Università degli Studi Roma Tre
Dipartimento di Matematica e Fisica
Corso di Laurea Magistrale in Scienze Computazionali

Sintesi della Tesi di Laurea Magistrale in Scienze Computazionali

Community-based Influence Maximization

Relatori:

Prof. Marco Liverani
Dott. Stefano Guarino

Candidato:

Daniele Salierno

Anno Accademico 2018/2019

Introduction

The quick technological advancement of the last decade brought us devices that are faster, cheaper and pervasive in everyday life, transforming social relations and interactions between individuals.

In particular, the growth of social networks has been unmatched: almost everyone has an account on some platform like Facebook, Instagram or Twitter, and often on all of them. Their great success is probably due to the possibility to share events and to remain connected easily with friends.

The expansion of socials leads the society, in particular, the younger generations, to consider them the privileged communication channels. Now people follow one another and see what others share, often in real-time.

Following this social development, social studies emerged interested in keeping pace with innovations: the analysis of such networks has become increasingly important and challenging, with the immense growth in the size of data sets. At first, it seemed crucial to study macroscopic properties of a network, as connectedness, flow and paths, but then the primary goal of the research has been shifting towards more mesoscopic properties, as motifs and communities, and, more recently, on microscopic aspects, as the importance of single nodes.

In social networks, single nodes are people and links are connections between them. Analysing a social network at this scale means detecting those nodes that are more important within the graph. However, what does *important* mean, exactly? Clarifying what we consider important in a specific application setting is the first step towards quantifying importance, and, ultimately, understanding how to compute and compare importance in practice.

We can roughly divide the ways of quantifying importance in two categories, criticality and vitality:

- A node is *critical*, if it is essential for the cohesiveness of the graph, i.e., if its deletion can endanger the connectedness of the graph, or cause a significant worsening in the communication properties of the network (e.g. extending the average length of minimum paths).
- A node is *vital* if it can actively and efficiently inject information within the network, taking advantage of its position or its neighbourhood. We often call this type of vertices "influencers", "spreaders" or "seeds".

Based on this primary distinction, two different problems arise: the *Critical Node Detection Problem* (CNDP) and the *Influence Maximisation Problem* (IMP). Within each of these categories, different metrics may be used to target a specific connotation of, respectively, criticality and vitality; for instance, a node can be critical either for safeguarding connectedness or for shortening connections (or for both).

An especially thorny aspect when dealing with importance is to distinguish between *individual* and *joint* importance. In many optimisation problems, selecting a single optimal instance is a relatively easy task, whereas choosing several objects that perform optimally together is much more troublesome. Albeit defining a suitable metrics may be itself a non-trivial task, if we aim at finding the best element in a dataset with respect to any chosen metrics, it suffices to compute that metrics for all elements and to select the one with the highest score. Unfortunately, for joint optimality, a similar greedy algorithm is typically either unfeasible or underperforming.

In the case of network criticality/vitality, identifying an exact solution to the problem of joint importance may require more-than-polynomial time, at least, on graphs worth considering. At a high level, this is because the importance of a node is rarely independent of that of others, so that choosing a set of nodes that work well together to achieve an objective is a different problem than selecting as many nodes that work well when considered alone. For instance, the "areas of importance" of a set of individually relevant nodes may overlap, thus decreasing their collective value, or predicting the combined actions of a given set of nodes may be impossible.

Theoretically, an optimal set of nodes could be found using a greedy algorithm that recomputes the ranking each time a new node is selected, in such a way to identify, at each step, the node that provides the best marginal gain. In some cases, this can also be done in practice to tackle and solve the problem, even with performance guarantees ([KKT03]).

However, in most cases, frequent recalculations of the metrics are computationally heavy and scale poorly with the input size. Some fix can make them acceptable solutions even on mid-range graphs, but they will ultimately fail when considering large networks.

In this thesis, we focus on the Influence Maximization Problem. While critical nodes usually come in sets (a real network can be hardly disconnected by removing a single vertex), vital nodes can be considered alone, since in many situations a single node can provide coverage for a significant fraction of the network. As a consequence, the literature seems to lack a detailed analysis of the difference between detecting a single influencer and concurrently selecting multiple ones.

Our aim is making a step towards understanding whether considering multiple influencers at once may be convenient with respect to sticking with a single-target approach.

In practice, we will elaborate on the following rationale. When we "select a node" in a real-life IMP instance, we ultimately aim at controlling it to use it as a source (of news, ads, etc.), and this comes at a cost, either in terms of money or resources. If we focus on a set of less influential nodes, which are most likely easier to control, can we achieve a result that is, proportionally, better than investing all of our resources on a single influencer? Moreover, is there a relatively simple strategy for choosing these "mini-influencers" that may be expected to maximise our profit, i.e., the ratio of gained influence over cost? For instance, using popular influencers on social networks for advertising campaigns is extremely expensive. If we rely on a team of less popular users, can we obtain similar/better results at a lower/comparable cost?

Formally, let us denote with f_u, f_S the value of the single best spreader u and the value of a chosen subset S, and c_u, c_S the cost of the single node and the subset, respectively. We can consider the proportions $f_{\%} = \frac{f_S}{f_u}$ and $c_{\%} = \frac{c_S}{c_u}$. We are then interested in $g = \frac{f_{\%}}{c_{\%}}$: if this quantity is at least 1, we consider convenient to select multiple influencers. When we need to compare two subsets, we consider better the one with the higher gain.

This process hides the actual value f_S, which in real situations can be (and will probably be) lesser than f_u, even a lot lesser. However, calculating f_u is an easy task, so it is always convenient and fast to consider it. Besides this, in our comparison process, it is necessary to know which is the single best spreader. Having both values is important to understand how far the chosen set is from the best spreader in terms of coverage and how much we gained.

Real networks, especially social networks, are sparse but mostly organized in sub-units, called communities or modules, which have stronger connections and denser edges. Detecting communities is of great importance to factorize a usually large graph into smaller units and to understand the general structure of the graph. Our strategy for detecting the influencers is based on communities.

However the task of detecting modules is not simple, both because of the ambiguity in the definition of what precisely a community is, and in the actual partitioning: besides the exponential number of possible partitions, there are other complications, such as the possible overlapping of some communities, i.e., the possibility that a node belongs to different communities at the same time.

Intuitively, a community (or module) is a subgraph more clustered within the graph, where the probability that an edge links two randomly chosen nodes is higher than on the whole graph. To extract these subgraphs, one needs to choose an objective function that describes this intuition.

We chose the optimisation of modularity.

Definition 1. Given a graph $G(V, E)$ and a partition of its nodes in s subsets, we call *modularity* of a module the quantity

$$Q_i = \frac{m_i}{m} - \left(\frac{d_i}{2m}\right)^2$$

and modularity of the partition the sum over the modules:

$$Q = \sum_{i=1}^{s} Q_i$$

where $m = |E(G)|$, $m_i = |G[V_i](E)|$ and $d_i = \sum_{v \in V_i} \deg v$.

The first term of the summand is the fraction of edges inside the module; the second term represents instead the expected fraction of links that would be inside the module, if they were located at random, under the constraint of conserving each node's degree.

We can understand this by referring to the configuration model ([Van13]). We imagine that we have a graph and that we cut each edge in halves, then we rewire uniformly at random the halves into full edges. Within the module, we have d_i half edges (where their total number is $2m$).

For each module, Q_i can be anywhere in $[-1, 1]$. If it is positive, it means that the module contains more edges than it would if their position were random; hence, it is more likely an actual cluster within the graph. We can conclude that a subgraph S with m_s internal edges and total degree d_s is a module if

$$\frac{m_s}{m} - \left(\frac{d_s}{2m}\right)^2 > 0$$

To select the multiple seeds, we compute a partition of the graph into communities with Louvain algorithm (proposed in [Blo+08]), which unfolds a complete hierarchical structure, through local optimization of modularity. Then we choose into each community a node as a seed, in such a way that the sum of their costs does not exceed the cost of the single spreader already selected. This heuristic approach, without a given number of influencers to select, is a variant of IMP not yet found in the literature.

We applied our community-based algorithm on several synthetic random graphs and real large networks. Results are, mostly, in favour of a multiple selection approach. Our algorithm has no approximation guarantee; hence, we expect that more sophisticated algorithms can achieve a better gain.

Influence Maximization Problem

The Influence Maximisation Problem was initially arisen in [DR01] in the context of viral marketing, and, due to its practical importance, has received much attention. In general, we want to detect a set of nodes in order to achieve a maximum spread of whatever we are going to spread, be it information, the adoption of a product or a behaviour, or even a disease.

Since we want to spread something, it seems reasonable to consider important those nodes that have more influence over the others. To evaluate the influence capacity of a selected set in such a situation, in which a process developing in time is involved, it is more convenient to choose a dynamic model ([PMM18]). Thus the spreading of information (often called *cascade*) can be simulated as a real process from a collective point of view.

In this type of applications, we are more interested in maximising the influence, given a spendable budget, than in minimising the cost in order to obtain a specific effect. Thus, the main goal of IMP is to find a set of cardinality k with maximum influence over the graph.

To quantify the information diffusion, we formally define the influence spread, as in [Li+18].

Definition 2. Given a graph G and a diffusion model M, the *influence spread* is a function $\sigma : \mathcal{P}(V) \to \mathbb{R}_{\geq 0}$ that maps each subset S in the expected number of influenced node for a diffusion process run under the model M using S as seeds set.

We chose the Independent Cascade Model, that describes accurately the word-of-mouth process viral marketing is based on.

The Independent Cascade Model and its variants are stochastic models, which means that the set of final active nodes can change in each simulation of the cascade.

We assign to every edge (u, v) an activation probability p_{uv}. During the simulation, when node u is first activated (suppose at time $t-1$), it has a single chance to activate each inactive neighbour v with probability p_{uv} independently of the story of the process so far. If u succeeds, v becomes active at time t, but if it fails, u cannot make further attempts to activate v. If more than one neighbour of a vertex v is activated at time $t-1$, each of them can attempt to activate v in time u (attempts are considered in arbitrary order).

In the most general case, the probability p_{uv} can depend on several factors, such as the number of steps from the source, the number of nodes that have already failed to activate v or time passed from the diffusion beginning.

A generalization can be obtained by allowing the probability to depend on the exact set A of neighbours that have already tried (and failed) to activate the node. The activation probabilities will then be $p_{uv}(A)$, where v is the target node and u the active one. The independent model is the special case of $p_{uv}(A) = p_{uv}$, constant and independent of A.

It is convenient to consider only functions that are *order-independent*, i.e., such that if r neighbours u_i try to activate a node v, the probability that v is active after the last try does not depend on the specific order in which attempts are made. Formally if $A = \{u_i, i \leq r\}$ and τ is an arbitrary permutation on A, we require that

$$\prod_{i=1}^{r} \left(1 - p_{u_{\pi(1)}v}(\{u_{\pi(i)} \mid i \leq r-1\})\right) = \prod_{i=1}^{r} \left(1 - p_{u_{\tau(1)}v}(\{u_{\tau(i)} \mid i \leq r-1\})\right)$$

The general cascade model makes influence maximisation intractable. A more tractable variant introduces a behaviour of diminishing return. This variant is called Decreasing Cascade Model, and the additional request is that $p_{uv}(S) \geq p_{uv}(T)$ if $S \subseteq T$, that is, the more attempts have been made to activate v, the more difficult it will be to influence the node.

Influence Maximization. Thus, the IMP receives in input a graph G, a diffusion model M and an integer k, and selects a set of k nodes in order to maximize the influence spread σ. The output is

$$\underset{S \subseteq V : |S| = k}{\operatorname{argmax}} \ \sigma(S)$$

A natural generalisation consists in taking as input a graph with nodes weighted with a cost and interpreting the integer k as the total cost of the nodes, instead of their number.

For IMP, there are some known hardness results ([KKT03]).

Theorem 3. *The Influence Maximisation Problem is NP-Hard to approximate within a factor of* $1 - \frac{1}{\varepsilon}$ *under the Independent Cascade Model for any* $\varepsilon > 0$.

Theorem 4. *The Influence Maximization Problem is NP-Hard to approximate within a factor of* $n^{1-\varepsilon}$ *under the general cascade model for any* $\varepsilon > 0$.

Thus, the IMP is at least NP-Complete under the independent cascade dynamic, but in the general formulation, it is also highly intractable, while under the other models is well approximable, for instance with the greedy algorithm proposed in [KKT03].

Theorem 5. *Let* σ *be a non-negative monotone and submodular function. Then the greedy algorithm produces a set* S *such that* $\sigma(S) \geq (1 - \frac{1}{e})\sigma(\bar{S})$, *where* $\bar{S}$ *is the optimal solution.*

The greedy algorithm is a simple hill-climbing: starting from the empty set, at each step adds to the seed set the node u that maximises the marginal gain $\sigma(S \cup \{u\}) - \sigma(S)$.

The influence spread cannot be computed exactly in polynomial time, both under the ICM ([WCW12]), but it is possible to obtain arbitrarily good approximations by simulating the diffusion sufficiently many times ([KKT03]).

Proposition 6. *If the diffusion process starting from* S *is simulated independently at least*

$$\Omega\left(\frac{n^2}{\varepsilon^2} \ln\left(\frac{1}{\delta}\right)\right)$$

times, then the average number of activated nodes over these simulations is a $(1 \pm \varepsilon)$*-approximation of* $\sigma(S)$ *with probability at least* $1 - \delta$.

Despite the excellent approximation result, the greedy algorithm runs in time $\mathcal{O}(knT)$, where T is the time needed to approximate the influence spread of a set. This complexity makes the algorithm not scalable on large graphs, which, speaking of social networks graphs with millions of nodes, is a significant drawback.

Comparing Influence Capabilities

We aim to compare the influence capabilities of a single high-degree vertex against several less important nodes. The fundamental question that arises this comparison is if, in a viral marketing application, it is more convenient to advertise on a social network a product through a single influencer or several.

Algorithm 1: Comparison Algorithm

```
  Input   : Graph $G$
  Output : Costs and values for single and multiple influencers
  /* Initialize Weights and Costs */
1 $\forall e = uv \in E$ compute $p_{uv}$
2 $\forall u \in V$ compute $c_u$
  /* Single Influencer Phase */
3 $u \leftarrow \operatorname{argmax}_u \deg u$
4 Simulate IC for $u$
5 Save cost and value of $u$
  /* Multiple Influencers Phase */
6 Compute partition $P$ of $V$ through Louvain
7 Select multiple influencers over $P$ and put them in $S$
8 Simulate IC for $S$
9 Save cost and value of $S$
```

Algorithm 1 shows the process of computing costs and values for a single influencer and for a multiple seeds selection, enabling us to compare them.

First of all, we need to compute the activation probabilities, $p_{uv} = \mathbb{P}(u \to v)$ for each $uv \in E$, and the cost of controlling each vertex, c_u.

The first phase of the algorithm selects the node with the highest degree as the best probable influencer and run an IC simulation starting from it. Each simulation is run a fixed amount of times, and we calculate the empirical average to evaluate the number of activated nodes.

The simulation of a single cascade follows algorithm 2, that is essentially a breadth-first visit, in which nodes are added to the queue only when activated. We know that a visit runs in $\mathcal{O}(n + m)$, so a single simulation costs at most the same time.

We know ([Kit+10]) that degree is not an accurate indicator of the popularity of a node in a network, but it is a good one: a node with significantly fewer neighbours is a less popular node. Because of this, we simulate the single influencer diffusion for a fixed number of high degree nodes (eight by default), and choose the best one. Being the graph scale-free, we know that the next node in degree order has significantly fewer neighbours than the most popular one. To select the candidates we do not actually sort vertices in degree order, but we compute the minimum degree that a node

Algorithm 2: Independent Cascade

```
Input   : Graph $G$, set of seeds $S$
Output  : Number of activated nodes
/* Initialize structures */
$Q \leftarrow \emptyset$, $a \leftarrow 0$      /* Queue and number of activated nodes */
foreach $v \in V$ do
    if $v \in S$ then
        $Q \leftarrow Q \cup \{v\}$
        Activated($v$) $\leftarrow$ true
        $a \leftarrow a + 1$
    else
        Activated($v$) $\leftarrow$ false
    end
end
/* Compute Spreading */
while $Q \neq \emptyset$ do
    Pop $u$ from $Q$
    foreach $v \in N(u)$ do
        if $v$ is not active and $u$ succeeds in activating it then
            Activated($v$) $\leftarrow$ true
            $Q \leftarrow Q \cup \{v\}$
            $a \leftarrow a + 1$
        end
    end
end
return $a$
```

must have to be a candidate with algorithm 3.

Algorithm 3: Degree-Based Single Influencer Selection

```
Input  : Graph $G$ with $n$ nodes, maximum number of candidates $h$
Output : Minimum degree $d$ to be a candidate
/* Initialize an array for degree distribution */
$i \leftarrow 0$
for $i \leq n$ do
    $D_i \leftarrow 0$
    $i \leftarrow i + 1$
end
/* Compute degree distribution */
foreach $v \in G$ do
    $D_{\deg v} \leftarrow D_{\deg v} + 1$
end
/* Compute minimum degree */
$d \leftarrow 0$
while $n > h$ do
    $n \leftarrow n - D_i$
    $d \leftarrow d + 1$
end
return $d$
```

After finding the best single influencer, the second phase of the computation begins, the search for multiple spreaders.

Firstly, the Louvain algorithm detects the hierarchical structure and partitions the graph. This algorithm appears to run in nearly linear time on sparse data ([Blo+08]), but to this, we need to add the time needed to read through the output and load to which module on each level each node belongs. This additional scan takes $\mathcal{O}(nl)$ operations, where l is the number of levels. For our instances, l was less than a half dozen, even on large graphs.

Given that, for each module, the algorithm adds to the seed set the node with maximum degree such that its cost is at most the cost of the single influencer over the number of communities. The condition on the cost is necessary in order to avoid exceeding the single influencer's cost.

It is possible to make this selection with a single loop on vertices followed by another loop on communities. During the first loop, we compute the vertex with the highest degree for all the communities, using a temporary array. This process (described in algorithm 4) requires an additional $\mathcal{O}(k)$ space but takes considerably less time than using only a variable and finding the node for one community at a time (which takes $\mathcal{O}(nk)$, where k is the number of communities). We know, however, from [FB07], that modularity has a resolution limit. Thus it is not able to solve small communities, which, in this case, keeps low the extra memory usage, at least for the last level of the hierarchy.

Algorithm 4: Multiple Selection

```
Input  : Graph $G$, partition $P$ into communities, $\Delta$, maximum
         degree in $G$
Output : A set $S$ of multiple spreaders, one for each module of $P$
   /* Initialize structures */
1  $S = \emptyset$
2  foreach $c \in P$ do
3      $\text{best}_c \leftarrow NULL$
4  end
   /* Loops on nodes */
5  foreach $v \in V$ do
6      $c \leftarrow \text{Community}(v)$
7      if $\deg v < \frac{\Delta}{|P|}$ and $(best_c = NULL$ or $\deg v > \deg(best_c))$ then
8          $\text{best}_c = v$
9      end
10 end
   /* Seeds selection */
11 foreach $c \in P$ do
12     $S \leftarrow S \cup \{\text{best}_c\}$
13 end
```

On top of all these considerations, it is possible to run the community-based multiple spreader selection for all the partitions of the hierarchy found by Louvain algorithm. This process takes a little more time but can be very rewarding, allowing for a better gain.

Lastly, we observe that an algorithm should run within time $\mathcal{O}(n \log n)$, in order to be successfully used on massive inputs, such as graphs with millions of edges. This scalability property is necessary when working with massive datasets, such as in applications to viral marketing.

All the process described runs under this time, assuming that $m \propto n$, which is true if the graph is sparse.

We recall the computational cost of the overall process. For computing the activation probability, we need m operations; for computing the community, the Louvain algorithm appears to run in nearly linear time on sparse data ([Blo+08]); during the single influencer phase, we compute a maximum among the n vertices (or we select h candidates with algorithm 3, that is also linear), and we simulate the independent cascade through a breadth-first search ($n + m$ operations in the worst case of massive activation); lastly, after

Louvain algorithm, we need n operations to add to the seed set a node for each community.

The total time is then the sum of all these operations (we omitted constants), $m + T + n + (n + m) + n$, where the important terms are m and T, that is the running time of Louvain algorithm (nearly linear on sparse data). Considering that the sparseness of the graph bounds m, the algorithm runs in nearly linear time.

Conclusions

The Influence Maximization Problem is of particular interest in applications in which one wants to spread information (or the like) within a graph, for instance, in viral marketing.

Since in many situations a single influencer provides coverage for a significant part of the network, for solving the IMP is often faster and easier to find the best single spreader than trying to coordinate many. Thus, the literature seems to lack an analysis of the advantages and disadvantages of detecting a single seed against selecting multiple ones.

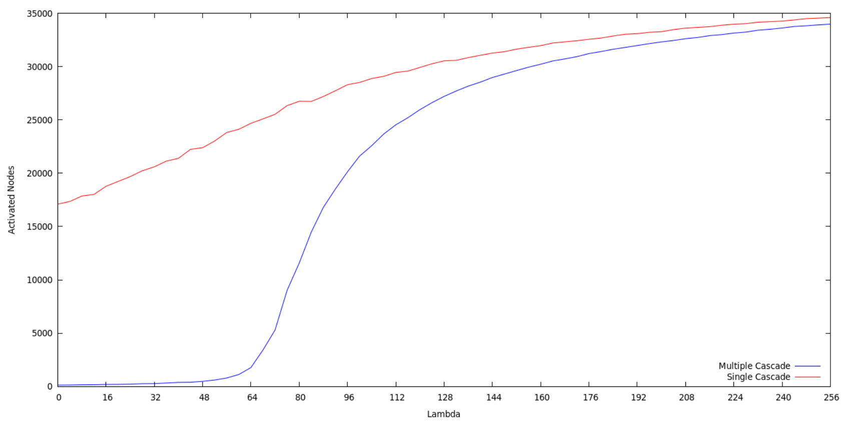
We defined a measure, the average number of activated nodes after an independent cascade starting from the seeds, and a cost, the vertex degree. Our aim is making a step towards understanding whether, proportionally, it is possible to achieve a better result, selecting many cheaper seeds instead of one more expensive.

We weighted edges with an activation probability, computed with dependence on the degree of both ends: the higher the degree of the already active node is with respect to the others, the higher the probability will be. The exact formula used is

$$p_{uv} = \max\left(0, \frac{\deg u - \deg v + \lambda}{\Delta}\right)$$

In this computation, we introduced a parameter, λ, representing the maximum degree distance between two nodes, such that the less popular of the two can activate the other during a cascade. We showed that the number of activated nodes in the multiple spreaders cascade has a phase-change-like phenomenon as λ varies, or at least a sudden growth, before which a negligible part of the graph is active, but after which a considerable fraction of it is covered. Figure 1 shows the average number of activated nodes as λ varies for one of our instances.

Moreover, we developed a simple way of choosing these multiple seeds. For this strategy, we took advantage of the fact that, in real networks, and, in particular, social networks, nodes tend to aggregate themselves in dense communities. We use the Louvain Algorithm to partition the graph into such modules and then we choose the node with the highest degree as an influencer for each community.



(a) socfb-Penn94

Figure 1: Number of Activated Nodes

This heuristic approach has no performance guarantees on the optimality of the solution, but it is incredibly fast, processing a graph with three million nodes and twenty-four million edges in less than ten minutes. Moreover, it does not require to know beforehand how many influencers there will be.

The simulations we made with our implementation of this community-based multiple influencers detection algorithm showed that it is often possible to obtain, proportionally, a broader coverage, spending considerably less using the multiple-seeds approach. Sometimes this gain was marginal, or even absent, but, usually, it was noticeable. Our simulations were run on a synthetic graph based on the Barabási-Albert model and several real-world graphs from the Facebook social network.

For better comparison, we used the same parameters for all the instances considered: in particular, we set $\lambda = 128$ and the number of possible single influencer candidates to 8.

Table 1 shows the general characteristics of the considered graphs used in the simulations, while table 2 summarises the results of the simulations. As we said in the previous section, it is possible to run the multiple spreaders selection several times, one for each level of the hierarchy found by Louvain algorithm. The second table shows both the last level and the best level results.

On the first table, $|V|$ is the number of vertices, $|E|$ of edges, $|L|$ of hierarchical levels and $|C_b|, |C_l|$ respectively the number of communities in the best and the last level. On the other table, there are f_S, average number of activated nodes with seed set S, f_u for the single spreader u, c_S and c_u, costs of the seed set S and of the vertex u; T is the running time when only the selection on last level is considered, while T_a is the total running time for the selection over all levels. Moreover, $f_\% = \frac{f_S}{f_u}$, $c_\% = \frac{c_S}{c_u}$ and $g = \frac{f_\%}{c_\%}$.

Our next aim is to explore the comparison further, firstly refining the

Graph Name	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert L\rvert$	$\lvert C_l\rvert$	$\lvert C_b\rvert$
BA5000m1	5000	4999	4	84	84
socfb-UF	35111	1465654	3	15	123
socfb-Penn94	41536	1362220	3	20	185
socfb-OR	63392	816886	4	53	53
socfb-A-anon	3097165	23667394	5	372	375

Table 1: Graph characteristics

	Last Level				Best Level			
Graph	$\frac{f_S}{f_u}$	$\frac{c_S}{c_u}$	g	T	$\frac{f_S}{f_u}$	$\frac{c_S}{c_u}$	g_a	T_a
BA5000m1	0.749	0.600	1.25	0	0.749	0.600	1.25	0
socfb-UF	0.714	0.924	0.77	18	0.678	0.710	0.95	21
socfb-Penn94	0.895	0.894	1.00	21	0.897	0.608	1.48	23
socfb-OR	0.992	0.472	2.10	13	0.992	0.472	2.10	15
socfb-A-anon	0.251	0.441	0.57	463	0.334	0.420	0.79	543

Table 2: Simulation results

selection of seed by calculating the k-shell value for all nodes. The k-shells (or cores) are an indicator of both the centrality and the degree of a node. The idea is to choose for each community the node with the highest degree among those with maximum k-coreness. This further computation could increase the spreading value for the multiple influencers, but we need to verify that the gain is worth the extra time required.

Besides this refining, we want to implement other types of dynamic models, in particular, the decreasing cascade. In the IC model considered, the probability p_{uv} is independent of the history of the process. However, the inclination of a node of being influenced may change, decreasing for each of its neighbours that have already attempted and failed to influence it.

About the results themselves, we would like to analyse two facts. Firstly, it can be interesting to study how the gain/loss varies with λ. Looking at how the number of activated nodes does, we would expect a phase-change phenomenon for the gain, too. For instance, for the A-anon graph, setting $\lambda = 160$ hields a gain $g \approx 2.00$. Then, we want to identify which structural property is more favourable for our algorithm. For instance, we expect that a large number of little communities could easily disrupt our approach since the algorithm would waste resources in selecting a multitude of nodes with a minimal degree.

Moreover, we do not know beforehand how many multiple influencers there will be, but we could make a guess and use a known algorithm, for instance, the greedy hill-climbing, that has some performance guarantee, in order to select the multiple seeds and make a comparison with our heuristic.

Lastly, and more importantly, we know that many social networks, like Instagram and Twitter, are naturally modelled as *directed* graphs. It is easy, from a directed graph, to derive the underlying undirected one, but in this process, we forget about the follower-leader relationship between nodes. Considering directions on edges makes the topic more complicated: for instance, it is not clear what a directed community is (while, at least intuitively, an undirected community is easy to understand). Moreover, the Louvain algorithm, in its original implementation, is not able to deal with directed graphs. Nevertheless, directed social networks are at least as common as undirected ones, and an attractive target for IMP instances, making of primary importance to adapt our comparison algorithm on directed networks.

Bibliography

[Aru+09] Ashwin Arulselvan et al. “Detecting critical nodes in sparse graphs”. In: *Computers & Operations Research* 36.7 (2009), pp. 2193–2200.

[BA99] Albert-László Barabási and Réka Albert. “Emergence of scaling in random networks”. In: *Science* 286.5439 (1999), pp. 509–512.

[BJP18] Suman Banerjee, Mamata Jenamani, and Dilip Kumar Pratihar. “A Survey on Influence Maximization in a Social Network”. In: *arXiv preprint arXiv:1808.05502* (2018).

[Blo+08] Vincent D. Blondel et al. “Fast unfolding of communities in large networks”. In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[BM08] John Adrian Bondy and Uppaluri Siva Ramachandra Murty. *Graph Theory*. Springer, 2008.

[Bor06] Stephen P. Borgatti. “Identifying sets of key players in a social network”. In: *Computational & Mathematical Organization Theory* 12.1 (2006), pp. 21–34.

[Boz+16] Arastoo Bozorgi et al. “INCIM: A community-based algorithm for influence maximization problem under the linear threshold model”. In: *Information Processing & Management* 52.6 (2016), pp. 1188–1199.

[Bra+06] Ulrik Brandes et al. “Maximizing modularity is hard”. In: *arXiv preprint physics/0608255* (2006).

[Bra+07] Ulrik Brandes et al. “On finding graph clusterings with maximum modularity”. In: *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2007, pp. 121–132.

[Cal+02] Guido Caldarelli et al. “Scale-free networks from varying vertex intrinsic fitness”. In: *Physical review letters* 89.25 (2002), p. 258702.

[Cor+10] Thomas H. Cormen et al. *Introduzione agli algoritmi e strutture dati 3/ed.* McGraw-Hill, 2010.

[CYZ10] Wei Chen, Yifei Yuan, and Li Zhang. "Scalable influence maximization in social networks under the linear threshold model". In: *2010 IEEE international conference on data mining*. IEEE. 2010, pp. 88–97.

[DB02] Zoltán Dezső and Albert-László Barabási. "Halting viruses in scale-free networks". In: *Physical Review E* 65.5 (2002), p. 055103.

[DR01] Pedro Domingos and Matt 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.

[ER59] P Erdős and A Rényi. "On random graphs I". In: *Publ. Math. Debrecen* 6 (1959), pp. 290–297.

[FB07] Santo Fortunato and Marc Barthelemy. "Resolution limit in community detection". In: *Proceedings of the National Academy of Sciences* 104.1 (2007), pp. 36–41.

[FC12] Santo Fortunato and Claudio Castellano. "Community structure in graphs". In: *Computational Complexity: Theory, Techniques, and Applications* (2012), pp. 490–512.

[GN02] Michelle Girvan and Mark E.J. Newman. "Community structure in social and biological networks". In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[HFC15] Jia-Lin He, Yan Fu, and Duan-Bing Chen. "A novel top-k strategy for influence maximization in complex networks with community structure". In: *PloS one* 10.12 (2015), e0145283.

[Ker+88] Brian W. Kernighan et al. *The C programming language*. Vol. 2. prentice-Hall Englewood Cliffs, NJ, 1988.

[Kit+10] Maksim Kitsak et al. "Identification of influential spreaders in complex networks". In: *Nature physics* 6.11 (2010), p. 888.

[KKT03] David Kempe, Jon Kleinberg, and Éva 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.

[LFH10] Andrea Landherr, Bettina Friedl, and Julia Heidemann. "A critical review of centrality measures in social networks". In: *Business & Information Systems Engineering* 2.6 (2010), pp. 371–385.

[LGC13] Etienne Lefebvre, Jean-Loup Guillaume, and Federico Campigotto. *Louvain's algorithm*. 2013. URL: `https://sourceforge.net/projects/louvain/`.

[Li+18] Yuchen Li et al. “Influence maximization on social graphs: A survey”. In: *IEEE Transactions on Knowledge and Data Engineering* 30.10 (2018), pp. 1852–1872.

[LLM10] Jure Leskovec, Kevin J. Lang, and Michael Mahoney. “Empirical comparison of algorithms for network community detection”. In: *Proceedings of the 19th international conference on World wide web*. ACM. 2010, pp. 631–640.

[LTK16] Mohammed Lalou, Mohammed Amin Tahraoui, and Hamamache Kheddouci. “Component-cardinality-constrained critical node problem in graphs”. In: *Discrete Applied Mathematics* 210 (2016), pp. 150–163.

[LTK18] Mohammed Lalou, Mohammed Amin Tahraoui, and Hamamache Kheddouci. “The critical node detection problem in networks: a survey”. In: *Computer Science Review* 28 (2018), pp. 92–117.

[Lü+16] Linyuan Lü et al. “Vital nodes identification in complex networks”. In: *Physics Reports* 650 (2016), pp. 1–63.

[New06] Mark E.J. Newman. “Modularity and community structure in networks”. In: *Proceedings of the national academy of sciences* 103.23 (2006), pp. 8577–8582.

[NG04] Mark E.J. Newman and Michelle Girvan. “Finding and evaluating community structure in networks”. In: *Physical review E* 69.2 (2004), p. 026113.

[Ngu+17] Hung T. Nguyen et al. “Social influence spectrum at scale: Near-optimal solutions for multiple budgets at once”. In: *ACM Transactions on Information Systems (TOIS)* 36.2 (2017), p. 14.

[NN10] Ramasuri Narayanam and Yadati Narahari. “A shapley value-based approach to discover influential nodes in social networks”. In: *IEEE Transactions on Automation Science and Engineering* 8.1 (2010), pp. 130–147.

[PG12] Lorenzo Pantieri and Tommaso Gordini. “L’arte di scrivere con LaTeX, 2012”. In: *URL http://www. lorenzopantieri. net* (2012).

[PMM18] Sen Pei, Flaviano Morone, and Hernán A. Makse. “Theories for influencer identification in complex networks”. In: *Complex Spreading Phenomena in Social Systems*. Springer, 2018, pp. 125–148.

[RA15] Ryan A. Rossi and Nesreen K. Ahmed. “The Network Data Repository with Interactive Graph Analytics and Visualization”. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015. URL: `http://networkrepository.com`.

[SS12] Siqian Shen and J. Cole Smith. “Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs”. In: *Networks* 60.2 (2012), pp. 103–119.

[Van13] Remco Van Der Hofstad. *Random graphs and complex networks*. Vol. 1. Cambridge University Press, 2013. URL: `https://www.win.tue.nl/~rhofstad`.

[Ven12] Mario Ventresca. “Global search algorithms using a combinatorial unranking-based problem representation for the critical node detection problem”. In: *Computers & Operations Research* 39.11 (2012), pp. 2763–2775.

[WCW12] Chi Wang, Wei Chen, and Yajun Wang. “Scalable influence maximization for independent cascade model in large-scale social networks”. In: *Data Mining and Knowledge Discovery* 25.3 (2012), pp. 545–576.

[WS98] Duncan J. Watts and Steven H. Strogatz. “Collective dynamics of ‘small-world’networks”. In: *Nature* 393.6684 (1998), p. 440.

[Yan81] Mihalis Yannakakis. “Node-deletion problems on bipartite graphs”. In: *SIAM Journal on Computing* 10.2 (1981), pp. 310–327.

[ZH17] Yangming Zhou and Jin-Kao Hao. “A fast heuristic algorithm for the critical node problem”. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM. 2017, pp. 121–122.