Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence

TONIC: Target Oriented Network Intelligence Collection for the Social Web

Roni Stern
SEAS
Harvard University
Cambridge MA, USA

Liron Samama, Rami Puzis, Tal Beja, Zahy Bnaya, Ariel Felner
Information Systems Engineering
Ben Gurion University of the Negev
Be'er Sheva, Israel

Abstract

In this paper we introduce the Target Oriented Network Intelligence Collection (TONIC) problem, which is the problem of finding profiles in a social network that contain information about a given target via automated crawling. We formalize TONIC as a search problem and a best-first approach is proposed for solving it. Several heuristics are presented to guide this search. These heuristics are based on the topology of the currently known part of the social network. The efficiency of the proposed heuristics and the effect of the graph topology on their performance is experimentally evaluated on the Google+ social network.

Introduction

Web-based *social networks* (SN) such as Facebook, Twitter, and Google+ are a part of everyday life for many people around the world. These SNs are a source of personal information of their users and may even contain sensitive commercial or security related information. Commercial companies, government agencies and even individual people often utilize this abundance of data to extract information about a given person of interest. For example, it is common practice for most commercial companies to inspect the Facebook and LinkedIn profiles of candidate employees in order to extract information about past projects, colleagues and managers. Government agencies may also explore the SN looking for information on terrorists and organized crime.

Information about a given person of interest, denoted as the *target*, can be automatically collected from its SN profile using information extraction (IE) techniques (Chang et al. 2006). However, awareness to privacy issues and to data leakage cause many users to tighten their SN privacy settings limiting the access to their profiles. This paper addresses the case where the target profile is inaccessible to third parties and therefore, information about the target cannot be collected from its profile. We assume that the only way to collect information about the target is through other profiles such as target's SN friends. Note that, while the target can easily modify its own privacy settings, it is prohibitively challenging to cause other SN profiles to conceal information about the target that they may expose. For example, a SN profile of a friend of the target may contain photos of the target, public posts made by the target or posts about the target.

Therefore, since the target profile itself is inaccessible, collecting information about the target consists of two tasks. The first task is to find SN profiles that potentially contain information about the target. The second task is to analyse these profiles and extract information about the target, if such information exists. The second task, referred to in this paper as *profile acquisition*, can be performed using standard IE and web scrapping techniques (Chang et al. 2006; Tang et al. 2010; Pawlas, Domanski, and Domanska 2012). In this paper we focus on optimizing the first task, i.e., finding profiles that contain information about the target while minimizing the number of acquisitions. We refer to the problem of finding such profiles as the Target Oriented Network Intelligence Collection (TONIC).

TONIC is reminiscent of the *link prediction* problem, where the goal is to predict whether pairs of nodes in a social network are directly connected (Fire et al. 2011). By contrast, in TONIC the relation between a profile and the target can be verified by acquiring that profile. The question is how to know which profile to acquire. Furthermore, TONIC operates in an initially unknown SN graph, while in topology-based link prediction, a large part of the network topology is given. Obtaining personal information from a SN by intelligent crawling (Chau et al. 2007; Mislove et al. 2007; Korolova et al. 2008) or other methods (Bonneau, Anderson, and Danezis 2009) was previously discussed, where the goal was to uncover large portions of a SN. In TONIC we wish to retrieve information about a specific target and avoid further crawling, requiring different problem formulation and heuristics.

We solve TONIC with Artificial Intelligence techniques. Specifically, TONIC is formalized as a heuristic search problem and several heuristics are introduced to guide a best-first search. In particular, we propose a robust *Bayesian Promising* heuristic that substantially outperforms other heuristics. Importantly, all investigated heuristics are based solely on the topology of the SN being crawled, and are thus orthogonal to the details of the profile acquisitions.

Problem Description

Next, we provide the technical background and terminology required to define the TONIC problem formally. The basic

entity in TONIC is the *profile*, which is associated with a specific SN (e.g., Facebook or Google+). Profiles are connected to each other via a "friendship" relation. The *list of "friends"* (LOF) of a given profile p is denoted by LOF(p).

We denote profiles that contain information about the target as *leads*. A profile p is a lead if (1) the target is contained in $LOF(p)$, or (2) p contains a post by the target, or (3) the target is tagged in one of the photos shown in p. Other criteria for determining whether a profile is a *lead*, e.g., using sophisticated IE techniques, are also possible. Information can be extracted from a profile in several ways. Scrapping, i.e. parsing the web pages that are exposed by the SN services, such as the time-line in Facebook. Application programming interface (API) exposed by SNs provide convenient way to collect information including the LOF of a profile. In general, IE from SN profiles is an active field of research (Tang et al. 2010; Pawlas, Domanski, and Domanska 2012, inter alia). In this work we assume that the IE aspects are encapsulated within an external function that we refer to as *profile acquisition*.

Definition 1 (Profile Acquisition) Acquiring *a profile p identifies if p is a lead or not. If p is a lead then $LOF(p)$ is extracted (for further use).*

Given a target profile, the task of TONIC is to find *leads* by searching through the SN graph. The input to TONIC, in addition to the target, is a set of *initial leads*. The expected output of TONIC is a set of other leads. In general, every SN profile may be a lead and thus might be considered for acquisition. However we assume that *friends* of leads (i.e., profiles that appear in leads' LOFs), are more likely to be leads than randomly selected SN profiles. This assumption is motivated by previous work that showed that SN profiles of friends tend to exhibit similar attributes (Altshuler et al. 2012; Fire et al. 2012). Thus, we refer to friends of leads that were not acquired yet as *potential leads*. In this work, we focus the search by only allowing acquisition of *potential leads* but not of any other profiles. The main challenge is thus to choose which potential lead to acquire next. Future work will investigate alternative approaches, where profiles that are not friends of leads can also be acquired.

Acquiring a profile incurs costs in terms of both computational resources and network activity. In addition, most SN services limit automatic web scrapping attempts as well as massive exploitation of their API. Moreover, one might wish to hide the fact that information is collected about a specific target. In this paper we encapsulate all aspects of the profile acquisition cost into a single constant associated with activating a profile acquisition. This leads to the following definition of the TONIC problem.

Definition 2 (The TONIC Problem) *Given a target profile, a set of initial leads and a budget b, the TONIC problem is to find maximum leads with at most b acquisitions.*

Other variants of the TONIC problem can also be formulated. For example, finding a fixed number of leads with minimum acquisitions. An "anytime" variant also exists, where an end-user may stop the search for leads unexpectedly. The algorithm and heuristics proposed in this paper are expected to be effective for all these variants.

Algorithm 1: Best-first search for a budgeted TONIC

```
   Input: Target, the target profile
   Input: Budget, the number of allowed focused queries
   Input: InitialLeads, the set of initial leads
   Output: FoundLeads, the leads found
1  FoundLeads ← ∅
2  OPEN ← InitialLeads
3  CLOSED ← ∅
4  while OPEN not empty AND Budget > 0 do
5      best ← ChooseBest(OPEN)
6      Add best to CLOSED
7      Acquire best
8      if best is a lead then
9          Add LOF(best) \ CLOSED to OPEN
10         Add best to FoundLeads
11     Decrease Budget by 1
12 return FoundLeads
```

TONIC as a Heuristic Search Problem

We model TONIC as a heuristic search problem on graphs, as follows. We denote the searched graph as $G = (V, E)$ where V is a set of SN profiles and E is a set of link such that $(v_1, v_2) \in E$ if $v_1 \in LOF(v_2)$. *Expanding* a node corresponds to acquiring a profile. Most heuristic search research deals with finding a path from an initial node to a predefined goal node. In contrast, the goal of TONIC is to find maximum number of leads while applying at most b acquisitions. While less common, such *utility* oriented search algorithms have been previously discussed in the literature (Edelkamp, Jabbar, and Lluch-Lafuente 2005; Stern, Puzis, and Felner 2011).

TONIC is a special case of *searching in an unknown graph* (Stern, Kalech, and Felner 2012), where expanding a node requires some external action, and the goal is to minimize the number of node expansions. Following previous work on searching in an unknown graph we propose to solve the TONIC problem with a best-first search approach, as outlined in Algorithm 1. Best-first search algorithms maintain two lists, an open list and a closed list, denoted by OPEN and CLOSED, respectively. We initiate OPEN with the initial leads (line 2) while CLOSED is initially empty (line 3). In every iteration, a single profile ($best$) is chosen from OPEN and is acquired (lines 5–7). If it is a lead then its LOF is added to OPEN (line 9). This process continues until the number of acquisitions exceeds the $Budget$ or until all reachable profiles have been acquired.

As stated above, expanding a node implies its acquisition. Following standard search terminology, the profiles in the social network can be partitioned to:
Expanded: profiles that were acquired and
Generated: potential leads.

Figure 1 presents an example execution of the best-first search shown in Algorithm 1. Here and in the rest of the paper black circles represent *leads*, gray – *potential leads*, white – *non-leads*, and the target is represented with a double border circle. The target is marked by t, and there are two

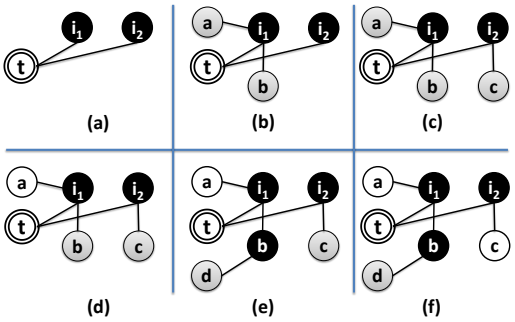

Figure 1: An example of the TONIC framework.

initial leads i_1 and i_2. After acquiring i_1 and i_2 we reveal a, b and c as potential leads. In the next step, a is acquired, revealing that a is not a lead. Then, b is acquired, revealing that b is a lead and discovering the potential lead d. The main challenge in a best-first search approach is choosing choosing the best node from OPEN to expand.

TONIC Heuristics

We now provide a number of heuristics that choose which potential lead to expand next. One baseline heuristic is to choose to expand profiles in a *first-in-first-out* (FIFO) manner. This means that the neighbors of the initial leads are expanded first, then their neighbors and hence forth. Figure 1 illustrates such an expansion order. Another baseline "heuristic" is to randomly choose which potential lead to expand next. These "heuristics", denoted *FIFO* and *RND*, serve as a baseline for more intelligent heuristics that are described next. In particular, Random and FIFO do not consider the topology of the SN.

While initially only the target and the initial leads are known, more parts of the SN topology are discovered as the search progresses. The discovered subgraph of the SN graph (i.e., the parts revealed when acquiring profiles) is referred to as the *currently known graph*, denoted by $CKG = (V_{CKG}, E_{CKG})$. V_{CKG} is composed of three mutually exclusive sets: set of profiles found to be *leads* (L), the set of profiles found to be *non-leads* (NL), and the set of *potential leads* (PL). The following heuristics choose which potential lead to acquire by analyzing the CKG in different ways.

Clustering Coefficient

Nodes in SNs tend to form tightly connected clusters in which most peoples are friends of each other. This phenomenon is quantified using the notion of local clustering coefficient (Watts and Strogatz 1998), which measures the density of edges between the neighbors of a given node. In TONIC, we expect a profile that is connected to a cluster of leads to be a part of that cluster and be a lead as well. An intuitive example of such case is a small university department where a member of that department is the target.

Building on this intuition we propose a heuristic that is based on computing the clustering coefficient (CC) of each of the potential leads, choosing to expand the potential lead with the highest CC. More formally, let $L(pl)$ be the set of leads that are friends of the potential lead pl in the CKG. The CC of pl in the CKG is the number of links between $L(pl)$ divided by the number of possible links among $L(pl)$:

$$CC(pl) = \frac{2 \cdot |\{(u,v) \in E_{CKG} | u,v \in L(pl)\}|}{|L(pl)| \cdot (|L(pl)| - 1)}$$

For example, a potential lead pl will have a CC of one if all its lead friends form a clique. The *CC heuristic* is the heuristic that chooses to expand the potential lead with the highest CC.

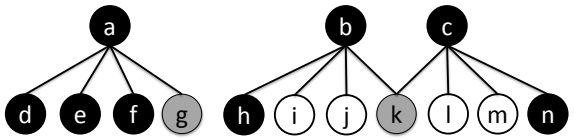

Figure 2: Example of the KD and Promising Heuristics.

Degree of Potential Leads

An obvious shortcoming to the CC heuristic is that it ignores the number of friends that a potential lead has. In fact, potential leads with a single friend will have a CC score of one. This is a disadvantage of CC, because it has been shown that degree of nodes in a social networks exhibits a power-law distribution (Barabási and Réka 1999), and previous work has shown that an effective strategy for searching in such graphs is to direct the search towards nodes with a high degree (Adamic et al. 2001). The intuition behind this is that high degree nodes are connected to many other nodes, and thus are better sources for searching than low degree nodes.

Identifying the potential lead with the highest degree is not possible in TONIC, since the problem solver does not know the true degree of potential leads before they are acquired. However, the degree of a potential lead in the CKG *is* known. This is called the *known degree* (KD) of a node and the corresponding heuristic, denoted as the *KD heuristic*, expands nodes in the order of their KD. This heuristic was previously used to find cliques in an unknown graph (Stern, Kalech, and Felner 2012). In TONIC, the KD of a potential lead pl is the number of expanded leads that are friends of pl, since only leads are acquired. The intuition behind considering KD for TONIC is based on the common-friends concept, which was extensively used in link-prediction research (Liben-Nowell and Kleinberg 2007). Simply put, we expect a potential lead that has many friends that are leads to also be a lead.

An example of KD is provided in Figure 2. Grayed nodes are potential leads, black nodes are leads, and white nodes are non-leads. $KD(g) = 1$ (connected to a) while $KD(k) = 2$ (connected to b and c). Thus, the KD heuristic will choose to expand node k before node g.

Promising-Leads Heuristic

The KD heuristic expands potential leads according to the number of expanded leads that are connected to them. This is reasonable if all leads have an equivalent effect on the likelihood that a potential lead connected to them is a lead. Following, we explore an alternative approach that considers

not just the amount of leads that a potential lead is connected to, as the KD heuristic, but also how "promising" these leads are, in the sense that potential leads connected to them are more likely to be leads.

The first step in creating such a heuristic is to define a measure of how "promising" a lead is. An ideal "promising" measure for a lead m would be the probability that a randomly drawn generated neighbor of m is a lead. This is the ratio of potential leads connected to m that are leads. We denote this ideal promising measure as $pm^*(m)$.

Unfortunately, $pm^*(m)$ cannot be known before *all* neighbors of m are acquired. As a practical alternative, we consider the ratio of leads among the *expanded* neighbors of m. Formally, we divided the LOF of an expanded lead m into three sets: leads, not leads and potential leads, denoted as $L(m)$, $NL(m)$ and $PL(m)$, respectively. The promising measure we propose, called the *promising factor* and denoted by $pf()$, is computed by $pf(m) = \frac{L(m)}{L(m)+NL(m)}$.[1]

Aggregating Promising Leads

If every potential lead was connected to a single expanded lead, then a straightforward TONIC heuristic that considers the promising factor would expand the potential lead that is a friend of the expanded lead with the highest promising factor. However, a potential lead may be connected to a set of expanded leads, each having a different promising factor.

Two simple ways to aggregate the promising factors of the leads is to take their maximum or their average. We call the corresponding TONIC heuristics $MaxP$ and $AvgP$, respectively. Formally, $MaxP$ chooses to expand the potential lead pl that maximizes $\max_{m \in L(pl)} pf(m)$, while $AvgP$ chooses to expand the potential lead pl that maximizes $\frac{1}{|L(pl)|} \sum_{m \in L(pl)} pf(m)$. Naturally, $MaxP$ only considers the lead that is most promising, and ignores all the other leads in $L(pl)$. $AvgP$ takes into consideration all the leads in $L(pl)$ but may diminish the effect of a very promising lead in $L(pl)$, if $L(pl)$ contains other less promising leads. Next, we consider a more sophisticated way to aggregate the promising factors of the leads.

Bayesian Aggregation The promising factor $pf(m)$ is designed to estimate $pm^*(m)$, which is the probability that a potential lead connected to m is a lead. We therefore propose another way to aggregate the promising factors that is based on a Naïve Bayes approach to aggregate probabilities.

$$BysP(pl) = 1 - \prod_{m \in L(pl)} (1 - pf(m))$$

The TONIC heuristic that chooses to expand the potential lead pl with the highest $BysP(pl)$ is denoted as the *Bayesian Promising* heuristic, or simply $BysP$. $BysP$ has the following desirable attributes. Unlike the $AvgP$, discovering a new lead m that is connected to pl is guaranteed to increase (or at least not decrease) $BysP(pl)$, since $pf(m) \leq 1$. Unlike $MaxP$, any change in the promising factor of each of the leads in $L(pl)$ affects the $BysP(pl)$: it will increase or decrease according to the increase or decrease of the promising factor of the leads in $L(pl)$.

As an example of the Bayesian Promising heuristic, consider again the graph in Figure 2. There are two potential leads, g and k. The only lead connected to g is a, which has $pf(a) = \frac{3}{3} = 1$, and thus $BysP(g) = 1 - (1 - pf(a)) = 1$. k is connected to two leads, b and c, with $pf(b) = pf(c) = \frac{2}{3}$, and thus $BysP(k) = 1 - (1 - pf(b))(1 - pf(c)) = \frac{5}{9}$. Therefore, according to the $BysP$ heuristic g will be expanded first.

[1] Initially, it is possible that $L(m) + NL(m) = 0$, making $pf(m)$ undefined. To avoid this, we set $pf(m) = 0.5$ in this case.

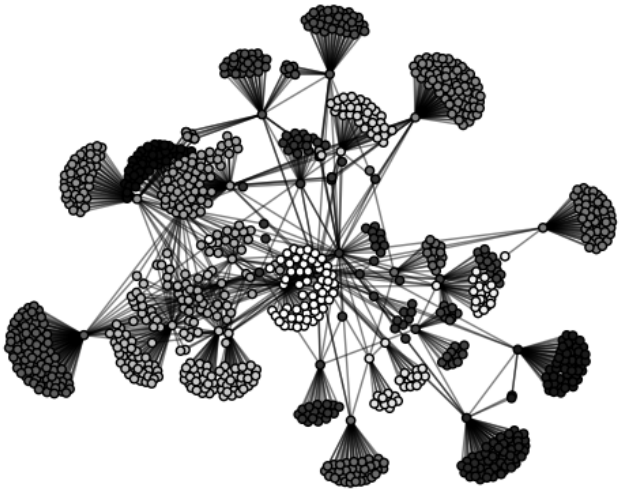

Figure 3: Sample network of the relevant neighborhood of a target.

Experimental Results

In this section we evaluate the performance of the different TONIC heuristics on the Google+ social network. Google+ is one of the largest social networks, having more than 500M registered users and 235M users that are active monthly (according to Wikipedia).

Data Collection

The data set we used for our experiments was obtained from the Google+ network and included 211K profiles with 1.5M links between them. This data set was collected by Fire et. al. (2011) and made available. From this data set we randomly selected a set of 100 profiles having at least 30 friends. These profiles were used as the targets in our experiments.

Since only potential leads can be acquired (Definition 1), the *relevant neighborhood* of each target is the leads and their neighbors. In our data set, the size of relevant neighborhood ranged from 233 to more than 4K profiles. Figure 3 presents the relevant neighborhood of one of the targets in our data set.

Algorithm 1 was executed for each target using the following heuristics: RND, FIFO, CC, KD, AvgP, MaxP and BysP. Three friends of every target were randomly chosen

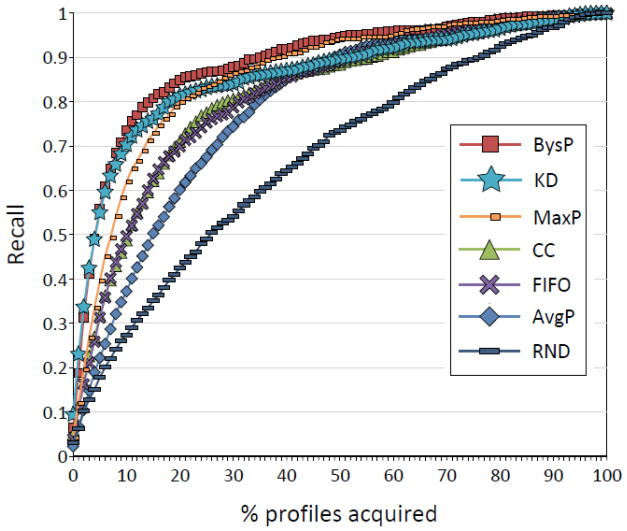

Figure 4: % of leads found vs. % of potential leads checked.

as the initial leads. The search process continued until *all* potential leads were acquired.

Cost-Benefit Analysis

Figure 4 shows the following cost-benefit analysis. The x-axis represents the percent of profiles acquired out of all reachable profiles. The y-axis represents the fraction of leads found out of all reachable leads, averaged over all the 100 targets in our data set. Naturally, all leads were found after 100% of the reachable profiles were acquired, regardless of which heuristic was used. However, it is easy to see that KD, MaxP, and BysP are able to find more leads earlier during the search process and in general outperform all other heuristics. In particular, BysP dominates all other heuristics.

TONIC as an Information Retrieval Task

An alternative cost-benefit analysis, often used in Information Retrieval (IR) and other fields, is the well-known receiver operating characteristic (ROC) curve (Croft, Metzler, and Strohman 2010), which measures the true positive rate (TPR) as a function of the false positive rate (FPR). TONIC can also be viewed as an IR task: the *query* is the target and the set of initial leads, and the *response* to that query is a sequence of profiles acquired up to any specific point during the search. Thus, for TONIC the TPR is the ratio of leads found and the FPR is the ratio of non-leads acquired. In TONIC, most potential leads are not leads, and thus in general the number of non-leads acquired is very similar to the total number of profiles acquired. Thus, the ROC curve for TONIC greatly resembles the cost-benefit curve shown in Figure 4 (this was also observed empirically).

Considering TONIC as an Information Retrieval task allows using known IR metrics for comparing the overall performance of TONIC heuristics. In particular, we consider two such metrics: the area under the ROC curve (AUC) and the discounted cumulative gain (DCG) (Jarvelin and Kekalainen 2002). DCG is computed by $DCG = rel_1 + \sum_{i=2}^{n} \frac{rel_i}{log_2 i}$, where rel_i in TONIC is one if the i^{th} profile acquired is a lead or zero otherwise, and n is the number of profiles in the relevant neighborhood. DCG provides an alternative view to AUC, giving higher value to finding leads earlier in the search. DCG has special importance in TONIC, as every profile acquisition increases the network footprint of the intelligence collection and the possibility that the SN service will block future acquisitions. Therefore, one can argue that it is important to spot the leads as early in the search process as possible, supporting the DCG metric.

Alg	BysP	KD	MaxP	CC	FIFO	AvgP
AUC	**0.904**	0.857	0.852	0.847	0.826	0.819
DCG	4.810	**4.907**	4.482	4.334	4.288	3.985

Table 1: AUC and DCG of the TONIC heuristics

Table 1 shows the average AUC and DCG of the same data set described previously. AUC results show the same trends as shown in Figure 4, as could be expected. Interestingly, DCG results provide a different view. KD is the best-performing heuristic in terms of DCG, with BysP has a slightly lower DCG. Interestingly, MaxP shows significantly lower DCG than KD, while having very similar AUC. These results are understandable, because the promising factor used by MaxP is based on information gathered in previous acquisitions. Thus, at the beginning of the search the information from the promising factor is less reliable, while as the search progress, the promising ratio becomes more meaningful. Since DCG gives higher weight to leads found at the beginning of the search, MaxP exhibits lower performance compared to KD. By contrast, KD is immediately informative once the initial leads are acquired, choosing to acquire the potential lead that is connected to the largest number of initial leads. Thus, the high DCG value of KD is reasonable. However, MaxP shows high AUC, as AUC gives a more balanced weight to leads found later in the search, when the promising factor becomes more informative.

Note that BysP is in general robust across both AUC and DCG, avoiding to some extent the shortcoming of MaxP by taking into considerations the number of leads connected to every potential profile, as discussed earlier in this paper.

Effects of the Initial Leads

Figure 5 compares the performance of the proposed heuristics in terms of DCG as a function of the number of initial leads. The same trends shown in Table 1 can be seen here, where on average BysP and KD perform best and RND and AvgP are the weakest. Naturally, the DCG increases with the number of initial leads, as the first profiles acquired are the initial leads, which are guaranteed to be leads.

Effects of Network Topology Characteristics

Next we investigate the network topology characteristics that influence the performance of the proposed TONIC heuristics. We have experimented with various topology attributes including the *clustering coefficient* (CC) of each tar-

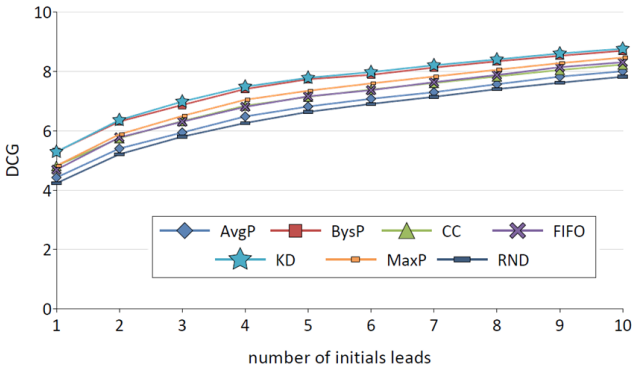

Figure 5: DCG as a function of # of initial leads.

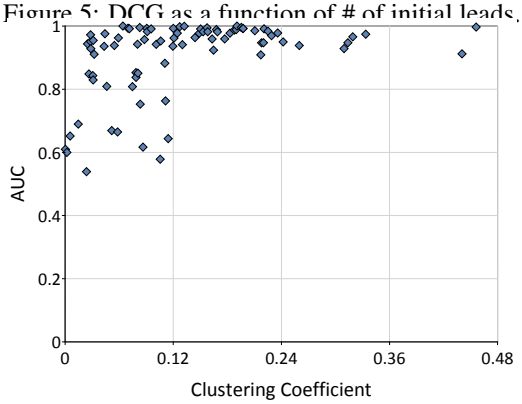

Figure 6: AUC of BysP vs. clustering coefficient

get, the size of the *relevant neighborhood* of the target, the density of this neighborhood and even using community detection algorithm (Blondel et al. 2008) that find densely connected regions in the relevant neighborhood. The topology characteristic that has the most distinct impact on the proposed heuristics was the CC of the target. The CC measure was described earlier in this paper, where the CC heuristics computes the CC in the CKG of every potential lead.

Figure 6 presents the AUC of BysP as the function of the CC of the target. Every data point corresponds to a the AUC and CC obtained by BysP for a single target. We can clearly see a phase transition in the AUC values around $CC = 0.12$. On 42 evaluated targets with *high CC* (larger than or equal to 0.12) the search performs well with very low variance. For these cases, BysP results in AUC ranges from 0.908 to 1.0 with an average of 0.969. By contrast, on the 58 evaluated targets with *low CC* (smaller than 0.12) the AUC is more noisy, ranging from 0.538 to 1.0 with an average of 0.846.

To gain deeper understanding of the effect CC has on the proposed TONIC heuristics, consider Figure 7, which shows the average AUC obtained by KD, BysP, MaxP and AvgP for targets with low CC (left) and high CC (right). As can be seen, for all of these heuristics the search performs better for targets with high CC. The reason is that a target with low CC means a low number of links between leads. As a result, OPEN is populated mostly with non-leads during each acquisition of a lead. In contrast, high CC means that more leads were added to the OPEN when acquiring a lead, be-

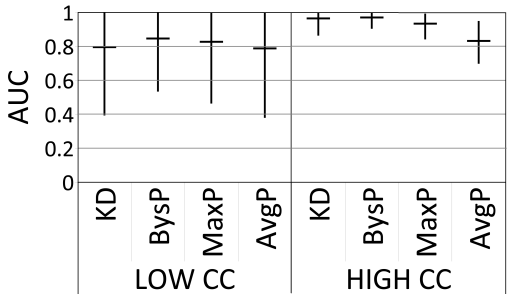

Figure 7: AUC of for targets with high and low CC

cause leads form tightly connected clusters. Thus, the search task in general becomes harder for targets with low CC.

Another notable fact is that the difference in AUC between BysP and KD is larger for low CC targets. This change is significant at the level of 0.01[2] Moreover, the lead interconnection (in cases with high CC) results in high KD values for leads, speeding up their acquisition by KD. In these cases the "promising" measure of leads does not contribute significantly to KD. On the other hand, for cases with low CC both potential leads that turn out to be leads and those that do not have same low KD values, making it hard for the KD heuristic to distinguish between them. In contrast, BysP and MaxP rely on the "promising" measure which does not lose its predictive power for targets with low CC. Notable, BysP is the best performing TONIC heuristic in both cases, demonstrating its robustness.

Conclusion and future work

In this paper we introduced and formalized the TONIC problem, where the goal is to find profiles in a social network that contain information about a given target profile. TONIC is an integral part of many commercial and governmental applications. To the best of our knowledge, this is the first academic work on this important problem. We formalize TONIC as a heuristic search problem and propose several heuristic functions to guide the search. Evaluation results show that (1) in general, heuristics that incorporate the promising measure are more robust to topological properties of the target and (2) in particular, the Bayesian Promising heuristic significantly outperforms other heuristics. Discussing the ethical aspects of TONIC is beyond the scope of this paper.

There are many exciting directions for future work. One direction is to consider acquiring also profiles that are not friends of leads. Another direction is exploiting demographic and textual information extracted from SN profiles to devise better, more intelligible heuristics. Additionally, we are currently pursuing how to combine TONIC heuristics to create an effective ensemble. Both approaches can utilize the power of Machine Learning (ML) to predict which potential leads will turn to be leads de facto. ML models can be trained using both data from past executions on various targets and data on profiles acquired during investigation of

[2]p-value of 0.0038 in Welch's t-test.

current target.

Acknowledgments

The research was supported by the Israeli Ministry of Industry Trade and Commerce, Office of the Chief Scientist.

References

Adamic, L. A.; Lukose, R. M.; Puniyani, A. R.; and Huberman, B. A. 2001. Search in power-law networks. *Phys. Rev. E* 64:046135.

Altshuler, Y.; Elovici, Y.; Cremers, A. B.; Aharony, N.; and Pentland, A. 2012. *Security and Privacy in Social Networks*. Springer.

Barabási, A.-L., and Réka. 1999. Emergence of scaling in random networks. *Science* 286(5439):509–512.

Blondel, V.; Guillaume, J.; Lambiotte, R.; and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008(10):P10008.

Bonneau, J.; Anderson, J.; and Danezis, G. 2009. Prying data out of a social network. In *Social Network Analysis and Mining, 2009. ASONAM'09. International Conference on Advances in*, 249–254. IEEE.

Chang, C.; Kayed, M.; Girgis, M.; Shaalan, K.; et al. 2006. A survey of web information extraction systems. *IEEE transactions on knowledge and data engineering* 18(10):1411.

Chau, D. H.; Pandit, S.; Wang, S.; and Faloutsos, C. 2007. Parallel crawling for online social networks. In *Proceedings of the 16th international conference on World Wide Web*, 1283–1284. ACM.

Croft, W.; Metzler, D.; and Strohman, T. 2010. *Search engines: Information retrieval in practice*. Addison-Wesley.

Edelkamp, S.; Jabbar, S.; and Lluch-Lafuente, A. 2005. Cost-algebraic heuristic search. In *AAAI*, 1362–1367.

Fire, M.; Tenenboim, L.; Lesser, O.; Puzis, R.; Rokach, L.; and Elovici, Y. 2011. Link prediction in social networks using computationally efficient topological features. In *SocialCom/PASSAT*, 73–80.

Fire, M.; Katz, G.; Elovici, Y.; Shapira, B.; and Rokach, L. 2012. Predicting student exam's scores by analyzing social network data. In *AMT*, 584–595.

Jarvelin, K., and Kekalainen, J. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. on Information Systems* 20(4):422?–446.

Korolova, A.; Motwani, R.; Nabar, S. U.; and Xu, Y. 2008. Link privacy in social networks. In *Proceedings of the 17th ACM conference on Information and knowledge management*, 289–298. ACM.

Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58(7):1019–1031.

Mislove, A.; Marcon, M.; Gummadi, K. P.; Druschel, P.; and Bhattacharjee, B. 2007. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 29–42. ACM.

Pawlas, P.; Domanski, A.; and Domanska, J. 2012. Universal web pages content parser. In *Computer Networks*, volume 291 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg. 130–138.

Stern, R.; Kalech, M.; and Felner, A. 2012. Finding patterns in an unknown graph. *AI Commun.* 25(3):229–256.

Stern, R.; Puzis, R.; and Felner, A. 2011. Potential search: A bounded-cost search algorithm. In *ICAPS*.

Tang, J.; Yao, L.; Zhang, D.; and Zhang, J. 2010. A combination approach to web user profiling. *ACM Trans. Knowl. Discov. Data* 5(1):2:1–2:44.

Watts, D. J., and Strogatz, S. 1998. Collective dynamics of 'small-world' networks. *Nature* 393(6684):440442.