A Community-Aware Framework for Social Influence Maximization

Abhishek K. Umrawal, Christopher J. Quinn, *Member, IEEE*, and Vaneet Aggarwal, *Senior Member, IEEE*

***Abstract*—We consider the problem of *Influence Maximization* (IM), the task of selecting k seed nodes in a social network such that the expected number of nodes influenced is maximized. We propose a community-aware divide-and-conquer framework that involves (i) learning the inherent community structure of the social network, (ii) generating candidate solutions by solving the influence maximization problem for each community, and (iii) selecting the final set of seed nodes using a novel progressive budgeting scheme.**

Our experiments on real-world social networks show that the proposed framework outperforms the standard methods in terms of run-time and the heuristic methods in terms of influence. We also study the effect of the community structure on the performance of the proposed framework. Our experiments show that the community structures with higher modularity lead the proposed framework to perform better in terms of run-time and influence.

***Index Terms*—Social networks, influence maximization, viral marketing, community detection, submodular maximization**

I. Introduction

A. Motivation

THE advent of social media has changed how traditional marketing strategies were used to be designed [1]. Companies are now preferring to allocate a significant proportion of their marketing budget to drive sales through large social media platforms. There are several ways in which social media can be leveraged for promotional marketing. For instance, advertising on the most visited social platforms, making social media pages for branding and spreading the word about the product, etc. A more sophisticated approach for promotional marketing would be to use the dynamics of the social network to identify the right individuals to be incentivized to get the maximum influence in the entire network.

In the context of social media marketing, Domingos and Richardson posed the *Influence Maximization* (IM) problem [2]: "if we can try to convince a subset of individuals in a social network to adopt a new product or innovation, and the goal is to trigger a large cascade of further adoptions, which set of individuals should we target?" Formally, it is the task of selecting k seed nodes in a social network such that the expected number of influenced nodes in the network (under some influence propagation model), referred to as the *influence*, is maximized. Kempe et al. [3] showed that the problem of influence maximization is NP-Hard. This problem has been widely studied in the literature and several approaches for solving it have been proposed. Some approaches provide near-optimal solutions but are costly in terms of run time. On the other hand, some approaches are faster but heuristics, i.e. do not have approximation guarantees.

Motivated by the idea of addressing this trade-off between accuracy and run-time, we propose a community-aware divide-and-conquer framework to provide a time-efficient solution. The proposed framework outperforms the standard methods in terms of run-time and the heuristic methods in terms of influence.

A. K. Umrawal is with the School of Industrial Engineering, Purdue University, West Lafayette, IN, 47907, USA (email: aumrawal@purdue.edu), and Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, Baltimore, MD, 21250, USA. C. J. Quinn is with the Department of Computer Science, Iowa State University, Ames, IA, 50011, USA (email: cjquinn@iastate.edu). V. Aggarwal is with the School of Industrial Engineering, and Elmore Family School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, 47907 USA (email: vaneet@purdue.edu). He is also with Computer Science, KAUST, Thuwal, 23955, KSA.

This material is based upon work supported in part by the National Science Foundation under Grants No. 1742847, 2149588, and 2149617.

B. Literature Review

In the literature, researchers have tried to solve the Influence Maximization (IM) problem using several approaches. We discuss the relevant approaches as follows.

1) Simple heuristics: Degree centrality is perhaps the simplest way to quantify the influence of an individual in the network [3]. Observing the fact that many of the most central nodes may be clustered, targeting all of them is not at all necessary, Chen et al. [4] proposed the degree discount heuristic. These heuristics are simple and time-efficient. However, they do not have any provable guarantees.

2) Simulation-based methods: The simulation-based methods assume an underlying model for the diffusion of information in the network and select the influential individuals by evaluating different sets of individuals using costly Monte Carlo simulations. Under the *independent cascade* [5], [6] and *linear threshold* [7], [8] models of diffusion (discussed in Section II-B), Kempe et al. [3] have shown that the problem of influence maximization is NP-Hard. They also proposed to use an efficient greedy algorithm [2] which due to a result by Nemhauser et al. [9] gives an $\left(1-\frac{1}{e}\right)$-approximation of the solution. The asymptotic run-time of this algorithm is O(nk). Asymptotically, this greedy algorithm is efficient but empirically the costly Monte Carlo simulations cause a huge overhead. Leskovec et al. [10] proposed the CELF algorithm which improves upon the empirical run-time of the simple greedy algorithm by further exploiting the property of submodularity. Goyal et al. [11] proposed the CELF++ algorithm which further improved upon the empirical run-time of the CELF algorithm by even further exploiting the property

of submodularity to avoiding unnecessary re-computations of marginal gains incurred by CELF. Note that both CELF and CELF++ are greedy algorithms with asymptotic run-time same as the one proposed by Kempe et al. [3], and the run-time gains are only empirical. Borgs et al. [12] proposed a greedy algorithm using reverse influence sampling (RIS) – an approach to efficiently estimate the influence of a seed set. CELF, CELF++, and [12] have the same worst-case run time O(nk) and approximation ratio $\left(1-\frac{1}{e}\right)$ as the one proposed by Kempe et al. [3]. Lotf et al. [13] proposed a genetic algorithm-based heuristic algorithm for dynamic (evolving over time) networks. This method involves Monte Carlo simulation and does not have any approximation guarantees. The framework proposed in this paper may also involve Monte Carlo simulations. But, the divide-and-conquer strategy allows us to significantly reduce the run-time. Asymptotic run-time for both CELF and CELF++ is still O(nk). The method proposed in this paper may involve Monte Carlo simulations. But, the divide-and-conquer strategy allows us to significantly reduce the run-time.

3) Community-based methods: As the proposed method utilizes the inherent community structure of the network, we discuss other community-based methods of influence maximization as follows. Chen et al. [14] proposed two methods called CDH-KCut and CDH-SHRINK under heat diffusion model [15]. They further improved their methods and proposed another method called CIM [16]. Bozorgi et al. [17] proposed a method called INCIM which works only for the linear threshold diffusion model. Moreover, the method involves overlapping community detection contrary to our work where the communities are non-overlapping. Bozorgi et al. [18] have also developed a method for competitive influence maximization [19] under the competitive linear threshold model. Shang et al. [20] have proposed a method called CoFIM under the independent cascade diffusion model and weighted cascade edge-weight model. Contrary to these methods, our method does not depend on the choice of the diffusion model. Huang et al. [21] proposed a data-based method called CTIM which requires a potential action log and item-topic relevance.

4) Data-based methods: In the presence of some observational data or action log involving real-world diffusion traces, the costly Monte Carlo simulations can be avoided completely by estimating the influence directly from the data. Goyal et al. [22], instead of using a propagation model, proposed a novel data-based-method to introduce a model called the credit distribution model, which directly leverages the propagation traces from real-world data and learns the flow of influence in the network. Pen et al. [23] and Deng et al. [24] have studied variants of the credit distribution model under time constraints and node features respectively. The proposed method does not involve any observational data. However, this is a potential future work.

5) Online methods: More recently, the focus has also been on solving the problem of influence maximization in an online manner where the goal is to maximize the cumulative observed influence of the seed sets chosen at different times while receiving instantaneous feedback. Approaches differ based on semi-bandit feedback [25]–[29] and full-bandit feedback [30], [31]. The proposed method is not an online method. However, this is a potential future work.

C. Contribution

In Section I-B, we discussed that the CELF++ [11] algorithm is faster compared to the simple greedy algorithm [2], [3]. But the costly aspect of performing a large number of diffusions in the entire network is still there. Motivated by the idea of solving the influence maximization problem in a time-efficient manner, we propose a community-aware divide-and-conquer framework that involves (i) learning the inherent community structure of the social network, (ii) generating candidate solutions by solving the influence maximization problem for each community, and (iii) selecting the final set of individuals to be incentivized from the candidate solutions using a novel progressive budgeting scheme. Our method may also use the Monte Carlo simulations but we are restricting them within each community as compared to the entire network which brings savings in terms of run-time as compared to the CELF++ algorithm.

Compared to the other community-based methods, the proposed framework is novel in the following ways. It is not limited to a specific diffusion and/or an edge-weight model. In Step 1, the set of candidate solutions is generated by all combinations of solutions from each community. In Step 2, the final seed selection is performed by solving an integer linear program (ILP) over candidate solutions subject to a budget constraint. We propose an efficient progressive budgeting scheme to efficiently solve the ILP in Step 3. We provide the proof of correctness of this scheme which leverages submodularity (defined in Section II) of the influence.

We provide experiments on real-world social networks, showing that the proposed framework outperforms simulation-based methods in terms of run-time and heuristic methods in terms of influence. We study the effect of the community structure on the performance of the proposed framework. Our experiments show that the community structures with higher 'modularity' (defined in Section II) lead the proposed framework to perform better in terms of run-time and influence.

D. Organization

The rest of the paper is organized as follows. In Section II, we discuss the preliminaries and formulate the problem. In Section III, we discuss our methodology. In Section IV, we discuss the experiments performed for different social networks. Section V concludes the paper and discusses future directions.

II. Preliminaries and Problem Formulation

In this section, we discuss some preliminaries and formulate the problem of interest in this paper. Refer to Table IV (in Appendix C) for the important notations used throughout the paper.

A. Submodularity

Let Ω denote the ground set of n elements and $\mathcal{O} = 2^{\Omega}$ to be the set of all subsets of Ω.

A set function $f : \mathcal{O} \rightarrow \mathbb{R}$ is said to be *submodular* if it satisfies a natural 'diminishing return' property: the marginal gain from adding an element v to a set $S \in \mathcal{S}$ is at least as high as the marginal gain from adding the same element v to a superset $T \in \mathcal{O}$ of S. Formally, for any sets $S, T \in \mathcal{O}$ such that $S \subseteq T$, f satisfies

$$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T). \tag{1}$$

A set function $f : \mathcal{O} \rightarrow \mathbb{R}$, is said to be *monotone* (non-decreasing) if for any sets $S, T \in \mathcal{O}$ such that $S \subseteq T$, f satisfies

$$f(S) \leq f(T). \tag{2}$$

B. Diffusion models and social influence

Diffusion models describe how the cascade takes place in a social network. For the purpose of our research, we focus on the *independent cascade* [5], [6] and *linear threshold* [7], [8] models of diffusion.

In the independent cascade model, given a graph $G = (V, E)$, the process starts at time 0 with an initial set of active nodes S, called the *seed set*. When a node $v \in S$ first becomes active at time t, it will be given a single chance to activate each currently inactive neighbor w, it succeeds with a probability $p_{v,w}$ (independent of the history thus far). If w has multiple newly activated neighbors, their attempts are sequenced in an arbitrary order. If v succeeds, then w will become active at time $t + 1$; but whether or not v succeeds, it cannot make any further attempts to activate w in subsequent rounds. The process runs until no further activation is possible.

In the linear threshold model, given a graph $G = (V, E)$, a node v is influenced by each neighbor w according to a weight $p_{v,w}$ such that $\sum_{w \in \partial v} p_{v,w} \leq 1$, where ∂v represents the set of neighbors of v. Each node v chooses a *threshold* θ_v uniformly from the interval [0,1]; this represents the weighted fraction of v's neighbors that must become active in order for v to become active. The process starts with a random choice of thresholds for the nodes, and an initial set of active nodes S, called the *seed set*. In step t, all nodes that were active in step $t - 1$ remain active, and we activate any node v for which the total weight of its active neighbors is at least θ_v. The process runs until no more activation is possible.

Note that both these processes of diffusion are *progressive*, i.e. the nodes can switch from being inactive to active, but do not switch in the other direction.

At any time t in the cascade, each node $v \in V$ can be either active or inactive. We denote the process as

$$y_t^{(v)} = \begin{cases} 1, & \text{if node } v \text{ was active at time } t, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The *influence* $\sigma(S)$ of a set S is defined as the expected number of active nodes at the end of the cascade (denoted by time T), given that S is the initial set of nodes.

$$\sigma(S) = \mathbb{E}\left[\sum_{v \in V} y_T^{(v)} \middle| y_0^{(v)} = 1 \ \forall v \in S, y_0^{(v)} = 0 \ \forall v \notin S\right]. \tag{4}$$

Kempe et al. [3] have shown that under independent cascade and linear threshold models of diffusion, $\sigma(S)$ is a monotone and submodular set function.

C. Problem statement

For a given integer budget k, we are interested in finding a $k-$node subset of the set of nodes V, which has the maximum influence over all possible $k-$node subsets of V. Formally, the problem of influence maximization (IM) is defined as

Problem 1.

$$\begin{aligned} & S^* \in \arg\max_{S \subseteq V} \ \sigma(S), \\ & \textit{s.t.} \qquad |S| \leq k. \end{aligned} \tag{5}$$

III. Methodology

As discussed earlier, the simulation-based methods suffer from the huge overhead of carrying out diffusions in the entire network to estimate the influence of different candidate solutions. Motivated by the idea of solving the influence maximization problem defined in (5) in a time-efficient manner, we propose a community-aware divide-and-conquer framework. The proposed framework tries to lower the cost of simulations by restricting the diffusions to some sub-networks of the original network instead of the entire network by partitioning the given network into several sub-networks. As most real-world networks exhibit some community structure, such a partitioning of a network can be obtained by learning its inherent community structure. The proposed framework involves (i) learning the inherent community structure of the social network, (ii) generating candidate solutions by solving the influence maximization problem for each community, and (iii) selecting the final set of individuals to be incentivized from the candidate solutions using a novel progressive budgeting scheme.

Algorithm 1 outlines the framework proposed in this paper. It uses three sub-routines which are explained in the following subsections.

Algorithm 1 Community-IM

1: **Input** G, k, `com-method`, `sol-method`.
2: $\{G_1, ..., G_c\}$ = Community-Detection(G, `com-method`)
3: **for** community i **do**
4: $\quad \mathcal{S}_i, \Sigma_i$ = Generate-Candidates(G_i, k, `sol-method`)
5: **end for**
6: $\mathcal{S} = \{\mathcal{S}_i : i = 1, ..., c\}$, $\Sigma = \{\Sigma_i : i = 1, ..., c\}$
7: S^* = Progressive-Budgeting($\mathcal{S}, \Sigma, k$)
8: **return** S^*

A. Learning the inherent community structure of the social network

For the given social network $G = (V, E)$, we obtain a hard partition $\{V_1, ..., V_c\}$ of V using some community detection method. By hard partitioning, we mean $V_i \cap V_j = \phi \ \forall i \neq j = 1, ..., c$ and $\bigcup_i V_i = V$. Let $|V_i| = n_i$ be the size of ith community, $i = 1, ..., c$, $\sum_{i=1}^{c} n_i = n$. Define $G_i = (V_i, E_i)$ where E_i is the set of edges from E which belong to the pairs of nodes in V_i. We call $\{G_1, ..., G_c\}$ a network-partition.

Most community detection methods try to find communities in the network such that the nodes within a community are more 'well-connected' than the nodes between communities. Methods usually differ as to how they measure the connectedness of the nodes in a network. Common methods are Louvain [32], label propagation [33], and Girvan-Newman algorithm [34]. Algorithm 2 outlines the community detection step.

Algorithm 2 Community-Detection

1: **Input** G, `com-method`.
2: Use `com-method` to partition G into $\{G_1, ..., G_c\}$
3: **return** $\{G_1, ..., G_c\}$

1) Quality of a network-partition: The quality of a network-partition can be measured using modularity score [35], [36]. The modularity score of a network-partition is defined as the fraction of the edges that fall within the given groups minus the expected fraction if edges were distributed at random. For a network-partition $\{G_1, ..., G_c\}$, modularity [36] is defined as

$$Q = \sum_{i=1}^{c} \left[\frac{L_i}{|E|} - \left(\frac{\delta_i}{|E|} \right)^2 \right]$$

where L_i is number of edges between the pairs of nodes in G_i and δ_i is the sum of degrees of nodes in G_i.

The modularity score measures how well a community detection algorithm partitions a network. A higher value of modularity corresponds to a network-partition with higher connectedness within each community.

2) Community detection methods: We discuss some commonly used community detection methods. The Louvain method [32] first obtains small communities by optimizing modularity locally on all of the nodes. Then each small community is treated as a single node and the previous step is repeated. Label propagation [33] starts with a (generally small) random subset of the nodes with community labels. The algorithm then iteratively assigns labels to previously unlabeled nodes. The Girvan-Newman method [34] method uses a measure known as 'betweenness.' Define the betweenness of an edge [34] as the sum of the 'weights' of the shortest paths between any pair of nodes that run along it. If there are d different shortest paths between any two nodes then the weight of each path is set as $1/d$. The Girvan-Newman method [34] method involves the following steps.

1) First, calculate the betweenness of all existing edges in the network.
2) Next, remove the edge(s) with the highest betweenness.
3) Finally, recalculate the betweenness of all edges affected by the removal at the previous step.
4) Repeat the previous two steps until no edge remains.

B. Generating candidate solutions by solving the influence maximization problem for each community

For every community, we find the subgraph of the G by keeping only the nodes in that community and all edges incident on them, then find 'best' seed sets of sizes up to k for that subgraph using a standard method. Let $S_{i,j}$ be the best seed set of size j ($j = 1, ..., k$) from community i, and $\sigma_i(S_{i,j})$ be its influence within community i ($i = 1, ..., c$).

Solving the influence maximization problem separately for different communities instead of the entire network brings gains in empirical run-time as the lengths of the diffusions get much shorter within communities as compared to the entire network. Algorithm 3 outlines the standard influence maximization step.

Algorithm 3 Generate-Candidates

1: **Input** G_i, k, `sol-method`.
2: Use `sol-method` to solve IM for G_i up to budget k
3: **return** $\mathcal{S}_i = \{S_{i,j} : j = 1, ..., k\}$, $\Sigma_i = \{\sigma_i(S_{i,j}) : j = 1, ..., k\}$

C. Selecting the final seed set

We select as many sets from $\{S_{i,j} : i = 1, ..., c; j = 1, ..., k\}$ with no repeating elements across them such that the sum of their influences is maximized and their union has exactly k elements. Use that union as a solution to (5). Formally, we are solving the following *integer linear program (ILP).*

$$\begin{aligned} x_{i,j} \ \in \ & \arg\max_{x_{i,j}} \ \sum_{i=1}^{c} \sum_{j=1}^{k} x_{i,j} \sigma_i(S_{i,j}), \\ \text{s.t.} \quad & \sum_{i=1}^{c} \sum_{j=1}^{k} x_{i,j} |S_{i,j}| = k, \quad \text{(budget constraint)}, \\ & \sum_{j=1}^{k} x_{i,j} = 1 \ \forall i, \quad \text{(no repetition constraints)}, \\ & x_{i,j} \in \{0, 1\}, \ \forall i, j, \quad \text{(integer constraints)}. \end{aligned} \tag{6}$$

After solving the above, the final solution is given as

$$S^* = \bigcup_{\substack{i=1,...c; j=1,...,k \\ x_{i,j}=1}} S_{i,j} \tag{7}$$

In general, solving an ILP is an NP-Complete problem [37]. However, the submodularity of the influence allows us to solve the ILP in (6) in polynomial time.

1) Progressive Budgeting: By the definition of submodularity, we know that the marginal gains in influence due to every additional node in the seed set are diminishing. Hence, we can progressively allocate the budget across the sets in $\mathcal{S}$ in the sense that once a set is allocated a budget, it remains there

Algorithm 4 Progressive-Budgeting

Input $\mathcal{S}, \Sigma, k$.
$\{\mathcal{S}_i : i = 1, ..., c\} = \mathcal{S}$, $\{\Sigma_i : i = 1, ..., c\} = \Sigma$
$\{S_{i,j} : j = 1, ..., k\} = \mathcal{S}_i$, $\{\sigma_i(S_{i,j}) : j = 1, ..., k\} = \Sigma_i$
$\delta_i = \sigma_i(S_{i,1})\ \forall i$ ▷ Initialize the marginal gains.
$b_i = 0\ \forall i$ ▷ Initialize the budget allocations.
$S^* = \phi$ ▷ Initialize the final set.
for $l = 1, ..., k$ **do**
  $m = \arg\max_i \delta_i$ ▷ Index of the community with maximum marginal gain.
  $b_m = b_m + 1$ ▷ Update the budget allocated to community m.
  $S^* = S^* \cup S_{m,b_m}$ ▷ Add the corresponding set to the final set.
  $\delta_m = \sigma_i(S_{m,b_m+1}) - \sigma_i(S_{m,b_m})$ ▷ Update the marginal gains.
end for
return S^* ▷ Final seed set.

in the list of all finally selected sets. Progressive-Budgeting sub-routine used in Algorithm 1 is outlined in Algorithm 4.

Theorem 1. *Progressive budgeting solves the ILP in (6).*

Proof. The proof follows due to the submodularity of the influence. The ILP in (6) is trying to select unique $S_{i,j}$'s such that the sum of the cardinalities of all of them is equal to k and the sum of their influences is maximized. We know that $\mathcal{S}_i = \{S_{i,j} : j = 1, ..., k\}$ is the greedy solution to the problem of influence maximization within community i. Hence, due to submodularity, we have $\sigma_i(S_{i,1}) - \sigma_i(S_{i,0}) \geq ... \geq \sigma_i(S_{i,k}) - \sigma_i(S_{i,k-1})\ \forall i$. Leveraging this property, the progressive budgeting algorithm is iteratively building up the set of unique $S_{i,j}$'s by comparing marginal improvements in influence between different choices. Hence, progressive budgeting indeed solves the ILP in (6). □

An illustrative example of progressive budgeting is provided in Appendix B.

D. Computational complexity analysis

We now analyze the computational complexity of the proposed framework (Algorithm 1). The run-time of the proposed framework is the sum of the times taken at the three steps. It depends on the choice of community detection method as well as the solution method to solve IM for each community. We analyze the run-time involved at each step as follows.

1) Learning the inherent community structure of the social network: The worst-case run-times of different community detection algorithms considered in this paper are given as follows: the Louvain method is $O(n \log n)$ [32], label propagation is $O(n + |E|)$ [33], and the Girvan-Newman method is $O(n|E|^2)$ [34].

2) Generating candidate solutions by solving the influence maximization problem for each community: If we use CELF++ to solve IM for c different communities then we are solving c problems of finding a k-node subset for each community from n_i nodes, $i = 1, \ldots, c$. For the ith community, CELF++ iteratively builds the k-node subset as follows. First, find the best individual node by evaluating all n_i subsets of cardinality one. Next, find the node with the highest marginal influence in the presence of the best individual node by evaluating (up to) all $n_i - 1$ subsets of the previously selected best individual and an additional node. CELF++ then keeps adding nodes to the previous set in the same manner until the size of the current set is k. The number of k-node subsets evaluated at the kth step is $n_i - (k-1)$ in the worst case. Thus, the number of subsets evaluated in the worst case is

$$\sum_{i=1}^{c} [n_i + (n_i - 1) + \cdots + (n_i - (k-1))] = nk - \frac{ck(k-1)}{2}. \quad (8)$$

On the contrary, if we use CELF++ for the entire network then the total number of subsets evaluated in the worst case is

$$n + (n-1) + \cdots + (n - (k-1)) = nk - \frac{k(k-1)}{2}. \quad (9)$$

By comparing (8) and (9), we observe that the Generate-Candidates step of the proposed framework achieves a lower run-time compared to using the `sol-method` for the entire network by an additive factor of $(c-1)k(k-1)/2$. Furthermore, as $n_i \leq n\ \forall i = 1, \ldots, c$, the length of the diffusion while evaluating a subset of the nodes using Monte Carlo simulations within any community will always be smaller as compared to doing the same in the entire network. This further reduces the run-time of the Generate-Candidates step.

3) Final seed set selection using progressive budgeting: The progressive budgeting method of final seed set selection solves 'finding the maximum of c elements' k times. Hence, the worst-case run-time of progressive budgeting is $O(ck)$.

In practice, solving IM for each community (using a simulation-based `sol-method`) is the step that takes the most amount of time due to the costly Monte Carlo simulations. In that sense, the worst-case run-time of the proposed framework (with a simulation-based `sol-method`) to solve IM for each community is lower compared to the same for solving IM for the original network using the same simulation-based `sol-method`.

IV. Experiments

We evaluated the performance of the proposed framework using real-world social networks. We discuss the network data used for our experiments, list the algorithms chosen for comparison, provide experimental details, present results, and discussion.

A. Network data

We used 4 real-world social networks for our experiments. The data is available at Stanford Large Network Dataset Collection [38]. The number of nodes, number of edges, and type of each network are provided in Table I.

The Facebook network is a dataset consisting of 'circles' (or 'friends lists') from Facebook [39]. Bitcoin network is a who-trusts-whom network of people who trade using Bitcoin on

TABLE I: Basic information of the networks used.

Network	Nodes	Edges	Type
Facebook [39]	4,039	88,234	Undirected
Bitcoin [40], [41]	5,881	35,592	Directed
Wikipedia [42], [43]	7,115	103,689	Directed
Epinions [44]	75,879	508,837	Directed

a platform called Bitcoin OTC [40], [41]. Wikipedia network is a who-votes-on-whom network to become an administrator [42], [43]. Epinions is a who-trust-whom online social network of a general consumer review platform called Epinions [44].

For undirected networks, each edge was replaced by two directed edges.

For edge-weights, two models are used which are *weighted cascade model* [3] where for each node $v \in V$, the weight of each edge entering v was set to $1/\text{in-degree}(v)$ and *trivalency model* [22] where each edge-weight was drawn uniformly at random from a small set of constants $\{0.1,\ 0.01,\ 0.001\}$. However, for the linear threshold model of diffusion, only the weighted cascade model is used for edge-weights as the trivalency model does not necessarily maintain the sum of weights of all edges incident on a node to be less than or equal to 1.

B. Algorithms

We compared the proposed community-aware framework (Community-IM) with the following algorithms.

1) CELF++ [11], a simulation-based greedy algorithm.
2) CoFIM [20], a community-aware heuristic algorithm.
3) DSGA [13], an genetic algorithm-based method.
4) Degree-Discount [4], a heuristic algorithm.
5) Out-Degree, a heuristic algorithm where for budget k, top-k out-degree nodes are selected.

We chose the above algorithms due to the following reasons.

- CELF++ is the state-of-the-art simulation-based algorithm.
- CoFIM is a community-aware heuristic with theoretic guarantees under the independent cascade diffusion model with the weighted-cascade [3] edge-weight model.
- DSGA [13] is a recent genetic algorithm-based method that uses Monte Carlo simulations.
- Degree-Discount and Out-Degree are some of the simplest yet powerful heuristics.

INCIM algorithm [17] discussed in the literature review subsection under Section I could have been a community-aware baseline for our comparisons under the linear threshold diffusion model but it uses overlapping community structure contrary to our method.

Note that the CoFIM algorithm was developed only for the independent cascade diffusion model with the weighted-cascade edge-weight model. However, for empirical comparisons, we have implemented it for other choices of diffusion models and edge-weight models as well.

As part of Community-IM, we used the Louvain method [32] as `com-method`, and CELF++ [11] as `sol-method` for Community-Detection and Generate-Candidates respectively.

We also studied the effect of the inherent community structure on the performance of the proposed framework. For this, we use the community structures learned using the community detection algorithms discussed in Section III-A2.

For brevity, we only consider the Facebook network under the weighted cascade edge-weight model to study the effect of the learned community structure of the social network.

C. Experimental details

We used the budget $k = 1, 5, 10, \ldots, 100$ for comparing different algorithms. However, for DSGA [13], we only used the budget $k = 1, 20, 40, \ldots, 100$ due to its high run-time. The influence of any seed set was estimated as the average number of active nodes from $1,000$ different Monte Carlo simulations of the underlying diffusion starting with the same seed set. For any network, if a community detection method returns some communities whose individual sizes are below 1% of the number of nodes in the network then we merged them all into a single community. We do this to avoid having too many small communities.

The experiments are carried out on a computer with 2.6 GHz 24-core Intel Xeon Gold Sky Lake processors and 96 GB of memory. We used Python for our implementation. The source codes of CELF++ and CoFIM provided by their authors are written in C++.

D. Results

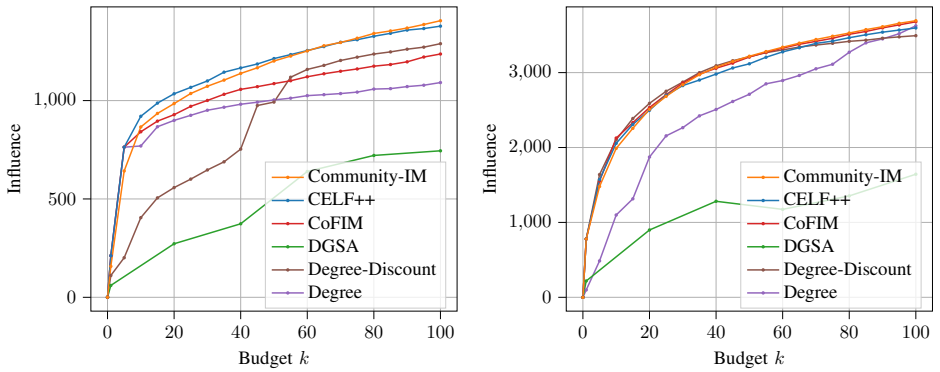
For different networks under different diffusion models and edge-weight models,

- Figure 1 - 3 show the influences of chosen seed sets using different algorithms for different values of k.
- Table II shows the influence of the seed set of size 100 chosen using CELF++, Community-IM, CoFIM, and DSGA.
- Table III shows the empirical run-times of CELF++, Community-IM, CoFIM, and DSGA for $k = 100$.

Further, the results for the Facebook network under different diffusion models and the weighted cascade edge-weight model using different community detection methods are provided in Appendix C.

E. Discussion

Figure 1(a) shows that for the Facebook network under the independent cascade diffusion model and the weighted cascade edge-weight model, for budget k up to 60, the influence for Community-IM is marginally lower than the influence for CELF++. However, for budget k larger than 60, the influence for Community-IM is the same or higher than the influence for CELF++. Furthermore, for all values of budget k, the influence for Community-IM is much higher compared to the influence for all the other methods except CELF++. Figure 1(d) provides a similar insight for the Epinions network under the independent cascade diffusion model and the weighted cascade edge-weight model. Figure 2(a) provides a similar insight for the Facebook network under the independent cascade diffusion model and the trivalency edge-weight model.



(a) Facebook network

(b) Bitcoin network

(c) Wikipedia network

(d) Epinions network

Fig. 1: Influence vs. k for different networks under independent cascade diffusion model and weighted cascade edge-weight model.

TABLE II: Comparison of influences for budget $k = 100$.

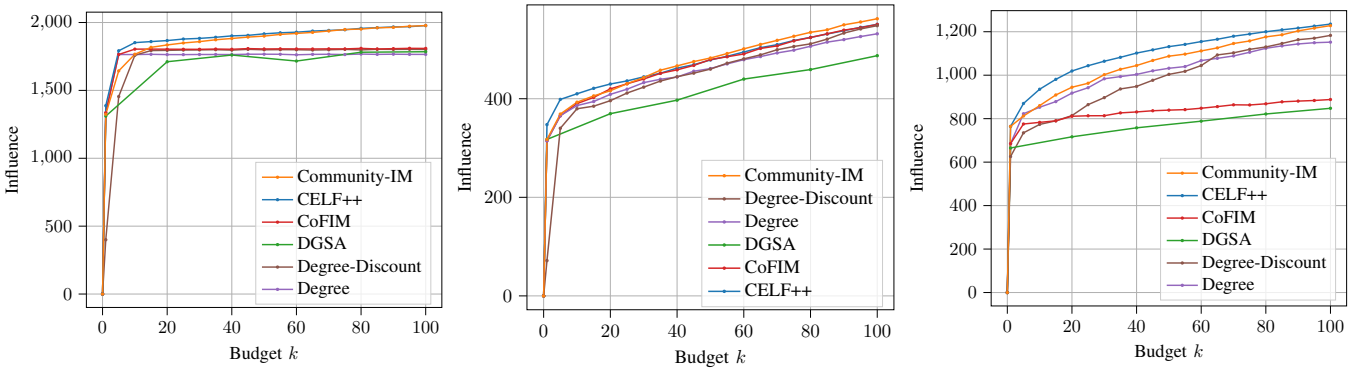
Diffusion model	Edge-weight model	Network	CELF++	Community-IM	CoFIM	DSGA	Degree	Degree-Discount
Independent cascade	Weighted cascade	Facebook	1,378.39	1,406.02	1,236.78	846.40	1,091.66	1,289.28
		Bitcoin	3,493.23	3,692.89	3,678.58	1,643.27	3,596.40	3,625.06
		Wikipedia	876.91	873.04	528.36	212.60	866.06	878.36
		Epinions	14,042.60	14,706.07	12,314.97	2,439.24	13,457.51	13,771.25
	Trivalency	Facebook	1,977.27	1,977.42	1,809.12	1,304.82	1,765.09	1,801.25
		Bitcoin	550.90	562.27	550.69	487.38	531.74	548.24
		Wikipedia	1,235.22	1,228.07	888.18	847.63	1,152.01	1,183.11
Linear threshold	Weighted cascade	Facebook	1,946.41	2,231.34	1,936.23	969.04	1,835.35	1,999.80
		Bitcoin	4,505.65	4,829.30	4,821.53	1,742.90	4,740.46	4,793.51
		Wikipedia	1,138.64	1,117.36	602.26	246.26	1,118.68	1,162.21

Figure 1(b) shows that for the Bitcoin network under the independent cascade diffusion model and weighted cascade edge-weight model, for budget k up to 60, the influences for all methods are very close to each other except that the influence for DSGA and Degree-Discount are lower than the rest. However, for budget k larger than 60, the influence for Community-IM tends to surpass the influence of all other methods except the influence for CoFIM.

Figure 1(c) shows that for the Wikipedia network, for all values of budget k, the influence for Community-IM is marginally lower than that for CELF++, and the influence for Community-IM is higher compared to that for CoFIM, DSGA, Degree-Discount and Degree under the independent cascade diffusion model and the trivalency edge-weight model.

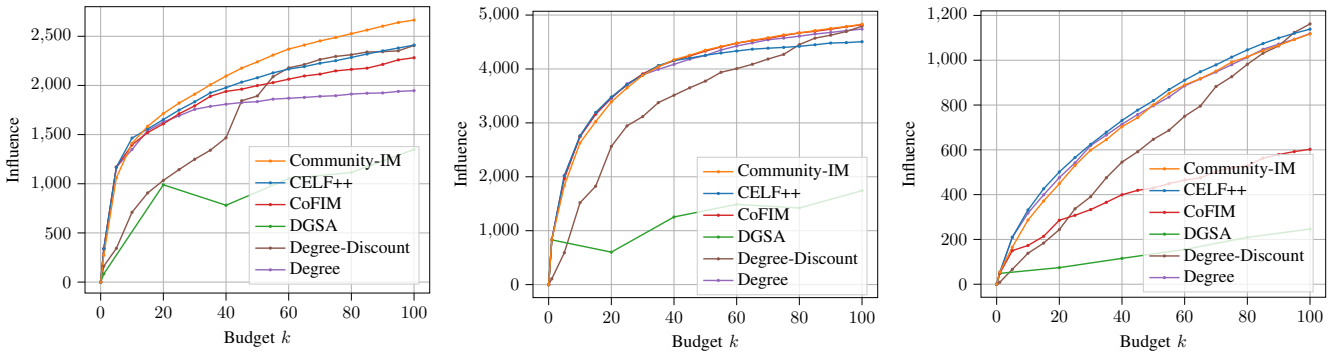
Figure 2(b) shows that for the Bitcoin network under the independent cascade diffusion model and the trivalency edge-weight model, for budget k up to 30, the influence for Community-IM is marginally lower than that for CELF++. However, for budget k larger than 30, the influence for Community-IM is higher than that for CELF++. Furthermore, the influence for Community-IM is higher compared to that for CoFIM, DSGA, Degree-Discount, and Degree.

Figure 2(c) shows that for the Wikipedia network under the independent cascade diffusion model and the trivalency edge-weight model, for all values of budget k, the influence for Community-IM is marginally lower than that for CELF++.



(a) Facebook network (b) Bitcoin network (c) Wikipedia network

Fig. 2: Influence vs. budget k for different networks under independent cascade diffusion model and trivalency edge-weight model.



(a) Facebook network (b) Bitcoin network (c) Wikipedia network

Fig. 3: Influence vs. budget k for different networks under linear threshold diffusion model and weighted cascade edge-weight model.

TABLE III: Comparison of empirical run-times (in seconds) for budget $k = 100$.

Diffusion model	Edge-weight model	Network	CELF++	Community-IM	CoFIM	DSGA
Independent cascade	Weighted cascade	Facebook	17,359	3,782	547	9,1267
		Bitcoin	10,859	850	35	20,825
		Wikipedia	2,660	3,477	213	18,447
		Epinions	250,241	16,465	7,397	267,796
	Trivalency	Facebook	74,684	7,195	567	312,948
		Bitcoin	7,818	576	35	34,453
		Wikipedia	35,227	3,771	211	77,241
Linear threshold	Weighted cascade	Facebook	46,771	8,545	554	65,391
		Bitcoin	45,747	1,077	36	62,184
		Wikipedia	5,940	4,628	224	23,307

However, the gap between the influence for Community-IM and that for CELF++ decreases as the budget k increases. Furthermore, the influence for Community-IM is higher compared to that for CoFIM, DSGA, Degree-Discount, and Degree. Figure 3(c) provides a similar insight for the Wikipedia network under the linear threshold diffusion model and the weighted cascade edge-weight model.

Figure 3(a) shows that for the Facebook network under linear threshold diffusion model and weighted cascade edge-weight model, for budget k up 15, the influence for Community-IM is marginally lower than that for CELF++. However, for budget k larger than 15, the influence for Community-IM is higher than that for CELF++. Furthermore, the influence for Community-IM is higher compared to that for CoFIM, DSGA, Degree-Discount, and Degree.

Figure 3(b) shows that for the Bitcoin network under linear threshold diffusion model and weighted cascade edge-weight model, for budget k up 40, the influence for Community-IM is marginally lower than that for CELF++. However, for budget k larger than 40, the influence for Community-IM is higher than that for CELF++. Furthermore, note that for budget $k \geq 30$, the influence for CoFIM, Degree-Discount

and Degree are higher than that for CELF++, but at the same time Community-IM performs better or as good as DSGA, Degree-Discount and Degree.

Moreover, Table II shows that for each network, the influence of the chosen seed set of size 100 using Community-IM is very close or even better to the same for CELF++ under different diffusion models and edge-weight models.

Furthermore, Table III shows that the proposed community-aware framework brings huge savings in terms of empirical run-time as compared to the CELF++ algorithm under all choices of diffusion models and edge-weight models. As noted Community-IM is much faster than CELF++, and DSGA across different networks, diffusion models, and edge-weight models.

From Table III, we also note that the gain in run-time varies across diffusion models and edge-weight models. The highest gains are for the independent cascade model with the trivalency edge-weight model and the least gains are for the independent cascade model with the weighted cascade edge-weight model.

Table VII (in Appendix C) shows that for the Facebook network, the influence of the chosen seed set of size 50 using Community-IM is approximately equal to the same for CELF++ for different choices of community detection methods under different diffusion models and weighted cascade edge-weight model. Moreover, Tables VII and VIII (in Appendix C) show that the performance of Community-IM compared to CELF++ in terms of influence and run-time improves as the modularity of the partition and the number of communities increase. From Table VII and Table VIII (in Appendix C), we also note that the Louvain method is the best choice of community detection method and the Girwan-Newman method performs the worst. The Louvain method partitions the graph into 18 communities with the largest community having 523 nodes which are approximately 10% of the size of the entire network. Hence, Community-IM does not come across any giant component and converges faster. Contrary to this, the Girwan-Newman algorithm partitions the network into just two communities with the largest community having 3,833 nodes which are very close to the size of the entire network, and hence the Community-IM method takes a lot of time in converging.

We also observe that for all values of budget k, the influence for Community-IM with the Girwan-Newman algorithm is very close to CELF++ which can be explained by the fact that the Girwan-Newman algorithm divides the entire network into just two communities with one community being a giant connected component of the entire network. On the other hand, for all values of budget k the influences for Community-IM with Louvain algorithm and Community-IM with Label propagation algorithm are very close to each other which can be explained by the fact that the modularity scores of the partitions obtained by these two methods are quite close.

Overall, we observe that the proposed framework brings savings in terms of run-time at the cost of minimal loss in terms of influence compared to the state-of-the-art simulation-based algorithm, CELF++ and improves in terms of influence compared to the rest of the algorithms.

V. Conclusion and Future Work

For solving the problem of influence maximization on social networks, we leveraged the inherent community structure of a network and proposed a novel community-aware framework for maximizing the spread of influence through a social network in a fast manner. Based on our experiments, we conclude that the proposed framework outperforms simulation-based algorithms in terms of empirical run-time and heuristic algorithms in terms of influence. As the proposed method leverages the inherent community structure of the network, we also studied the effect of the community structure on the performance of our framework. Based on our experiments, we conclude that the community structures with higher modularity lead the proposed framework to perform better in terms of run-time and influence. Among the methods considered in this paper, we find the Louvain algorithm [32] to be the best for the problem of influence maximization.

We point out two limitations of our method. First, our method requires the communities learned during step (i) to be non-overlapping. However, in general, a real-world social network may have overlapping communities. Second, our method does not explicitly account for the inter-community influence while generating the candidate solutions during step (ii). In the future, we want to extend our method so that it can handle overlapping community structures and also explicitly accounts for the inter-community influence. Other future directions are to extend the proposed community-aware framework to competitive influence maximization [45], data-based influence maximization [22], and full-bandit online influence maximization [30], [31].

References

[1] D. Evans, *Social Media Marketing: The Next Generation of Business Engagement*. John Wiley & Sons, 2010.

[2] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2001, pp. 57–66.

[3] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2003, pp. 137–146.

[4] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 199–208.

[5] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," *Marketing Letters*, vol. 12, no. 3, pp. 211–223, 2001.

[6] ——, "Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata," *Academy of Marketing Science Review*, vol. 9, no. 3, pp. 1–18, 2001.

[7] M. Granovetter, "Threshold models of collective behavior," *American Journal of Sociology*, vol. 83, no. 6, pp. 1420–1443, 1978.

[8] T. C. Schelling, *Micromotives and Macrobehavior*. WW Norton & Company, 2006.

[9] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—I," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[10] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007, pp. 420–429.

[11] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++: Optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th International Conference Companion on World Wide Web*, 2011, pp. 47–48.

[12] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2014, pp. 946–957.

[13] J. J. Lotf, M. A. Azgomi, and M. R. E. Dishabi, "An improved influence maximization method for social networks based on genetic algorithm," *Physica A: Statistical Mechanics and its Applications*, vol. 586, p. 126480, 2022.

[14] Y.-C. Chen, W.-C. Peng, and S.-Y. Lee, "Efficient algorithms for influence maximization in social networks," *Knowledge and Information Systems*, vol. 33, no. 3, pp. 577–601, 2012.

[15] H. Ma, H. Yang, M. R. Lyu, and I. King, "Mining social networks using heat diffusion processes for marketing candidates selection," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, 2008, pp. 233–242.

[16] Y.-C. Chen, W.-Y. Zhu, W.-C. Peng, W.-C. Lee, and S.-Y. Lee, "CIM: Community-based influence maximization in social networks," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 2, pp. 1–31, 2014.

[17] A. Bozorgi, H. Haghighi, M. S. Zahedi, and M. Rezvani, "INCIM: A community-based algorithm for influence maximization problem under the linear threshold model," *Information Processing & Management*, vol. 52, no. 6, pp. 1188–1199, 2016.

[18] A. Bozorgi, S. Samet, J. Kwisthout, and T. Wareham, "Community-based influence maximization in social networks under a competitive linear threshold model," *Knowledge-Based Systems*, vol. 134, pp. 149–158, 2017.

[19] S. Bharathi, D. Kempe, and M. Salek, "Competitive influence maximization in social networks," in *Internet and Network Economics*, X. Deng and F. C. Graham, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 306–311.

[20] J. Shang, S. Zhou, X. Li, L. Liu, and H. Wu, "CoFIM: A community-based framework for influence maximization on large-scale networks," *Knowledge-Based Systems*, vol. 117, pp. 88–100, 2017.

[21] H. Huang, H. Shen, Z. Meng, H. Chang, and H. He, "Community-based influence maximization for viral marketing," *Applied Intelligence*, vol. 49, no. 6, pp. 2137–2150, 2019.

[22] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "A data-based approach to social influence maximization," *Proceedings of the VLDB Endowment*, vol. 5, no. 1, pp. 73–84, 2011.

[23] Y. Pan, X. Deng, and H. Shen, "Credit distribution for influence maximization in online social networks with time constraint," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. IEEE, 2015, pp. 255–260.

[24] X. Deng, Y. Pan, Y. Wu, and J. Gui, "Credit distribution and influence maximization in online social networks using node features," in *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*. IEEE, 2015, pp. 2093–2100.

[25] S. Lei, S. Maniu, L. Mo, R. Cheng, and P. Senellart, "Online influence maximization," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 645–654.

[26] Z. Wen, B. Kveton, M. Valko, and S. Vaswani, "Online influence maximization under independent cascade model with semi-bandit feedback," in *Advances in Neural Information Processing Systems*, 2017, pp. 3022–3032.

[27] S. Vaswani, B. Kveton, Z. Wen, M. Ghavamzadeh, L. Lakshmanan, and M. Schmidt, "Diffusion independent semi-bandit influence maximization," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[28] S. Li, F. Kong, K. Tang, Q. Li, and W. Chen, "Online influence maximization under linear threshold model," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1192–1204, 2020.

[29] P. Perrault, J. Healey, Z. Wen, and M. Valko, "Budgeted online influence maximization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7620–7631.

[30] M. Agarwal, V. Aggarwal, A. K. Umrawal, and C. J. Quinn, "Stochastic top k-subset bandits with linear space and non-linear feedback with applications to social influence maximization," *ACM/IMS Transactions on Data Science (TDS)*, vol. 2, no. 4, pp. 1–39, 2022.

[31] G. Nie, M. Agarwal, A. K. Umrawal, V. Aggarwal, and C. J. Quinn, "An explore-then-commit algorithm for submodular maximization under full-bandit feedback," in *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[32] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[33] G. Cordasco and L. Gargano, "Community detection via semi-synchronous label propagation algorithms," in *2010 IEEE International Workshop on: Business Applications of Social Network Analysis (BASNA)*. IEEE, 2010, pp. 1–8.

[34] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[35] M. Newman, *Networks*. Oxford University Press, 2018.

[36] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, p. 066111, 2004.

[37] R. Kannan and C. L. Monma, "On the computational complexity of integer programming problems," in *Optimization and Operations Research*. Springer, 1978, pp. 161–172.

[38] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[39] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 539–547.

[40] S. Kumar, F. Spezzano, V. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 2016, pp. 221–230.

[41] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 2018, pp. 333–341.

[42] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 1361–1370.

[43] ——, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th International Conference on World Wide Web*, 2010, pp. 641–650.

[44] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *International Semantic Web Conference*. Springer, 2003, pp. 351–368.

[45] S. Bharathi, D. Kempe, and M. Salek, "Competitive influence maximization in social networks," in *International Workshop on Web and Internet Economics*. Springer, 2007, pp. 306–311.

Abhishek K. Umrawal is currently a Ph.D. Candidate in the School of Industrial Engineering at Purdue University, West Lafayette, IN 47907, USA, and a Visiting Lecturer in the Department of Computer Science and Electrical Engineering at the University of Maryland, Baltimore County, MD 21250, USA. He received an M.Sc. degree in Statistics from the Indian Institute of Technology Kanpur, India, and an M.S. degree in Economics from Purdue University. His research interests include causality, reinforcement learning, optimization, and network science with applications to social networks and intelligent transportation.

Christopher J. Quinn (Member, IEEE) is currently an Assistant Professor in the Department of Computer Science at Iowa State University, Ames, IA 50011, USA. He received a B.S. degree in Engineering Physics from Cornell University, and M.S. and Ph.D. degrees in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign. His current research interests include machine learning, information theory, and network science, with applications to neuroscience and social networks.

Vaneet Aggarwal (Senior Member, IEEE) is currently a Full Professor at Purdue University, West Lafayette, IN 47907, USA. He received a B.Tech. degree from the Indian Institute of Technology Kanpur, India, and M.A. and Ph.D. degrees from Princeton University, all in Electrical Engineering. His current research interests include machine learning and its applications in networking, transportation, and quantum systems.

Appendix to the article, Umrawal, A. K., Quinn, C. J., and Aggarwal, V. (2023). A Community-Aware Framework for Social Influence Maximization. *IEEE Transactions on Emerging Topics in Computational Intelligence (TETCI).*

APPENDIX A
TABLE OF NOTATIONS

TABLE IV: Table of notations.

Symbol	Explanation
Ω	Ground set.
$\mathcal{O}$	Set of all subsets of Ω.
$G = (V, E)$	Directed graph.
$V = (v_1, \ldots, v_n)$	Set of vertices or nodes.
n	Size of V.
$E = (e_1, \ldots, e_n)$	Set of directed edges where $e_i, i = 1, \ldots, n$ are ordered pairs of nodes.
$p_{v,w}$	Weight of the edge $v \to w$.
∂v	Set of neighbors of node v.
$y_t^{(v)}$	Activation/state of node v at time t.
k	Budget.
$\sigma(S)$	Influence of a set S of nodes.
c	Number of communities.
`com-method`	Community detection method.
`sol-method`	Influence maximization method.
$\{G_1, \ldots G_c\}$	A partition of G with c sub-graphs that are $G_1, \ldots G_c$.
$\{V_1, \ldots V_c\}$	Set of sets of vertices for all sub-graphs in the partition $\{G_1, \ldots G_c\}$.
n_i	Size of $V_i, i = 1, ..., c$.
$\{E_1, \ldots E_c\}$	Set of sets of edges for all sub-graphs in the partition $\{G_1, \ldots G_c\}$.
Q	Modularity of a network-partition.
$S_{i,j}$	Best seed set of size j ($j = 1, ..., k$) from community i ($i = 1, ..., c$).
$\sigma_i(S_{i,j})$	Influence of $S_{i,j}$ within community i ($i = 1, ..., c$).
$\mathcal{S}_i$	Set of all candidate solutions from community i = $\{S_{i,j} : j = 1, ..., k\}$.
Σ_i	Influences of all candidate solutions from community i = $\{\sigma_i(S_{i,j}) : j = 1, ..., k\}$.
$\mathcal{S}$	Set of sets of all candidate solutions from all communities = $\{\mathcal{S}_i : i = 1, ..., c\}$.
Σ	Set of sets of influences of all candidate solutions from all communities = $\{\Sigma_i : i = 1, ..., c\}$.
S^*	Final solution of our framework.

APPENDIX B
AN ILLUSTRATIVE EXAMPLE OF PROGRESSIVE BUDGETING

In this section, we provide an illustrative example of progressive budgeting. After executing the Community-Detection and the Generate-Candidates steps of the proposed framework, we obtain the following output.

$$S_{i,j} = \text{Candidate set of size } j \text{ from community } i,$$
$$\sigma_{i,j} := \sigma_i(S_{i,j}) = \text{Influence of } S_{i,j} \text{ within community } i,$$
$$i = 1, \ldots, j = 1, \ldots, k.$$

Let the budget, $k = 4$. No. of communities, $c = 5$. The influences of different candidate sets within different communities are given in Table V(a). For every $i = 1, \ldots, c; j = 1, \ldots, k$, we calculate the marginal influences as $m_{i,j} := \sigma_i(S_{i,j}) - \sigma_i(S_{i,j-1})$, where $\sigma_i(S_{i,0}) = 0, \forall i$. The marginal influences for the influences given in Table V(a) are provided in Table V(b).

Influence

Community i	$\sigma_{i,1}$	$\sigma_{i,2}$	$\sigma_{i,3}$	$\sigma_{i,4}$
1	8	14	18	21
2	5	10	14	15
3	9	14	16	17
4	7	12	16	18
5	5	9	11	11

(a) Influences of candidate sets after step (ii).

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1	8	6	4	3
2	5	5	4	1
3	9	5	2	1
4	7	5	4	2
5	5	4	2	0

(b) Marginal Influences of candidate sets after step (ii).

TABLE V: An example input for progressive budgeting.

The progressive budgeting scheme for the example in Table V is explained in Table VI. At any iteration, the circled cells ($\textcircled{\cdot}$) are the ones whose maximum is to be obtained. An asterisk ($*$) is placed before the maximum value at the current iteration. The superscript(s) on any community label represents the nodes selected from that community (ordered based on the ordering in the corresponding candidate set obtained in step (ii)).

For the example we considered, the final seed set is $\{1^{1,2}, 3^1, 4^1\}$ which is equivalent to $S_{1,2} \cup S_{3,1} \cup S_{4,1}$.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1	⑧	6	4	3
2	⑤	5	4	1
3^1	$*$ ⑨	5	2	1
4	⑦	5	4	2
5	⑤	4	2	0

(a) Iteration 1: Allocating the first unit.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1^1	$*$ ⑧	6	4	3
2	⑤	5	4	1
3^1	9	⑤	2	1
4	⑦	5	4	2
5	⑤	4	2	0

(b) Iteration 2: Allocating the second unit.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1^1	8	⑥	4	3
2	⑤	5	4	1
3^1	9	⑤	2	1
4^1	$*$ ⑦	5	4	2
5	⑤	4	2	0

(c) Iteration 3: Allocating the third unit.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
$1^{1,2}$	8	$*$ ⑥	4	3
2	⑤	5	4	1
3^1	9	⑤	2	1
4^1	7	⑤	4	2
5	⑤	4	2	0

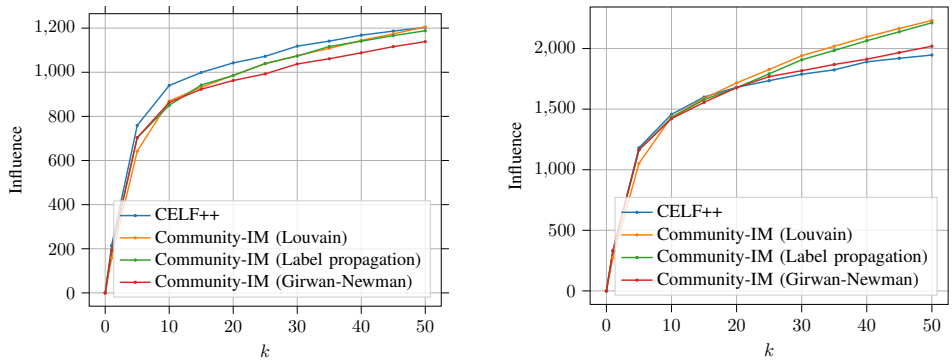
(d) Iteration 4: Allocating the fourth unit.

TABLE VI: An illustration of progressive budgeting.

APPENDIX C
EFFECT OF THE COMMUNITY STRUCTURE ON THE PERFORMANCE OF COMMUNITY-IM

For the Facebook network under different diffusion models and weighted cascade edge-weight models using different community detection methods,

- Figure 4 shows the influences of chosen seed sets using different algorithms for different values of k.
- Table VII shows the modularity score, no. of communities, the influence of the seed set of size 50 chosen using CELF++ and Community-IM, and their ratios.
- Table VIII shows the modularity score, no. of communities, the empirical run-times of CELF++ and Community-IM for $k = 50$. The run-times for Degree and Degree-Discounts in all cases were just a few seconds.



(a) Independent cascade model

(b) Linear threshold model

Fig. 4: influence vs. k for the Facebook network under different diffusion models and weighted cascade edge-weight model.

TABLE VII: Comparison of influences for $k = 50$ for the Facebook network under weighted cascade edge-weight model for different community detection methods.

Diffusion model	Community detection method	No. of communities	Modularity score	CELF++	Community-IM
Independent cascade	Louvain	18	0.8678	1203.42	1205.12
	Label propagation	11	0.7368	1203.42	1188.16
	Girvan-Newman	2	0.0439	1203.42	1138.93
Linear threshold	Louvain	18	0.8304	1946.41	2231.34
	Label propagation	11	0.7368	1946.41	2212.57
	Girvan-Newman	2	0.0439	1946.41	2019.08

Table VII shows that for the Facebook network, the influence of the chosen seed set of size 50 using Community-IM is approximately equal to the same for CELF++ for different choices of community detection methods under different diffusion models and weighted cascade edge-weight model. Moreover, Table VII and Table VIII show that the performance of Community-IM compared to CELF++ in terms of influence and run-time improves as the modularity of the partition and

TABLE VIII: Comparison of run-times (in seconds) for $k = 50$ for the Facebook network under weighted cascade edge-weight model for different community detection methods.

Diffusion model	Community detection method	No. of communities	Modularity score	CELF++	Community-IM
Independent cascade	Louvain	18	0.8678	14077	3069
	Label propagation	11	0.7368	14077	4068
	Girvan-Newman	2	0.0439	14077	14221
Linear threshold	Louvain	18	0.8304	38968	7224
	Label propagation	11	0.7368	38968	12961
	Girvan-Newman	2	0.0439	38968	34606

the number of communities increase. From Table VII and Table VIII, we also note that the Louvain method is the best choice of community detection method and the Girwan-Newman method performs the worst. The Louvain method partitions the graph into 18 communities with the largest community having 523 nodes which are approximately 10% of the size of the entire network. Hence, Community-IM does not come across any giant component and finishes faster. Contrary to this, the Girwan-Newman algorithm partitions the network into just two communities with the largest community having 3,833 nodes which are very close to the size of the entire network hence the Community-IM method takes a lot of time in finishing.