LEBANESE AMERICAN UNIVERSITY

EFFICIENT HEURISTIC ALGORITHMS FOR INFLUENCE PROPAGATION IN SOCIAL NETWORKS

By

Karine H. Lamaa

A thesis

Submitted in partial fulfillment of the requirements

for the degree of Master of Science in Computer Science

School of Arts and Sciences

April 2018

LAU
الجامعة اللبنانية الأميركية
Lebanese American University

THESIS APPROVAL FORM

Student Name: Karine Lamaa I.D. #: 201508431

Thesis Title : Efficient Heuristic Algorithms for Influence Propagation in Social Networks

Program: MS in Computer Science

Department: Computer Science and Mathematics

School: Arts and Sciences

The undersigned certify that they have examined the final electronic copy of this thesis and approved it in Partial Fulfillment of the requirements for the degree of:

Master of Science in the major of Computer Science

Thesis Advisor's Name Faisal Abu Khzam	Signature	[redacted]	DATE: 10 / 4 / 2018 (Day / Month / Year)
Committee Member's Name May Hamdan	Signature		DATE: 13 / 4 / 2018 (Day / Month / Year)
Committee Member's Name Azzam Mourad	Signature		DATE: 10 / 4 / 2018 (Day / Month / Year)

LAU
الجامعة اللبنانية الأميركية
Lebanese American University

THESIS COPYRIGHT RELEASE FORM

LEBANESE AMERICAN UNIVERSITY NON-EXCLUSIVE DISTRIBUTION LICENSE

By signing and submitting this license, you (the author(s) or copyright owner) grants the Lebanese American University (LAU) the non-exclusive right to reproduce, translate (as defined below), and/or distribute your submission (including the abstract) worldwide in print and electronic formats and in any medium, including but not limited to audio or video. You agree that LAU may, without changing the content, translate the submission to any medium or format for the purpose of preservation. You also agree that LAU may keep more than one copy of this submission for purposes of security, backup and preservation. You represent that the submission is your original work, and that you have the right to grant the rights contained in this license. You also represent that your submission does not, to the best of your knowledge, infringe upon anyone's copyright. If the submission contains material for which you do not hold copyright, you represent that you have obtained the unrestricted permission of the copyright owner to grant LAU the rights required by this license, and that such third-party owned material is clearly identified and acknowledged within the text or content of the submission. IF THE SUBMISSION IS BASED UPON WORK THAT HAS BEEN SPONSORED OR SUPPORTED BY AN AGENCY OR ORGANIZATION OTHER THAN LAU, YOU REPRESENT THAT YOU HAVE FULFILLED ANY RIGHT OF REVIEW OR OTHER OBLIGATIONS REQUIRED BY SUCH CONTRACT OR AGREEMENT. LAU will clearly identify your name(s) as the author(s) or owner(s) of the submission, and will not make any alteration, other than as allowed by this license, to your submission.

Name: Karine Lamaa

Signature: [redacted]

Date: 17-03-2018

LAU
الجامعة اللبنانية الأميركية
Lebanese American University

PLAGIARISM POLICY COMPLIANCE STATEMENT

I certify that:

1. I have read and understood LAU's Plagiarism Policy.
2. I understand that failure to comply with this Policy can lead to academic and disciplinary actions against me.
3. This work is substantially my own, and to the extent that any part of this work is not my own I have indicated that by acknowledging its sources.

Name: Kaxine Lamaa

Signature: [redacted]

Date: 17-03-2018

ACKNOWLEDGMENT

This project would not have been possible without the support of many people.
Many thanks to my advisor, Dr. Faisal Abu-Khzam, who read my numerous revisions and helped make some sense of the confusion. Also thanks to my committee members, Dr. Azzam Mourad and Dr. May Hamdan, who offered guidance and support.

And finally, thanks to my parents and my friends who endured this long process with me, always offering support and love.

Efficient heuristic algorithms for influence propagation in social networks

Karine H Lamaa

ABSTRACT

The study of how fast advertisements and ideas propagate across a social network started to gain notable attention recently. In this context, the notion of an influencer has been considered: an influencer is an individual capable of affecting the behavior, character and/or social opinion of others. Our objective in this work is to find a set of individuals that can collectively serve as influencers. We model the problem using the previously studied notion of a positive influence dominating set. The problem seeks a smallest set of positive-influencers assuming that an individual becomes positively influenced when the majority of his/her friends are influenced. We start by presenting and studying efficient heuristic algorithms for this problem and show how different types of social networks require different heuristic methods. Then we introduce the notion of an influence propagation function and use it to design an efficient algorithm across all types of networks. Finally, we introduce a new model that allows the maximization of influence propagation while selecting a much smaller set of influencers. Our experiments on a variety of social (sub) networks show that our algorithms can almost always manage to extract a small set of influencers through which we can effectively propagate a message throughout the whole network.

Keywords: Greedy Algorithms, Online Social Networks, Positive Influence Dominating Set, Influence Propagation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table **Page**

Chapter One

Introduction

Currently there are 1.94 billion monthly active Facebook users, with almost 18 percent increase year after year [1]; such a number can only reflect the importance of social networks and their impact. As Internet and social media are becoming easily accessible, people are speaking more freely and directly. As individuals read articles or view advertisements they are allowing others to influence their sociological and political views. When we are hiding behind a virtual wall on the Internet it becomes easier to be aggressive and negative in our comments. This rise in (the use of) technology triggers an imminent need for appropriate data analysis techniques that focus on behavioral aspects of social networks, thus our attempt (in this work) is to find algorithmic methods to identify influencers in social networks and utilize them to spread information across the network.

In reality, we are less likely to be (directly) influenced by an advertisement or a post when we see it once. It is only after multiple appearances of the same post that we are intrigued to check it. In real life we become similar to our family and the people we befriend, same approach happens in social networks as users start adopting each others opinions and interests [27]. Big companies are able to advertise their products on different social medias by paying a fortune. However, this is not an option for small businesses. With the ongoing change in social networks, it is becoming harder to reach random people knowing that algorithms are favoring family, friends, and big

companies. A natural question to pose in this context is how can we spread a certain advertisement or opinion so that it reaches and influences all the network, or a sub-network without having to spend (a large amount of) money? In an attempt to model this question, the graph-theoretic Positive Influence Dominating Set problem (PIDS) has been recently introduced in [34] as follows: given a graph (or network) G and an integer k, can we find a set P consisting of k or less vertices (or nodes) in G such that every other vertex of G has at least half of its neighbors in P? One may require that elements of P also have half of their neighbors in P, which gives rise to the Total Positive Influence Dominating Set problem (TPIDS).

In this work we do not consider TPIDS since we assume that a network member who is entrusted to spread positive influence is already positively influenced. In a real setting, each element of the PIDS solution would post a message, article, or even opinion about the product in question. When nodes not in PIDS see the same topic multiple times by different individuals it should gain their attention and potentially (indirectly) influence their opinion. A typical (social campaign) application of PIDS would be to select a set of non-smokers and ask them to share their different opinions about the dangers of smoking. Thus, even without spreading a common message we will have them spreading the idea that smoking is dangerous and influencing people in their networks to stop smoking. In this work, we address the issue from a different perspective where instead of creating/injecting positivity, we would require the selected nodes (in the PIDS solution) to spread a common message in order to advertise it to the entire network. This allows the use of the positive set in social, marketing and political campaigns. Influential nodes selected by the algorithm can be approached directly by the company seeking the results, eliminating the need to invest in advertisements.

In the first part of our work, we observe that graphs corresponding to online social networks (OSN) differ in structure such as density and degree distribution. For example, a Facebook graph's density and degree distribution is different from a Twitter or a Google+ graph. Facebook is more about connections between friends so the degrees

of corresponding vertices do not vary too much, whereas in a Twitter graph popular nodes would have very high degree compared to other nodes. Therefore, we observe that each of the known heuristic algorithms might not be effective on all the different graph structures. In order to address those differences, we examine four heuristic algorithms and report on the effectiveness of each for every network structure.

Moving forward with our research we highlight the importance of the "word of mouth" effect in OSNs. Most of the times a post makes its way to our home-pages because a friend in our network reacted by liking, commenting, sharing, or re-tweeting the post. Hence, we distinguish the importance of selecting a node through which we can propagate the maximum influence if someone from the influencer's network interacted with the post. To measure the influence of each individual we introduce the notion of an influence propagation function, which calculates the propagation of a node's influence from the nearest to the most distant neighbors. Using this measure we introduce an additional algorithm to compute a minimal positive set on all types of social networks. This is made possible with few additional computations that could take more time. We note that solving the PIDS problem by extracting less seeds is far more important than finding a (larger) set of seeds in less time.

Finally, we notice that requiring half of a node's neighbors to follow a trend or advertise a common topic will definitely leave a clear impact on that node. Nevertheless, this leads to a sizable (T)PIDS, and in the case of a marketing campaign will cost a great deal of investment and might not be a feasible answer for all companies. Therefore, we need to ask ourselves do we really need to see the majority of our friends adopting an idea or product before we do? The notable recent "Me too" movement, which took over the whole Internet in 2017 [32], started with one person posting a tweet to encourage girls to post "Me too" in their statuses to indicate sexual harassment. This single post was followed by 1.7 million tweets and 12 millions Facebook posts and comments, leaving a huge impact on society. A multimedia artist created the below network visualization of the movement [21]. The graph contains 25,218

nodes, 16,183 edges and 10,709 communities. This movement started from the blue community representing the person that posted the first tweet and her neighbors.

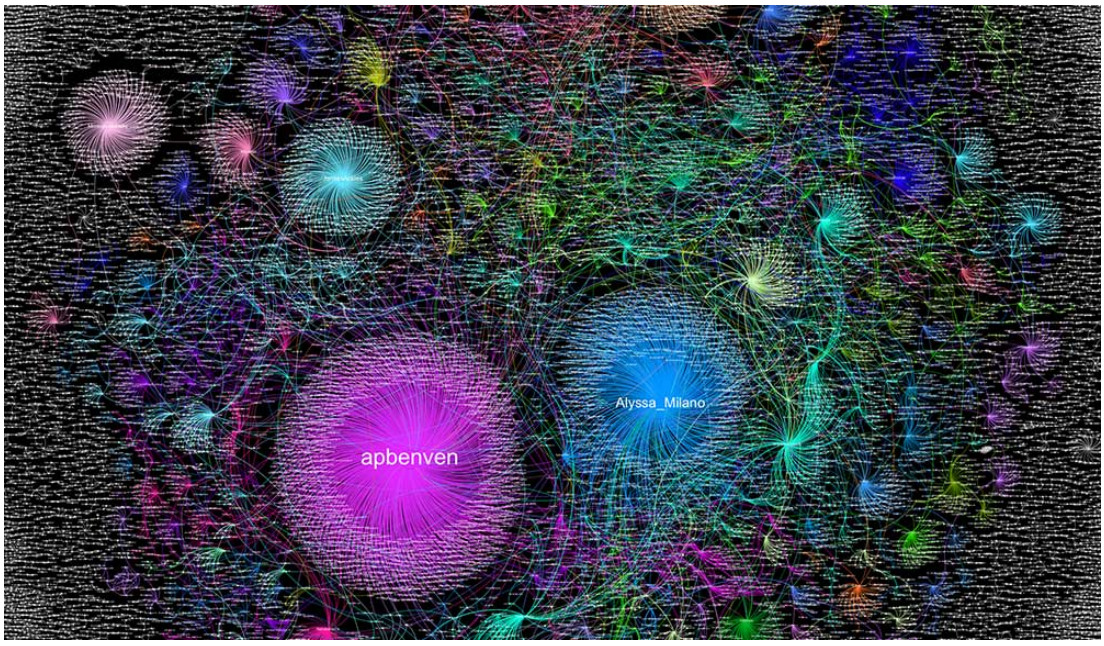

Figure 1: Metoo hashtag network visualization

With the above example in mind, we introduce a new influence propagation model inspired by PIDS. We consider that a node v becomes active/positive when either one of its neighbors is in the set or a majority of its neighbors are directly connected to a node in the set, hence active. The latter shall be called an *influenced* node. This new definition allows the generation of even smaller sets while guaranteeing maximal influence propagation across the network.

The thesis is structured as follows. Chapter 2 defines some preliminaries; chapter 3 overviews related work; chapter 4 presents a number of efficient heuristic algorithms; chapter 5 describes the influence function along with the new proposed model, followed by experimental analysis and results in chapter 6; chapter 7 concludes with a summary and future directions.

Chapter Two

Preliminaries

We model a social network as a simple unweighted graph. Typically a graph is denoted by $G = (V, E)$ where V is the set of vertices/nodes representing individuals, and E the set of edges or direct links representing friendship in the network (such as friends on Facebook or followers on Twitter). We adopt common graph theoretic terminologies such as those found in [19]. This research is based on the notion of a *dominating set*. Formally speaking, a dominating set D in an undirected graph G is a subset of V such that any vertex not in D is a neighbor of at least one vertex in D. The set of neighbors of a vertex v is denoted by $N(v)$, and for a subset D of V, $N(D)$ is the union of the neighborhoods of the vertices of D while $N_D(v)$ denotes the number of elements of D that are neighbors of the vertex v. If D is a dominating set, the elements of D will be called dominator vertices/nodes, and their neighbors will be the *dominated* vertices. The degree of a vertex v, denoted $deg(v)$, is defined as the number of edges incident on v and is equivalent to $|N_G(v)|$ in this text.

Other aspects of domination include total domination (each and every vertex, even an element of the dominating set, must have a neighbor in the dominating set) [24]; k-domination where a vertex must be dominated by at least k of its neighbors [17]; colored domination: the dominating set is required to be of one color while the vertices to be dominated are of the other color [15]. Moreover, one may also require (any of the above versions of) the dominating set to induce a connected subgraph [23, 11].

Another variation of total domination is the harmless set problem defined by obtaining a small total dominating set through vertex deletions [6].

As mentioned earlier, *positive influence dominating set* of a graph G is a set D of vertices of G such that every vertex that is not in D has at least half (i.e., majority) of its neighbors in D. A *total positive influence dominating set* is a set D of G such that every vertex of G is dominated by at least half of its neighbors. At the initial state of our algorithm we assume that all nodes in the graph are *negative* or *inactive*.

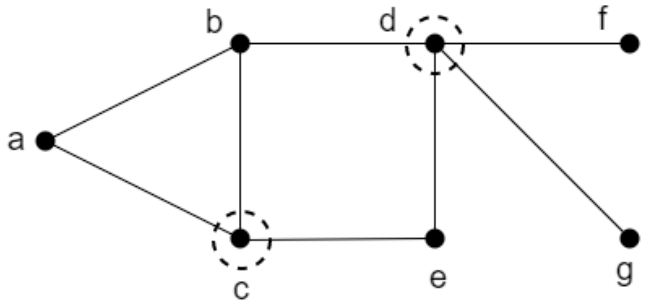

Figure 2: Graph with a PIDS solution of size two.

While building a positive influence dominating set D, elements of D are said to be positive, so is any vertex that has half of its neighbors in D. In other words, D is a positive influence dominating set of G when all the vertices of G are positive, in which case we say that the graph is positively satisfied by D. An example is depicted in Figure 2. Circled nodes in the graph are the ones included in a positive influence dominating set ($D = \{c,d\}$). Each of the other nodes has half of its neighbors in D. Note that, in this example, a minimum solution to TPIDS must consist of at least four vertices because even nodes c and d would require the majority of their neighbors to be in D. A possible solution for TPIDS would therefore be $D = \{b,c,d,e\}$.

An important measure when dealing with positive influence domination is the *need*

degree of a vertex. This is the number of neighbors still needed for the vertex to become positive. Therefore it is calculated as half of its degree minus the number of its neighbors that are already in the (partial) positive dominating set D.

$$Need(v) = \frac{deg((v))}{2} - |N_D(v)|$$

The *utility* of a vertex v is the sum of its neighbors' needs. A vertex of high utility is capable of satisfying a large number of needy neighbors.

$$Utility(v) = \sum_{u \in N(v)} Need(u)$$

Throughout the rest of this thesis we take into consideration the following three different types of social network structures.

1. **Dense:** A dense graph is a graph where the number of edges is close to maximal. Such that the number of edges is greater than or equal to $(VlgV)$, where V is the number of vertices [4].

2. **Sparse:** A sparse graph is a graph with only few edges. Such that the number of edges is less than $(VlgV)$ [4].

3. **Hub structured:** A hub structured graph is a graph with a small number of high-degree vertices (hubs). Connections are mainly established with those hub nodes.

Chapter Three

Positive Influence Dominating Set

Spreading influence in online social networks (OSN) appeared first in the literature when Kempe et al. [26] introduced the problem of finding a small subset of individuals in a network through which we can spread an information among maximum others. From this approach, Wang et al. [34] proposed having the majority of a node's neighbors in the subset to maximize the influence. The (T)PIDS problem was mainly inspired by the well-known dominating set problem. This notion has been extensively studied in the literature [12, 33], as well as the corresponding minimization problem [22].

Kempe et al. [26] presented a polynomial-time approximation algorithm that can deliver solutions that are within 63% of an optimum solution allowing the detection of a seemingly reasonable set of influential nodes in a network. Two models were introduced: independent cascade model (IC) and linear threshold model (LT). In both models the algorithm starts with a set of active nodes and considers that a node becomes active as more of its neighbors become active. In IC, when a node v becomes active it has one chance to activate each neighbor u according to a parameter determined by the system. On the other hand, in LT, each node u chooses a random threshold from 0 to 1 at which it becomes active. Each node is influenced by its neighbors according to a weight, and the sum of weights is at most 1. When the sum of weights from active neighbors of an inactive node u reaches the threshold, u becomes active. These models

were later on used in multiple papers such as [28] where they used the IC model to find a set of seeds with limited budget and a fixed cost per node.

In the literature, this problem is denoted as the *influence maximization problem (IMP)* or *maximum influence propagation* (MIP) [14]. Many other papers studied this topic and introduced algorithms to detect and select influential nodes in the network [31, 20, 13, 38, 36, 16]. Most of these algorithms are based on either [34] or [26]. In this thesis we start by addressing the PIDS problem and finish by introducing a new model inspired by PIDS. The following previously introduced algorithms were the basis of our work. Therefore we will start by explaining the relevant details of the approaches that have been adopted.

3.1 Dominating Set Based Approach

Wang et al. first introduced the (T)PIDS problem in 2009 as a solution to social problems such as drinking, smoking and drugs. Two distinct groups of people were considered: abstainers and binge drinkers, and the objective is to find a set of abstainers to influence the rest of the network "positively". A set of initially-known as positive nodes (the abstainers, in this case) is directly inserted into the positive set D and removed from further computations, a (T)PIDS selection algorithm is then used. Their first approach was to find a 1-dominating set over the vertices not in D, insert them into D and continue until the network is satisfied.

In a more recent paper by Wang et al. [35], a different approach was used by introducing the following function, allowing the selection of the node able to satisfy maximum need in the network.

$$f(A) = \sum_{v \in V} min(h(v), n_A(v))$$

In the above function A represents the positive set; hence $n_A(v)$ represents the neighbors of v in A. Whereas $h(v)$ represents the number of neighbors v needs in the set to

become positive, it is equal to $\frac{deg(v)}{2}$. Knowing that the function is computed even for nodes already in the set, we note that the main focus in the above mentioned work was on the TPIDS problem.

Let us consider the example depicted in Figure 2 (of the previous chapter). We assume that node a is added into the set and calculate $min(h(v), n_A(v))$ for its neighbors v. This allows us to assess the value or influence of adding a into the set. For node b $min(h(b), n_A(b)) = min(2,1) = 1$, for node c $min(h(c), n_A(c)) = min(2,1) = 1$, and so on. The sum of all the neighbors forms $f(A)$. The node with the maximum function value is selected according to the following algorithm. In other words, Wang's greedy algorithm focuses on inserting into A the node that influences the most number of unsatisfied neighbors. As explained in [29] this algorithm runs in $O(n^3)$ time and does not guarantee a smallest set size in literature.

Algorithm 1: Wang's et al. Algorithm

$A \leftarrow \varnothing$
while $f(A) < \sum_{v \in V} h(v)$ **do**
    choose $u \in V$ to maximize $f(A \cup \{u\})$
    set $A \leftarrow A \cup \{u\}$
    compute $f(A)$
end
Output A

3.2 Cover Degree Based Algorithm

Based on the work described above, Raie et al. [29] introduced an improved greedy algorithm that runs in $O(n^2)$ and generates sets of smaller size. Their algorithm uses the notions of *need* and $utility$ (they call it *cover-degree*). Their heuristic computes the utility of each node and inserts into the set the node with maximum utility. Hence, they seek to add into the set the node that can benefit and satisfy the maximum number of needy neighbors in the graph. Both algorithms answer the TPIDS problem. However, Raie's algorithm was able to generate an average of 5.5% smaller set size than Wang's

algorithm. This result was further improved in [18].

Algorithm 2: Raie's et al. Algorithm

$A \leftarrow \varnothing$
Compute the *cover-degree* for each node
while *there exists a vertex* $v \in V$ *such that* $(n_A(v) < \frac{deg(v)}{2})$ **do**
    choose $u \in V$ with maximum *cover-degree*
    set $A \leftarrow A \cup \{u\}$
    Subtract 1 from the *need-degree* of neighbors of u
    Revise the *cover-degree* of each node
end
Output A

3.3 Maximum Needy Neighbors Algorithm

Paper [18] focuses on solving the PIDS problem. In order to compare to [35] and [29] they apply their algorithm to the PIDS instead of TPIDS problem. They introduce the following functions.

$$s(u) = \begin{cases} 1 & \text{if } n_P(u) \geqslant h(u) \text{ or } u \in P \\ 0 & \text{otherwise} \end{cases}$$

$$g(u) = s(u) + \sum_{w \in n(u)} 1 - s(w)$$

P denotes the positive set and $h(u)$ denoted the degree of vertex u divided by 2. For the remaining notations the same definitions as previously explained are used. Function $s(u)$ is equal to 1 if the number of u's neighbors in the set is bigger than or equal to $h(u)$ or if u is in the set. Hence, it is 1 if u is satisfied and 0 otherwise. Function $g(u)$ on

the other hand represents the number of unsatisfied nodes in u's neighborhood. They introduce an algorithm that selects into P the nodes with maximum $g(u)$. The algorithm shares similarities with [29] but instead of calculating the sum of needs (cover-degree) they calculate and select the node with the maximum **number** of unsatisfied neighbors. They also take into consideration the influence a node u has on itself by adding $s(u)$ to the sum of unsatisfied neighbors in $g(u)$. This allows the selection of an unsatisfied node even if it doesn't have a noticeable impact on its direct neighbors. This latter notion was not taken in consideration in previous work.

Algorithm 3: AltGreedy Algorithm

$P \leftarrow \varnothing$
Compute $g(u)$ for all $u \in V$
while *(P is not a PIDS)* **do**
    select $u \in V - P$ to maximize $g(u)$
    set $P \leftarrow P \cup \{u\}$
    Revise $g(w)$ values for all $w \in V - P$
end
output A ;

The above algorithm was capable of generating an average of 5% smaller set size than Raie et al. algorithm and therefore a 10.5% smaller set size that Wang et al. algorithm. We shall focus on Raie et al. and AltGreedy when examining results to demonstrate the effectiveness of our algorithms.

Chapter Four

Basic Heuristic Algorithms

In this chapter we present four different heuristic algorithms, two of which are new to the best of our knowledge and two existing. We highlight 3 main graph structures; dense, sparse and hub-structured. We consider the following four heuristic methods that can be applied to extract a positive influence dominating set. Again, our motivation is that a single heuristic cannot be effective on different types of social networks, as we shall see in the next section.

1. **Maximum utility**: select a node with maximum utility into the positive set.
2. **Minimum need maximum utility**: find a node with minimum need and select its maximum utility neighbor into the positive set.
3. **Maximum need**: select a node with maximum need directly into the positive set.
4. **Maximum need maximum utility**: locate a node with maximum need and select its maximum utility neighbor into the positive set.

4.1 Modeling Social Networks

Social networks have different ways of creating connections between individuals. In Facebook it is enough that an individual adds another to become friends and see each

others posts. Whereas in Twitter, if an individual follows an account he will be able to see its posts but not the opposite. Therefore, we model the Facebook network as an undirected graph, while Twitter is modeled as directed graph. It is important to differentiate between those two types because in a directed graph influence can only be propagated in one direction, on the contrary to undirected graphs.

Our algorithms guarantee effective results on all types of social networks by taking into consideration the above. When we parse an undirected graph both endpoints u and v of edge e are listed as each others neighbors. Hence the algorithm will compute the propagation of influence from u to v and from v to u. On the other hand, when we parse a directed graph only endpoint v of edge e is added as a neighbor of u (if e's direction is from u to v, the opposite otherwise). In this case, the algorithm will compute the propagation of influence only from u to v. Such that if node u added a post then node v will be able to see it, but not the opposite.

4.2 The Maximum Utility Heuristic

The following $MaxUtility$ heuristic algorithm applies the maximum utility defined above repeatedly until a solution is obtained. This approach promises to be effective in dense graphs where the propagation of positivity is faster and easier when selecting a node of maximum utility since (obviously) it helps in satisfying multiple needy neighbors. An example of a dense graph is presented in figure 3.
The below pseudo-code describes the Maximum Utility algorithm. Note that the sum of needs is initialized to be the sum of degrees in the graph because the positive set is initially empty. As the set gets filled, the value of sumNeeds reaches zero when all the nodes become positive. This is when our algorithm terminates.

Algorithm 4: Maximum Utility

Input: Graph $G = (V, E)$
Output: Influential set $S \subset V$
$S \leftarrow \varnothing$
$sumNeeds(G) \leftarrow \sum_{v \in G} need(v)$
foreach *v in G* **do**
    $Need(v) = \frac{deg(v)}{2}$
    $Utility(v) = \sum_{u \in N(v)} need(u)$
end
while $sumNeeds(G) > 0$ **do**
    $u \leftarrow maxUtility(G)$
    Insert u into S
    $Need(u) = Utility(u) = 0$
    $sumNeeds(G) = sumNeeds(G) - (Need(u) + |N(u)|)$
    foreach *v in* $N(u)$ **do**
        if $Need(v) > 0$ **then**
            $Need(v) = Need(v) - 1$
        end
        $Utility(v) = Utility(v) - Need(u)$
        foreach *w in* $N(v)$ **do**
            $Utility(w) = Utility(w) - 1$
        end
    end
end
Output S

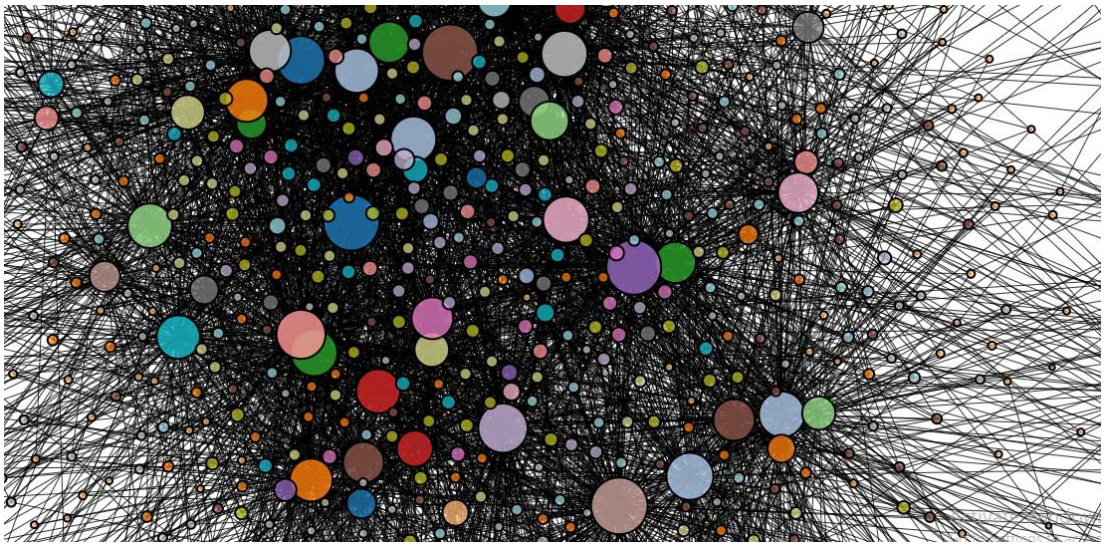

Figure 3: Dense subgraph (Facebook)

4.3 The Maximum Need Heuristic

The "Maximum Need" algorithm promises to be the most effective on sparse graphs. In fact, when a graph is very sparse the utility becomes of less importance (as the positivity propagation becomes harder): fewer edges are present in the graph and thus a node with maximum utility would barely satisfy neighboring nodes. Thus it would be better to select a needy vertex directly so it would satisfy itself as well as its neighbors. Note that a needy vertex is not necessarily a vertex of low degree. The algorithm for "Maximum Need" is the same as the one for "Maximum Utility" except that we replace $maxUtility$ by $maxNeed$ (on the first line in the body of the while loop). An example of a sparse graph is presented in figure 4.

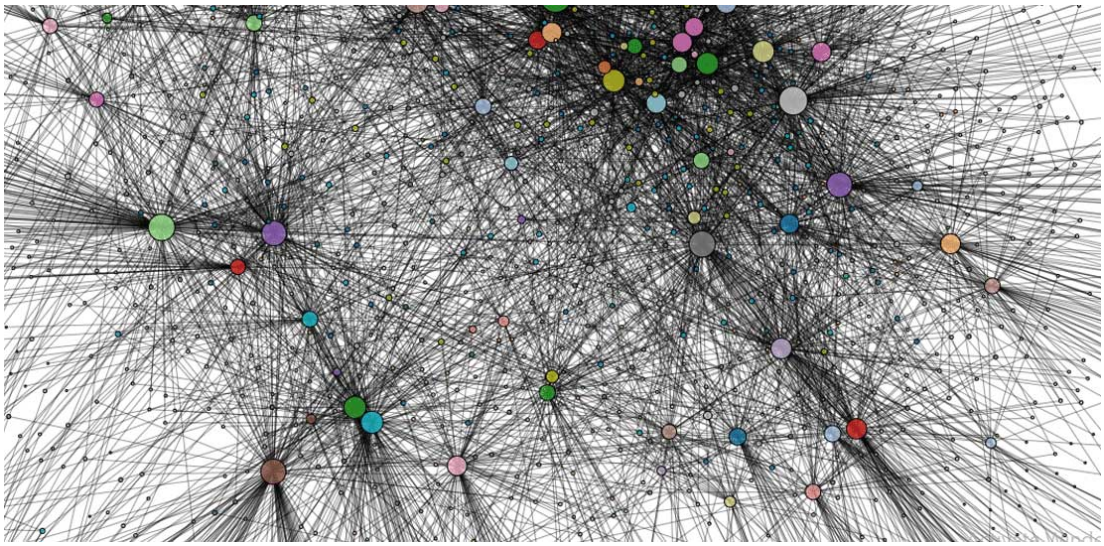

Figure 4: Sparse subgraph (Facebook)

4.4 The Minimum Need Maximum Utility Heuristic

The “Minimum Need Maximum Utility” promises to be effective on graphs where a small number of high utility nodes can dominate a large set of neighbors. Such graphs are characterized by having many hubs (high-degree vertices) without being of high density. An example of a hub-structured graph is presented in figure 5. The below pseudo-code describes the corresponding algorithm.

Algorithm 5: Minimum Need Maximum Utility

Input: Graph $G = (V, E)$
Output: Influential set $S \subset V$
$S \leftarrow \varnothing$
foreach *v in G* **do**
    $Need(v) = \frac{deg(v)}{2}$
    $Utility(v) = \sum_{u \in N(v)} need(u)$
end
$sumNeeds(G) \leftarrow \sum_{v \in G} need(v)$
while $sumNeeds(G) > 0$ **do**
    $z \leftarrow$ *minNeed(G)*
    $u \leftarrow$ *maxUtility u* neighbor of z
    Insert u into S
    $Need(u) = Utility(u) = 0$
    $sumNeeds(G) = sumNeeds(G) - (Need(u) + |N(u)|)$
    foreach *v in* $N(u)$ **do**
        if $Need(v) > 0$ **then**
            $Need(v) = Need(v) - 1$
        end
        $Utility(v) = Utility(v) - Need(u)$
        foreach *w in* $N(v)$ **do**
            $Utility(w) = Utility(w) - 1$
        end
    end
end
Output S

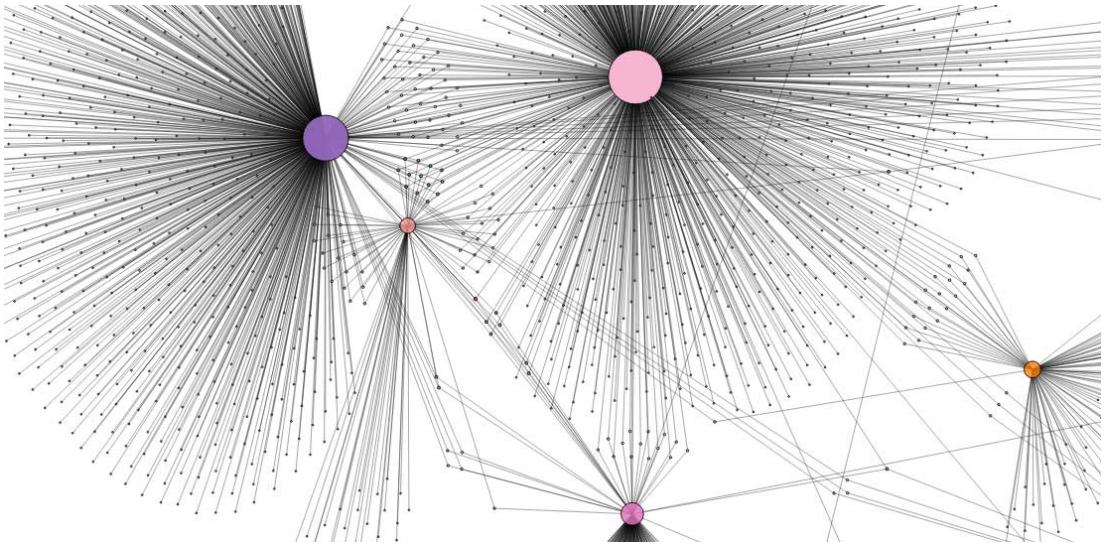

Figure 5: Hub structured graph

4.5 The Maximum Need Maximum Utility Heuristic

Finally, we study the possibility of selecting a high utility neighbor of a node that has a maximum need. The motivation behind this approach stems from the intuition that a node of high need should either be selected or have a set of neighbors that can satisfy a largest possible set of vertices. However, as we shall see, the high need makes a few neighbors not enough for satisfying the selected (max-need) vertex, which causes the resulting solution to be larger than those produced by the other algorithms. The algorithm for "Maximum Need Maximum Utility" is the same as the one for "Minimum Need Maximum Utility" except that we replace *minNeed* by *maxNeed* (on the first line in the body of the while loop).

4.6 Analysis

A binary heap data structure is used in the above heuristic algorithms to efficiently extract a vertex of maximum/minimum need or a vertex of maximum utility. Thus it takes $O(\log n)$ to locate such vertices. Selecting a vertex and updating the need and utility takes linear time while the time needed to update the needs of all the vertices is in the worst-case the sum of degrees which is $O(|E|)$ (by the Handshaking Lemma). On the other hand, updating the utility of all the vertices is in the worst-case $O(n^2)$ as it requires a nested loop. Therefore the total running time is in $O(n^2)$.

Chapter Five

Influence Propagation Algorithms

In this chapter we start by minimizing the positive influence set size (PIDS) by introducing a deceasing function to measure the influence. We then introduce a new model called influence propagation dominating set (IPDS).

5.1 The Influence Function

Most of the times a certain post reaches our home-pages when someone from our network interacts with it by liking, commenting, sharing or re-tweeting. Therefore, influence is not only propagated through the person initially posting the information. Carefully choosing a node with a high propagation range allows the maximization of influence propagation through user interactions. In fact, according to the "Six degrees of separation theory" [37] we are all connected through six or fewer steps. These relations are reflected in social networks such that any two individuals (accounts) are at a distance of 6 or less "friend of a friend" chain. Hence, if a message is posted from nodes with high influence propagation the probability of this message reaching a maximum number of individuals becomes more important. Taking this factor in consideration, we use a new influence propagation measure (dubbed influence function) introduced by Abu-Khzam in a recent grant proposal [5].

To calculate the influence throughout all distance neighbors of v we use "Breadth-First Search" algorithm n times with n being the number of nodes [19]. As we traverse

the graph using "BFS" we compute the influence of each node from its first to the last distant neighborhood. As we further move from the root node the coefficient of the influence is decreased.

5.1.1 A Maximum Influence Algorithm

As mentioned earlier, we focus on solving the PIDS problem. First we define the t^{th} neighborhood of a vertex as follows: $N_t(v)$ denotes the set of vertices at distance t from v. For example, $N_1(v) = N(v)$ is the set of (direct) neighbors of v in the network. From this point on, we denote by S the set of nodes selected by our algorithm into the positive set. We shall denote by d the maximum distance between any two vertices (AKA. graph diameter). Thus $N_d(v)$ is the most distant neighborhood of v. Starting with an empty set, the algorithm stops when the graph becomes entirely active (influenced). We use need to determine the influence because the more needy v's neighbors are the more important it is for v to be in S. Therefore, we calculate the influence of a vertex v based on the need of v itself, plus the need of its direct and indirect neighbors, assuming the influence of v decreases as the distance from v increases. For simplicity, we denote by $Need(N_d(v))$ the sum of needs of v's neighborhood at distance d:

$$Need(N_d(v)) = \sum_{u \in N_d(v)} Need(u)$$

The influence function employs a decreasing sequence whose t^{th} term is multiplied by the sum of needs of the elements of $N_t(v)$. We assume that if node v is added to the positive set then the influence it has on itself is 100%, the influence it has on its direct neighbors is 50% and its influence on its indirect neighbors is 25% and so on. Therefore we compute the influence function using the geometric sequence of common ratio 0.5 as follows.

$$Influence(v) = Need(v) + \tfrac{1}{2}Need(N_1(v)) + \tfrac{1}{4}Need(N_2(v)) + \cdots + \tfrac{1}{2^d}Need(N_d(v))$$

$Influence(v)$ is calculated for all v in G. We proceed to select the vertex of maximum influence because it will satisfy more nodes in the network, thus allowing faster prop-

agation. The below pseudo-code of the "Maximum Influence" algorithm describes the influence propagation algorithm used. We note that the sum of needs is initialized to be the sum of degrees of all v in G because S is initially empty. As the set gets filled, the value of *sumNeeds* reaches zero when all the nodes become positive. This is when our algorithm terminates.

Algorithm 6: Maximum Influence

Input: Graph $G = (V, E)$
Output: Influential set $S \subset V$
$S \leftarrow \varnothing$
foreach *v in G* **do**
  $Need(v) = \frac{deg(v)}{2}$
end
$sumNeeds(G) \leftarrow \sum_{v \in G} need(v)$
while $sumNeeds(G) > 0$ **do**
  foreach *v in G* **do**
    $Influence(v) = Need(v) + \frac{1}{2}Need(N_1(v)) + ... + \frac{1}{2^d}Need(N_d(v))$
  end
  $u \leftarrow maxInfluence(G)$
  $need[u] = 0$
  $sumNeeds(G) = sumNeeds(G) - Need(maxInf)$
  foreach *w in* $N_1(u)$ **do**
    if $Need(w) > 0$ **then**
      $Need(w) = Need(w) - 1$
      $sumNeeds(G) = sumNeeds(G) - 1$
    end
  end
end
Output S

The algorithm loops over all nodes in the graph and calculates the function $Influence(v)$, the node with the maximum influence is selected (line 4 of the pseudo-code). After inserting v into the influential set S, its need is set to 0 as it is considered satisfied. All vertices in $N_1(v)$ now have a neighbor in the positive set, therefore their needs are decremented by 1. We shall see in the following section that this algorithm generates a set of smaller size, thus requires less cost.

5.2 Influence Propagation Dominating Set (IPDS)

As we previously discussed, most posts on social networks make their way to our timelines through the interaction of others. A friend might comment, like, re-tweet or share a certain post, similar to the "Metoo" hash-tag. Moreover, the influence function previously introduced allowed us to select a node not only capable of satisfying a maximum number of direct neighbors, but also capable of satisfying more distant neighbors if the post gets any interactions. If an appropriate vertex shares the post it allows it to have more visibility in the network.

Inspired by the PIDS model we introduce a new influence propagation model such that a node u becomes positive if it satisfies at least one of the following conditions.

1. Node u belongs to the positive set S
2. At least **one** of u's direct neighbors belong to S
3. At least **half** of u's neighbors are positive (either in S or influenced by a node in S).

We only require one of u's direct neighbors to be in the set because we have a high probability of someone interacting with that post, thus making it appear more on u's homepage. However in point 3 we require the majority of u's neighbors to be positive. In fact, such nodes are not the ones directly posting the information but they are connected to someone who is. Requiring the majority guarantees that at least one of them will interact with the post so it can make its way to u's timeline. We can easily spot how the information is being propagated through a domino effect across the entire network.

We adopt the previously explained influence function f to select nodes into the positive set. When a node is inserted into the set both its direct and indirect neighbors will be influenced. This model, along with the use of f, allows a maximization of the influence while guaranteeing a much smaller set size. Such a model is mostly efficient for companies looking to select the smallest positive set to propagate their message.

Algorithm 7 describes the adopted approach. We notice that the selection of nodes into the positive set is similar to the previous model. When a node u is added into the set, the need of its direct neighbors w is set to zero (satisfied). On the other hand, the need of its second level neighbors z is decremented by one to indicate that they are *influenced* by someone in S.

Algorithm 7: IPDS

Input: Graph $G = (V, E)$
Output: Influential set $S \subset V$
$S \leftarrow \varnothing$
foreach v *in* G **do**
  $Need(v) = \frac{deg(v)}{2}$
end
$sumNeeds(G) \leftarrow \sum_{v \in G} need(v)$
while $sumNeeds(G) > 0$ **do**
  foreach v *in* G **do**
    $Influence(v) = Need(v) + \frac{1}{2}Need(N_1(v)) + ... + \frac{1}{2^d}Need(N_d(v))$
  end
  $u \leftarrow maxInfluence(G)$
  Insert u into S
  $need[u] = 0$
  $sumNeeds(G) = sumNeeds(G) - Need(maxInf)$
  foreach w *in* $N_1(u)$ **do**
    if $Need(w) > 0$ **then**
      $sumNeeds(G) = sumNeeds(G) - Need(w)$
      $Need(w) = 0$
    end
    foreach z *in* $N_1(w)$ **do**
      if $Need(z) > 0$ **then**
        $Need(z) = Need(z) - 1$
        $sumNeeds(G) = sumNeeds(G) - 1$
      end
    end
  end
end

Chapter Six

Experimental Analysis

We implemented all the previously discussed algorithms using the JAVA language and we conducted our experiments on a 64-bit operating system, 2.40 GHZ CPU, 8 RAM and SSD hard disk machine. In this chapter we present the results of our experiments on a number of synthetic as well as real graphs.

6.1 Datasets

To test the proposed algorithms we used random graphs from the DIMACS benchmark generated using different tools [2] [25] and real social subgraphs from the network repository and Stanford dataset collection [30] [3]. We selected graphs of different densities and sizes to highlight how the effectiveness of the various algorithms is affected by the structure of the input graph. Table 1 presents the details of the random graphs, whereas table 2 presents the details of the social subgraphs. To be able to compare our algorithm with the work in [18], we used (just like they did) the Barabasi-Albert algorithm to generate graphs with 500 to 1000 nodes (with increments of 100). All graphs were created with an average degree of 10.

Graph	Nodes	Edges	Average degree
dsjc1000.5	1000	249825	499
le450-5a	450	5713	25
school1-nsh	352	14611	83
dsjc250.5	250	15667	125
nasa1824	1825	18692	20.48
fpsol2.i.3	363	8687	47.86
r250.1	250	866	6.94
2FullIns5	416	3584	17.02
130bit	584	6119	20.9

Table 1: Random graphs details

Graph	Nodes	Edges	Average degree
Fb-Brandeis99	3898	137566	70
Fb-Bucknell39	3826	158864	83
Fb-Mich67	3748	81902	43
Fb-Bowdoin47	2252	84387	74
Fb-Swarthmore42	1659	61050	73
Fb-Simmons81	1518	32988	43
Fb-Nips-Ego	2887	2980	2
Twitter-subgraph	1543	107305	69
Google-subgraph	23627	39241	3.32

Table 2: Social network sub-graphs details

6.2 Results for PIDS Heuristics

We started by running the four heuristic algorithms explained in section 3 on both random and social graphs. Table 3 presents the results obtained. For simplicity we denote by *A* the "Minimum Need Maximum Utility", *B* for "Maximum Need Maximum Utility", *C* for "Maximum Utility" and *D* for "Maximum Need".

We notice that the obtained solution of PIDS was constantly close to half of the network no matter how big the graph grew, which might be explained by the fact that each node must have at least half of its neighbors in the solution. Note that graphs nasa1824, le450-5a and fpsol2.i.3 are sparse. In this particular case, we noticed that vertex utility becomes less important and it was best to use the "Maximum Need" heuristic to quickly satisfy the needy vertices. Graphs dsjc250.5 and dsjc1000.5 are notably dense,

Graph	A	B	C	D
nasa1824	968	1007	928	**876**
le450-5a	225	219	216	**212**
fpsol2.i.3	142	181	162	**50**
r250.1	**113**	127	124	116
school1-nsh	**158**	159	159	163
2FullIns5	**142**	160	146	148
dsjc250.5	123	123	**122**	123
dsjc1000.5	502	498	**496**	497
130bit	196	233	**186**	192

Table 3: Heuristics' results on random graphs

which made the "Maximum Utility" heuristic more effective: selecting a number of maximum utility vertices would potentially lead to satisfying multiple needy ones. Finally, graphs r250.1, school1-nsh and 2-FullIns-5 have a relatively medium density (sparser than dsjc250.5 and denser than nasa1824). In this latter case it was better to use the "Minimum Need Maximum Utility" heuristic: selecting max-utility neighbors of vertices that are close to becoming "satisfied" seems to allow for faster propagation of positive influence.

We now study the various heuristics on actual social media graphs. We have four graphs of different types. Facebook's social network is based on connections between friends, leading to a dense network graph. Twitter is a social network that focuses on following popular and advertisement accounts, thus forming nodes with a higher degree than the average. Google+ graphs are very sparse with a few hubs (nodes of much higher degree than average). Note the difference in degree distribution, which suggests that a single heuristic would not be effective on both graphs, as witnessed with the above random graphs.

We notice that Fb-Mich67 and Fb-Brandeis99 have a relatively high density. In this case, the best heuristic was "Maximum Utility". Furthermore, we used a smaller Fb-nips-ego, extracted from a small circle on the social network, thus some nodes became of smaller degree and the graph became sparser. In this case using the "MinNeed-

Graph	A	B	C	D
Fb-Mich67	1411	1430	**1386**	1454
Fb-Brandeis99	1422	1530	**1393**	1452
Fb-Busknell39	1506	1519	**1480**	1535
Fb-Bowdoin47	853	857	**827**	858
Fb-Swarthmore42	625	618	**610**	633
Fb-Simmons81	549	574	**533**	567
Fb-nips-ego	**10**	1152	13	11
Twitter	**555**	612	1523	652
Google	223	6613	1584	**130**

Table 4: Heuristics' results on social subgraphs

MaxUtility" allowed us to retrieve a smaller positive set, confirming our previous observation on random graphs. The Google+ network is considerably sparse because it is not based on making friends or growing networks. It mainly has a few important nodes (hubs) that people connect to. In this type of graphs, as observed also with random graphs, we found out that it is best to use the "MaxNeed" algorithm. Additionally, Twitter is known to have some popular accounts with more connections than others, thus forming a less dense graph than Facebook, but not as sparse than Google+. Hence, as in the case for Fb-nips-ego, results suggest that we should apply "MinNeedMaxUtility" heuristic to extract a smaller positive set.

6.3 Results for Maximum Influence Algorithm

After testing the proposed heuristics on random and social graphs, we proceed to test the proposed influence propagation algorithm. In order to assess the efficiency of our algorithm, we compare our test results with the existing work of [29] and [18]. As we previously discussed, [29] proposed an algorithm that selects the node with maximum utility. Afterwards, [18] proposed an algorithm that selects the node with maximum number of needy vertices, allowing the generation of smaller positive sets. Similar to [18] we calculated the average positive set size after running the Raie et al. algorithm denoted RaieGreedy, AltGreedy, and Maximum Influence algorithms on 1000 graphs. We focus only on the PIDS variation of the problem. Therefore the RaieGreedy is

altered to detect PIDS rather than (T)PIDS for comparison.

Figure 6 presents the average size of PIDS obtained over 1000 iterations for each graph size. We notice that AltGreedy shows a 5% decrease of the PIDS size from RaieGreedy. However, our algorithm shows an average of 50% decrease from AltGreedy thus 55% decrease from RaieGreedy. After having tested the dataset used by AltGreedy, we continued our testing on a selection from the random and social graphs previously introduced. We will be comparing to AltGreedy knowing that they generated a smaller set size.

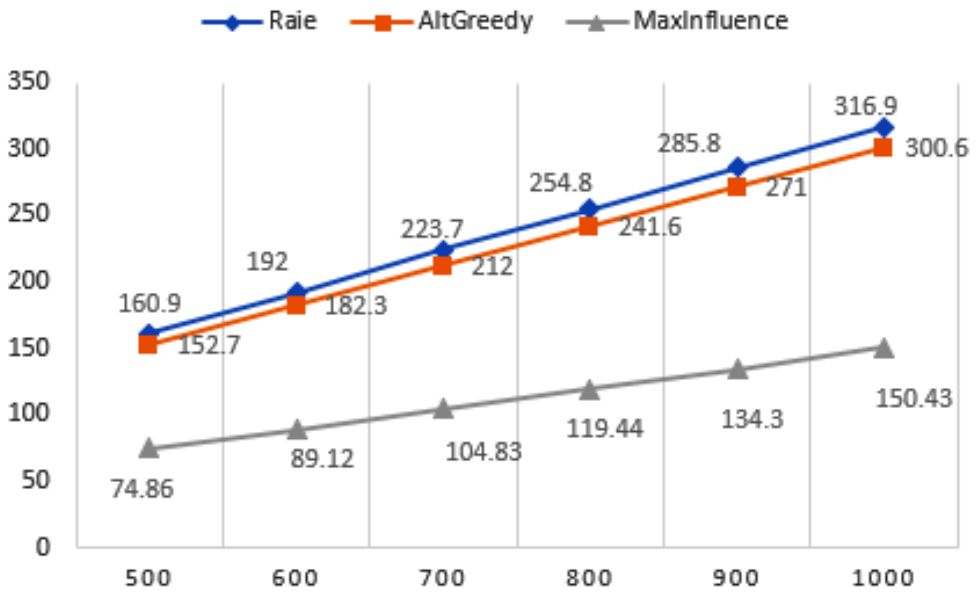

Figure 6: Average PIDS size on Barabasi-Albert generated graphs

Figure 7 presents the size of the set obtained when running both our algorithm and AltGreedy. We notice that over this selection of different graph types our algorithm was also able to generate smaller set sizes. In fact, we show an average of 50.2% decrease in set size from AltGreedy.

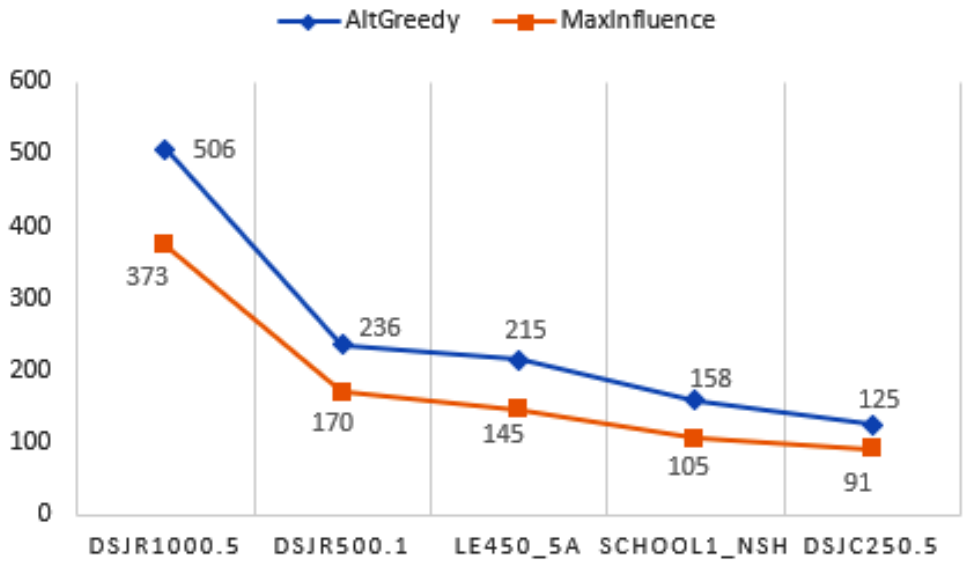

Figure 7: PIDS size on random graphs

Finally, we ran the test results on a selection of the Facebook subgraphs previously introduced.

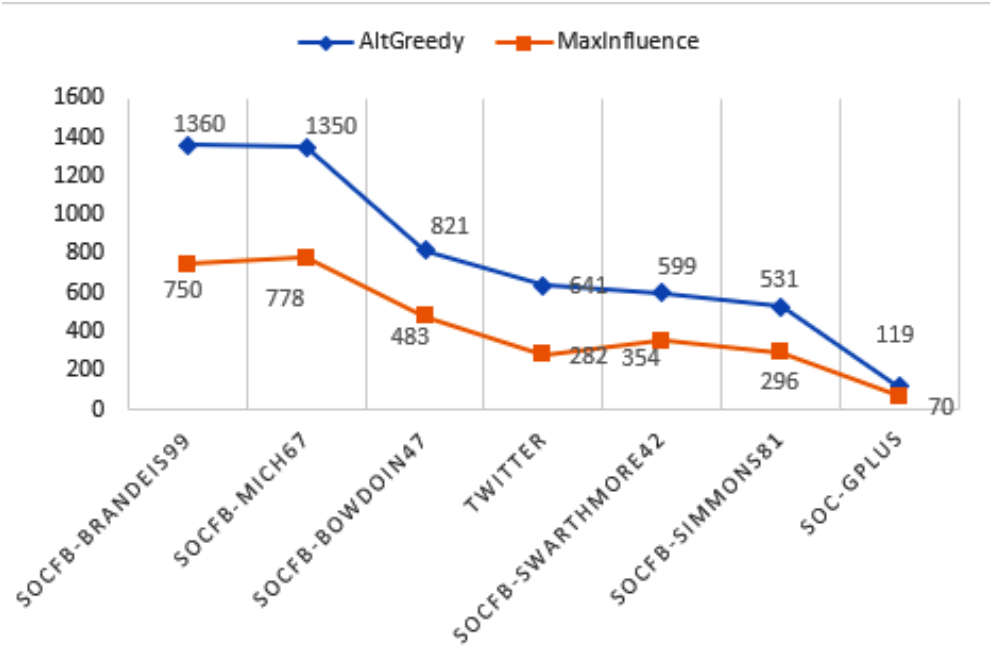

Figure 8: PIDS size on social subgraphs

Figure 8 shows the results obtained. Similar to the random graphs, our algorithm was

able to generate better set size. We show a decrease of 50% in size from AltGreedy. We also notice the efficiency of our algorithm by comparing results obtained on FB-Brandeis99 and FB-Mich67. In fact, FB-Brandeis99 contains 137566 edges over 3898 nodes, whereas FB-Mich67 contains 81902 edges on approximately the same number of nodes 3748. FB-Brandeis99 is considered more dense than FB-Mich67 and therefore allows faster influence propagation according to our decreasing function. From the results, we observe that running the "Maximum Influence" algorithm on both graphs caused a 44.8% decrease for FB-Brandeis99 and a 42.7% decrease for FB-Mich67. Both results are positive and important, however the dense aspect of FB-Brandeis99 allowed a faster propagation of influence, thus minimizing the positive set size.

This influence propagation algorithm requires more computation as it needs to traverse the graph to compute the influence. However, it generates a much smaller set size, which makes it more effective especially since these algorithms are expected to run offline. Usually when dealing with social networks we care more about the end result. Therefore we believe that the influence propagation is best used if we are seeking minimum set size and willing to compromise running time. Nevertheless, if the aim is to generate a small set with minimum time than it is best to use the previously introduced algorithms.

6.4 Results for IPDS

In this section we shall compare the set size obtained using the proposed influence propagation model to the sizes obtained using the original PIDS model. Knowing that this is a new model, we will compare to the random as well as social graphs we used in previous sections. We also selected graphs of different structures.
We started with random graphs, figure 9 presents the results obtained. It shows a comparison between the set size obtained by applying the PIDS model using our "Maximum Influence" algorithm and the set size obtained by applying the new model using our "IPDS" algorithm.

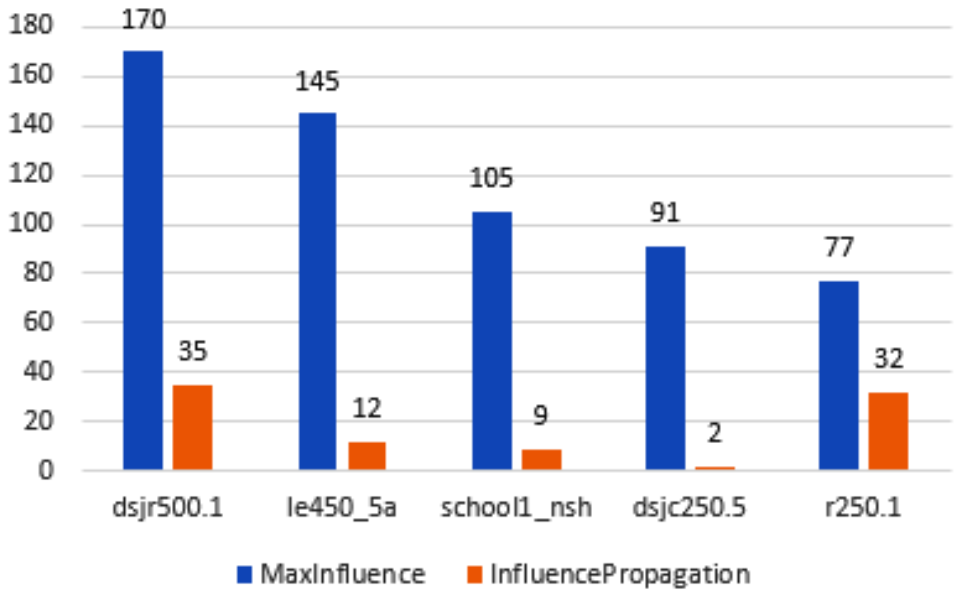

Figure 9: Influence propagation set size on random graphs

First of all we notice a huge decrease in set size due to the applied domino effect. In fact, the algorithm is shown to be effective because the denser a graph is the smaller the set size. Graph dsjr250.5 has 15667 edges connecting 250 nodes, so it is a dense graph. The influence propagation algorithm extracted 2 influential nodes instead of 91 due to the connections between nodes. If those 2 nodes posted the message and some of their neighbors interacted with it would be enough to positively affect the whole network. On the other hand, graph r250.5 shares the same number of nodes as dsjr250.5 but only connected through 886 edges. The propagation of a message through this graph becomes harder than denser ones. The influence propagation selected 32 nodes into the set, thus highlighting the lack of propagation.

Figure 10 presents the results obtained on social graphs. Similar to random graphs we notice the intuitive fact that dense graphs need less influential nodes. In fact, graph socfb-Brandeis99 has 137566 edges covering 3898 nodes, and it required 27 nodes to positively affect the graph. Whereas, graph socfb-Mich67 has 81902 covering 3748 nodes. Even though they both almost share the same number of nodes, graph socfb-Mich67 required 113 nodes to be covered. This is due to its less dense structure.

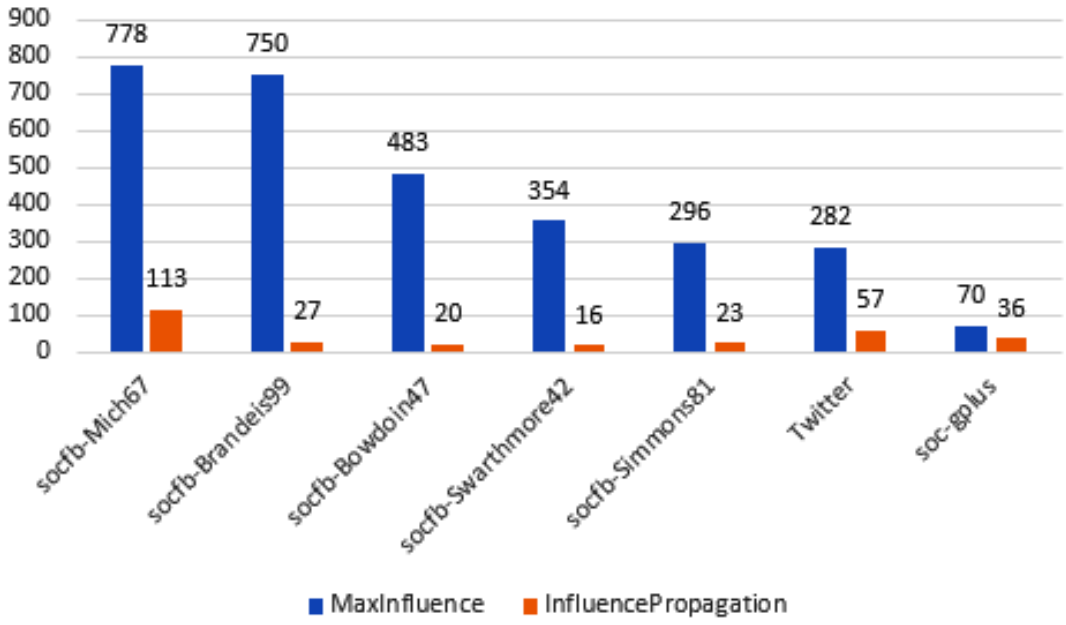

Figure 10: Influence propagation set size on social graphs

The Google+ graph soc-gplus also presents interesting results. As we previously explained this graph is characterized by its hub structure where a small number of vertices have high degrees. We notice that the algorithm selected 36 nodes into the set even though the graph has 39241 edges only covering 23627 nodes. This can be explained through the influence function as we recall that we add the need of a node itself to a decreasing coefficient of its neighbors need. Hub nodes will therefore have a high influence measure even though their propagation mainly covers the direct neighbors.

Chapter Seven

Concluding Remarks

In this thesis, we presented different algorithms able to detect influential individuals in a social network using the Positive Influence Dominating Set problem (PIDS) as a model. We started with four different algorithms out of which three showed meaningful results in satisfying different types of social networks. "Maximum Utility" for dense graphs, "Maximum Need" for sparse graphs, and "Minimum Need Maximum Utility" for graphs with multiple hub nodes. The experiments on these basic heuristic algorithms have been presented in a recent conference [10]. We then used a decreasing-sequence approach to measure the influence propagation and we used it in our "IPDS" algorithm. Lastly, we introduced a new influence propagation model that loosens the requirements enforced by PIDS while still maintaining maximum propagation.

Social networks are dynamic environments with constant changes in nodes and relations. Hence, the influence propagation problem can be studied while taking in consideration the frequency of interactions between nodes, and how those interactions change over time. In addition, an initially detected positive influence dominating set at time t might not satisfy the network at time $t+1$ if nodes or edges were added or removed. Therefore, we believe that a dynamic approach would be important, in future work, to repair the set according to the changes [8, 9]. Additionally, a turbo-charging approach similar to the work done in [7] on dominating set can be applied to the PIDS model.

Bibliography

[1] *Online social networks statistics, https://zephoria.com/top-15-valuable-facebook-statistics/.*

[2] *Random graphs. DIMACS, the Center for Discrete Mathematics and Theoretical Computer.*

[3] *Twitter graph - Stanford University, https://snap.stanford.edu/data/egonets-Facebook.html.*

[4] Graph definitions, 2005.

[5] F. N. Abu-Khzam, 2018. Personal Communication.

[6] F. N. Abu-Khzam, C. Bazgan, M. Chopin, and H. Fernau. Data reductions and combinatorial bounds for improved approximation algorithms. *J. Comput. Syst. Sci.*, 82(3):503–520, 2016.

[7] F. N. Abu-Khzam, S. Cai, J. Egan, P. Shaw, and K. Wang. Turbo-charging dominating set with an FPT subroutine: Further improvements and experimental analysis. In T. V. Gopal, G. Jäger, and S. Steila, editors, *Theory and Applications of Models of Computation - 14th Annual Conference, TAMC 2017, Bern, Switzerland, April 20-22, 2017, Proceedings*, volume 10185 of *Lecture Notes in Computer Science*, pages 59–70, 2017.

[8] F. N. Abu-Khzam, J. Egan, M. R. Fellows, F. A. Rosamond, and P. Shaw. On the parameterized complexity of dynamic problems with connectivity constraints. In Z. Zhang, L. Wu, W. Xu, and D. Du, editors, *Combinatorial Optimization and*

Applications - 8th International Conference, COCOA 2014, Wailea, Maui, HI, USA, December 19-21, 2014, Proceedings, volume 8881 of *Lecture Notes in Computer Science*, pages 625–636. Springer, 2014.

[9] F. N. Abu-Khzam, J. Egan, M. R. Fellows, F. A. Rosamond, and P. Shaw. On the parameterized complexity of dynamic problems. *Theor. Comput. Sci.*, 607:426–434, 2015.

[10] F. N. Abu-Khzam and K. H. Lamaa. Efficient heuristic algorithms for positive-influence dominating set in social networks. In *Proceedings of the 10th International Workshop on Hot Topics in Pervasive Mobile and Online Social Networking (HotPOST)*, 2018. Accepted for publication.

[11] F. N. Abu-Khzam, A. E. Mouawad, and M. Liedloff. An exact algorithm for connected red–blue dominating set. *Journal of Discrete Algorithms*, 9(3):252 – 262, 2011.

[12] R. B. Allan and R. Laskar. On domination and independent domination numbers of a graph. *Discrete Mathematics*, 23(2):73 – 76, 1978.

[13] S. Bhagat, A. Goyal, and L. V. Lakshmanan. Maximizing product adoption in social networks. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 603–612, New York, NY, USA, 2012. ACM.

[14] F. Bonchi. Influence propagation in social networks: A data mining perspective. *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, 1:2–2, 2011.

[15] M. Chellali and F. Maffray. Dominator colorings in some classes of graphs. *Graphs and Combinatorics*, 28(1):97–107, Jan 2012.

[16] N. Chen. On the approximability of influence in social networks. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA

'08, pages 1029–1037, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[17] E. DeLaViña, W. Goddard, M. A. Henning, R. Pepper, and E. R. Vaughan. Bounds on the k-domination number of a graph. *Applied Mathematics Letters*, 24(6):996 – 998, 2011.

[18] A. Dhawan and M. Rink. Positive influence dominating set generation in social networks. In *2015 International Conference on Computing and Network Communications (CoCoNet)*, pages 112–117, Dec 2015.

[19] R. Diestel. *Graph Theory. Graduate Texts in Mathematics*. vol. 173, 3rd edn. Springer, Berlin.

[20] Y. Fernandess and D. Malkhi. On spreading recommendations via social gossip. In *Proceedings of the Twentieth Annual Symposium on Parallelism in Algorithms and Architectures*, SPAA '08, pages 91–97, New York, NY, USA, 2008. ACM.

[21] E. Gallagher. MeToo hashtag network visualization, 2017.

[22] F. Grandoni. A note on the complexity of minimum dominating set. *Journal of Discrete Algorithms*, 4(2):209 – 214, 2006.

[23] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, Apr. 1998.

[24] M. A. Henning and N. J. Rad. Locating-total domination in graphs. *Discrete Applied Mathematics*, 160(13):1986 – 1993, 2012.

[25] B. Instances. Random graphs. Benchmark Instances, 2017.

[26] D. Kempe, J. Kleinberg, and E. 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*, KDD '03, pages 137–146, New York, NY, USA, 2003. ACM.

[27] K. Lewis, M. Gonzalez, and J. Kaufman. Social selection and peer influence in an online social network. *Proceedings of the National Academy of Sciences*, 109(1):68–72, 2012.

[28] H. Nguyen and R. Zheng. On budgeted influence maximization in social networks. *IEEE Journal on Selected Areas in Communications*, 31(6):1084–1094, June 2013.

[29] H. Raei, N. Yazdani, and M. Asadpour. A new algorithm for positive influence dominating set in social networks. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 253–257, Aug 2012.

[30] R. A. Rossi and N. K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[31] K. Saito, M. Kimura, K. Ohara, and H. Motoda. Super mediator - a new centrality measure of node importance for information diffusion over social network. *Inf. Sci.*, 329(C):985–1000, Feb. 2016.

[32] J. Thorpe. What the astounding number of 'Me Too' posts really mean, 2017.

[33] J. M. M. Van Rooij and H. L. Bodlaender. Exact algorithms for dominating set. *Discrete Appl. Math.*, 159(17):2147–2164, Oct. 2011.

[34] F. Wang, E. Camacho, and K. Xu. *Positive Influence Dominating Set in Online Social Networks*, pages 313–321. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[35] F. Wang, H. Du, E. Camacho, K. Xu, W. Lee, Y. Shi, and S. Shan. On positive influence dominating sets in social networks. *Theoretical Computer Science*, 412(3):265 – 269, 2011. Combinatorial Optimization and Applications.

[36] G. Wang, H. Wang, X. Tao, and J. Zhang. *Finding Weighted Positive Influence Dominating Set to Make Impact to Negatives: A Study on Online Social Networks in the New Millennium*, pages 67–80. Springer US, Boston, MA, 2014.

[37] Wikipedia. Six degrees of separation, 2018.

[38] W. Zhang, W. Wu, F. Wang, and K. Xu. Positive influence dominating sets in power-law graphs. *Social Network Analysis and Mining*, 2(1):31–37, Mar 2012.