R}, k)$
8: **if** $\frac{n}{\theta} \cdot M_{\mathcal{R}}(S_i) \geq (1 + \epsilon') \cdot x$ **then**
9: $LB = \frac{n}{\theta} \cdot M_{\mathcal{R}}/(1 + \epsilon')$
10: Break;
11: **end if**
12: **end for**
13: $\mathcal{R} = \emptyset$
14: **while** $|\mathcal{R}| \leq \lambda/LB$ **do**
15: Add a random RR set to $\mathcal{R}$
16: **end while**

	NetHEPT	Douban-Book	Douban-Movie	Orkut	Twitter
# nodes	15.2K	23.3K	34.9K	3.07M	41.7M
# edges	31.4K	141K	274K	117M	1.47G
avg. deg.	4.13	6.5	7.9	77.5	70.5
type	undirected	directed	directed	undirected	directed

Table 2: Network Statistics

the network adopts the superior item i_m, $u_{\max}$ is the utility of i_m. Thus the binary search ranges from 1 to UB (Line 2).

A good lower bound is found when the condition of Line 8 is satisfied. Different from [44], this condition directly operates on welfare which is shown in Lemma 7.

Lemma 7. *Let $x \in [1, UB]$, ϵ' and $\delta \in (0, 1)$, then if we invoke NodeSelection with $|\mathcal{R}| = \theta$, where*

$$\theta \geq \frac{(2 + \frac{2}{3}\epsilon') \cdot (log\binom{n}{b'} + log(1/\delta))}{\epsilon'^2} \cdot \frac{n}{x} \tag{2}$$

and S is the output NodeSelection returns, then if $OPT < x$, $\frac{n}{\theta} \cdot M_{\mathcal{R}}(S) < (1 + \epsilon') \cdot x$, w.p. at least $(1 - \delta)$.

Then by setting λ' using Eq. (3), we get Theorem 2 of [44]

$$\lambda' = \frac{(2 + \frac{2}{3}\epsilon') \cdot (\log\binom{n}{b'} + \ell' \cdot \log\ n + \log\log_2\ n) \cdot n}{\epsilon'^2}, \tag{3}$$

The rest of the proof is similar to that of [44], which gives us the following result.

Theorem 5.3. *Let $\boldsymbol{\mathcal{S}}^P$ be a partial allocation on the inferior items. Let $\boldsymbol{\mathcal{S}}^{Grd}$ be the allocation of the superior item produced by* SupGrd. *Given $\epsilon, \ell > 0$, we have $\rho(\boldsymbol{\mathcal{S}}^{Grd} \cup \boldsymbol{\mathcal{S}}^P) \geq (1 - \frac{1}{e} - \epsilon)\rho(\boldsymbol{\mathcal{S}}^A \cup \boldsymbol{\mathcal{S}}^P)$ w.p. at least $1 - \frac{1}{|V|^\ell}$, where $\boldsymbol{\mathcal{S}}^A$ is any arbitrary allocation.*

Running time: Let w_{min} be the minumum weight of an RR set. Then using Lemma 9 of [44], the expected total time to generate $\mathcal{R}$ is given by $O((b' + \ell)(n + m) \log\ n \cdot \epsilon^{-2}/w_{min})$.

Notice that generating an RR-set from scratch for the final node selection (line 13), following the fix of [17], only adds a multiplicative factor of 2. Hence the overall asymptotic running time to generate $\mathcal{R}$ remains unaffected.

6 EXPERIMENTS

6.1 Experiment Setup

All our experiments are run on a Linux machine with Intel Xeon 2.6 GHz CPU and 128 GB RAM.

6.1.1 Networks. Our experiments were conducted on five real social networks: NetHEPT, Douban-Book, Douban-Movie, Twitter, and Orkut, whose characteristics are summarized in Table 2. Of these, NetHEPT, Douban-Book, and Douban-Movie are benchmarks in IM literature [36], while Twitter and Orkut are two of the largest public networks available at [1].

6.1.2 Algorithms compared. In the experiments our four algorithms – SeqGRD, SeqGRD-NM, MaxGRD, and SupGRD are compared against three baselines – TCIM, Balance-C and greedyWM. There is no previous work that can deal with both arbitrary degree of competition and multiple items in propagation. Our first two baselines each covers one aspect. TCIM [34] in particular assumes a propagation model which is an extension of the IC model under pure competition. It can, however, handle more than two items. Given fixed seed sets of other competing items, TCIM selects seeds of an item under a budget constraint, such that the number of adoptions of that item is maximized. When we run TCIM for multiple items, we select seeds for each of the items one by one, while keeping the seeds of other items fixed and then report the allocation that produces the maximum welfare.

In contrast, Balance-C [23] does not assume pure competition, but it works only when number of items in propagation is two. Given an initial seed placement of the two items, Balance-C chooses the remaining seeds such that at the end of the propagation, the number of nodes seeing either both the items or none, is maximized. Thus for competing ideas, Balance-C ensures that there is a balanced exposure of the two ideas to the most number of nodes. It is non-trivial to extend Balance-C for more than two items hence we compare against it only in two item set up.

Both TCIM and Balance-C aim to maximize adoption count, not social welfare. Our third baseline greedyWM maximizes the social welfare directly. It greedily selects iteratively the (node, item) pair that maximizes the marginal social welfare, till the budgets are exhausted. Below, by deterministic utility of an itemset I, we mean $\mathcal{V}(I) - \mathcal{P}(I)$, i.e., its utility with the noise term ignored.

6.1.3 Default parameters. Following previous works [25, 39] we set probability of edge $e = (u, v)$ to $1/d_{in}(v)$, where $d_{in}(v)$ is the in-degree of node v. Unless otherwise specified, we use $\epsilon = 0.5$ and $\ell = 1$ as our default in all the algorithms that use these parameters. We test the algorithms across a wide variety of utility configurations to cover different aspects of competition. We will describe the configurations as we present the corresponding experiments. Whenever marginal gains are required, we run 5000 simulations and take the average result.

6.2 Experiments with two items

For our first set of experiments we restrict the number of items to two so that we can compare against all of the mentioned baselines. We also consider four different configurations to capture different kinds of competition. The details of the configurations are given in Table 3. In configurations C1 and C2, the items exhibit pure competition. In C1, items have comparable individual utility. In C2, the difference between individual utility is high: i's deterministic utility is 1, 10 times higher than that of j. C3 and C4 exhibit partial competition. Except for C4, in all configurations we consider the same budget for both items (uniform); budget is varied from 10 to

No	Price	Value	Noise	Budget
C1	$i = 3$	$i = 4, j = 4.9$ $\{i, j\} = 4.9$	$i: N(0, 1)$	Uniform
C2	$j = 4$	$i = 4, j = 4.1$ $\{i, j\} = 4.1$		Uniform
C3	$\{i, j\} = 7$	$i = 4, j = 4.9$	$j: N(0, 1)$	Uniform
C4		$\{i, j\} = 8.7$		Nonuniform

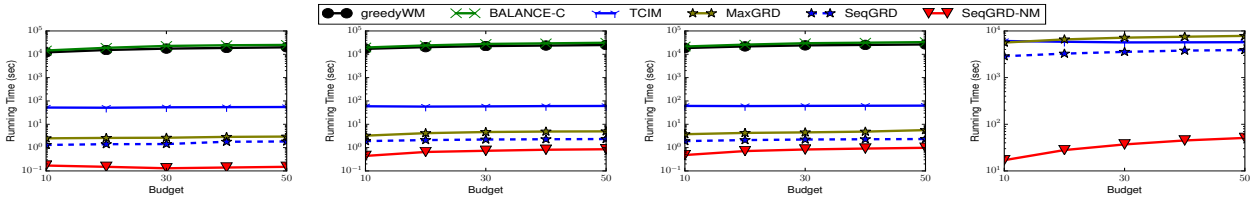
Table 3: Two item configurations

$\mathcal{U}(i) = 2$	$\mathcal{U}(\{i, j\}) < 0$
$\mathcal{U}(\{j\}) = 0.11$	$\mathcal{U}(\{j, k\}) < 0$
$\mathcal{U}(\{k\}) = 0.1$	$\mathcal{U}(\{i, j, k\})$
$\mathcal{U}(\{i, k\}) = 2.1$	< 0

Table 4: Three item configuration

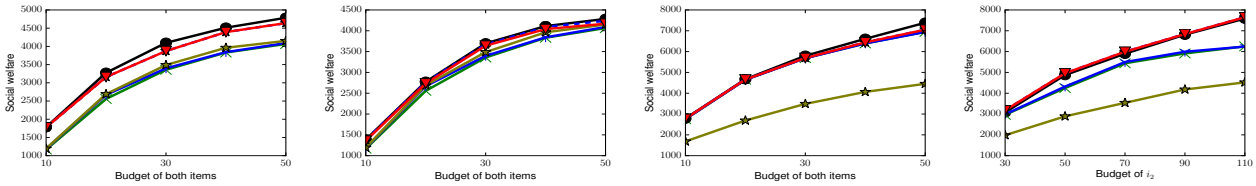
item	p	q	$\mathcal{U}_D$
$\{indie\}$	0.107	na	7.0
$\{rock\}$	0.091	na	6.8
$\{industrial\}$	0.015	na	5.0
$\{progressive\_metal\}$	0.011	na	4.7

Table 5: Learned parameters



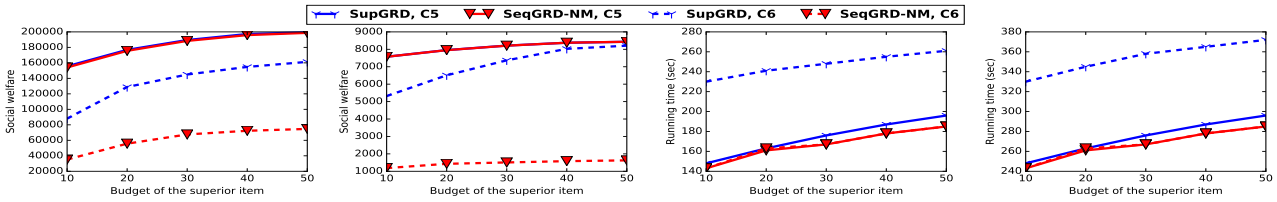
(a) NetHEPT (b) Douban-Book (c) Douban-Movie (d) Orkut

Figure 1: Running times of greedyWM, Balance-C, TCIM, MaxGRD, SeqGRD **and** SeqGRD-NM **(on Configuration** 1**)**



(a) Configuration 1 (b) Configuration 2 (c) Configuration 3 (d) Configuration 4

Figure 2: Expected social welfare in four configurations (on the Douban-Movie network)



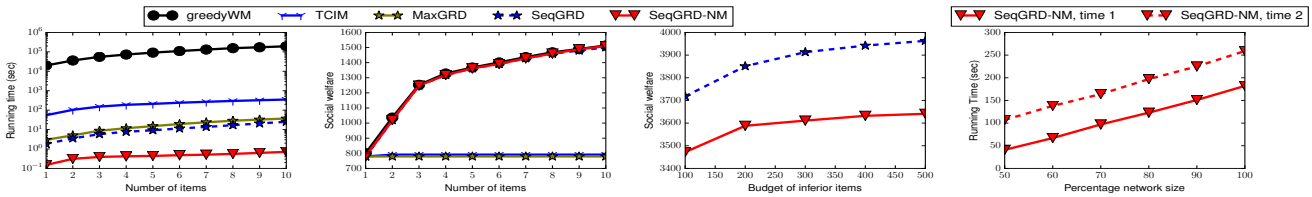
(a) Orkut (b) Twitter (c) Orkut (d) Twitter

Figure 3: Comparison between SupGRD and SeqGRD on C2 and C3 (a-b) Social welfare, (c-d) Running time

50 in steps of 10. In C4, we fix the budget of i to 50 and vary j's budget (non-uniform) from 30 to 100 in steps of 20. We assume $\mathcal{S}^p = \emptyset$ in these configurations. Since it does not meet constraints required by SupGRD, we defer the comparison until §6.2.3.

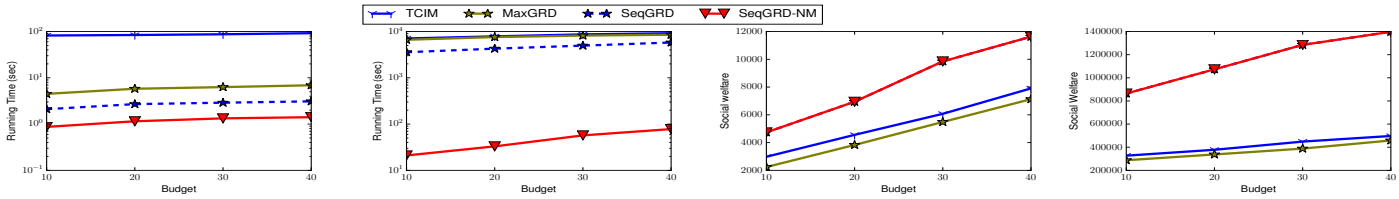
6.2.1 Running time. First we compare the running time of the algorithms using C1 as a representative case. Fig. 1 shows the result on four networks. SeqGRD-NM is orders of magnitude faster than other algorithms in every network. The reason is that SeqGRD-NM does not compute any marginal. Each marginal computation requires iterating over 5000 samples, which significantly increases the running time. For the same reason greedyWM and Balance-C exhibit exorbitantly high running time: they do not in fact complete in 6 hours on a large network like Orkut. Hence they are not included in Fig. 1(d). Except for SeqGRD-NM, none of the other algorithms scale to the largest network Twitter. We will compare SeqGRD-NM and SupGRD on Twitter later. Performance on other configurations show similar trends, and hence omitted for brevity.

6.2.2 Social welfare. We now compare the expected social welfare achieved by the algorithms on the four configurations (Fig. 2). We show the results only for Douban-Movie , since the trend of the results is similar on other networks. In all configuration SeqGRD, SeqGRD-NM and greedyWM outperform all other algorithms. The difference in welfare is up to 3× higher. MaxGRD in particular allocates just one of the two items. Thus when items exhibit partial competition (C3 and C4), ,MaxGRD performs significantly worse. Balance-C performs comparatively better under partial competition (C3), however for a non-uniform budget again its performance drops. TCIM on the other hand aims to maximize the adoption count of the item being allocated. Thus it also ends up allocating both the items in same seed nodes. This reduces the overall social welfare for configuration such C1, where both Balance-C and TCIM perform comparatively worse. Social welfare produced by greedyWM is consistently good, but its running time is exorbitantly high, which prohibits its applicability on any decently sized network. SeqGRD-NM on the other hand is the fastest algorithm,



(a) Running Time (b) Social Welfare (c) Social welfare (d) Running time

Figure 4: Multi-item experiments: Impact of number of items on (a) Running time, (b) Social welfare on NetHEPT. (c) Comparing performance of SeqGRD **and** SeqGRD-NM **on NetHEPT. (d) Scalability on Orkut**



(a) NetHEPT (b) Orkut (c) NetHEPT (d) Orkut

Figure 5: Performance of TCIM, MaxGRD, SeqGRD **and** SeqGRD-NM **on real utility configurations (Table 5)**

which produces similar welfare across all these four configurations. However, notice that in none of these four configurations, item blocking is effective. We will show later in §6.3.2 that in the presence of multiple items, when avoiding item blocking is critical, the performance of SeqGRD-NM deteriorates.

6.2.3 Comparison against SupGRD**.** In this experiment we compare SupGRD and SeqGRD-NM on the two largest networks, Orkut and Twitter. We use utility configurations of C1 and C2, but adopt the seed placements needed to meet the constraints required for SupGRD. Recall that for SupGRD the seeds for the inferior items need to be fixed. Hence we select the top 50 nodes using IMM and set them as seeds of j. Then, the seeds of i are selected using SupGRD and SeqGRD-NM with the budget being varied from 10 to 50 in steps of 10. We call these new configurations C5 and C6 respectively.

Since the top nodes in terms of the spread are given to j, these two cases pose a unique challenge of dealing with arbitrary degree of competition when maximizing welfare. When items' utilities are similar, in C5, new seeds of i should be chosen in a way that minimizes i's overlap with j propagation. Instead in C6, when i has much higher utility, it should be allocated to the top seed nodes. That way, the number of nodes that can be reached by i is much higher and that helps boost the overall social welfare. As can be seen from our results next, that SupGRD can navigate through these varied "strategies", while SeqGRD-NM cannot.

Fig. 3 (a) and (b) shows the result on the expected social welfare on Orkut and Twitter respectively. "SeqGRD-NM-C5" (resp. "SupGRD-C5") refers to SeqGRD-NM (resp. SupGRD) on C5 and "SeqGRD-NM-C6" (resp. "SupGRD-C6") on C6. Notice that in C5 the welfare produced by the two algorithms are comparable. However in C6, where the gap between the individual utilities of the two items is higher, difference between the welfare of SupGRD and SeqGRD-NM is also larger. The reason for that is as follows. SeqGRD-NM uses PRIMA$^+$ to select the seeds of i. Consequently to maximize the marginal gain in spread, it minimizes the overlap in the spread of i and j and hence allocates i to lower ranked nodes in terms of spread. However i is the superior item, so allocating lower ranked nodes to i decreases the overall welfare.

Fig. 3(c) and (d) compares the running time of the two algorithms on Orkut and Twitter. Both the algorithms scale on these large networks. Unlike SeqGRD-NM, running time SupGRD depends on the utility configurations as well. As our running time analysis (§5.3) suggests, when the minimum utility of an item is lower, the running time of SupGRD is higher. However as can be seen, even on large networks, the difference in the running times is not very high: e.g., in configuration C6, the running time of SupGRD is only a 2× that of SeqGRD-NM, whereas in C5 the running times are similar. To summarize, SupGRD addresses this unique challenge of dealing with an arbitrary degree of competition, with a slightly higher running time.

6.3 More than two items

Except for Balance-C, all the algorithms can deal with multiple items. In this section, we study their performances when the number of items is more than two. First, we show the impact of increasing the number of items on the running time and social welfare produced. Then we study how the algorithms behave under some challenging configurations designed using multiple items.

6.3.1 Impact of number of items. For this experiment, the configuration we test is as follows. Each individual item has expected utility of 1 and the items exhibit pure competition. Every item has budget 50 and $\boldsymbol{\mathcal{S}}^p = \emptyset$.

Fig. 4(a) and (b) show respectively, the running time and social welfare produced by the five algorithms for 10 items. Since Balance-C cannot run on more than two items, it is omitted. Running time of algorithms greedyWM, TCIM, MaxGRD, and SeqGRD increases significantly w.r.t the number of items. As the number of items increase, the number of times marginal check is needed for these algorithms, also increases. The marginal check is the most

time consuming portion in their running time. SeqGRD-NM on the other hand relies solely on RR-sets and does not do any marginal checks. Hence the growth in running time is not high. With higher number items, the difference between the running time of SeqGRD-NM and other algorithms, increases.

In terms of social welfare, TCIM and MaxGRD perform worse than the other algorithms. MaxGRD selects only one item in the final allocation, hence it misses out on the additional welfare that could come from allocating the remaining items. Similarly TCIM tries to maximize the spread of the last allocated item, at cost of propagation of other items. Thus their welfare does not increase with more items, unlike the other algorithms.

6.3.2 Effect of marginal check. In our experiments so far, social welfare of SeqGRD-NM has been similar to other algorithms that perform marginal checks. One exception being SupGRD (§6.2.3), but SupGRD assumes specific constraints that are not general. By not performing the marginal check, SeqGRD-NM runs much faster compared to other algorithms. This begs the question if there is any advantage of using the marginal check altogether. In this experiment we show how marginal check helps avoid item-blocking that SeqGRD-NM fails to circumvent.

For this experiment, we consider three items in the propagation. Their expected utilities are specified in Table 4. i has the highest expected utility, followed by j and k has the least. i and k exhibit partial competition hence bundle $\{i, k\}$ has a positive utility, but all other item bundles have negative utilities, exhibiting pure competition. We set the budget of i to 500, and increase the budget of j and k from 100 to 500 each in steps of 100 and study the effect on the welfare produced by SeqGRD-NM and SeqGRD.

Fig. 4(c) shows the result on the NetHEPT network. Both algorithms first allocate i as it has the highest individual utility. Then SeqGRD-NM allocates j next, however this allocation is "adjacent" to i since NetHEPT is small, and blocks propagation of i more. Since the utility of i is significantly higher than j, allocating j this way in fact causes a negative marginal. SeqGRD, using marginal check, postpones allocation of j. After i, it instead allocates k. Although k also has a low individual utility, because of partial competition, it does not block propagation of i and the marginal is non-negative. It later allocates j, which is now further apart from i, hence cannot block i's propagation. Thus SeqGRD produces a social welfare which is higher than that of SeqGRD-NM. Further, as the budget of j increases, the amount of blocking also increases, hence the welfare difference between the two algorithms also goes up.

6.3.3 Scalability of SeqGRD-NM**.** Our next experiment shows the impact of network size on SeqGRD-NM using Orkut with two types of edge probabilities: (1) $1/d_{in}(v)$ and (2) fixed 0.01. We use a uniform budget of 50 for all three items. Instead of using the full network, we use breadth-first-search to progressively increase the network size so that it includes a certain percentage of the total nodes in the network. At 100%, the full network is used. Fig. 4(d) shows the results. "SeqGRD-NM, time 1" and "SeqGRD-NM, time 2" depict the running time of SeqGRD-NM on the two types of edge probabilites respectively. As the network size increases, the running time in both cases roughly has a linear increase.

6.4 Real item experiments

In this section for our experiments we learn the utilities of items from real dataset instead of the synthetic utilities used in earlier experiments. The dataset used is the LastfmGenres generated from the listening behavior of the users on the music streaming service Last.fm [14, 30]. This dataset was used in [7] to learn the adoption probabilities of different items, where each genre is treated as an item. This dataset also echos our first motivating example presented in the introduction. We next establish the connection between the adoption probabilities and the utilities, which enables us to learn the parameters using [7].

6.4.1 Learning the utilities. In [7], every item i is associated with an adoption probability p_i. Adoption probability of an itemset $I = \{i, ..., k\}$ is $p_I = \gamma_{|I|} \prod_{j \in I} p_j + q_I$, where q_I is a correction received depending on the way items in I interact with each other: if the items are complementary, then the correction is positive, if competing then it is negative, and 0 if the items are independent. These probabilities and corrections are learnt in [7] from the dataset of how frequently items are selected together by the users.

According to Observation 2.2 of [7], $p_i = e^{v_i} / \sum_j e^{v_j}$, where v_i is the expected utility of item i as per our utility model. Given a set of learnt p_i, we first set $\sum_j e^{v_j} = 10000$. Then for every i, we set $v_i = \ln(10000 \cdot p_i)$. We choose the number 10000 to ensure that the corresponding utilities are positive. Finally we set the expected utility of item i, $\mathcal{U}(i) = v_i$. Next for an itemset $I = \{i, ..., k\}$, [7] learns two parameters $\gamma_{|I|}$ and q'_I. By using $q_I = \gamma \cdot q'_I$ the probability of adopting the bundle, p_I is derived. The expected utility of the bundle is similarly set to be $\mathcal{U}(I) = \ln(10000 \cdot p_I)$. Notice that the exact values of utilities are not as important as the relative order of utilities of different itemsets. The way utilities are learnt is in correspondence with the adoption probabilities learned in [7].

Table 5 shows the utilities of four different items (i.e., genres) in the dataset: *rock, indie, industrial and progressive_metal*, learned using the above described method. Larger bundles are either not present in the dataset or have smaller learned utilities compared to the individual items in the bundle, suggesting that items are in pure competition in our utility model.

6.4.2 Results using real parameters.

We use the learned utility configuration to compare the social welfare produced by the algorithms on two networks, namely NetHEPT and Orkut. For the experiment we set uniform budget for all the four items, which varies from 10 to 40 in steps of 10. The algorithms compared are TCIM, MaxGRD, SeqGRD and SeqGRD-NM. The results are shown in Fig 5.

In terms of running time the results are similar to our previous experiments (Fig 5(a)-(b)). SeqGRD-NM outperforms the other algorithms by orders of magnitude, since it does not require the time consuming marginal gain computation. For social welfare, notice that the real utility configuration exhibits pure competition. As noted earlier, under pure competition, social welfare produced by SeqGRD and SeqGRD-NM coincide. MaxGRD and TCIM on the other hand typically encourage the adoption of one single item. Hence the difference in social welfare produced by these algorithms compared to SeqGRD-NM is higher since the number of items are also more than the previous configurations we used.

Network	Budget	Algorithm	Real Utility Configuration (as shown in Table 5)					Synthetic Utility Configuration (as shown in Table 4)			
			indie	rock	industrial	progressive_metal	welfare	i	j	k	welfare
NetHEPT	10	RR	203	217	191	196	4473.08	277	244	234	513.2
		Snake	204(+0.005)	201(-0.081)	207(+0.083)	195(-0.005)	4458.64(-0.003)	258(-0.068)	246(+0.008)	261(+0.115)	478.4(-0.068)
		SGRD-NM	255(+0.252)	199(-0.082)	188(-0.016)	**165(-0.158)**	4951.8 (+0.112)	306(+0.105)	220(-0.098)	227(-0.030)	577.4(+0.125)
NetHEPT	40	RR	496	493	491	475	10795.3	667	576	645	1227.3
		Snake	483(-0.026)	496(+0.006)	488(-0.004)	488(+0.027)	10758.2(-0.003)	648(-0.028)	581(+0.009)	669(+0.037)	1194.5(-0.027)
		SGRD-NM	673(+0.357)	499(+0.012)	419(-0.147)	**365(-0.189)**	11264.5(+0.043)	800(+0.199)	510(-0.114)	514(-0.203)	1510.6(+0.230)
Orkut	10	RR	37790	38888	38331	34711	828368.2	69151	49730	67405	110032.5
		Snake	38241(+0.012)	37401(-0.038)	39818(+0.039)	34260(-0.013)	828235.4(-0.002)	67648(-0.021)	50511(+0.016)	68510(+0.016)	107227.7(-0.026)
		SGRD-NM	50800(+0.344)	40837(+0.050)	31189(-0.186)	26895(-0.225)	864154.3(+0.040)	76784(+0.110)	50219(+0.010)	57199(-0.151)	124210.9(+0.129)
Orkut	40	RR	58142	58586	59939	54607	1276650.6	119039	83291	113359	183853.2
		Snake	57211(-0.016)	56922(-0.028)	61603(+0.028)	55538(+0.017)	1272190.7(-0.035)	117454(-0.013)	82937(-0.043)	115338(+0.018)	180269.4(-0.020)
		SGRD-NM	106876(+0.838)	54909(-0.063)	42218(-0.296)	**27272(-0.501)**	1397770.8(+0.095)	150926(+0.268)	**63577(-0.237)**	87480(-0.228)	**253427.9(+0.378)**

Table 6: Adoption count of different items and the overall social welfare

6.4.3 Social welfare vs adoption.

Our final set of experiments compare the relationship between the social welfare and item adoptions. In particular, we want to investigate whether maximizing welfare for competing items could result in a significant drop in the number of item adoptions. For this experiment, we focus on two utility configurations – (i) Real utility of Table 5, which exhibits pure competition and (ii) Synthetic utility of Table 4, which exhibits a mix of partial and pure competition. NetHEPT and Orkut are the two networks used and each item's budget is set to two different values, 10 and 40.

We compare our algorithm SeqGRD-NM against two baselines. After selecting the seed nodes, the first baseline allocates items to the nodes in a round robin manner, hence it is called Round-robin. The second baseline, called Snake, is similar to Round-robin, but it flips the order for every successive sequence of allocations. To illustrate, if there are 4 seed nodes $s_1, ..., s_4$, in order, and two items i, j, SeqGRD-NM algorithm allocates as $s_1 : i, s_2 : i, s_3 : j, s_4 : j$, Round-robin algorithm allocates as $s_1 : i, s_2 : j, s_3 : i, s_4 : j$ and Snake algorithm allocates as $s_1 : i, s_2 : j, s_3 : j, s_4 : i$. Table 6 shows the adoption count of each item and the social welfare produced by these algorithms under different configurations.

In terms of the social welfare objective, SeqGRD-NM dominates across all different configurations. Round-robin produces the next highest welfare. Hence we report the fractional change (+ denotes increase and − denotes decrease), in comparison to Round-robin, next to each entry of the table. The entries that deserve more attention are highlighted in green.

As can be seen, the total number of adoptions of all the items remains the same across all three algorithms. However, SeqGRD-NM generally increases the adoption of the superior product to increase the welfare, while reducing the adoption of the inferior item. On NetHEPT, for budget 10, the maximum drop in adoptions happens for the most inferior item (progressive_metal), by 15.8%. For a higher budget, the drop increases (to 18.9%), because when budget increases for the superior item, SeqGRD-NM allocates lower ranked seeds for the inferior item.

The highest drop in adoption i.e., 50.1%, also happens for the item progressive_metal for budget 40 on Orkut. This is because when items are purely competing, number of items is high and each item has a large budget, the inferior items' seeds are in fact much lower ranked. However, if it exhibits partial competition with a superior item, then leveraging it the adoption does not decrease that much. That is why in Orkut even when the budget is 40, for the synthetic utility configuration, the highest drop in adoptions for the inferior items is only 23.7%. Also notice SeqGRD-NM produces significantly higher social welfare compared to the baselines, the increase being up to 37.8%. In summary, we see that our welfare maximization algorithm provide more adoptions to the superior items and fewer adoptions to the inferior items, but the amount of change is not too drastic. We argue that this is the "price" of enhancing the overall user satisfaction; also the drop in the adoptions of the inferior items is exactly because they are not as competitive.

To conclude this section, we generally observe that: (a) when the conditions required by SupGRD are met, it is the best option providing the best social welfare and competitive running time; (b) in the general case, SeqGRD-NM performs well in most cases and has the best running time, but when item blocking is significant, its marginal-checking version SeqGRD could provide better social welfare, at the cost of higher running time; (c) MaxGRD could be used to enhance the theoretical guarantee when the utility difference is high, but its superiority is not typically observed in large networks; (d) our algorithms outperform all baselines on social welfare and running time and scale to large networks. Our algorithms achieve superior welfare at the expense of a reasonable drop in the adoption count of inferior items, keeping the total adoption count unchanged.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we study the problem of maximizing social welfare over competing items under the UIC model. The problem is not only NP-hard but is also NP-hard to approximate within a constant factor. Further we find that due to conflicting requirements, it is challenging to design a single algorithm that can work effectively for all different utility configurations. Yet we propose a cohort of efficient algorithms that not only provide approximation guarantees but also scale well to real large networks, and their performance is validated through extensive experiments on real-world networks.

Although welfare maximization under competition ensures that users' total utility from adoptions is maximized, it does not directly ensure fairness. For a campaigner who often pays for advertising, ensuring that her item is seen at least by a certain number of users is critical. While fairness in IM has been studied recently, incorporating fairness in social welfare maximization will be an interesting challenge. Further, this paper and [6] studied competition and complementarity in isolation. Designing algorithms for an arbitrary mix of competing and complementary items is an intriguing problem.

REFERENCES

[1] [n. d.]. Twitter and Orkut Dataset. https://snap.stanford.edu/data/. Accessed: 2020-05-30.

[2] Ben Abramowitz and Elliot Anshelevich. 2018. Utilitarians Without Utilities: Maximizing Social Welfare for Graph Problems Using Only Ordinal Preferences. In *AAAI*. 894–901.

[3] Cigdem Aslay, Wei Lu3 Francesco Bonchi, Amit Goyal, and Laks VS Lakshmanan. 2015. Viral Marketing Meets Social Advertising: Ad Allocation with Minimum Regret. *Proceedings of the VLDB Endowment* 8, 7 (2015).

[4] Cigdem Aslay, Francesco Bonchi Laks VS Lakshmanan, and Wei Lu. 2017. Revenue Maximization in Incentivized Social Advertising. *Proceedings of the VLDB Endowment* 10, 11 (2017).

[5] Prithu Banerjee et al. [n. d.]. Maximizing Social Welfare in a Competitive Diffusion Model. https://arxiv.org/abs/2012.03354.

[6] Prithu Banerjee et al. 2019. Maximizing Welfare in Social Networks under A Utility Driven Influence Diffusion model. In *SIGMOD*.

[7] Austin R Benson, Ravi Kumar, and Andrew Tomkins. 2018. A discrete choice model for subset selection. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 37–45.

[8] Shishir Bharathi et al. 2007. Competitive influence maximization in social networks. In *IWWIE*.

[9] Sayan Bhattacharya et al. 2017. Welfare Maximization with Friends-of-Friends Network Externalities. *Theory of Computing Systems* 61, 4 (2017), 948–986.

[10] Robin W Boadway and Neil Bruce. 1984. *Welfare economics*. B. Blackwell New York.

[11] Christian Borgs et al. 2014. Maximizing social influence in nearly optimal time. In *SODA*.

[12] Ceren Budak et al. 2011. Limiting the spread of misinformation in social networks. In *WWW*.

[13] Robert Carbaugh. 2016. *Contemporary Economics: An Applications Approach* (8th ed.). Routledge.

[14] O. Celma. 2010. Last.fm Dataset – 1K users. http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html.

[15] Parinya Chalermsook et al. 2015. Social network monetization via sponsored viral marketing. In *SIGMETRICS*.

[16] Shuo Chen et al. 2015. Online topic-aware influence maximization. In *VLDB*. 666–677.

[17] Wei Chen. 2018. An Issue in the Martingale Analysis of the Influence Maximization Algorithm IMM. *arXiv preprint arXiv:1808.09363* (2018).

[18] Wei Chen et al. 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*.

[19] Wei Chen et al. 2010. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*.

[20] Wei Chen et al. 2013. *Information and influence propagation in social networks*. Morgan & Claypool Publishers.

[21] Samik Datta et al. 2010. Viral marketing for multiple products. In *ICDM*.

[22] Uriel Feige and Jan Vondrák. 2010. The Submodular Welfare Problem with Demand Queries. *TOC* (2010).

[23] Kiran Garimella, Aristides Gionis, Nikos Parotsidis, and Nikolaj Tatti. 2017. Balancing information exposure in social networks. In *Advances in Neural Information Processing Systems*. 4663–4671.

[24] Xinran He et al. 2012. Influence blocking maximization in social networks under the competitive linear threshold model. In *ICDM*.

[25] Keke Huang et al. 2017. Revisiting the Stop-and-stare Algorithms for Influence Maximization. *VLDB* (2017).

[26] Kyomin Jung et al. 2012. Irie: Scalable and robust influence maximization in social networks. In *ICDM*.

[27] Michael Kapralov et al. 2013. Online Submodular Welfare Maximization: Greedy is Optimal. In *SODA*.

[28] David Kempe et al. 2003. Maximizing the spread of influence through a social network. In *KDD*.

[29] Nitish Korula et al. 2015. Online submodular welfare maximization: Greedy beats 1/2 in random order. In *TOC*.

[30] Paul Lamere. 2008. LastFM-ArtistTags2007 dataset. http://musicmachinery.com/2010/11/10/lastfm-artisttags2007/.

[31] Hui Li et al. 2015. Conformity-aware influence maximization in online social networks. In *VLDB*. 117–141.

[32] Yuchen Li et al. 2018. Influence Maximization on Social Graphs: A Survey. *TKDE* (2018).

[33] Yuchen Li, Ju Fan, George Ovchinnikov, and Panagiotis Karras. 2019. Maximizing multifaceted network influence. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 446–457.

[34] Yishi Lin and John CS Lui. 2015. Analyzing competitive influence maximization problems with partial information: An approximation algorithmic framework. *Performance Evaluation* 91 (2015), 187–204.

[35] Wei Lu et al. 2013. The bang for the buck: fair competitive viral marketing from the host perspective. In *KDD*.

[36] Wei Lu et al. 2016. From Competition to Complementarity: Comparative Influence Diffusion and Maximization. In *VLDB*.

[37] Wei Lu, Francesco Bonchi, Amit Goyal, and Laks VS Lakshmanan. 2013. The bang for the buck: fair competitive viral marketing from the host perspective. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 928–936.

[38] Roger B Myerson. 1981. Optimal auction design. *Mathematics of operations research* (1981).

[39] Hung T Nguyen et al. 2016. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *SIGMOD*.

[40] Noam Nisan et al. 2007. *Algorithmic game theory*. Cambridge university press.

[41] Nishith Pathak et al. 2010. A generalized linear threshold model for multiple cascades. In *ICDM*.

[42] Tao Sun et al. 2011. Participation Maximization Based on Social Influence in Online Discussion Forums. In *ICWSM*.

[43] Jing Tang et al. 2018. Online Processing Algorithms for Influence Maximization. In *SIGMOD*.

[44] Youze Tang et al. 2015. Influence maximization in near-linear time: A martingale approach. In *SIGMOD*.

[45] Yuqing Zhu et al. 2016. Minimum cost seed set for competitive social influence. In *INFOCOM*.