Maximizing Welfare in Social Networks under a Utility Driven Influence Diffusion Model

Prithu Banerjee†  Wei Chen‡  Laks V.S. Lakshmanan†

University of British Columbia, {prithu,laks}@cs.ubc.ca †, Microsoft Research, weic@microsoft.com ‡

ABSTRACT
Motivated by applications such as viral marketing, the problem of influence maximization (IM) has been extensively studied in the literature. The goal is to select a small number of users to adopt an item such that it results in a large cascade of adoptions by others. Existing works have three key limitations. (1) They do not account for economic considerations of a user in buying/adopting items. (2) Most studies on multiple items focus on competition, with complementary items receiving limited attention. (3) For the network owner, maximizing social welfare is important to ensure customer loyalty, which is not addressed in prior work in the IM literature. In this paper, we address all three limitations and propose a novel model called UIC that combines utility-driven item adoption with influence propagation over networks. Focusing on the mutually complementary setting, we formulate the problem of social welfare maximization in this novel setting. We show that while the objective function is neither submodular nor supermodular, surprisingly a simple greedy allocation algorithm achieves a factor of $(1 - 1/e - \epsilon)$ of the optimum expected social welfare. We develop bundle-GRD, a scalable version of this approximation algorithm, and demonstrate, with comprehensive experiments on real and synthetic datasets, that it significantly outperforms all baselines.

CCS CONCEPTS
• Information systems → Data mining; Social networking sites; Computational advertising;

KEYWORDS
Influence maximization, Welfare maximization

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION
Motivated by applications such as viral marketing, the problem of influence maximization has been extensively studied in the literature [29]. The seminal paper of Kempe et al. [25] formulated influence maximization (IM) as a discrete optimization problem: given a directed graph $G = (V, E, p)$, with nodes $V$, edges $E$, a function $p : E \rightarrow [0, 1]$ associating influence weights with edges, a stochastic diffusion model $M$, and a seed budget $k$, select a set $S \subset V$ of up to $k$ seed nodes such that by activating the nodes $S$, the expected number of nodes of $G$ that get activated under $M$ is maximized. Two fundamental diffusion models are independent cascade (IC) and linear threshold (LT) [25]. Most of the work on IM has focused on a single item or phenomenon propagating through the network, and has developed efficient and scalable heuristic and approximation algorithms for IM [7, 14, 15, 39]. Subsequent work studies multiple campaigns propagating through a network [4, 8, 10, 19, 30, 31, 36], mostly focusing on competing campaigns. One exception is the Com-IC model by Lu et al. [31], which studied the effect of complementary products propagating through a network. A significant omission from the literature on IM and viral marketing is a study with item adoptions grounded in a sound economic footing.

Adoption of items by users is a well-studied concept in economics [33, 35]: item adoption by a user is driven by the utility that the user can derive from the item (or itemset). Precisely, a user’s utility for an item(set) is the difference between the valuation that the user has for the item(set) and the price she pays. A rich body of literature in combinatorial auctions (e.g., see [18, 24, 27]) studies the optimal allocation of goods to users, given the users’ valuation for various sets of goods. These studies are not concerned with the influence propagation in networks, whereby users’ desire of items arises due to the influence from their network neighbors who already adopted items, and then these users may in
We develop a novel block accounting method for modeling user valuation for mutually complementary items. The marginal value-gain of an item w.r.t. a set of items increases as the set grows. Many companies offer complementary products, e.g., Apple offers iPhone, and AirPod. The marginal value-gain of AirPod is higher for a user who has bought an iPhone, compared to a user who hasn’t. Complementary items have been well studied in the economics literature and supermodular function is a typical way for modeling their valuations (e.g., see [9, 40]). As a preview, our experiments show that complementary items are natural and that their valuation is indeed supermodular (Section 6.4). We study adoptions of complementary items, by combining a basic stochastic diffusion model with the utility model for item adoption.

In practice, prices of items may be known, but our knowledge of users’ valuation for items may be uncertain. Thus, we further add a random noise to the utility function. We formulate the optimization problem of finding the optimal allocation of items to seed nodes under item budget constraints so as to maximize the expected social welfare. The task is NP-hard, but more challenging is our result that the expected social welfare is neither submodular nor supermodular under the reasonable assumption that price and noise are additive. We show that we can still design an efficient algorithm that achieves an \((1 - 1/\epsilon - \epsilon)\)-approximation to the optimal expected social welfare, for any small \(\epsilon > 0\). While our main algorithm is still based on the greedy approach for solving submodular function maximization, its analysis is far from trivial, because the objective function is neither submodular nor supermodular. As part of our proof strategy, we develop a novel block accounting method for reasoning about expected social welfare for properly defined blocks of items.

An important feature of our algorithm is that it does not require the valuations or prices of items as the input, and merely the fact that item valuation is supermodular while price and noise are additive is sufficient to guarantee the approximation ratio. This means that we do not need to obtain the valuations or marginal valuations of items, which may not be straightforward to get in practice.

To summarize, in this paper, we study the problem of optimal allocation of items to seeds subject to item budgets, such that after network propagation the expected social welfare is maximized, and we make the following contributions:

1. We incorporate utility-based item adoption with influence diffusion into a novel multi-item diffusion model, Utility-driven IC (UIC) model. UIC can support any mix of competing and complementary items. In this paper, we study the social welfare maximization problem for mutually complementary items (§3).

2. We propose a greedy allocation algorithm, and show that the algorithm achieves a \((1 - 1/\epsilon - \epsilon)\)-approximation ratio, even though the social welfare function is neither submodular nor supermodular (§4 and §5). Our main technical contribution is the block accounting method, which distributes social welfare to properly defined item blocks. The analysis is highly nontrivial and may be of independent interest to other studies.

3. We design a prefix-preserving seed selection algorithm for multi-item IM that may be of independent interest, with running time and memory usage in the same order as the scalable approximation algorithm IMM [39] on the maximum budgeted item, regardless of the number of items (§5).

4. We conduct detailed experiments comparing the performance of our algorithm with baselines on five large real networks, with both real and synthetic utility configurations. Our results show that our algorithm significantly dominates the baselines in terms of running time or expected social welfare or both (§6).

All proofs and additional examples that are omitted here for the lack of space, can be found in our full report [3].

2 BACKGROUND & RELATED WORK

Single Item IM: A social network is represented as a directed graph \(G = (V, E, p)\) as described in §1. Two of the classic diffusion models are independent cascade (IC) and linear threshold (LT).

We briefly review the IC model. Given a set \(S \subseteq V\) of seeds, diffusion proceeds in discrete time steps. At \(t = 0\), only the seeds are active. At every time \(t > 0\), each node \(u\) that became active at time \(t - 1\) makes one attempt at activating each of its inactive out-neighbors \(v\), i.e., it tests if the edge \((u, v)\) is “live” or “blocked”. The attempt succeeds (the edge \((u, v)\) is live) with probability \(p_{uv} = p(u, v)\). The diffusion stops when no more nodes become active. We refer
the reader to [16, 25] for details. The influence spread of a seed set $S$, denoted $\sigma(S)$, is the expected number of active nodes after the diffusion that starts from the seed set $S$ ends.

**Influence maximization (IM)** is the problem of finding, for a seed budget $k$ and a diffusion model, a set $S \subseteq V$ of at most $k$ seeds that maximizes the influence spread $\sigma(S)$ [25]. A set function $f: 2^U \rightarrow \mathbb{R}$ is monotone if $f(S) \leq f(T)$ whenever $S \subseteq T \subseteq U$; submodular if for any $S \subseteq T \subseteq U$ and any $x \in U \setminus T$, $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$; $f$ is supermodular if the inequality above is reversed; and $f$ is modular if it is both submodular and supermodular. Under both IC and LT models, IM is intractable [14, 15, 25], but $\sigma(\cdot)$ is monotone and submodular for both models. Thus, a simple greedy seed selection algorithm together with Monte Carlo simulation for estimating the spread, achieves an $(1-1/e-\epsilon)$-approximation, for any $\epsilon > 0$ [24, 25]. Several heuristics for IM and its extensions were proposed over the years [12, 14, 15, 23, 26, 28]. Building on the notion of RR sets, proposed by Borgs et al. [7], a family of scalable approximation algorithms such as IMM and SSA have been developed for IM [21, 34, 39].

**Multi-item IM:** Recent studies on IM allow for multiple campaigns, covering both independent (thus, non-competing) items [17], and competing items [4, 8, 19, 30, 36] (see [16] for a survey), where a user adopts at most one item from the set of items being propagated.

Lu et al. [31] introduced a model called Com-IC capturing both competition and complementarity between a pair of items. Their model subsumes perfect complementarity and pure competition as special cases. However, they do not consider item adoption driven by utility considerations, and only use probability parameters for adoption. Their main study is confined to the diffusion of two items, and a straightforward extension to multiple items would need an exponential number of probability parameters in the number of items. Different from us, the above works on multi-item IM focus on maximizing expected number of item adoptions.

**Combinatorial Auctions:** In economics, adoption of items by users is modeled in terms of the utility that the user derives from the adoption [33, 35]. A classic problem is given $m$ users and $n$ items and the utility functions of users for various subsets of items, find an allocation of items to users such that the social welfare, i.e., the sum of utilities of users for allocated itemsets, is maximized. This is intractable and approximation algorithms have been developed [18, 24, 27]. These works are not concerned with the interaction of utility-maximizing item adoption with recursive propagation through a network.

**Welfare maximization on social networks:** There are a few studies related to welfare maximization on social networks, but they all have significant differences with our model and problem setting. Sun et al. [37] study participation maximization in the context of online discussion forums. An item in that context is a discussion topic, and adopting an item means posting or replying on the topic. Item adoptions do propagate in the network, but (a) item propagations are independent (i.e., valuation of itemsets is additive rather than supermodular or submodular), and (b) they have a budget on the number of items each seed node can be allocated with, rather than on the number of seeds each item can be allocated to as studied in our model. Bhattacharya et al. [5] consider item allocations to nodes for welfare maximization in a network with network externalities, but the major differences with our problem are: (a) they use network externalities to model social influence, i.e., a user’s valuation of an item is affected by the number of her one- or two-hop neighbors in the network adopting the same item, but network externalities do not model the propagation of influence and item adoptions, our main focus in modeling the viral marketing effect; (b) they consider unit demand or bounded demand on each node, which means items are competing against one another on every node, while our study focuses on the case of complementary items rather than competing items, and item bundling is a key component in our solution; (c) they do not have budget on items so an item could be allocated to any number of nodes, while we have a budget on the number of nodes that can be allocated to an item as seeds and we rely on propagation for more nodes to adopt items. Despite these major differences, we will do an empirical comparison of our algorithm versus theirs algorithms. Abramowizt and Anshelevich [2] study network formation with various constraints to maximize social welfare, but it has no item allocation, no item complementarity, and no influence propagation, and thus is further away from our work. In summary, to our knowledge, our study is the only one addressing social welfare maximization in a network with influence propagation, complementary items, and budget limits on items.

## 3 UIC Model

In this section, we propose a novel model called utility driven independent cascade model (UIC for short) that combines the diffusion dynamics of the classic IC model with an item adoption framework where decisions are governed by utility.

Table 5 (Appendix A.1) summarizes the notations.

**Utility based adoption.** Utility is a widely studied concept in economics and is used to model item adoption decisions of users [18, 33, 35].

We let $I$ denote a finite universe of items. The utility of a set of items $I \subseteq I$ for a user is the pay-off of $I$ to the user and depends on the aggregate effect of three components: the price $\mathcal{P}$ that the user needs to pay, the valuation $\mathcal{V}$ that the user has for $I$ and a random noise term $\mathcal{N}$, used to model the uncertainty in our knowledge of the user’s valuation on items, where $\mathcal{P}$, $\mathcal{V}$ and $\mathcal{N}$ are all set functions over items. For an item $i \in I$, $\mathcal{P}(i) > 0$ denotes its price. We assume that
price is additive, i.e., for an itemset $I \subseteq I$, $\mathcal{P}(I) = \sum_{i \in I} \mathcal{P}(i)$. Although UIC can handle any generic valuation function, in this paper we focus on complementary products. Hence we assume that $\mathcal{V}$ is supermodular (definition in §2), meaning that the marginal value of an item with respect to an itemset $I$ increases as $I$ grows. We also assume $\mathcal{V}$ is monotone since it is a natural property for valuations. For $i \in I$, $N(i) \sim D_i$ denotes the noise term associated with item $i$, where the noise may be drawn from any distribution $D_i$ having a zero mean. Every item has an independent noise distribution. For a set of items $I \subseteq I$, we assume the noise is additive, i.e., the noise of $I$, $N(I) := \sum_{i \in I} N(i)$. Similar assumptions on additive noise are used in economics theory [20, 22].

Finally, the utility of an itemset $I$ is $\mathcal{U}(I) = \mathcal{V}(I) - \mathcal{P}(I) + N(I)$. Since noise is a random variable, utility is also random. Since noise is drawn from a zero mean distribution, $\mathbb{E}[\mathcal{U}(I)] = \mathcal{V}(I) - \mathcal{P}(I)$. We assume $\mathcal{U}(\emptyset) = 0$.

**Seed allocation.** Let $b = (b_1, \ldots, b_H)$ be a vector of natural numbers representing the budgets associated with the items. An item’s budget specifies the number of seed nodes that may be assigned to that item. We sometimes abuse notation and write $b_i \in b$ to indicate that $b_i$ is one of the item budgets. We denote the maximum budget as $\bar{b} := \max\{b_1 | b_i \in b\}$. We define an allocation as a relation $\mathcal{S} \subset V \times I$ such that $\forall i \in I : |\{(v, i) \mid v \in V\}| \leq b_i$. In words, each item is assigned a set of nodes whose size is under the item’s budget.

We refer to the nodes $S_i^\mathcal{S} := \{v \mid \langle v, i \rangle \in \mathcal{S}\}$ as the seed nodes of $\mathcal{S}$ for item $i$ and to the nodes $S_i^\mathcal{S} := \bigcup_{i \in I} S_i^\mathcal{S}$ as the seed nodes of $\mathcal{S}$. We denote the set of items allocated to a node $v \in V$ as $I_v^\mathcal{S} := \{i \in I \mid \langle v, i \rangle \in \mathcal{S}\}$.

**Desire and adoption.** Every node maintains 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 the node adopts. At any time a node selects, from its desire set at that time, the subset of items that maximizes the utility, and adopts it. If there is a tie in the maximum utility between itemsets, then it is broken in favor of larger itemsets. We later show in Lemma 1 of §4 that breaking ties in this way results in a well-defined adoption behavior of the nodes. We consider a progressive model: once a node desires an item, it remains in the node’s desire set forever; similarly, once an item is adopted by a node, it cannot be unadopted later.

For a node $u$, $\mathcal{R}^\mathcal{S}(u, t)$ denotes its desire set and $\mathcal{A}^\mathcal{S}(u, t)$ denotes its adoption set at time $t$, pertinent to an allocation $\mathcal{S}$. We omit the time argument $t$ to refer to the sets at the end of diffusion. We now present the diffusion under UIC.

**Diffusion model.** In the beginning of any diffusion, the noise terms of all items are sampled, which are then used till the diffusion terminates. The diffusion then proceeds in discrete time steps, starting from $t = 1$. Given an allocation $\mathcal{S}$ at $t = 1$, the seed nodes have their desire sets initialized: $\forall v \in S, \mathcal{R}^\mathcal{S}(v, 1) = I_v^\mathcal{S}$. Seed nodes then adopt the subset of items from the desire set that maximizes the utility, breaking ties if needed in favor of sets of larger cardinality. Thus, a seed node may adopt just a subset of items allocated to it.

Once a seed node $u$ adopts an item $i$, it influences its out-neighbors $u$ with probability $P_{u', u}$, and if it succeeds, then $i$ is added to the desire set of $u$ at time $t = 2$. The rest of the diffusion process is described in Fig. 1.

**Figure 1: Diffusion dynamics under UIC model**

We illustrate the diffusion under UIC using an example scenario shown in Figure 2. The graph $G$ with edge probabilities and the utilities of the two items after sampling the noise terms, are shown on the left side. At time $t = 1$, node $v_1$ is seeded with item $i_1$ and $v_3$ with $i_2$, hence they desire those items respectively. Since $i_1$ (resp. $i_2$) has a positive (resp. negative) individual utility, $v_1$ adopts $i_1$ (resp. $v_3$ does not adopt $i_2$). However $i_2$ remains in the desire set of $v_3$. Then at $t = 2$, outgoing edges of $v_1$ are tested for transition: edge $(v_1, v_3)$ fails (shown as red dotted line), but edge $(v_1, v_2)$ succeeds (green solid line). Consequently $v_2$ desires and adopts $i_2$. Next at $t = 3$, $v_2$’s outgoing edge $(v_2, v_3)$ is tested. As it succeeds, $v_3$ desires $i_1$. Since it already had $i_2$ in its desire set, it adopts the set $\{i_1, i_2\}$. Propagation ends at $v_3$.

**Social welfare Maximization.** Let $G = (V, E, p)$ be a social network, $I$ the universe of items under consideration. Here, we consider a novel utility-based objective called social welfare, which is the sum of all users’ utilities of itemsets adopted by them after propagation converges. Formally, $\mathbb{E}[\mathcal{U}(\mathcal{A}(u))]$ is the expected utility that a user $u$ enjoys for a seed allocation $\mathcal{S}$ after propagation ends. Then the expected social welfare (also known as “consumer surplus” in algorithmic game theory) for $\mathcal{S}$, is $P(\mathcal{S}) = \sum_{u \in V} \mathbb{E}[\mathcal{U}(\mathcal{A}(u))]$, where the expectation is over both the randomness of propagation and randomness of noise.
We define the problem of maximizing expected social welfare (WelMax) as follows. We refer to $V, \mathcal{P}, N$, as the model parameters and denote them collectively as Param.

**Problem 1 (WelMax).** Given $G = (V, E, p)$, the set of model parameters Param, and budget vector $\mathbf{b}$, find a seed allocation $S^*$, such that $\forall i \in 1, |S_i^*| \leq b_i$ and $S^*$ maximizes the expected social welfare, i.e., $S^* = \arg\max_S \rho(S)$.

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

**Function types.** Notice that the functions $\mathcal{V}$ and $\mathcal{U}$ are functions over sets of items, whereas $\sigma$ is a function over sets of network nodes, and $\rho$ is a function over allocations, which are sets of (node, item) pairs. When we speak of a certain property (e.g., submodularity) of a function of a given type, the property is meant w.r.t. the applicable type. E.g., $\sigma$ is monotone and submodular w.r.t. sets of nodes.

**Design choices.** In the UIC model, the desire set of a user is triggered either by seeing or by the influence of a user as her peers adopt items. Following standard practices in IM models, we keep it progressive: a desire set never shrinks. On the other hand, the adoption decisions are driven by a standard assumption in economics [6], that users aim to maximize the utility when they adopt items. This assumption governs adoption decisions of the users. In UIC, we assume price is additive. There are different ways of pricing a bundle of items: additivity is a simple and natural pricing model in the absence of discounts [11]. Further, we use supermodular value functions to model the effect of complementarity among products, again following standard practice in the economics literature [32, 40]. Finally, our way of modeling the noise can be viewed as reflecting the uncertainty in the population’s reaction to an item. One may further introduce personalized noise to model individual uncertainty, but this would make algorithm design and analysis more difficult. Our approximation bound would not hold when noise is personalized and when valuation is not supermodular. Although we make specific design choices in this paper for simplicity and tractability of the model, the UIC model can encompass any general form of value, price, and noise parameters and works for any triggering model [25].

**4 PROPERTIES OF UIC**

Since WelMax is NP-hard, we explore properties of the welfare function – monotonicity and submodularity, which can help us design efficient approximation strategies. We begin with an equivalent possible world model to help our analysis.

**Possible world model.** Given an instance $(G, \text{Param})$ of UIC, where $G = (V, E, p)$, we define a possible world associated with the instance, as a pair $W = (W^E, W^N)$, where $W^E$ is an edge possible world (edge world), and $W^N$ is a noise possible world (noise world). $W^E$ is a sample graph drawn from the distribution associated with $G$ by sampling edges, and $W^N$ is a sample of noise terms for each item in $I$, drawn from the corresponding item’s noise distribution in Param. As all the random terms are sampled, propagation and adoption in $W$ is fully deterministic. For nodes $u, v \in V$, we say $v$ is reachable from $u$ in $W$ if there is a directed path from $u$ to $v$ in the deterministic graph $W^E$. $N_W(t)$ denotes the sampled noise for item $i$ and $\mathcal{U}_W(t)$ denotes the (deterministic) utility of itemset $I$, in world $W$. For a node $u$ and an allocation $S$, we denote its desire and adoption sets at time $t$ as $D(u, t)$ and $A_{\mathcal{U}}(u, t)$ respectively. When only the noise terms are sampled, i.e., in a noise world $W^N$, the utilities are deterministic, but the propagation remains random.

Given a possible world $W = (W^E, W^N)$ and an allocation $S$, a node $v \in V$ adopts a set of items as follows: (i) if $v$ is a seed node, then it desires $I^S_v$ at time $t = 1$ and adopts an itemset $A^S_{\mathcal{U}}(v, 1) := \arg\max_i \{U_W(l) \mid l \subseteq I^S_v\}$; (ii) if $v$ is a non-seed node, and $t > 1$, then it desires the itemset $R^S_{\mathcal{U}}(v, t) := (\bigcup_{u \in N_W^{\mathcal{U}}(v)} A^S_{\mathcal{U}}(u, t - 1)) \cup A^S_{\mathcal{U}}(v, t - 1)$, where $N_{\mathcal{U}}^W(v)$ denotes the in-neighbors of $v$ in the deterministic graph $W^E$, i.e., at time $t$, node $v$ desires items that it desired at $(t - 1)$ as well as items any of its in-neighbors in $W^E$ adopted at $(t - 1)$; node $v$ then adopts the itemset $A^S_{\mathcal{U}}(v, t) := \arg\max_{l \subseteq I} \{U_W(l) \mid l \subseteq R^S_{\mathcal{U}}(v, t) \cup A^S_{\mathcal{U}}(v, t - 1) \subseteq I\}$. If there is more than one itemset in $R^S_{\mathcal{U}}(v, t)$ with the same maximum utility, we assume that $v$ breaks ties in favor of the set with the larger cardinality.

$\mathcal{V}(\cdot)$ is supermodular while $\mathcal{P}(\cdot)$ and $N_{\mathcal{U}}(\cdot)$ are additive and hence modular, so it immediately follows that $\mathcal{U}_{\mathcal{W}}(\cdot)$ is supermodular with respect to sets of items. Thus the expectation of utility w.r.t. edge worlds is supermodular. However, $\mathcal{U}_{\mathcal{W}}(\cdot)$ is not monotone, because adding an item with a very high price may decrease the utility.
desires, there is a unique set of items that it adopts. Specifically, if there are multiple sets tied for utility, the node will adopt their union. For a set function \( f : 2^U \rightarrow R \), we define \( f(T \mid S) = f(S \cup T) - f(S) \). We say that an itemset \( A \) is a local maximum w.r.t. the utility function \( \mathcal{U}_W \), if the utility of \( A \) is the maximum among all its subsets, i.e., \( \mathcal{U}_W(A) = \max_{A' \subseteq A} \mathcal{U}_W(A') \). The following lemma is based on simple algebraic manipulations on the definitions of supermodularity and local maximum.

**Lemma 1.** (Local maximum). Let \( W \) be a possible world and \( A, B \subseteq I \) be any itemsets such that \( A \) and \( B \) are local maximum with respect to \( \mathcal{U}_W \). Then \( (A \cup B) \) is also a local maximum with respect to \( \mathcal{U}_W \), i.e., \( \mathcal{U}_W(A \cup B) = \max_{C \subseteq A \cup B} \mathcal{U}_W(C) \). An immediate consequence of Lemma 1 is that when two itemsets have the same largest utility, their union must also have the largest utility, and thus our tie-breaking rule is well-defined. Another consequence is the following lemma.

**Lemma 2.** For any node \( u \) and any time \( t \), the itemset adopted by \( u \) at time \( t \), \( A_W^u(u, t) \), must be a local maximum.

Our next result shows that in any given possible world, adoption of items propagates through reachability. Reachability is a key property to be used later in Lemmas 5 and 7 while establishing the approximation guarantee of our algorithm.

**Lemma 3.** (Reachability). For any item \( i \) and any possible world \( W \), if a node \( u \) adopts \( i \) under allocation \( \mathcal{S} \), then all nodes that are reachable from \( u \) in the world \( W \) also adopt \( i \).

The social welfare of an allocation \( \mathcal{S} \) in a possible world \( W = (W_E, W_N) \) is defined as the sum of utilities of itemsets adopted by nodes, i.e., \( \rho_W(\mathcal{S}) := \sum_{v \in V} \mathcal{U}(A_W^v(v)) \). The expected social welfare of an allocation \( \mathcal{S} \) is \( \rho(\mathcal{S}) := \mathbb{E}_{W_E}[\mathbb{E}_{W_N}[\rho_W(\mathcal{S})]] = \mathbb{E}_{W_N}[\mathbb{E}_{W_E}[\rho_W(\mathcal{S})]] \). It is straightforward to show that the expected social welfare of allocation \( \mathcal{S} \) defined in §3 is equivalent to the above definition.

**Properties of social welfare.** The following theorem summarizes the property of social welfare function. It is not submodular because the valuation is supermodular, and it is not supermodular because the propagation based on IC model would have submodular influence coverage.

**Theorem 1.** Expected social welfare is monotone with respect to the sets of node-item allocation pairs. However it is neither submodular nor supermodular.

## 5 APPROXIMATION ALGORITHM

### 5.1 Greedy algorithm overview

Given that the welfare function is neither submodular nor supermodular, designing an approximation algorithm for WellMax is challenging. Nevertheless, in this section we show that for any given \( \epsilon > 0 \) and number \( \ell \geq 1 \), a \( (1 - \frac{1}{e} - \epsilon) \)-approximation to the optimal social welfare can be achieved with probability at least \( 1 - \frac{1}{|V|} \), using a simple greedy algorithm. To the best of our knowledge, this is the first instance in the context of viral marketing where an efficient approximation algorithm is proposed for a non-submodular objective, at the same level as submodular objectives. We first present our algorithm and then analyze its correctness and efficiency.

Our algorithm, called bundleGRD (for bundle greedy) and shown in Algorithm 1, is based on a greedy allocation of seed nodes to items. Given a graph \( G \), the universe of items \( I \), item budget vector \( \vec{b} \), \( \epsilon \), and \( \ell \), bundleGRD first selects (line 2) the top-\( \vec{b} \) seed nodes \( S^{Grd} := S_G \) for the IC model (disregarding item utilities), where \( \vec{b} = \max(b_i \mid b_i \in \vec{b}) \). Then, (line 4) for each item \( i \) with budget \( b_i \), it assigns the top-\( b_i \) nodes from \( S^{Grd} \) to \( i \). We will show that this allocation achieves a \( (1 - \frac{1}{e} - \epsilon) \)-approximation to the optimal expected social welfare. For this to work, bundleGRD must ensure that the \( \vec{b} \) seeds selected, \( S_G \), satisfy a prefix-preserving property.

We define the property and present the PRIMA algorithm (invoked in line 2 of Algorithm 1), in §5.3. The following is the main result for the bundleGRD algorithm.

**Theorem 2.** Let \( S^{Grd} \) be the greedy allocation generated by bundleGRD, and \( S^{OPT} \) be the optimal allocation. Given \( \epsilon > 0 \) and \( \ell > 0 \), with probability at least \( 1 - \frac{1}{|V|} \), we have

\[
\rho(S^{Grd}) \geq (1 - \frac{1}{e} - \epsilon) \cdot \rho(S^{OPT}).
\]

The running time is \( O((\vec{b} + \ell + \log n) |\vec{b}|(m+n) \log n/e^2) \).

We note that our bundleGRD algorithm has the interesting property that it does not need the valuation functions, prices, and the distributions of noises as input, and thus works for all possible utility settings. It reflects the power of bundling in the complementary setting.

### 5.2 Block accounting for bundleGRD

The analysis of the algorithm is highly non-trivial, because it needs to consider all possible seed allocations, propagation scenarios, with budgets possibly being non-uniform among items. Our main idea is a “block” based accounting method: we break the set of items into a sequence of “atomic” blocks, such that each block has non-negative marginal utility given previous blocks, and it can be counted as an atomic unit in the diffusion process. Then we account for each block’s
contribution to the social welfare during a propagation, and argue that for every block, the contribution of the block achieved by the greedy allocation is always at least \((1 – 1/\epsilon – \epsilon)\) times the contribution under any allocation. In §5.2.1 we first introduce the block generation process. Using block based accounting, in §5.2.2 we establish the welfare produced by bundleGRD, and in §5.2.3, show an upper bound on the welfare produced by any arbitrary allocation. The technical subtlety includes properly defining the blocks, showing why each block can be accounted for as an atomic unit separately, dealing with partial item propagation within blocks, etc.

In the rest of the analysis, we fix the noise world \(W^N\), and prove that \(\rho_{W^N}(S^{\text{Grd}}) \geq (1 – \frac{1}{\epsilon} – \epsilon) \cdot \rho_{W^N}(S^{\text{OPT}})\), where \(\rho_{W^N}\) denotes the expected social welfare under the fixed noise world \(W^N\). We then take another expectation over the distribution of \(W^N\) to obtain Inequality (1). Let \(U_{W^N}\) be the utility function under the noise possible world \(W^N\).

Given \(W^N\), let \(I_{W^N}^*\) be the subset of items that gives the largest utility in \(W^N\), with ties broken in favor of larger sets. By Lemma 1, \(I_{W^N}^*\) is unique. This implies that the marginal utility of any (non-empty) subset of \(I \setminus I_{W^N}^*\) given \(I_{W^N}^*\) is strictly negative. Further recall that \(U_{W^N}\) is supermodular. Hence the marginal utility of any subset of \(I \setminus I_{W^N}^*\) given any subset of \(I_{W^N}^*\) is strictly negative, which means no items in \(I \setminus I_{W^N}^*\) can be ever accepted by any user under the noise world \(W^N\). Thus, once we fix \(W^N\), we can safely remove all items in \(I \setminus I_{W^N}^*\) from consideration. In the rest of §5.2, for simplicity we use \(I^*\) as a shorthand for \(I_{W^N}^*\).

5.2.1 Block generation process. We divide items in \(I^*\) into a sequence of disjoint blocks such that each block has a non-negative marginal utility w.r.t. the union of all its preceding blocks. We also need to carefully arrange items according to their budgets for later accounting analysis. We next discuss how the blocks are generated.

Let \(I^* = \{i_1, \ldots, i_{|I^*|}\}\). We order the items in non-increasing order of their budgets, i.e., \(b_1 \geq b_2 \geq \cdots \geq b_{|I^*|}\). Figure 3 shows the process of generating the blocks. Note that this block generation process is solely used for our accounting analysis and is not part of our seed allocation algorithm. Given \(I^*\) and \(W^N\), we first generate a global sequence \(I\) of all non-empty subsets of \(I^*\), following a precedence order \(\prec\) (Step 2).

For two distinct subsets \(S, S' \subseteq I^*\), arrange items in each of \(S, S'\) in decreasing order of item indices. Compare items in \(S, S'\), starting from the highest indexed items of \(S\) and \(S'\). If they match, compare the second highest indexed items and so on until one of the following rules applies:
1. One of \(S\) or \(S'\) exhausts. If say \(S\) exhausts first, then \(S \prec S'\).
2. The current pair of items in \(S\) and \(S'\) do not match. Then \(S \prec S'\) if the current item of \(S\) has a lower index than the current item of \(S'\).

Figure 3: The block generation process

Example 1 (Generation of \(I\)). Suppose we have three items \(I^* = \{i_1, i_2, i_3\}\) with \(b_1 \geq b_2 \geq b_3\), then we order the subsets in the following way: \(I = \{\{i_1\}, \{i_2\}, \{i_1, i_2\}, \{i_1, i_3\}, \{i_1, i_2, i_3\}\}\). Between subsets \(\{i_1\}\) and \(\{i_1, i_3\}\), \(\{i_1, i_2, i_3\}\) is ordered first according to rule 1, whereas between \(\{i_1, i_2\}\) and \(\{i_3\}\), \(\{i_1, i_2, i_3\}\) is ordered first according to rule 2. \(\Box\)

The sequence \(I\) has the following useful property:

Property 1. For any subsets \(S\) and \(T\) in the sequence \(I\), if \((a)\) \(T\) is a proper subset of \(S\), or \((b)\) the highest index among all items in \(T\) is strictly lower than the highest index among all items in \(S\), then \(T\) appears before \(S\) in \(I\).

From \(I\), blocks are selected following an iterative process, as shown in Step 3 of Figure 3.

By the fact that \(I^*\) is a local maximum, it is easy to see that the blocks generated form a partition of \(I^*\). Let \(B = \{B_1, B_2, \ldots, B_t\}\) be the sequence of blocks generated, where \(t\) is the number of blocks in the block partition. We define the marginal gain of each block \(B_i\) as

$$\Delta_i = U_{W^N}(B_i | \bigcup_{j=1}^{i-1} B_j).$$

Marginal gains have the following properties.

Property 2. \(\forall i \in [t], \Delta_i \geq 0\), and \(U_{W^N}(I^*) = \sum_{i=1}^{t} \Delta_i\).

Let \(A \subseteq I^*\) be an arbitrary subset of items. We partition \(A\) based on block partition \(B\). Define \(A_j = A \cap B_j, \forall i \in [t]\). If \(A_j = B_j\), we call \(A_j\) a full block, if \(A_j = \emptyset\), then it is an empty block, otherwise, we call it a partial block. Define \(A_j^A = U_{W^N}(A_j | A_j \cup \ldots \cup A_{j-1})\). By Property 1 and the fact that \(B_1\) is the first block in \(I\) with non-negative marginal utility w.r.t. \(\bigcup_{j=1}^{t} B_j\), it follows that

Property 3. \(\forall i \in [t], \Delta_i^A \leq \Delta_i\), and \(U_{W^N}(A) = \sum_{i=1}^{t} \Delta_i^A\).

Using this property, we devise our accounting where each \(A_j\) contributes \(\Delta_i^A\) in its social welfare.

5.2.2 Social welfare under greedy allocation. We are now ready to analyze the social welfare of our greedy allocation (Algorithm 1) using block accounting. We first show that, before the propagation starts, each seed node would adopt exactly the prefix of all blocks allocated until the first
non-full block, and then show that all these adopted full blocks will propagate together, so we can exactly account for the contribution of each block to the expected social welfare. The following lemma gives the exact statement of the first part.

**Lemma 4.** Under the greedy allocation, suppose that at a seed node \( v \), \( \sigma_i \) is the first non-full block assigned to \( v \), then before the propagation starts, \( v \) adopts exactly \( B_1 \cup \cdots \cup B_{i-1} \).

**Effective budget of blocks.** For a block \( B_i \), we define its effective budget \( e_i = \min_{j \in B_i \cup \cdots \cup B_j} b_j \). In bundleGRD (Algorithm 1), the first \( e_i \) seed nodes of \( S^{Grd} \) are assigned all the full blocks \( \{ B_1 \cup \cdots \cup B_i \} \). By Lemma 4, only those nodes actually adopt the block \( B_i \) before the propagation starts. Such seed nodes are called effective seed nodes of block \( B_i \) and denoted as \( S_{B_i}^{Grd} \). Thus in summary, under the greedy allocation, before the propagation starts, all seed nodes in \( S_{B_i}^{Grd} \) adopt \( B_i \) together with \( B_1 , \ldots, B_{i-1} \), and none of the seed nodes outside \( S_{B_i}^{Grd} \) adopts any items in \( B_1 , B_{i+1}, \ldots, B_i \).

We now show the social welfare of bundleGRD.

**Lemma 5.** Let \( S^{Grd} \) be the greedy allocation obtained using Algorithm 1. Then the expected social welfare of \( S^{Grd} \) in \( W^N \) is \( \rho_{W^N}(S^{Grd}) = \sum_{i \in [1]} \sigma(S_{B_i}^{Grd}) \cdot \Delta_i \), where \( S_{B_i}^{Grd} \) are the effective seed nodes of block \( B_i \) under allocation \( S^{Grd} \), \( \sigma(\cdot) \) is the expected spread function under the IC model, and \( \Delta_i \) is as defined in Eq. (2).

### 5.2.3 Social welfare under an arbitrary allocation.

Unlike greedy, in an arbitrary allocation, for the effective seed nodes, we cannot conclude that a block \( B_i \) is offered with all previous full blocks \( B_1, \ldots, B_{i-1} \). Thus our accounting method needs to be adjusted. Our idea is to define the key concept of an anchor item \( a_i \) for every block \( B_i \), which appears in \( B_1 \cup \cdots \cup B_i \). We want to show that only when \( B_i \) is co-adopted with \( a_i \) by any node, \( B_i \) could contribute positive marginal social welfare (Lemma 6), and in this case its marginal contribution is upper bounded by \( \Delta_i \) (Property 3). Hence we only need to track the diffusion of the anchor item \( a_i \) to account for the marginal contribution of \( B_i \). Finally by showing that the budget of \( a_i \) is exactly the effective budget \( e_i = |S_{B_i}^{Grd}| \) of \( B_i \), we conclude that \( \sigma(S_{a_i}) \leq (1 - 1/e - \epsilon) \sigma(S_{B_i}^{Grd}) \) by the prefix-preserving property explained in §5.1.

We define the budget of a block to be the minimum budget of any item in the block. Then the anchor block \( B_i^a \) of a block \( B_i \) is the block from \( B_1, \ldots, B_i \) that has the minimum budget. In case of a tie, the block having highest index is chosen as the anchor block. Notice that anchor item \( a_i \) is the highest indexed and consequently minimum budgeted item in its corresponding anchor block \( B_i^a \). Notice that, by definition, if block \( B_j \) is the anchor block of block \( B_i \) with \( j < i \), then block \( B_j \) is also the anchor block for all blocks \( B_j, B_{j+1}, \ldots, B_i \). Moreover, the effective budget \( e_i \) of a block \( B_i \), is the budget of its anchor item \( a_i \), i.e., the minimum budget of all items in \( B_1 \cup \cdots \cup B_i \).

**Lemma 6.** Let \( a_i \) be the anchor item of \( B_i \), and suppose \( a_i \) appears in \( B_j \), \( j \leq i \). During the diffusion process from an arbitrary seed allocation \( S \), let \( A \) be the set of items in \( B_j \cup \cdots \cup B_i \) that have been adopted by \( v \) by time \( t \). If \( a_i \notin A \) and \( A \neq \emptyset \), then \( U_{W^N}(A | B_1, \ldots, B_{i-1}) < 0 \).

Using the above result, we establish the following lemma, which upper bounds the welfare produced by an arbitrary allocation.

**Lemma 7.** For any arbitrary seed allocation \( S \), the expected social welfare in \( W^N \) is \( \rho_{W^N}(S) \leq \sum_{i \in [1]} \sigma(S_{a_i}) \cdot \Delta_i \), where \( S_{a_i} \) is the seed set assigned to the anchor item \( a_i \) of block \( B_i \), and \( \Delta_i \) is as defined in Eq. (2).

Notice in Lemma 7, \( |S_{a_i}| \leq e_i \), whereas in Lemma 5 \( |S_{B_i}^{Grd}| = e_i \). Hence the combination of Lemma 5 and Lemma 7, together with the fact that \( S_{B_i}^{Grd} \) is a \((1 - 1/e - \epsilon)\)-approximation of the optimal solution with \( e_i \) seeds (by the prefix-preserving property), leads to the approximation guarantee of bundleGRD (Eq. (1) of Theorem 2). In the next section, we explain the component PRIMA that provides the prefix-preserving property.

### 5.3 Item-wise prefix preserving IMM

We first formally define the prefix-preserving property.

**Definition 1.** (Prefix-Preserving Property). Given \( G = (V, E, p) \) and budget vector \( \bar{b} \), an influence maximization algorithm \( \hat{A} \) is prefix-preserving w.r.t. \( \bar{b} \), if for any \( e > 0 \) and \( \ell > 0 \), \( \hat{A} \) returns an ordered set \( S^e_{\bar{b}} \) of size \( e \), such that with probability at least \( 1 - \frac{1}{|V^e\sigma(\bar{b})|} \), for every \( b \in S^e_{\bar{b}} \), the top-\( b \) nodes of \( S^e_{\bar{b}} \), denoted \( S_{B_i}^{Grd} \), satisfies \( \sigma(S_{B_i}^{Grd}) \geq (1 - 1/e - \epsilon) \cdot OPT_{\bar{b}} \), where \( OPT_{\bar{b}} \) is the optimal expected spread of \( b \).

Unfortunately, state-of-the-art IM algorithms such as IMM [39], SSA [34], and OPIM [38] are not prefix-preserving out-of-the-box. In this section, we present a non-trivial extension of IMM [39], called PRIMA (PRefix preserving IM Algorithm) (Algorithm 2), to make it prefix-preserving.

State-of-the-art IM algorithms including IMM use reverse influence sampling (RIS) approach [7] governed by reverse-reachable (RR) sets. An RR set is a random set of nodes sampled from the graph by (a) first selecting a node \( v \) uniformly at random from the graph, and (b) then simulating the reverse propagation of the model (e.g., IC model) and adding all visited nodes into the RR set. The main property of a random RR set \( R \) is that: influence spread \( \sigma(S) = n \cdot \mathbb{E}[|S \cap R \neq \emptyset|] \) for any seed set \( S \), where \( \mathbb{I} \) is the indicator function. After finding a large enough number of RR sets, the original influence maximization problem is turned into a \( k \)-max coverage...
problem – finding a set of $k$ nodes that covers the most number of RR sets, where a set $S$ covers an RR set $R$ if $S \cap R \neq \emptyset$. All RIS algorithms use the same well-known coverage procedure, denoted as NodeSelection($\mathcal{R}, k$) in [39], and thus we omit its description here. Different RIS algorithms differ in estimating the number of RR sets needed for the approximation guarantee. The number of RR sets generated is in general not monotone in the budget $k$, making them not prefix-preserving. Our PRIMA algorithm carefully addresses this issue for IMM, even with nonuniform item budgets, while keeping the efficiency of the algorithm. In §A.3, we discuss how other RIS algorithms can be so extended.

PRIMA ingests four inputs, namely the budget vector $\vec{b}$ sorted in non-increasing order, graph $G$, $\epsilon$ and $\ell'$ (with $\ell'$ derived from $\ell$). Extending the bounding technique of [39], for each budget $k$, we set

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

$$\lambda_k = 2n \cdot ((1 - 1/e) \cdot \alpha + \beta_k)^2 \cdot e^{-2}$$

where, $\alpha = \sqrt{\ell' \log n + \log 2}$ is a constant independent of $k$, and $\beta_k = \sqrt{(1 - 1/e) \cdot (\log (\frac{n}{\lambda'_k}) + \ell' \log n + \log 2)}$. Note that we use log without a base to represent the natural logarithm.

Algorithm 2: PRIMA ($\vec{b}, G, \epsilon, \ell'$)

1. Initialize $\mathcal{R} = \emptyset$, $s = 1$, $n = |V|$, $i = 1$, $\epsilon' = \sqrt{2} - \epsilon$, $\text{budgetSwitch} = \text{false}$.
2. $\ell = \ell + \log 2 / \log n$, $\ell' = \log_2 (n \cdot |\vec{b}|)$.
3. while $i \leq \log_2 (n) - 1$ and $s \leq |\vec{b}|$ do
   4. $k = b_s$, $\ell B = 1$.
   5. $s = \frac{n}{\ell B} + \theta_i = \ell' / x$, where $\ell'_x$ is defined in Eq. (3);
   6. while $|\mathcal{R}| \leq \theta_i$ do
      7. Generate an RR set for a randomly selected node $v$ of $G$ and insert in $\mathcal{R}$;
     8. if budgetSwitch then $S_k = \text{the first } k \text{ nodes in the ordered set } S_{b_{s-1}}$ returned from the previous call to NodeSelection;
    9. else
      10. $S_k = \text{NodeSelection} (\mathcal{R}, k)$;
    11. if $n - \text{OPT}(S_k) \geq (1 + \epsilon') \cdot x$ then $\ell B = n - \text{OPT}(S_k) / (1 + \epsilon')$;
     12. $\theta_i = \ell'_x / \ell B$, where $\ell'_x$ is defined in Eq. (4);
    13. while $|\mathcal{R}| < \theta_i$ do
       14. Generate an RR set for a randomly selected node $v$ of $G$ and insert in $\mathcal{R}$;
       15. $s = s + 1$; budgetSwitch = true;
     16. else $i = i + 1$; budgetSwitch = false;
    17. if $s \leq |\vec{b}|$ then $\theta_k = \lambda'_k / \ell B$;
    18. $\mathcal{R} = \emptyset$.
    19. while $|\mathcal{R}| < \theta_k$ do
      20. Generate an RR set for a randomly selected node $v$ of $G$ and insert in $\mathcal{R}$;
      21. $s = s + 1$; budgetSwitch = true;
    22. $S_k = \text{NodeSelection} (\mathcal{R}, \vec{b})$;
    23. return $S_k$ as the final seed set.;

---

6 EXPERIMENTS

6.1 Experiment Setup

We perform extensive experiments on five real social networks. We first experiment with synthetic utility (value and price) functions. For real utility functions, we learn the value and noise distributions of items from the bidding data in eBay, and obtain item prices from Craigslist and Facebook groups to make them compatible with used items auctioned in eBay. All experiments are performed on a Linux machine with Intel Xeon 2.6 GHz CPU and 128 GB RAM.

6.1.1 Networks. Table 1 summarizes the networks and their characteristics. Flixster is mined in [31] from a social movie site and a strongly connected component is extracted. Douban is a Chinese social network, where users rate books, movies, music, etc. In [31] all movie and book ratings of the users in the graph are crawled separately to derive two datasets from book and movie ratings: Douban-Book and Douban-Movie. Twitter and Orkut are two of the largest public network datasets that can be obtained from [1].

<table>
<thead>
<tr>
<th>Network</th>
<th># nodes</th>
<th># edges</th>
<th>avg. degree</th>
<th>type</th>
<th>avg. activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flixter</td>
<td>7.7K</td>
<td>23.3K</td>
<td>6.5</td>
<td>undirected</td>
<td>7.9</td>
</tr>
<tr>
<td>Douban-Book</td>
<td>141K</td>
<td>274K</td>
<td>7.9</td>
<td>directed</td>
<td>70.5</td>
</tr>
<tr>
<td>Douban-Movie</td>
<td>34.7K</td>
<td>41.7M</td>
<td>7.9</td>
<td>directed</td>
<td>77.5</td>
</tr>
<tr>
<td>Twitter</td>
<td>1.47G</td>
<td>234M</td>
<td>7.9</td>
<td>directed</td>
<td>77.5</td>
</tr>
<tr>
<td>Orkut</td>
<td>3.07M</td>
<td>234M</td>
<td>7.9</td>
<td>directed</td>
<td>77.5</td>
</tr>
</tbody>
</table>

Table 1: Network Statistics

The basic idea of PRIMA is to generate enough RR sets such that for any budget $k \in \vec{b}$, $|\mathcal{R}| \geq \lambda'_k / \text{OPT}_k$, with probability at least $1 - 1/n^\ell$. Since $\text{OPT}_k$ is unknown, we rely on a good lower bound $LB_k$ of $\text{OPT}_k$. PRIMA iterates through all budgets in $\vec{b}$ to find the $|\mathcal{R}| = \max_{k \in \vec{b}} \lambda'_k / LB_k$. In that process PRIMA efficiently reuses the RR-sets and the prefix of the already found seed set, avoiding redundant calls to the NodeSelection procedure.

Lastly, after determining $|\mathcal{R}|$, those many RR sets are generated from scratch (line 23) on which the final NodeSelection is invoked. This addresses a recently found issue of the original IMM algorithm [13]. PRIMA then returns the top-$\vec{b}$ seeds obtained from NodeSelection (line 25).

The correctness and the running time of the PRIMA algorithm mainly follow the proof of the IMM algorithm [13, 39]. Intuitively, there are two changes in PRIMA’s running time. The budget $k$ of a single item of IMM is replaced with $\vec{b}$, the maximum budget of any item. Secondly, by applying union bound on every individual item’s failure probability, a factor of $\log n |\vec{b}|$ is added to the sample complexity. Our main Theorem 2 follows from the correctness of Algorithms bundleGRD and PRIMA.

---

6.2 Algorithms compared. We compare bundleGRD against six baselines – RR-SIM, RR-CIM, item-disj, bundle-disj, BDHS-Step and BDHS-Convex. RR-SIM$^+$ and RR-CIM are two state-of-the-art algorithms designed for
Figure 4: Expected social welfare in four configurations (on the Douban-Movie network)

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

Figure 5: Running times of bundleGRD, RR-SIM⁺, RR-CIM, item-disj and bundle-disj (on Configuration 1)

(a) Flixster  (b) Douban-Book  (c) Douban-Movie  (d) Twitter

Figure 6: Expected social welfare in four configurations (on the Twitter network)

(a) Configuration 5  (b) Configuration 6  (c) Configuration 7  (d) Configuration 8

Figure 7: (a) Impact of number of items on the running time and (b-d) Experiments using real Param (on the Twitter network)

(a) Effect of number of items  (b) Welfare  (c) Running time  (d) Budget skew

Figure 8: (a-c) Comparison against BDHS algorithms and (d) Scalability of bundleGRD

complementary products in the context of IM [31]. However, they work only for two items. Extending the Com-IC framework and the RR-SIM⁺ and RR-CIM algorithms for more than two items is highly non-trivial as that requires dealing with automata with exponentially many states. Hence in comparing the performance of bundleGRD against RR-SIM⁺
and RR-CIM, we limit the number of items to two. Later we experiment with more than two items. 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.

### 6.1.2.1 Com-IC baselines.
For two items $i_1$ and $i_2$, given seed set of item $i_2$ (resp. $i_1$), RR-SIM$^+$ (resp. RR-CIM) finds seed set of item $i_1$ (resp. $i_2$) such that expected number of adoptions of $i_1$ is maximized. Initial seeds of $i_2$ (resp. $i_1$) are chosen using IMM [39].

### 6.1.2.2 Item-disjoint.
Our next baseline item-disj allocates only one item to every seed node. Given the set of items $I$, item-disj finds $\sum_i b_i$ nodes, say $L$, using IMM [39], where $b_i$ is the budget of item $i$. Then it visits items in $L$ in non-increasing order of budgets, assigns item $i$ to first $b_i$ nodes and removes those $b_i$ nodes from $L$. By explicitly assigning every item to different seeds, item-disj does not leverage the effect of supermodularity. However it benefits from the network propagation: since the utilities are supermodular, if more neighbors of a node adopt some item, it is more likely that the node will also adopt an item. Thus, when individual items have positive utility and hence can be adopted and propagate on their own, by choosing more seeds, item-disj makes use of the network propagation to encourage more adoptions.

### 6.1.2.3 Bundle-disjoint.
Baseline bundle-disj, aims to leverage both supermodularity and network propagation. It first orders the items $I$ in non-increasing budget order and determines successively minimum sized subsets with non-negative deterministic utility, maintaining these subsets ("bundles") in a list. Items in each bundle $B$ are allocated to a new set of $b_B := \min(b_i \mid i \in B)$ seed nodes. The budget of each item in $B$ is decremented by $b_B$, and items with budget 0 are removed. When no more bundles can be found, we revisit each item $i$ with a positive unused budget and repeatedly allocate it to the seeds of the first existing bundle $B$ which does not contain $i$. If $b_B > b_i$ (where $b_i$ is the current budget of $i$ after all deductions), then the first $b_i$ nodes from the seed set of $B$ are assigned to $i$. If an item $i$ still has a surplus budget, we select $b_i$ fresh seeds using IMM and assign them to $i$.

### 6.1.2.4 Welfare maximization baselines.
Our last two baselines, BDHS-Concave and BDHS-Step are two state-of-the-art welfare maximization algorithms under network externalities [5]. As discussed in § 2, their study has significant differences from our study, but we still make an empirical comparison with their algorithms with the goal to explore what fraction of the budget is needed by our model with network propagation to achieve the same social welfare as their model which has network externality but no network propagation. We defer the details of the comparison to § 6.4.

### 6.1.3 Default Parameters.
Following previous works [21, 34] we set probability of edge $e = (u, v)$ to $1/d_{in}(v)$. Unless otherwise specified, we use $\epsilon = 0.5$ and $\ell = 1$ as our default for all five methods as recommended in [31, 39]. The Com-IC algorithms RR-SIM$^+$ and RR-CIM use adoption probabilities, called GAP parameters [31], to model the interaction between items. The GAP parameters can be simulated within the UIC framework using utilities shown in Eq. (5). The derivation follows simple algebra. Here, $q_{i_j\mid i} (\text{resp.}, q_{i_j\mid i'})$ denotes the probability that a user adopts item $i_j$ given that it has adopted nothing (resp., item $i_j$).

$$q_{i_j\mid i} = \Pr[N(i_1) \geq \mathcal{P}(i_1) - \mathcal{V}(i_1)],$$

$$q_{i_j\mid i} = \Pr[N(i_1) \geq \mathcal{P}(i_1) - (\mathcal{V}(\{i_j, i_2\}) - \mathcal{V}(i_2))],$$

$$q_{i_j\mid i} = \Pr[N(i_1) \geq \mathcal{P}(i_1) - (\mathcal{V}(\{i_j, i_2\}) - \mathcal{V}(i_1))].$$

### 6.2 Experiments on two items
We explore four different configurations corresponding to the choice of the values, prices, noise distribution parameters, and item budgets (see Table 2). While UIC does not assume any specific distribution for noise, in our experiments we use a Gaussian distribution for illustration.

In Configurations 1 and 2, individual items have non-negative deterministic utility. In this setting item-disj and bundle-disj are equivalent. In Configurations 3 and 4 one item has a negative deterministic utility while the other item has a non-negative one. In this setting, however, bundleGRD and bundle-disj are equivalent. One may also consider configurations where every individual item has negative deterministic utility. In such a setting, item-disj produces 0 welfare, which makes the comparison degenerates.

For every parameter setting, we consider two budget settings, uniform (e.g., Configuration 1) and non-uniform (resp. Configuration 2). In case of uniform budget, both items have the same budget $k$, where $k$ is varied from 10 to 50 in steps of 10. For non-uniform budget, $i_1$’s budget is fixed at 70, and $i_2$’s budget is varied from 30 to 110 in steps of 20.

#### 6.2.1 Social Welfare.
We compare the expected social welfare achieved by all algorithms on all four configurations (Fig. 4). We show the results only for Douban-Movie, since the trend of the results is similar on other networks. In terms of social welfare, bundleGRD achieves an expected social welfare up to 5 times higher than item-disj (Fig. 4d). A similar remark applies when bundle-disj and bundleGRD are not equivalent (e.g., Fig. 4b). Further, notice that RR-SIM$^+$ and RR-CIM produce welfare similar to bundleGRD. It follows from Table 4 of [31] (full arxiv version) that under this configuration, RR-SIM$^+$ and RR-CIM end up copying the seeds of the other item. Hence their allocations are similar to bundleGRD. However, as shown next, bundleGRD is much more efficient than the other two algorithms, and easily supports more than two items, which makes bundleGRD more suitable in practice for multiple items over large networks.
6.2.2 Running time. We study the running time of all algorithms using Configuration 1 as a representative case. The results are shown in Fig. 5. As can be seen, bundleGRD and bundle-disj are equivalent and hence have the same running time. However, bundleGRD significantly outperforms all other baselines on every dataset. RR-SIM* and RR-CIM are particularly slow. In fact, on the large Twitter network, they could not finish even after our timeout after 6 hours (hence they are omitted from Fig. 5(d)). In comparison with the baselines, bundleGRD is up to 5 orders of magnitude (resp. 1.5 times) faster than RR-CIM (resp. item-disj). Running times on other configurations show a similar trend, and are omitted.

6.3 More than two items

We use the largest dataset Twitter for tests in this subsection.

6.3.1 The configurations. Having established the superiority of bundleGRD for two items, we now consider more than two items. Recall that RR-SIM* and RR-CIM cannot work with more than two items, so we confine our comparison to item-disj and bundle-disj. We gauge the performance of the algorithms on social welfare and running time. We also study the effect of budget distribution on social welfare. We design four configurations corresponding to the choice of budget and utility (see Table 3). For all configurations, we sample noise terms from $N(0, 1)$. Price and value are set in such a way as to achieve certain shapes for the set of itemsets in the lattice that have a positive utility (see below).

6.3.1.1 Configurations 5-7. Configuration 5 is the simplest: every item has the same budget; price and value are set such that every item has the same utility of 1 and utility is additive. Thus, by design, this configuration gives minimal advantage to any algorithm that tries to leverage supermodularity. The next two configurations (6 and 7) model the situation where a single "core" item is necessary in order to make an itemset’s utility positive. E.g., a smartphone may be a core item, without which its accessories do not have a positive utility. We set the core item’s utility to 5. The addition of any other item increases the utility by 2. Thus, all supersets of the core item have a positive utility, while all other subsets have a negative utility. Hence, the set of subsets with positive utility forms a “cone” in the itemset lattice. In Configuration 6 (resp. 7), the core item is the item with maximum (resp. minimum) budget. Finally, we design a more general configuration where the set of itemsets with positive utility forms an arbitrary shape (see Configuration 8 below).

6.3.2.1 Configuration 8. We consider the itemset lattice, with level $t$ having subsets of size $t$. We randomly set the prices and values of items in level 1 such that a random subset of items have a non-negative utility. Let $A_t$ be any itemset at level $t > 1$ and $i \in A_t$ any item. We choose a value uniformly at random, $\epsilon \sim U[1, 5]$, and define

\[ V(i|A_t \setminus \{i\}) = \max_{\emptyset \neq B \subseteq A_t \setminus \{i\}} V(B|B) + \epsilon \]  

where $V(A, q)$ denotes the set of subsets of $A$ of size $q$. That is, the marginal gain of an item $i$ w.r.t. $A_t \setminus \{i\}$ is set to be the maximum marginal gain of $i$ w.r.t. subsets of $A_t$ of size $t - 2$, plus a randomly chosen boost ($\epsilon$). E.g., let $A_t = \{i, j, k, l\}, t = 4$ then $V(i|\{j, k, l\}) = \max \{V(i|\{j, k\}), V(i|\{k, l\}), V(i|\{j, l\})\} + \epsilon$.

Recall that the value computation proceeds level-wise starting from level $t = 0$. Thus, for any itemset $A_t$ in Eq.(6), $V(i|B)$ for subsets $B$ is already defined.

Finally, we set $V(A_t) = \max_{s \in A_t} \{V(A_t \setminus \{s\}) + V(i|A_t \setminus \{i\})\}$. Notice that this way of assigning values ensures that the value function is well-defined and supermodular.

6.3.2 Social welfare. First, we study the social welfare achieved by the algorithms, in each of the above configurations, with the total budget varying from 500 to 1000 in steps of 100. For Configurations 7 and 10, we set the budget uniformly for every item. For other configurations, the max and min budget is set to 20% and 2% of the total budget. Remaining budget is split uniformly. The results of the experiment on Twitter network are shown in Fig. 6. Under Configurations 8 and 9, bundleGRD and bundle-disj produces the same allocation, hence the welfare is the same. However in general bundleGRD outperforms every baseline in all the four configurations by producing welfare up to 4 times higher than baselines.

6.3.3 Running time vs number of items. Next, we study the effect of the number of items on the running time of the algorithms. For this experiment, we use Configuration 5. We set the budget of every item to $k = 50$ and vary the number of items $s$, from 1 to 10. Fig. 7(a) shows the running times on the Twitter dataset. As the number of items increases the number of seed nodes to be selected for item-disj and bundle-disj increases. Notice both item-disj and bundle-disj select the same number of seeds, which is $k \times s$. item-disj selects it by one invocation of IMM, with budget $ks$, while bundle-disj invokes IMM $s$ times with budget $k$ for every invocation. So their overall running times differ. By contrast, the running time of bundleGRD only depends on the maximum budget and is independent of the number of items. E.g.,

<table>
<thead>
<tr>
<th>No</th>
<th>Price</th>
<th>Value</th>
<th>Noise</th>
<th>GAP Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$i_1 = 3$</td>
<td>$i_2 = 4$</td>
<td>$i_1 + i_2 = 8$</td>
<td>$i_1 \in N(0, 1)$</td>
</tr>
<tr>
<td>2</td>
<td>$i_1 = 4$</td>
<td>$i_2 = 8$</td>
<td>$i_1 + i_2 = 12$</td>
<td>$i_1 \in N(0, 1)$</td>
</tr>
<tr>
<td>3</td>
<td>${i_1, i_2} = 7$</td>
<td>${i_1, i_2} = 15$</td>
<td>${i_1, i_2} \in N(0, 2)$</td>
<td>$q_{i_1} = 0.5$, $q_{i_2} = 0.16$</td>
</tr>
<tr>
<td>4</td>
<td>${i_1, i_2} = 8$</td>
<td>${i_1, i_2} = 16$</td>
<td>${i_1, i_2} \in N(0, 2)$</td>
<td>$q_{i_1} = 0.98$, $q_{i_2} = 0.84$</td>
</tr>
</tbody>
</table>

Table 2: Two item configurations

<table>
<thead>
<tr>
<th>No</th>
<th>Value</th>
<th>Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Additive</td>
<td>Uniform</td>
</tr>
<tr>
<td>6</td>
<td>Cone-max</td>
<td>Non-uniform</td>
</tr>
<tr>
<td>7</td>
<td>Cone-min</td>
<td>Non-uniform</td>
</tr>
<tr>
<td>8</td>
<td>Level-wise</td>
<td>Uniform</td>
</tr>
</tbody>
</table>

Table 3: Multiple item configurations
when number of items is 10, bundleGRD is about 8 times faster than bundle-disj and 2.5 times faster than item-disj.

6.4 Experiment with real value, price, and noise parameters

In this section, we conduct experiments on parameters (value, price, and noise) learned from real data. We consider the following 5 items: (1) Playstation 4, 500 GB console, denoted \( ps \), (2) Controller of the Playstation, denoted \( c \), and (3-5) Three different games compatible with \( ps \), denoted \( g_1 \), \( g_2 \) and \( g_3 \) respectively. We next describe the method by which we learn their parameters from real data.

### 6.4.1 Learning the value, price, and noise

Jiang et al. [22] showed that learning user’s valuations of items improves the prediction accuracy of bids in auctions. Given the bidding history of an item, their method learns a value distribution of the item, by taking into account hidden/unobserved bids. We use it to learn the values of itemsets from bidding histories. In our model value is not random, instead noise models the randomness in valuations. Hence we take the mean of the learned distribution to be the value and the noise is set to have 0 mean and the same variance as the learnt distribution. While UIC does not assume specific noise distributions, for concreteness, we fit a Gaussian distribution to noise by taking 10,000 independent random samples.

### Table 4: Learned parameters

We mine the bidding histories of itemsets from eBay. To match the used products bidden in eBay, we use prices for the used products in Craigslist and Facebook groups.

The price obtained is C$260 for \( ps \), C$20 for \( c \), and C$5 each for \( g_1 \), \( g_2 \) and \( g_3 \). For some of the itemsets, we show the learned parameters and the links to the corresponding eBay bidding histories used in the learning, in Table 4. The rest of the itemsets are omitted from the table for brevity. We describe the parameters of those omitted itemsets here. Firstly, any of \( c, g_1, g_2, g_3 \), without the core item \( ps \), is useless. Hence values of those items are set to 0. Secondly, we did not find any bidding record for an itemset consisting of \( ps, c \) and a single game. This is perhaps because typically owners of \( ps \) own multiple games and while selling they sell all the games together with \( ps \). Hence, we consider the itemset with \( ps, c \) and a single game to have negative deterministic utility. However, as the table shows, itemsets with \( ps, c \) and two games have non-negative deterministic utility. Finding the bidding history for the exact same games is difficult, so since games \( g_1, g_3 \) are priced similarly and valued similarly by users, we assume that any itemset with \( ps, c \) and any two games has the same utility as that shown in the fourth row of Table 4. From the value column, we can see that the items indeed follow supermodular valuation, confirming that in practice complementarity arises naturally. Lastly, the only itemsets that have positive deterministic utility are itemsets with \( ps, c \) and at least two games. All other itemsets including the singleton items, have negative deterministic utility. Consequently, we know that the allocation produced by item-disj will have 0 expected social welfare, so we omit item-disj from our experiments, discussed next.

### 6.4.2 Effect of total budget size

We compare bundleGRD with bundle-disj on the Twitter dataset with different sizes of total budgets. Given a total budget, we assign 30%, 30%, 20%, 10%, 10% of that to \( ps, c, g_1, g_2, g_3 \) respectively. Then we vary the total budget from 100 to 500 in steps of 100. Fig. 7(b) shows the welfare: as can be seen, bundleGRD outperforms bundle-disj in both high and low budgets. In fact with higher budget, bundleGRD produces welfare more than 2 times that of bundle-disj. Next we report the running time of the two algorithms in Fig. 7(c). Since bundle-disj makes multiple calls to IMM, its running time is 1.5 times higher than bundleGRD.

### 6.4.3 Effect of different item budget given the same total budget

Our next experiment studies the following question. Suppose we have a fixed total budget which we must be divided up among various items. How would the social welfare and running time vary for different splits? Since we have seen that in terms of social welfare bundleGRD dominates all baselines, we use it to measure the welfare. Given a total budget of 500, we split it across 5 items following three different budget distributions, namely (i) Uniform: each item has the same budget 100, (ii) Large skew: one item, \( ps \) has 82% of the total budget and the remaining 18% is divided evenly among the remaining 4 items; and (iii) Moderate skew: Budgets of the 5 items are, \( [ps = 150, c = 150, g_1 = 100, g_2 = 50, g_3 = 50] \). Fig. 7(d) shows the expected social welfare and the running time of bundleGRD under the three budget distributions on the Twitter dataset. The welfare is the highest under uniform and worst under large skew, with moderate skew in between. Running time shows consistent trend, with uniform being the fastest and large skew being the slowest. The findings are consistent with the observation that with large skew, the number of seeds to be selected increases and the allocation cannot take full advantage of supermodularity.

### 6.4.4 Effect of propagation vs. network externality

We next compare our bundleGRD against the other two baselines, BDHS-Concave and BDHS-Step (referred to as BDHS algorithms for simplicity). BDHS-Concave and BDHS-Step correspond to the concave and step externality algorithms respectively (i.e. Alg 1 and 3 of [5]). Our overall approach is, despite the differences between our model and BDHS model
as highlighted in §2, we try to convert our model in a reasonable way to their model by means of restriction, and use their algorithms to find the total social welfare that they can achieve. Then we gradually increase the budget of items in our model to see at which budget the social welfare achieved by our solution reaches the social welfare achieved by their solution that has no budget and assigns items to every node directly. This would demonstrate the budget savings due to our consideration of network propagation.

We now describe how we convert our model to their model. First, our model uses network propagation with the UIC model while their model uses network externality without propagation. To align the two models, we try two alternatives. The first alternative is to sample 10,000 live-edge graphs, and the propagation on one live-edge graph bears similarity with the 1-step function, and thus we use 1-step externality function on each live-edge graph to compute the total social welfare and then average over all live-edge graphs. We refer to this alternative BDHS-Step. The second alternative works when we restrict our UIC model such that every edge has the same probability p. In this case, the activation probability of a node v is 1 − (1 − p)^k, where k is the number of active neighbors of v which is at most the size s of its 2-neighborhood support set. This resembles the concave function case in the BDHS model, and thus we use the concave function 1 − (1 − p)^s in their 2-hop model. We refer to this alternative BDHS-Concave.

Second, to align their unit demand model with our model, we treat each item subset as a virtual item in their model, so that they can assign item subsets to all nodes. Finally, their model has no budget, so they are free to assign all item subsets to all nodes. We use this as a benchmark of the total social welfare they can achieve, and see at what fraction of the budget we can achieve the same social welfare due to the network propagation effect.

We use the Orkut as one of the large networks in this study, which also enables the study of the performance of bundleGRD on a large network other than Twitter (which is already used in Figure 5(d), 6, and 7). Fig. 8(a-c) shows the results on Orkut, Douban-Book and Douban-Movie networks respectively. The x axis shows the fraction of the budget needed by bundleGRD, where 100% corresponds to a budget of n, i.e., #nodes in the network, which corresponds to the setting of [5]. As can be seen, for dense networks like Orkut, bundleGRD needs less than 35% as the budget. We found a similar result on Flixster, not included here for the lack of space. For a sparse graph like Douban-Book it needs 82%, which is still less than the budget of BDHS. Further, since propagation has a submodular growth, much of the budget is used to increase the latter half of the welfare. E.g., even on Douban-Book, 75% of BDHS’ welfare is obtained by only using 50% budget. This test clearly demonstrates that our bundleGRD could leverage the power of propagation, compared to the BDHS approach that only considers externality.

6.4.5 Scalability test. Our next experiment shows the impact of network size on bundleGRD using Orkut with two types of edge probabilities: (1) 1/di(n(v)) and (2) fixed 0.01. We use a uniform budget of 50 for all items. We then use breadth-first-search to progressively increase the network size such that it includes a certain percentage of the total nodes. The results are shown in Fig. 8(d). With increasing network size, the running time in both cases roughly has a linear increase, whereas the welfare depicts a sublinear growth. It is worth noticing that even for the entire million-sized network and fixed probability, bundleGRD requires mere 129 (time 2) seconds to complete, which again attests to its scalability.

6.4.6 Memory usage. Our experiment shows that bundleGRD has a similar memory requirement as IMM. It generates the same number of RR sets as IMM. We omit the details for the lack of space.

7 SUMMARY & DISCUSSION

We propose a novel model combining influence diffusion with utility-driven item adoption, which supports any mix of competing and complementary items. Focusing on complementary items, we study the problem of optimizing expected social welfare. Our objective function is monotone, but neither submodular nor supermodular. Yet, we show that a simple greedy allocation guarantees a (1 − 1/e − ε)-approximation to the optimum. Based on this, we develop a scalable approximation algorithm bundleGRD, which satisfies an interesting prefix preserving property. With extensive experiments, we show that our algorithm outperforms the state of the art baselines.

Our results and techniques carry over unchanged to any triggering propagation model [25]. We assumed that price is additive and valuations are supermodular. If we use submodular prices, that would further favor item bundling. In this case, utility remains supermodular and our results remain intact. Independently of this, we could study competition using submodular value functions. Orthogonally, we can study the UIC model under personalized noise terms.

Acknowledgments. The research of the first and last author was supported in part by a Discovery grant and a Discovery Accelerator Supplement grant from the Natural Sciences and Engineering Research Council of Canada (NSERC). The second author was partially supported by the National Natural Science Foundation of China (Grant No. 61433014). We would like to thank Wei Lu and Ritika Jain for stimulating discussions during the early stages of this research.
REFERENCES


A PROOFS AND ADDITIONAL THEORETICAL ANALYSIS

We restate the theorems and lemmas with the original numbering for convenience.

A.1 Proofs for Properties of UIC (Section 4)

Lemma 1. (Local maximum). Let W be a possible world and A, B ⊆ I be any itemsets such that A and B are local maximum with respect to U_W. Then (A∪B) is also a local maximum with respect to U_W, i.e., U_W(A∪B) = max_{C⊆A∪B} U_W(C).

Proof. For any subset C ⊆ A∪B, we have

U_W(C) = U_W(C \ B | B \ C) + U_W(B \ C)
\leq U_W(C \ B | B) + U_W(B)
= U_W(C \ B) = U_W(B \ C \ B) + U_W(C \ B)
\leq U_W(B \ A) + U_W(A) = U_W(A \ B).

Lemma 2. For any node u and any time t, the itemset adopted by u at time t, A_S^u(u, t), must be a local maximum.

Proof. We prove by an induction on t. The base case of t = 1 is true because by the model, node u adopts the local maximum among all subsets of items allocated to it. For the induction step, suppose for a contradiction that A_S^u(u, t) is not a local maximum but A_S^u(u, t−1) is a local maximum. Then there must exist a C ⊆ A_S^u(u, t) that is a local maximum and U_W(C) > U_W(A_S^u(u, t−1)). By Lemma 1, C ∪ A_S^u(u, t−1) is also a local maximum, and thus C ∪ A_S^u(u, t−1) cannot be A_S^u(u, t). Since U_W(C ∪ A_S^u(u, t−1)) ≥ U_W(C) > U_W(A_S^u(u, t−1)), u should adopt C ∪ A_S^u(u, t−1) instead of A_S^u(u, t), a contradiction. □
Lemma 3. (Reachability). For any item \( i \) and any possible world \( W \), if a node \( u \) adopts \( i \) under allocation \( S \), then all nodes that are reachable from \( u \) in the world \( W \) also adopt \( i \).

Proof. Consider a possible world \( W \) and a node \( u \) that adopts item \( i \). Consider any node \( v \) reachable from \( u \) in \( W \) that does not adopt \( i \). Let \( (u, v_1, \ldots, v_r, v) \) be a path in \( W^E \). Assume w.l.o.g. that \( v \) is the first node on the path that does not adopt \( i \). \( A^S_W(v, t) \) and \( A^S_W(v, t + 1) \) respectively are the itemsets adopted by \( v \) at time \( t \) and by \( v \) at time \( t + 1 \). If \( J \in A^S_W(v, t) \cup A^S_W(v, t + 1) \) for some \( J \subseteq S \), then by Lemma 1, \( J \) is also a local maximum, hence \( u \) will adopt \( i \). Also, \( |J| \geq |A^S_W(v, t + 1)| \), as \( A^S_W(v, t + 1) \subset J \). Thus expected individual utility of \( u \) at time \( t \) should adopt the larger cardinality set \( J \). Hence \( i \) is adopted by \( v \).

Theorem 1. Expected social welfare is monotone with respect to the sets of node-item allocation pairs. However it is neither submodular nor supermodular.

Proof. To prove monotonicity, we show by induction on propagation time that the social welfare in any world \( W \) is monotone. The result follows upon taking expectation. Consider allocations \( S \leq S' \) and any node \( v \).

Base Case: At \( t = 1 \), desire happens by seeding. By assumption, \( v \in S \) and \( v \in S' \). Thus, \( A^S(v, t) \subseteq A^{S'}(v, t) \), and \( A^S(v, t + 1) \neq \emptyset \). From the semantics of adoption of itemsets, we have \( U_W(J, A^S(v, t + 1)) \geq 0 \). Now, \( A^{S'}(v, t + 1) \subseteq A^S(v, t + 1) \). By supermodularity of utility, \( U_W(J | A^{S'}(v, t + 1)) \geq 0 \). Since \( J \subseteq A^{S'}(v, t + 1) \subseteq A^S(v, t + 1) \), by the semantics of itemset adoption, the set \( J \cup A^S(v, t + 1) \) will be adopted by \( v \) at time \( t \). By assumption, \( \sum_{v \in V} U_W(A^S(v)) = \rho_W(S) \). This shows that the social welfare in any possible world is monotone.

For submodularity and supermodularity, we give counterexamples. Consider a network with single node \( u \) and two items \( i_1 \) and \( i_2 \). Let \( \mathcal{P}(i_1) > \mathcal{V}(i_1) \) and \( \mathcal{P}(i_2) < \mathcal{V}(i_2) \). However \( \mathcal{V}(\{i_1, i_2\}) > \mathcal{P}(i_1) + \mathcal{P}(i_2) \). Assume that noise terms are bounded random variables, i.e., \( |\mathcal{N}(i_j)| \leq |\mathcal{V}(i_j) - \mathcal{P}(i_j)| \), \( j = 1, 2 \). Thus expected individual utility of \( i_1 \) or \( i_2 \) is negative, but when they are offered together, the expected utility is positive. Now consider two seed allocations \( S = \emptyset \) and \( S' = \{(u, i_1, i_2)\} \). Let the additional allocation pair be \( (u, i_2) \). Now \( \rho(S \cup \{(u, i_2)\}) - \rho(S) = 0 - 0 > 0 \); for \( S \), no items are adopted and for \( S \cup \{(u, i_2)\} \) the noise \( \mathcal{N}(i_2) \) cannot affect adoption decision in any possible world, so \( i_2 \) will be adopted by \( u \) in any world. However, \( \rho(S' \cup \{(u, i_2)\}) - \rho(S') > 0 \), as under allocation \( S' \), \( i_1 \) is not adopted by \( u \) in any world, while under allocation \( S' \cup \{(u, i_2)\} \), \( u \) will adopt \( \{i_1, i_2\} \) in any possible world, resulting in positive social welfare and breaking submodularity.

For supermodularity, consider a network consisting of two nodes \( v_1 \) and \( v_2 \) with a single directed edge from \( v_1 \) to \( v_2 \), with probability \( 1 \). Let there be one item \( i \) whose deterministic utility is positive, i.e., \( \mathcal{V}(i) > \mathcal{P}(i) \). Assume that the noise term \( \mathcal{N}(i) \) is a bounded random variable, i.e., \( |\mathcal{N}(i)| \leq |\mathcal{V}(i) - \mathcal{P}(i)| \). Now consider two seed allocations \( S = \emptyset \) and \( S' = \{(v_1, i_1, i_2)\} \). Let the additional pair be \( (v_2, i_2) \). Under allocation \( S' \), both nodes \( v_1 \) and \( v_2 \) will adopt \( i \) in any possible world. Hence adding the additional pair \( (v_2, i_2) \) does not change item adoption in any world and consequently the expected social welfare is unchanged. Thus we have, \( \rho(S \cup \{(v_2, i_2)\}) - \rho(S) = \mathbb{E}[\mathcal{U}(i_2)] > 0 \), which breaks supermodularity.

A.2 Proofs for Block Accounting (Section 5.2)

Lemma 7. For any arbitrary seed allocation \( S \), the expected social welfare in \( W^N \) is \( \rho_{W^N}(S) \leq \sum_{i} |\mathcal{S}_a_i| \cdot \Delta_i \), where \( \mathcal{S}_a_i \) is the seed set assigned to the anchor item \( a_i \) of block \( B_i \), and \( \Delta_i \) is as defined in Eq. (2).

Proof. For an edge possible world \( W^E \), suppose that after the diffusion process under \( W^E \), every node \( v \) adopts item set
where $\epsilon > 0$ and $\ell > 0$, the expected social welfare $\rho(S^{Grd}) \geq (1 - \frac{1}{\ell} - \epsilon) \cdot \rho(S)$ with at least $1 - \frac{1}{|\Gamma'|}$ probability.

Proof. From Lemma 5, we have for a possible world $W^N = (W^E, W^N)$, $\rho_W(S^{Grd}) = \sum_{i \in [\ell]} \sigma(S^{Grd}_B_i) \cdot \Delta_i$, where the size of $S^{Grd}$ is the effective budget of $B_i$.

For an arbitrary allocation $S$, since $a_i$ is the anchor item of $B_i$, by its definition we know that $|S_{a_i}| = |S^{Grd}_B_i|$. By the correctness of the prefix-preserve influence maximization algorithm we use in line 2 (Definition 1, to be instantiated in §5.3), we have that with probability at least $1 - \frac{1}{|\Gamma'|}$, $\sigma(S^{Grd}_B_i) \geq (1 - \frac{1}{\ell} - \epsilon)\sigma(S_{a_i})$, for all blocks $B_i$’s and their corresponding anchors $a_i$’s.

Let $\rho(W^N)$ be $D^N$. Then, together with Lemma 7, we have that with probability at least $1 - \frac{1}{|\Gamma'|}$,

$$\rho(S^{Grd}) \geq \rho(W^N(S^{Grd})) = \frac{\mathbb{E}_{W^N} \left[ \sum_{i \in [\ell]} \sigma(S^{Grd}_B_i) \cdot \Delta_i \right]}{\mathbb{E}_{W^N} \left[ \sum_{i \in [\ell]} \Delta_i \right]} \geq \frac{\mathbb{E}_{W^N} \left[ \sum_{i \in [\ell]} (1 - \frac{1}{\ell} - \epsilon)\sigma(S_{a_i}) \cdot \Delta_i \right]}{\mathbb{E}_{W^N} \left[ \sum_{i \in [\ell]} \Delta_i \right]} \geq (1 - \frac{1}{\ell} - \epsilon)\rho(W^N(S))$$

Therefore, the theorem holds. \(\square\)

A.3 Proofs for Item-wise prefix preserving IMM (Section 5.3)

Lemma 8. Let $\mathcal{R}$ be the final set of RR sets generated by PRIMA at the end and let $k \in \mathcal{B}$ be any budget. Then $|\mathcal{R}| \geq \lambda^*_k / OPT_k$ holds with probability at least $1 - 1/n^{e'}$.

Proof. Given $x \in [1, n]$, $e'$ and $\delta_3 \in (0, 1)$ and a budget $k$. Let $\mathcal{S}_k$ be the seed set of size $k$ obtained by invoking NodeSelection($\mathcal{R}, k$), where,

$$|\mathcal{R}| \geq \frac{2}{e'} \left(\log \left(\frac{2}{\delta_3}\right) + \log(1/\delta_3)\right) \cdot \frac{n}{x^{e'}}$$

Then, from Lemma 6 of [39], if $OPT_k < x$, then $n \cdot F_k(\mathcal{S}_k) < (1 + e') \cdot x$ with probability at least $(1 - \delta_3)$. Now let $\ell = \log_2 \frac{n}{OPT_k}$. By union bound, we can infer that PRIMA has probability at most $(j - 1)/(n^{e'} \cdot \log_2 n)$ to satisfy the coverage condition of line 12 for the budget $k$. Then by Lemma 7 of [39] and the union bound, PRIMA will satisfy $LB_k \leq OPT_k$ with probability at least $1 - n^{e'}$. We know that for any $k \in \mathcal{B}$, $|\mathcal{R}| \geq \lambda^*_k / LB_k$, hence the lemma follows. \(\square\)

We are now ready to prove the correctness of PRIMA.

Lemma 9. PRIMA returns a prefix preserving $(1 - 1/e - \epsilon)$-approximate solution $\mathcal{S}_\mathcal{R}$ to the optimal expected spread, with probability at least $1 - 1/n^{e'}$. 
Proof. We know from Lemma 8 that the RR set sampling for any budget can result in the coverage condition (Algorithm 2, line 12) failing with probability at most \(1/n^{\ell'}\). By applying union bound over all the budgets, we have that the failure probability of the coverage condition in PRIMA is at most \(\sum_{b \in R} 1/n^{\ell'} = |\tilde{b}| \cdot 1/n^{\ell'}\). By setting \(\ell' = \log_n (n \cdot |\tilde{b}|)\), we bound this failure probability to at most \(1/n^{\ell'}\). Thus \(\ell'\) is used for computing \(a\) and \(\beta_k\) in Eq. (4). Further once \(\theta_k\) is determined, we generate those many RR set from scratch.

This follows the fix proposed in [13] for a bug in Theorem 1 of [39]. Without the fix, the top \(S_\ell\) nodes returned by the last call to NodeSelection (line 25), cannot be shown to have a \((1 - 1/e - \epsilon)\)-approximate solution with probability at least \(1 - 1/n^{\ell'}\). For every budget \(b_i \in \tilde{b}\), we can then choose the prefix of top-\(b_i\) nodes of \(S_\ell\) and use that as a solution \(S_{b_i}\) for that budget, with the guarantee that with probability at least \(1 - 1/n^{\ell'}\), each \(S_{b_i}\) is a \((1 - 1/e - \epsilon)\)-approximate solution to \(OPT_{b_i}\). By union bound, PRIMA returns a \((1 - 1/e - \epsilon)\)-approximate prefix preserving solution with probability at least \(1 - 2/n^{\ell'}\).

Finally by increasing \(\ell\) to \(\ell + \log 2/\log n\) in line 2, we raise PRIMA’s probability of success to \(1 - 1/n^{\ell'}\).

**Running time**

The running time of PRIMA essentially involves two parts: the time needed to generate the set of RR sets \(R\) and the total time of all NodeSelection invocations. From Lemma 9 of [39], we have for any budget \(k\), the set of RR sets generated for that budget \(R_k\) satisfies,

\[
\mathbb{E}[|R_k|] \leq \frac{3\max(\lambda^+_{b_i}, \lambda^-_{b_i}) \cdot (1 + \epsilon')^2}{(1 - 1/e)} \cdot OPT_{\min} = O((\tilde{b} + \ell')n \log n \cdot e^{-2}/OPT_{\min}).
\]

Further since PRIMA reuses the RR sets instead of generating them from scratch for every budget, for the RR set \(R\) generated by PRIMA,

\[
\mathbb{E}[|R|] = \max_{k \in \mathcal{F}} \mathbb{E}[|R_k|] = O((\tilde{b} + \ell')n \log n \cdot e^{-2}/OPT_{\min}). \quad (10)
\]

For an RR set \(R \in \mathcal{F}\), let \(w(R)\) denote the number of edges in \(G\) pointing to nodes in \(R\). If \(EPT\) is the expected value of \(w(R)\), then we know, \(n \cdot EPT \leq m \cdot OPT_{\min}\) [39]. Hence using Eq. (10), the expected total time to generate \(R\) is determined by,

\[
\mathbb{E}\left[ \sum_{R \in \mathcal{F}} w(R) \right] = \mathbb{E}[|R|] \cdot EPT = O((\tilde{b} + \ell')(n + m) \log n \cdot e^{-2}). \quad (11)
\]

Notice that generating RR set from scratch for the final node selection, following the fix of [13], only adds a multiplicative factor of 2. Hence the overall asymptotic running time to generate \(R\) remains unaffected. Using Lemma 9 and Eq. (11) we now prove the correctness and the running time result of PRIMA.

**Theorem 4.** PRIMA is prefix preserving and returns a \((1 - 1/e - \epsilon)\)-approximate solution to IM with at least \(1 - 1/n^{\ell'}\) probability in \(O((\tilde{b} + \ell' + \log n |\tilde{b}|)(n + m) \log n \cdot e^{-2})\) expected time.

Proof. From Lemma 9, we have that PRIMA returns a prefix preserving \((1 - 1/e - \epsilon)\)-approximate solution with at least \(1 - 1/n^{\ell'}\) probability. In that process PRIMA invokes NodeSelection, \(\log_2 n - 1\) times in the while loop and once to find the final seed set \(S_{\ell}\). Note that, we intentionally avoid redundant calls to NodeSelection when we switch budgets, which saves \(\tilde{b}\) additional calls to NodeSelection.

Let \(\mathcal{R}_i\) be the subset of \(\mathcal{R}\) used in the \(i\)-th iteration of the loop. Since NodeSelection involves one pass over all RR set, on a given input \(\mathcal{R}_i\), it takes \(O(\sum_{R \in \mathcal{R}_i} |R|)\) time. Recall \(|\mathcal{R}_i|\) doubles with every increment of \(i\). Hence it is a geometric sequence with a common ratio of 2. Now from Theorem 3 of [39] and the fact that there is no additional calls to NodeSelection during budget switch, we have total cost of invoking all NodeSelection is \(O(\mathbb{E}[\sum_{R \in \mathcal{R}} |R|])\).

Since \(|R| \leq w(R)\), for any \(R \in \mathcal{R}\), then using Eq. (11) we have,

\[
O(\mathbb{E}[\sum_{R \in \mathcal{R}_i} |R|]) = O(\mathbb{E}[\sum_{R \in \mathcal{R}_i} w(R)])
\]

\[
= O((\tilde{b} + \ell')n + m) \log n \cdot e^{-2})
\]

\[
= O((\tilde{b} + \ell + \log n |\tilde{b}|)(n + m) \log n \cdot e^{-2}).
\]

Hence the theorem follows.

Combining Theorems 3 and 4 we get our main Theorem 2.

**Prefix-preserving extensions to general RIS algorithms.** RIS algorithms have the following two steps:

1. Generate a sufficiently large set of random RR sets.
2. Find \(k\) nodes that covers the most number of RR sets.

In RIS algorithms that are designed for a single item propagation, the stopping criterion of step 1 is tested for a single budget. However, in the UIC model, we have a budget vector. Hence we need to ensure that for any budget in the given budget vector, the stopping criterion is met, i.e., there are enough RR-sets sampled for every budget in the budget vector. A straightforward extension is to make a linear pass on the vector, and sample more RR sets for a budget whenever the number of RR sets is less than required for the current budget. This increases the running time by a factor of the size of the budget vector. In PRIMA, we have made a careful extension to the IMM algorithm to avoid this overhead, by reusing the seed set found in step 2 for previous budgets. Investigation of similar efficient extensions for other RIS algorithms is left for future research.