The handle [http://hdl.handle.net/1887/29764](http://hdl.handle.net/1887/29764) holds various files of this Leiden University dissertation.

**Author:** Takes, Frank Willem  
**Title:** Algorithms for analyzing and mining real-world graphs  
**Issue Date:** 2014-11-19
In this chapter, the structural properties of the friendship graph of a large online social network consisting of 8 million users and close to 1 billion links are investigated. The main focus is on characterizing the prominent users that reside within the online social network based on their position in the graph. The derived structural node properties will then be used to steer an automated classification algorithm based on biased random walks for distinguishing between prominent and regular nodes (users). The effectiveness of the proposed approach is assessed using the online social network at hand, for which it is known which nodes have been manually identified as prominent individuals. It turns out that using the proposed random walk algorithm, it is possible to efficiently identify a large portion of the prominent nodes in the network, outperforming standard web-inspired measures such as HITS and PageRank. This chapter is based on:

6.1 Introduction

Nowadays, online social networks (OSNs) such as Facebook, Twitter and LinkedIn are extremely popular, with numbers as high as hundreds of millions of users and billions of friendship links. The main concept of these online social networks is simple: a user creates a profile with some personal attributes and then links this profile to other users, the so-called friends, creating a very large graph of befriended users: the friendship graph. In order to better understand the structure of social networks, the friendship graph is extensively being measured, modeled and mined [2, 76, 103].

In this chapter, we consider the (former) Dutch online social network HYVES, which was online for about 9 years between 2004 and 2013. We will investigate one full snapshot of the network which was obtained in September 2010. At that time, little over 8 million users participated in the friendship graph of the network, together forming well over 900 million friendship links. The main question discussed in this study is how we can automatically identify the most important users in this online social network, based only on the structure of the network. Being able to identify such prominent actors has various useful applications. For example, companies nowadays frequently use online social networks for their viral marketing [83] campaigns, in which they want to deliver a message to as many people as possible through the linking structure of the social network. Prominent nodes may just be the places where such a campaign should start in order to efficiently reach a large number of people [61].

A relevant question is then how prominence or importance of a node within a network should actually be defined for the network that we consider. While various definitions may be correct, we will assume that someone is prominent, or important, or influential, if he or she has some celebrity status (famous politicians, soccer players, artists, movie actors, etc.) in the real world. We believe that this definition can be justified based on the fact that both online and in the real world, celebrities have a certain status, or reputation. If a celebrity promotes a certain brand, people are far more likely to identify with that brand, compared to when a regular person would promote the brand [105]. Within the online social network Twitter, tweets originating from people like president of the United States are far more likely to be “retweeted” than when they would come from a normal person in the network. Thus, the president could be seen as a more prominent actor in the network.

In this chapter we first study the difference in characteristics between regular and prominent nodes. We consider various existing methods of determining the importance of nodes in the friendship graph of an online social network, and then introduce a new method which is based on the characteristics of the prominent nodes that we obtained. Next, we verify the performance of the discussed techniques empirically on
a large complete dataset of the online social network. For this network, we know exactly which users are considered to be prominent, allowing us to verify the obtained results against a predefined ground truth.

The rest of the chapter is structured as follows. In Section 6.2 we formally define the main problem, after which we discuss related and previous work in Section 6.3. In Section 6.4 we discuss the characteristic properties of prominent nodes, and how these properties can be incorporated in a random walk algorithm. Next, in Section 6.5 we describe the dataset that we will use in Section 6.6 to compare the performance of the discussed methods. Section 6.7 concludes.

6.2 Preliminaries

In this section we will discuss some basis concepts, formulate the problem statement and describe the application domain.

6.2.1 Definitions

We are given a friendship graph $G = (V, E)$, where $V$ is the set of $n = |V|$ nodes (individuals) and $E \subseteq V \times V$ is the set of $m = |E|$ edges (connections). The graph is undirected, meaning that the set of edges is symmetric, so if $(u, v) \in E$ then also $(v, u) \in E$. A path or walk from $u$ to $v$ is a sequence of edges, starting with an edge containing $u$ and ending with an edge containing $v$. The distance $d(u, v)$ between two nodes $u$ and $v$ is defined as the length of a shortest path between these nodes. Because the graph is undirected, $d(u, v) = d(v, u)$ for all $u, v \in V$. A connected component is a maximal subset of the set of nodes $V$ such that the any pair of nodes within this subset is connected via one or more edges.

We define the neighborhood $N(v)$ of a node $v$ as the set of nodes at distance 1 of $v$, more specifically: $N(v) = \{u \in V \mid (u, v) \in E\}$. The degree of a node $v$ is defined as the number of edges starting (or ending) at node $v$, i.e., the size of the neighborhood of that node: $\deg(v) = |N(v)|$.

6.2.2 Problem statement

Amongst the nodes in the network, there is an initially unknown set $W \subseteq V$, of size $k = |W|$, which contains the nodes that are considered to be “prominent”. Logically, $k \leq n$, but in practice, $k$ is much smaller than $n$, as only a small portion of the nodes is typically considered to be prominent. The main goal is to find, given only the graph $G = (V, E)$, an as small as possible subset $I \subseteq V$ such that $|I \cap W|$ is maximal, i.e., we are trying to find as many prominent nodes as possible.
In this chapter we describe various existing, derived, and new methods for determining node importance. For each of these methods \( M \) we assign a normalized value \( C_M(v) \in [0, 1] \) to each node \( v \in V \) which determines its importance. We will assume that higher values indicate a higher level of importance. In order to determine the performance of a method \( M \), we sort the list of nodes by their importance value \( C_M(v) \) in descending order, and define \( I \) to be the top \( \ell \) nodes of this sorted list. The precision and recall, \(|W \cap I|/|I|\) and \(|W \cap I|/|W|\), respectively, will ultimately determine the performance of a method \( M \). More generally, we can say that the F-measure, \((2 \times \text{precision} \times \text{recall})/(\text{precision} + \text{recall})\), measures the balance between the two. Note that if \( \ell = k \), precision, recall and F-measure are equal.

An example of a network with 18 nodes of which 2 nodes (\( F \) and \( L \)) are manually labeled as “prominent”, is given in Figure 6.1. If some method would determine that nodes \( F \) and \( J \) are the prominent nodes, then \( W \cap I = \{F, L\} \cap \{F, J\} = \{F\} \) and the performance of this method (in terms of both precision and recall) would be 50%.

### 6.2.3 Online social networks

Topological properties of online social networks have been studied in great detail [103]. Social networks are usually sparse and contain one large connected component consisting of the majority of the nodes, called the giant component. Often, there are a few smaller isolated communities, as well as various singletons [2]. Furthermore, it is well-known that the structure of online social networks resembles that of real-life social networks [139]. Online social networks generally belong to the class of small-world networks [140]. Such networks are characterized by relatively small pairwise distances between nodes, i.e., the average distance between two nodes is very small.
Chapter 6. Identifying Prominent Actors in Online Social Networks

(typically around four to eight) compared to the total number of nodes (easily more than one million). Online social networks also tend to exhibit a node degree distribution that follows a power law: there are relatively few nodes with an exceptionally high degree, and many nodes with a low(er) degree. The high degree nodes function as hubs, and are often grouped in a densely connected core, realizing the short pairwise distances between the more “peripheral” nodes.

6.3 Related work

Various studies have addressed the problem of identifying the prominent actors within large (online) (social) networks. Some related work deals with finding experts, or who can be trusted within some semantic social network [50, 148]. We will distinguish from these methods by not considering semantics, but only structural properties of the nodes.

Centrality measures have been popularized by social scientists as possible measures for the importance, or “prestige” of a person within a social network [139]. These measures identify nodes that have a central position based on the structure of the network. Such a central position usually means that the node is connected to many other nodes, possibly indirectly via some (short) path(s). Degree centrality is by far the simplest and most common measure, and is in case of an online social network simply equal to the number of friends of a user. As we will see later on, the number of friends is a good indication of the prominence of a node, but definitely not perfect. The complexity of computing other more complex distance-based centrality measures is in the order of $O(mn)$ or worse [27], and therefore not considered in this chapter.

Propagation-based methods such as PageRank [107] are known to be very successful in determining the importance of web pages [80], citation networks [31] and Wikipedia articles [69]. Therefore, in this study we will also consider the PageRank measure $C_{PR}$ as defined in Section 5.4.1 as a method for identifying the prominent actors in our online social network. We will furthermore consider HITS:

Hyperlink Induced Topic Search (HITS) $C_{HITS}$. HITS [70] is a technique for assessing node centrality which assigns a hub score $h(v)$ and an authority score $a(v)$ to every node $v$ in the graph, both initialized to 1. Then, for a certain number of iterations (100 iterations is usually enough for convergence), for each node $v$ the value of $a(v)$ is set to the sum of the (normalized) $h(u)$ values of the nodes $u$ for which there exists a link $(u, v)$, after which for each node $v$ the value of $h(v)$ is set to the sum of the (normalized) $a(w)$ values of the nodes $w$ for which there exists a link $(v, w)$. For the centrality measure $C_{HITS}$, we use the authority score $a(v)$. 
The NODERanking algorithm was proposed in [113] as a method based on random walks for determining the importance of authors in a directed citation network. Random walk algorithms generally traverse the graph, moving to a random neighbor with probability $1 - p$, and jumping to a random node with probability $p$. The distinguishing property of the NODERanking algorithm is that it jumps with a probability depending on the degree of the current node, where a low degree indicates a high jumping probability. In Section 6.6.2 we will compare approach discussed in the next section with each of the algorithms discussed above.

6.4 Prominent nodes

We will now outline the proposed approach for finding the prominent nodes in an online social network. First we sketch the expected characteristic properties of the target nodes. After that, we will describe an algorithm based on random walks which uses these node properties to guide the walk towards the prominent nodes.

6.4.1 Node properties

The simplest intuition that we have about prominent people, is that they have a large number of connections. Therefore we expect the degree of a node to play a great role in determining the importance of a node. So we could state that degree centrality, determining the importance of a node $v$ based on its number of connections, could be a good first indication of importance, formally:

$$C_d(v) = \frac{|N(v)|}{n-1}$$

However, there may be nodes in the graph with many connections, that are not prominent, or vice versa, prominent people with a smaller number of connections.

Let us recall several observations regarding social networks in general, which have been described in literature. People tend to use social networks for two reasons: social searching, and social browsing [78]. These two terms refer to reconfirming real-life friendships online, and browsing for completely new relationships, respectively. Another common concept is that of triadic closure: the vast majority of all friendships formed within a social network takes place between two people who have at least one friend in common. The probability of a link being formed has been shown to increase with the number of common acquaintances [75] as well as with the degree of a node, a phenomenon called preferential attachment [104]. For example, in the graph in Figure 6.1, the connection $(A, B)$ would be more likely to appear than the connection $(A, K)$, as $A$ and $B$ have node $C$ as a common friend, and $A$ and $K$ have
no direct common friends. The connection \((A, E)\) would in turn be more likely than \((A, B)\), as \(E\) already has a higher degree.

Based on the observations above, we expect that a user adding someone like the president of the United States, is not within the circle of friends of the president, making this friendship more like a result of the aforementioned social browsing instead of searching. More generally, we argue that the friends of prominent nodes have fewer connections in common than regular nodes. We call this concept the neighborhood density \((nd)\) of a node:

\[
C_{nd}(v) = 1 - \sum_{w \in N(v), |N(w)| > 1} \frac{|N(w) \cap N(v)|}{(|N(w)| - 1) \cdot |N(v)|}
\]

Here, the numerator defines the number of common connections, whereas the denominator normalizes the result so that it is independent of the degree of \(v\) or the degree of \(w\). If \(|N(v)| > 1\), \(C_{nd}(v)\) is minimal in case \(N(v)\) is fully connected, and becomes larger as a smaller fraction of the neighborhood is interconnected. The proposed measure differs from related measures such as the clustering coefficient in a sense that this measure normalizes for both the neighborhood size of the considered node as well as the neighborhood of each of the adjacent nodes.

### 6.4.2 BiasedRandomWalk

We believe that a combination of the two measures from Section 6.4.1 will be able to identify a large portion of the various prominent actors. Therefore we devised an algorithm based on random walks, which has a parameterized bias towards each of these properties. The random walk algorithm has as an advantage that it only needs local information to determine the next state of the algorithm, allowing the algorithm to run efficiently even when the entire graph can not be stored in main memory.

The proposed algorithm, called \textsc{BiasedRandomWalk} (BRW), takes as input an unweighted graph \(G = (V, E)\) and parameters \(N\), \(p\) and \(\alpha\), and outputs a function value \(C_{BRW}(v)\) for each node \(v \in V\) in the graph, determining its importance. Here \(N\) is the number of steps in the random walk algorithm, \(p\) is the jumping probability (which we fix at 0.15 as suggested in literature [84]), and \(\alpha\) is used to define the focus on either one of the two measures that we discussed.

The procedure is outlined in Algorithm 1, and works as follows. After setting some initialization values in lines 3–6, the algorithm starts by selecting a random node from \(V\) (line 7). After that, for \(N\) iterations, the algorithm repeatedly increases the function value (line 9) of the current node \(v\) by \(1/N\) (to keep the final function value within \([0; 1]\)). Then, the algorithm either selects a new node from the neighborhood \(N(v)\) of
Algorithm 6.1 BiasedRandomWalk

1: **Input:** Graph $G = (V, E)$, $N, p, \alpha$
2: **Output:** List $C_{BRW}$, containing $C_{BRW}[v]$ for each node $v \in V$
3: **for** $v \in V$ **do**
4: \hspace{1em} $C_{BRW}[v] \leftarrow 0$
5: **end for**
6: $i \leftarrow 0$
7: $v \leftarrow$ RANDOMNODEFROM($V$)
8: **while** ($i < N$) **do**
9: \hspace{1em} $C_{BRW}[v] \leftarrow C_{BRW}[v] + (1/N)$
10: \hspace{1em} **if** ($\text{rand}(0, 1) > p$) **then**
11: \hspace{2em} $v \leftarrow \text{BIASSELECTFROM}(N(v), \alpha)$
12: \hspace{1em} **else**
13: \hspace{2em} $v \leftarrow$ RANDOMNODEFROM($V$)
14: \hspace{1em} **end if**
15: \hspace{1em} $i \leftarrow i + 1$
16: **end while**
17: **return** $C_{BRW}[v]$ using the function $\text{BIASSELECTFROM}()$ with probability $1 - p$ (line 11), or jumps to a completely random node with probability $p$, denoted by the $\text{RANDOMNODEFROM}()$ function (line 13).

If in the function $\text{BIASSELECTFROM}()$ a random neighbor is selected, the algorithm would be a plain random walk algorithm. However, in this specific case, the function $\text{BIASSELECTFROM}()$ selects a node with a probability dependent on different function values of the prominence measures, as we expect that each of these functions tells us something about the probability of that node being prominent. This means that given current node $v$, the probability $P(w)$ of selecting node $w \in N(v)$ in the next step, is equal to:

$$P(w) = \frac{\alpha C_d(w) + (1 - \alpha)C_{nd}(w)}{\sum_{u \in N(v)} (\alpha C_d(u) + (1 - \alpha)C_{nd}(u))}$$

Here, $\alpha \in [0, 1]$ defines the focus on either one of the two measures. Not surprisingly, setting the value of $\alpha$ to 1 resulted in roughly the same result as degree centrality. A value of 0 for $\alpha$ did not find any of the prominent actors, which we believe is due to the fact that even though $C_{nd}$ is normalized, the degree plays a significant role in identifying the prominence of a node, and very low degree nodes can still get
a high neighborhood density score. In an attempt to linearly tune parameter $\alpha$ with steps of 0.1, it turned out that any value between 0.2 and 0.8 gave consistently better results than a lower or higher value. Thus apparently, both of the discussed measures to some extent influence the final result. Therefore we fixed the parameter to 0.5 to give equal focus to both measures. Finally, $N$, the number of iterations, should be set to a value significantly larger than the number of nodes $n$. We investigate the value of $N$ more precisely in Section 6.6.2 when we look at the convergence of the BiasedRandomWalk algorithm.

## 6.5 Dataset

In this chapter, we consider an anonymized full snapshot of the friendship graph of the Dutch online social network HYVES from September 2010. Some statistics such as the number of nodes and edges, the average degree and the density (defined as the number of edges divided by the maximum number of edges, i.e., $m/(n(n-1))$) of this graph are given in Table 6.1.

Although the graph has almost 10,000 connected components (see Figure 6.2 and note the logarithmic vertical axis), the vast majority of the nodes resides within the largest connected component. According to the statistics provided on the website of the social network at the time the snapshot of the network was made, the website

<table>
<thead>
<tr>
<th>Data</th>
<th>Full network</th>
<th>Largest component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>8,113,017</td>
<td>8,083,964</td>
</tr>
<tr>
<td>Links</td>
<td>912,120,070</td>
<td>912,067,984</td>
</tr>
<tr>
<td>Average degree</td>
<td>112</td>
<td>99.6%</td>
</tr>
<tr>
<td>Density</td>
<td>$1.386 \cdot 10^{-5}$</td>
<td>$1.396 \cdot 10^{-5}$</td>
</tr>
<tr>
<td>Connected components</td>
<td>9,926</td>
<td>9,926</td>
</tr>
</tbody>
</table>

Table 6.1: Friendship graph statistics.
had over 11 million members. This means that there were roughly 3 million users that were not participating in the friendship graph at all. The node degree distribution of the graph is shown in Figure 6.3. This distribution follows a clear power law, and has an even longer tail than visible, going all the way up to one node with a degree of 285,827. Note that the social network had a maximum number of friends at 1,000, 1,500 and 2,000 which could only be removed upon request with the network administrators, causing some noise in the tail of the degree distribution.

The node-to-node distance distribution shown in Figure 6.4 demonstrates how the network adheres to the small-world property (see Section 6.2.3). This distribution was obtained by sampling 100,000 node pairs \( u, v \in V \), computing the value of distance \( d(u, v) \), and then counting for each obtained distance value how frequently it was observed. This distance distribution was also used to derive the average distance of 4.75 listed in Table 6.1. The radius and diameter of the graph were computed using the algorithms described in Chapter 2 and Chapter 4.

A set of nodes \( W \) of size \( |W| = 4,867 \) (0.06%) has been manually labeled by the network administrators as “prominent”. This subset consists of various Dutch politicians, artists, athletes and actors and will be considered as a ground truth for assessing the performance of different measures of prominence. We note that all prominent nodes are part of the giant component.

Figure 6.2: Component size distribution (excluding the largest component of 8,083,964 nodes).
Figure 6.3: Degree distribution.

Figure 6.4: Distance distribution.
To the best of our knowledge, the only other study of a full snapshot of the full HYVES graph is provided in [35]. In this work, similar observations regarding the structural properties of the network are reported, and the distribution of various node attributes such as the age of the user are given.

6.6 Experiments

In this section we will compare the proposed algorithm to various existing approaches for determining node importance in networks. The algorithm as well as other discussed measures have been implemented in C++. Experiments were run on a 3.2GHZ machine with 10GB memory, allowing us to keep the large network dataset in memory. We start with a verification of the different node properties, after which we assess the performance of the BiasedRandomWalk algorithm.

6.6.1 Node properties

We have verified the two measures discussed in Section 6.4.1 on the discussed online social network dataset (see Section 6.5 for a description of the dataset). From the degree distribution shown in Figure 6.3, we can conclude that prominent users indeed have many more friends than regular users. A simple strategy for identifying prominent users would be to say that any user with more than for example 2,000 friends is prominent. However, this would not only be incorrect because such a cut-off may be domain-specific and dependent on the type of social network, but it will also not help to identify the significant number of prominent users with anywhere between 0 and 2,000 friends. For this degree range, there is also a (much larger) number of regular users with the same degree (notice the logarithmic vertical axis of Figure 6.3).

For the neighborhood density $C_{nd}$, we computed this value for 1,000 randomly selected regular nodes and 1,000 randomly selected prominent nodes, and found values of $1 - 0.131 = 0.869$ and $1 - 0.035 = 0.965$, respectively (the one minus notation is used to indicate the significant difference in the density summation of the measure of neighborhood density, see Section 6.4.1). This result is consistent with the intuition of regular users having a relatively more dense neighborhood than prominent users. Clearly, both of the properties that we discussed are related to the prominence of a user in the considered network.

6.6.2 BiasedRandomWalk

To verify the applicability of the proposed random walk algorithm, we first consider its convergence in terms of whether or not the set of identified prominent nodes becomes
consistent as the random walk algorithm runs. The result is displayed in Figure 6.5 for a 1 million node sample of the original graph. Clearly, the obtained value of the precision of the algorithm converges. We experimented with various sample sizes (10,000, 100,000 and 1 million nodes) of the original 8 million node dataset, and consistently found that after $N = 10 \cdot n$ iterations, the precision did not show any significant improvement. Thus, for this network we conclude the parameter $N$ can be set to $10 \cdot n$ to ensure a suitable result is obtained, which means that the running time would scale linearly with the number of nodes.

### 6.6.3 Results

The results of applying the various algorithms to the full friendship graph are outlined in Table 6.2. Here we compare the results of each of the methods based on $k = \ell$, meaning that we select exactly as many prominent people as there are in the dataset. Recall from Section 6.4 that we thus select the top $\ell = k$ nodes from the list of nodes sorted by their prominence function value. The BiasedRandomWalk algorithm was executed with a budget of $N = 10 \cdot n = 81$ million iterations. Except for degree centrality which is fully deterministic, results are averaged over 10 runs in order to flatten the effect of outliers due to the inherent randomness of the approaches, resulting in standard deviations of less than 3% for each of the methods.
<table>
<thead>
<tr>
<th>Measure</th>
<th>Precision</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.06%</td>
<td>0sec</td>
</tr>
<tr>
<td>HITS</td>
<td>51.4%</td>
<td>10min</td>
</tr>
<tr>
<td>PageRank</td>
<td>56.4%</td>
<td>10min</td>
</tr>
<tr>
<td>RandomWalk</td>
<td>63.9%</td>
<td>1min</td>
</tr>
<tr>
<td>NODERANKING</td>
<td>64.0%</td>
<td>1min</td>
</tr>
<tr>
<td>Degree Centrality</td>
<td>64.2%</td>
<td>0sec</td>
</tr>
<tr>
<td>BIASED RANDOM WALK</td>
<td>70.1%</td>
<td>13min</td>
</tr>
</tbody>
</table>

Table 6.2: Precision and indication of computation time of various importance measures with \( k = \ell \).

As a baseline for comparison we could say that if we were to select \( \ell \) random nodes from the complete set of nodes \( V \) to form the set \( W \), we would on average find 0.06% of the prominent nodes in the network. Degree centrality, RandomWalk and the NODERANKING algorithm have roughly equal performance, and greatly improve upon this baseline by already identifying about 64.0% correctly. It turns out that HITS and PageRank performed significantly worse, which might be due to the fact that these three algorithms were at least initially designed for directed graphs. The proposed BIASED RANDOM WALK method improves another 6 percentage points upon degree centrality, the best performing existing method, demonstrating the advantage of looking at both the degree and the neighborhood density during the random walk.

As for the running time, obviously random selection and degree centrality require no additional computation time. PageRank and HITS both iterate over the set of edges 100 times to update the node values based on their neighboring nodes, each taking roughly 10 minutes in doing so. The random walk algorithms each run for \( 10 \cdot n \) steps, where in case of the plain random walk and the NODERANKING algorithm, no additional computation is done in each node. The BIASED RANDOM WALK method picks the neighbor with the highest \( C_{BRW} \) value which takes some computation time (but, assuming the graph is static, can be cached), running in little over 13 minutes in total.

One may argue that the number of prominent actors in a network is not always known in advance. Therefore we also did experiments in which we varied \( \ell \) between 0.01 \( k \) and 2 \( k \) on the 1 million node sample of the full dataset, allowing the study of the precision, recall and F-measure curves. Assuming that we want to take a number of false positives for granted as is often permitted in practical applications, we may choose to focus solely on maximizing the recall value. Therefore, the recall value for each of the approaches as a function of the fraction of \( k \) is presented in Figure 6.6. In Figure 6.7 we furthermore present a comparison of the different F-values. At 0.75 \( k \), the F-value appears optimal for BIASED RANDOM WALK, and we would have a good
Figure 6.6: Recall for each of the methods.

Figure 6.7: F-value for each of the methods.
balance between recall and precision. This optimum lies slightly lower around $0.70 \cdot k$ for the measure of degree centrality (we left out the other two identically performing measures because they both performed somewhat equal to degree centrality). Finally, note that for small values of $\ell$ (up to $0.3 \cdot k$), the obtained result is always perfect regardless of the method considered: apparently the top of the list is the same for each of the measures. It turns out these nodes simply had an enormously high degree (over 2,000, see Figure 6.3), and were therefore selected by each of the methods. In general, we can conclude that the proposed BiasedRandomWalk method works well, and is able to identify a significant portion of the prominent actors of the friendship graph of the considered online social network.

6.7 Conclusion

We have outlined various characteristic node properties of prominent actors in an online social network, and used these properties to create an algorithm for identifying prominent actors. Our algorithm, called BiasedRandomWalk, combines the measures of degree centrality and neighborhood density in a random walk algorithm by having a bias towards nodes with high values for these two measures. Neighborhood density can be seen as a measure of the percentage of triadic closure, which is significantly lower for prominent actors as compared to regular nodes. On the other hand, the degree of prominent nodes is typically high. Experiments show that the proposed method works quite well, as standard centrality measures such as degree centrality, HITS and PageRank are outperformed in terms of precision, recall and F-measure.

In future work we would like to verify the extent to which the proposed random walk technique can be applied to other types of (social) networks, and how we can make the method parameter-free, or determine good parameter values based on properties of the network. We also want to consider the temporal aspect of importance, and study how properties of prominent nodes within a social network change over time.

Acknowledgment

We thank HYVES for making the structure of their (anonymized) friendship graph available.