Fast Dominating Set Algorithms for Social Networks

Alina Campan, Traian Marius Truta, and Matthew Beckerich
Computer Science Department
Northern Kentucky University
Highland Heights, KY 41099, U.S.A.
campana1@nku.edu, trutat1@nku.edu, beckerichm1@mymail.nku.edu

Abstract

In this paper we introduce two novel algorithms that are able to efficiently determine an approximation to the minimum dominating set problem, and at the same time, they will preserve the quality of the solution to an acceptable level. We compare these two algorithms with three existing algorithms, for a large number of synthetic datasets, and for several real world social networks. For experiments, we use social network generators that create both power-law and random networks, and a few real network datasets made available by the Stanford Network Analysis Project. Our experiments show that the proposed algorithms are viable, and in many instances, preferable for determining the minimum dominating set of a social network.

Introduction

Today, online social networks are used by an ever increasing number of people. For instance, Facebook has currently more than 1.35 billion users, LinkedIn has over 332 million users, and 1 out of 4 people from the entire world is an active user in at least one existing social network (Statista 2014). It is not a surprise that researchers from various domains such as sociology, economics, physics, mathematics, and computer science are analyzing different problems and proposing various solutions related to social networks. While a lot of research has been performed in the past on networks/graphs, nowadays the main focus is moving toward social issues (as the name implies, social networks usually represent relationships between people) and toward big social networks (as previously noted, these networks tend to be very large).

Social networks existed long before the Internet. As early as around 1200-1450 such social networks existed in pre-Hispanic Southwest in the current United States. The growth, collapse, and change of such social networks are reflected in thousands of ceramic and obsidian artifacts (Mills et al. 2013). More recently, in the first half of the 20th century, the social network analysis discipline was started by the sociology community (Freeman 2011). Beginning from the 1990s, other fields, in particular computer science and physics, brought a lot of new results to this field (Freeman 2011). With the advent of Myspace, Friendster, and more recently, Facebook and LinkedIn, social networks have grown to a new dimension -- the online social network -- which allows larger numbers of people to participate. Many existing methods for social network analysis did not envision such a data explosion. Thus, developing fast and accurate solutions for large social networks is becoming more essential in present days.

An important feature in a social network is the ability to communicate quickly within the network. For instance, in an emergency situation, we may need to be able to reach to all network participants, but only a small number of individuals in the network can be contacted directly. However, if all individuals from the network are connected to at least one such individual who can be contacted directly (or is one of those individuals) then the emergency message can be quickly sent to all network participants. This scenario can be modeled by what is known as the *dominating set problem*. This is formally described below.

A set $\mathcal{DS} \in \mathcal{N}$ of nodes in a network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ is a dominating set if every node $x \in \mathcal{N}$ is either in $\mathcal{DS}$ or adjacent to $\mathcal{DS}$ (Berge 1962). The dominating set with the smallest numbers of elements is known as the minimum dominating set ($\mathcal{MDS}$). To find such a minimum dominating set is a well-known NP-complete problem, thus an exact efficient solution is unlikely to be found (Garey and Johnson 1979).

Approximation algorithms for determining the minimum dominating set exist in the literature, with the most common one being an adaptation of the greedy algorithm for determining a set cover (Lovasz 1975). However, as we will show in the following sections, this greedy algorithm is slow for large networks, and therefore more efficient solutions are needed. A very efficient algorithm has also been proposed (Eubank et al. 2004), but this algorithm has a low result quality as we will show in the future sections.

Extensive experimental results have been performed for scale-free networks (that model social networks well) using either the basic greedy algorithms for approximating the minimum dominating set size (Molnar et al. 2013) or a different algorithm based on a binary integer programming-based method (Nacher et al. 2012). These works do not compare the results of these algorithms with any other algorithms, and, in addition, they target social networks with low cardinality (up to 10,000 nodes), and therefore they do not discuss how the used algorithms will scale to large social networks.

In this paper we ***introduce two novel algorithms*** that are able to determine efficiently an approximation to the minimum dominating set problem and, simultaneously, they will preserve the quality of the solution to an acceptable level. Those two algorithms: one extends the greedy algorithm mentioned earlier, and the other is an improvement of an algorithm that is also introduced in (Eubank et al. 2004). We ***compare those two algorithms with three existing algorithms*** for a large number of synthetic datasets, and for several real world social networks. For the experimental part, we use social network generators that create both power-law and random networks, as well as real network datasets made available by the Stanford Network Analysis Project (Leskovec and Krevl 2014).

The remaining of this paper is structured as follows. Section "Dominating Set Algorithms" introduces the two new algorithms that we use in this paper. In addition, in this section we also present three existing algorithms that we use for experimental evaluation. Section "Datasets" describes how we generate the synthetic social networks, and provides a description of the used real datasets. In section "Experiments and Results" we perform a detailed experimental evaluation of our algorithms using the datasets presented in the previous section. Section "Discussion and Future Extension" presents conclusions and a summary of future extension for our work.

Dominating Sets Algorithms

In this section we describe the greedy algorithms we use to approximate the minimum dominating set. Algorithms 3 and 4 are new, while the rest are existing algorithms against which we compare ours.

The first algorithm (***Algorithm 1*** or ***Alg. 1***) is the standard greedy algorithm that choses at each step, one node that covers the maximum number of currently uncovered nodes (Eubank et al. 2004, Molnar et al. 2013). In this algorithm we start with an empty set, $\mathcal{DS}$, which will at the end store the nodes in the dominating set. Nodes from the network are colored according to their state during the execution of the algorithm. Nodes in $\mathcal{DS}$ are called *black*, nodes which are covered (because they neighbor one or more nodes in $\mathcal{DS}$) are called *gray*, and all uncovered nodes are *white*. $\mathcal{W}(v)$ denotes the set of white nodes among the direct neighbors of v, including v itself. $w(v) = |\mathcal{W}(v)|$ is called the span of v. Note that in the Alg. 1, we delete the black nodes, and the next selected node is either gray or white. This algorithm is named ***RegularGreedy*** in (Eubank et al. 2004). Its pseudocode is presented next:

```
Algorithm 1 ($\mathcal{G}$, $\mathcal{DS}$) is
  Input:  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ - a social network;
  Output: $\mathcal{DS}$ - a dominating set for $\mathcal{G}$;
  $\mathcal{DS} = \varnothing$;
  while ∃ white nodes do
    choose $v \in \{x \in \mathcal{N} \mid w(x) = \max_{u \in \mathcal{N}} |\mathcal{W}(u)|\}$;
    $\mathcal{DS} := \mathcal{DS} \cup \{v\}$;
    // Delete the vertex $v$ and
    // its adjacent edges from G.
    $\mathcal{G} = (\mathcal{N} \setminus \{v\}, \mathcal{E} \setminus \{(x, y) \in \mathcal{E} \mid x = v \text{ or } y = v\})$;
  end while;
end Algorithm 1.
```

The second algorithm (***Algorithm 2*** or ***Alg. 2***) starts by finding the set of all nodes of degree exactly one ($\mathcal{D}_1$). Then finds the set of nodes adjacent to $\mathcal{D}_1$ (*Neighbors*($\mathcal{D}_1$)). Obviously, any dominating set must contain *Neighbors*($\mathcal{D}_1$). Next, we run the Alg. 1 on the subgraph induced by $\mathcal{N} \setminus$ *Neighbors*($\mathcal{D}_1$). This second algorithm is referred in previous work as ***VRegularGreedy*** (Eubank et al. 2004). Its pseudocode is presented next:

```
Algorithm 2 ($\mathcal{G}$, $\mathcal{DS}$) is
  Input:  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ - a social network;
  Output: $\mathcal{DS}$ - a dominating set for $\mathcal{G}$;
  $\mathcal{D}_1 = \{x \in \mathcal{N} \mid degree(x) = 1\}$;
  $Neighbors(\mathcal{D}_1) = \{x \in \mathcal{N} \mid \exists\, y \in \mathcal{D}_1 \text{ and } (x, y) \in \mathcal{E}\}$
  $\mathcal{DS} = Neighbors(\mathcal{D}_1)$;
  // $\mathcal{E}'$- contains all edges from $\mathcal{E}$ that have
  // both ends in $\mathcal{N} \setminus \{Neighbors(\mathcal{D}_1) \cup \mathcal{D}_1\}$.
  $\mathcal{G}' = (\mathcal{N} \setminus \{Neighbors(\mathcal{D}_1) \cup \mathcal{D}_1\}, \mathcal{E}')$;
  Algorithm 1 ($\mathcal{G}'$, $\mathcal{DS}'$);
  $\mathcal{DS} = Neighbors(\mathcal{D}_1) \cup \mathcal{DS}'$;
end Algorithm 2.
```

The third algorithm (***Algorithm 3*** or ***Alg. 3***) improves the running time of Algorithm 1 by removing both the black and gray nodes from the remaining network, thus reducing significantly at each step the size of the network. Since all the time the remaining nodes are white, the use of node coloring is no longer useful. This algorithm is described next:

```
Algorithm 3 ($\mathcal{G}$, $\mathcal{DS}$) is
  Input:  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ - a social network;
  Output: $\mathcal{DS}$ - a dominating set for $\mathcal{G}$;
  $\mathcal{DS} = \varnothing$;
  while $\mathcal{N} \neq \varnothing$ do
    choose $v \in \{x \in \mathcal{N} \mid$
              $degree(x) = \max_{u \in \mathcal{N}} (degree(u))\}$;
    $\mathcal{DS} := \mathcal{DS} \cup \{v\}$;
    // Delete the vertex $v$, $Neighbors(\{v\})$, and
    // their corresponding edges.
    // The remaining edges are labeled $\mathcal{E}'$
    $\mathcal{G} = (\mathcal{N} \setminus \{v\} \setminus Neighbors(\{v\}), \mathcal{E}')$;
  end while;
end Algorithm 3.
```

The fourth algorithm (***Algorithm 4*** or ***Alg. 4***) combines the Algorithms 2 and 3, by including in the dominating set all neighbors of nodes of degree 1, and then by applying Algorithm 3 to the remaining graph. This algorithm is described as:

```
Algorithm 4 ($\mathcal{G}$, $\mathcal{DS}$) is
   Input:  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ - a social network;
   Output: $\mathcal{DS}$ - a dominating set for $\mathcal{G}$;

   while $\mathcal{N} \neq \emptyset$ do
      $\mathcal{D}_1 = \{ x \in \mathcal{N} \mid degree(x) = 1\}$;
      $Neighbors(\mathcal{D}_1) = \{ x \in \mathcal{N} \mid \exists\, y \in \mathcal{D}_1$ and $(x, y) \in \mathcal{E} \}$
      $\mathcal{DS} = Neighbors(\mathcal{D}_1)$;
      // $\mathcal{E}'$- contains all edges in $\mathcal{E}$ with both ends
      // in $\mathcal{N} \setminus \{Neighbors(\mathcal{D}_1) \cup Neighbors\ (Neighbors(\mathcal{D}_1))\}$.
      $\mathcal{G}' = (\mathcal{N} \setminus \{Neighbors(\mathcal{D}_1) \cup Neighbors\ (Neighbors(\mathcal{D}_1)),\ \mathcal{E}')$;
      Main step in Algorithm 3 ($\mathcal{G}'$, $v$);
      $\mathcal{DS} = Neighbors(\mathcal{D}_1) \cup \{v\}$;
   end while;
end Algorithm 4.
```

Note that in the Alg. 4, we also delete all nodes dominated by $Neighbors(\mathcal{D}_1)$ prior to using Algorithm 3. This is equivalent with deleting the gray nodes along with the black node which is executed in Alg. 3 and it is not performed in Alg. 1.

The last algorithm known as FastGreedy in (Eubank et al. 2004) is a very simple and efficient algorithm for computing a dominating set. Consider the nodes in $\mathcal{N}$ in non-increasing order of the degree $v_1, v_2, \ldots, v_{|\mathcal{N}|}$, with $degree(v_1) \geq degree(v_2) \geq \ldots \geq degree(v_{|\mathcal{N}|})$, where $degree(\cdot)$ is the given degree-sequence. Pick the smallest i such that $|\cup_{j \leq i} Neighbors\ (v_j)| = |\mathcal{N}|$ and take the subset $\{v_1, v_2, \ldots, v_i\}$ to be the dominating set of the graph. In this algorithm, we do not take those v_i nodes for which $Neighbors\ (v_i) \subseteq \cup_{j<i} Neighbors\ (v_j)$. We refer to this algorithm as ***Algorithm 5*** or ***Alg. 5***. The pseudocode algorithm is presented next:

```
Algorithm 5 ($\mathcal{G}$, $\mathcal{DS}$) is
   Input:  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ - a social network;
   Output: $\mathcal{DS}$ - a dominating set for $\mathcal{G}$;

   $\mathcal{DS} = \emptyset$;
   Order vertices in $\mathcal{N} = \{v_1, v_2, \ldots v_{|N|}\}$
      such that $degree(v_1) \geq degree(v_2) \geq \ldots \geq degree(v_{|\mathcal{N}|})$.
   $i = 1$;
   while $(|\cup_{j \leq i} Neighbors\ (v_j)| < |\mathcal{N}|)$ do
      if (not $(Neighbors\ (v_i) \subseteq \cup_{j<i} Neighbors\ (v_j))$)
         $\mathcal{DS} = \mathcal{DS} \cup \{v_i\}$;
      $i$++;
end Algorithm 5.
```

Datasets

In order to compare the size of dominating sets and execution times for the above 5 algorithms, we use both synthetic and real network datasets. All synthetic graphs were generated using the SNAP library (Leskovec and Sosic 2014).

We generate synthetic datasets using the well-known Erdos-Renyi random graph model (Bollobás 2001) and the configuration model (Molloy and Reed 1995, Britton et al. 2005) that generates a scale free network (Catanzaro et al. 2005). Details about the generation algorithms as well as the properties of the generated networks can be found in (Leskovec and Sosic 2014). We refer to those networks as ***ERNetworks*** and ***ConfNetworks*** respectively.

To generate *ERNetworks* we chose the following parameters. First we fix the size of the network, $|\mathcal{N}|$, to 50,000, and we generate *ERNetworks* for 10 different average degree values (AVG = 5, 10, 15, 20, .., 50). Second we chose a fixed AVG value (AVG = 10) and we use 10 distinct values for the size of $\mathcal{N}$($|\mathcal{N}|$ = 10K, 20K, …, 100K). For each combination of $|\mathcal{N}|$ and AVG values we generate 5 distinct networks. The total number of *ERNetworks* generated is 95 (19 combinations x 5 samples). Note that the size 50K and AVG = 10 is common between the two generation approaches, and therefore it is considered only once.

We generate the same number of *ConfNetworks* but we change the parameters as follows. For the size of the networks we use exactly the same values, but as a second parameter we use the power-law exponent γ from the power-law distribution ($P(k) \sim k^{-\gamma}$). For the fixed $|\mathcal{N}|$ value (50K) we use the values for γ = 1.7, 1.8, …, 2.6. When we vary the size of networks, we use γ = 2.0. As before, for each combination of $|\mathcal{N}|$ and γ we generate 5 district networks and the total number of generated *ConfNetworks* is 95.

We also use 10 real networks in this work. These networks are from the SNAP datasets website (Leskovec and Krevl 2014). The number of nodes and edges of these networks, as well as a short description, are shown in Table 1. For a full description of the networks and additional properties please consult (Leskovec and Krevl 2014).

Table 1. Real Networks' Characteristics.

Name	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{E}\rvert$	Description
ego-Facebook	4,039	88,234	Social circles from Facebook
email-Enron	36,692	183,831	Email communication network from Enron
ca-AstroPh	18,772	198,110	Collaboration network of Arxiv Astro Physics
ca-CondMat	23,133	93,497	Collaboration network of Arxiv Condensed Matter
ca-GrQc	5,242	14,496	Collab.network of Arxiv General Relativity
ca-HepPh	12,008	118,521	Collab. network of Arxiv High Energy Physics
ca-HepTh	9,877	25,998	Collab. network of Arxiv High Energy Physics Theory
com-Amazon	334,863	925,872	Amazon product network
com-DBLP	317,080	1,049,866	DBLP collaboration network
com-Youtube	1,134,890	2,987,624	Youtube online social network

Experiments and Results

To execute our experiments, we implemented the Algorithms 1 – 5 described in Section "Dominating Set Algorithms" and the graph random generators described Section "Datasets" in C++ using the SNAP framework (Leskovec and Sosic 2014). To allow a meaningful comparison between running times, all the experiments were performed on an Intel® Xeon® E5430@2.66 GHz dual CPU machine with 4 GB memory running on 32 bit Windows Server 2007 operating system. In the experiments executed on the synthetic networks (*ERNetworks* and *ConfNetworks*), the reported results are the average between the 5 sample datasets.

Figures 1 – 14 show the results of all our experiments. For figures that report the dominating set size, the *y* axis shows the number of nodes from the graph in the dominating set. For figures that report the running time, the *y* axis shows the number of seconds needed by our algorithms to compute the dominating sets.

Figures 1 – 6 show the results for *ERNetworks*. These networks gave predictable results in terms of both minimum dominating set size and running time. For dominating set sizes, we notice that the size of the network does not influence which algorithm performs best. However, the average degree is very important. When the average degree is below 25, the best algorithm is Alg. 4, followed in order by Alg. 3, Alg. 5, Alg. 2, and Alg. 1. When the average degree is over 25, Alg. 1 and Alg. 2 outperform the other 3 algorithms. Also it is worth noting that even in this case, Alg. 3 and, in particular, Alg. 4 find dominating sets of very close size with Alg. 1 and Alg. 2 (~15% more nodes in their dominating sets). For running time, Alg. 1 and Alg. 2 perform very poorly, and they are not suitable for very large networks. By comparing the other three algorithms (Figures 3 and 6), we draw the following two conclusions: First, Alg. 5 is always the fastest algorithm. Second, Alg. 4 is faster than Alg. 5 when the average degree is lower than 20, and the order changes for higher average degrees. It is worth noting that these three algorithms are suitable for very large networks. Based on these experiments, we conclude that for Erdos-Renyi random graphs the best algorithm to use in most cases is Alg. 4.

Figures 7 – 10 illustrate the results for *ConfNetworks*. The results are quite different compared to the *ERNetworks*. In terms of finding small dominating sets (Figures 7 and 9), Alg. 4 and 5 perform the best, with very similar results. Alg. 2 was close behind. Alg. 1 and 3 perform the worst. The reason for those results is that there are many vertices with degree = 1 that are identified early by Alg. 2 and 4, and their only neighbors are added to the dominating set in the beginning of these algorithm, thus reducing the size of the graph. When power exponent values are increased (Figure 9), the dominating set sizes are more similar than for low values of γ. This is expected since there are fewer nodes with low degree and the degree distribution became more equal. In terms of running time (Figures 8 and 10), Alg. 1 performs the worst by far, and all the other algorithms run in a relatively small amount of time. Looking more carefully at the data, Alg. 2, 3 and 4 perform very similarly. Alg. 3 is slightly better; Alg. 2 and 4 are very close, while Alg. 5 outperforms them significantly. Alg. 5 is between 20 times and 80 times faster than Alg. 3 in all our experiments.

To summarize, for power-law networks, the best algorithm is Alg. 5. While this algorithm is sometime worse than Alg. 4 in terms of finding the minimum dominating set size, it still performs best on average in this category. It is by far the fastest algorithm to execute, thus being very suitable for very large networks.

Figures 11 – 14 illustrate the results for *RealNetworks*. The smallest 7 datasets (shown in Figure 12) do not have any nodes of degree 1, and the results for Alg. 1 and Alg. 2 are identical. This is also true of Alg. 3 and Alg. 4 results. For those sets, Alg. 1 (and Alg. 2) outperform the other algorithms and Alg. 5 is close behind. For the larger sets, Alg. 4 and 5 perform the best, while Alg. 1 performs the worst. The reason for this sudden change is the distribution of nodes' degree; the large datasets have more of a power-law distribution, and the results are more similar to the results obtained for *ConfNetworks*. In terms of running time, Alg. 1 and 2 (and Alg. 3 and 4) are very similar for the smallest 7 datasets for the same reason as above. For the larger datasets, Alg. 1 performs the worst. As expected, Alg. 5 is by far the fastest.

To summarize, for real networks, determining the best algorithm to use is not as straight-forward. If the running time is an important factor in this decision, then Alg. 5 is again the winner. If the time is not a factor, then Alg. 1 (or Alg. 2) is the choice for datasets without high variance in the average degree, and with no nodes of degree 1 (such as the 7 smallest datasets); while Alg 4 or 5 is the choice for the datasets that follow a power-law distribution (com-amazon, com-dblp, com-youtube).

The experiments performed in this paper show that Alg. 4 is a viable algorithm to find a dominating set of a small size in large networks of various types. Also, the experiments show that Alg. 5 is surprisingly accurate, and it can be used successfully, in particular when the running time is a very important factor. To our surprise, Alg. 1 and Alg. 2 (which are frequently used in the literature) do not rise to the expected level, being outperformed in terms of dominating set size in most of the experiments by the other algorithms.

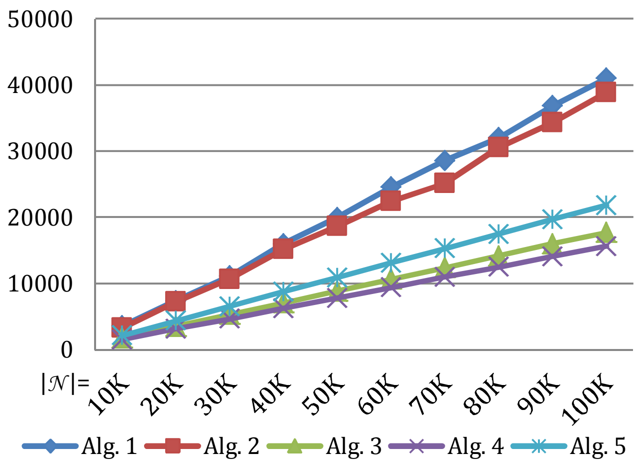

Figure 1. Dominating Set Size for ERNetworks when AVG = 10.

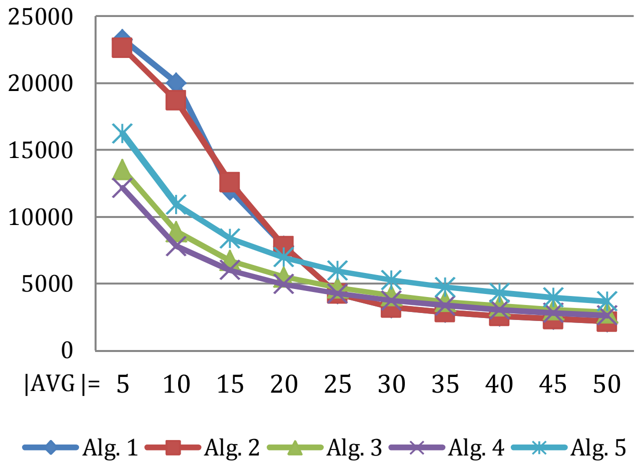

Figure 4. Dominating Set Size for ERNetworks when |𝒩| = 50K.

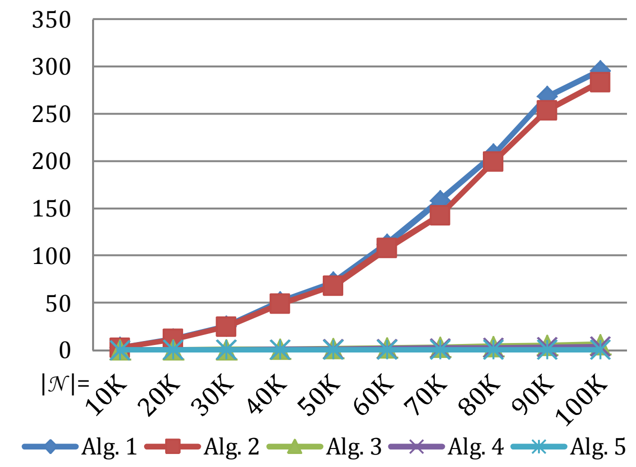

Figure 2. Running Time for ERNetworks when AVG = 10.

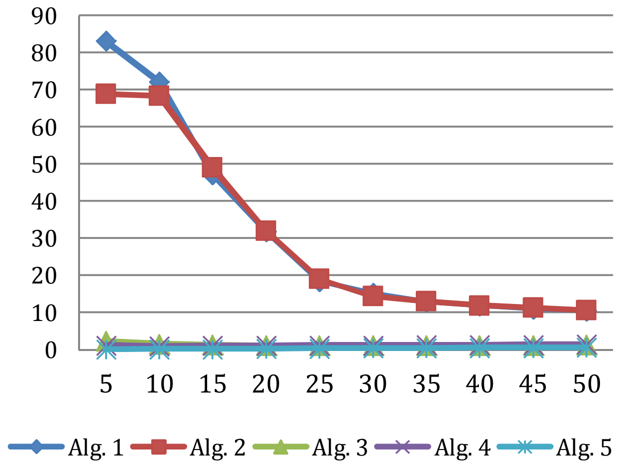

Figure 5. Running Time for ERNetworks when |𝒩| = 50K.

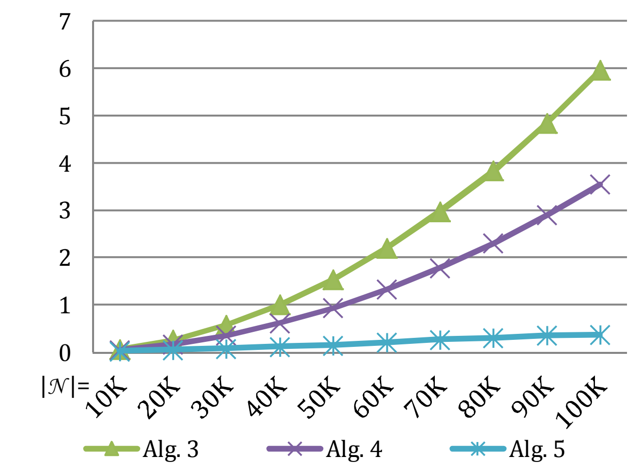

Figure 3. Running Time for ERNetworks when AVG = 10 (fastest algorithms only).

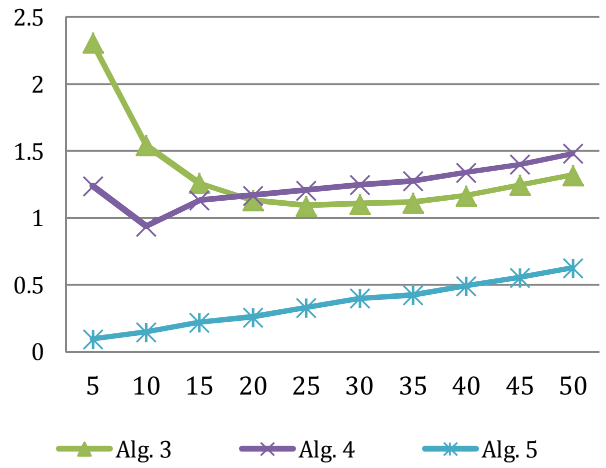

Figure 6. Running Time for ERNetworks when |𝒩| = 50K (fastest algorithms only).

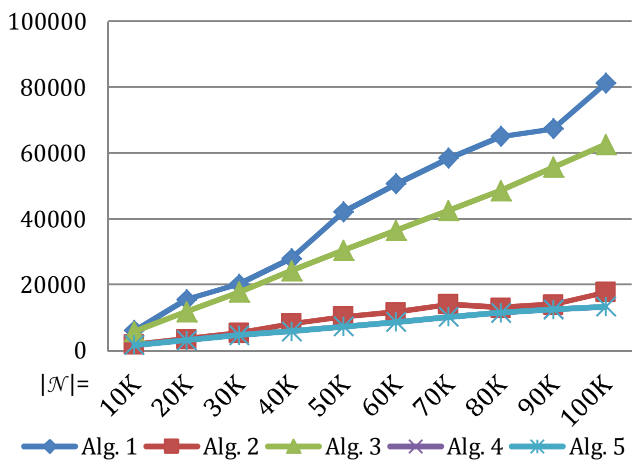

Figure 7. Dominating Set Size for ConfNetworks when γ = 2.0.

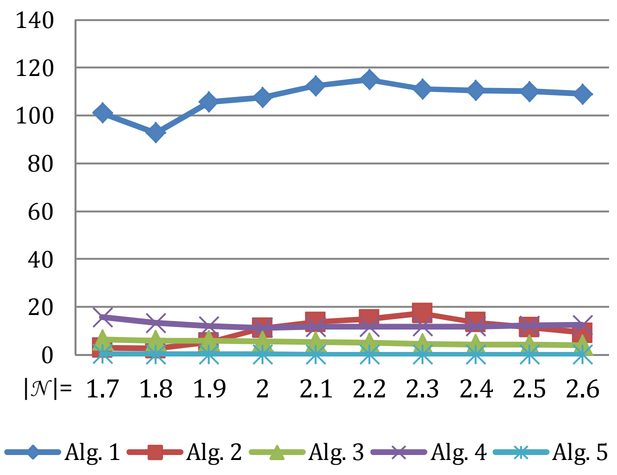

Figure 10. Running Time for ConfNetworks when |𝒩| = 50K.

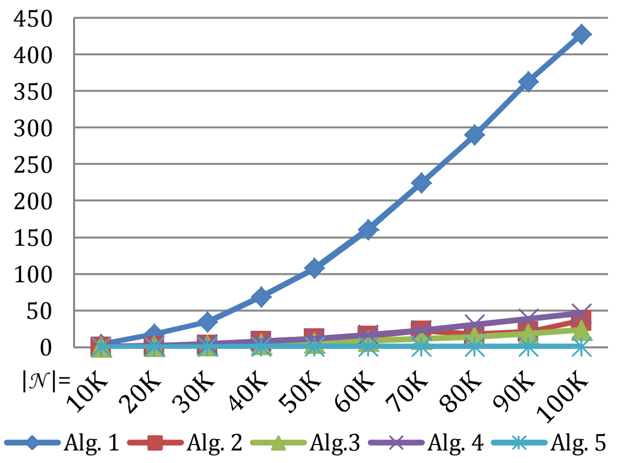

Figure 8. Running Time for ConfNetworks when γ = 2.0.

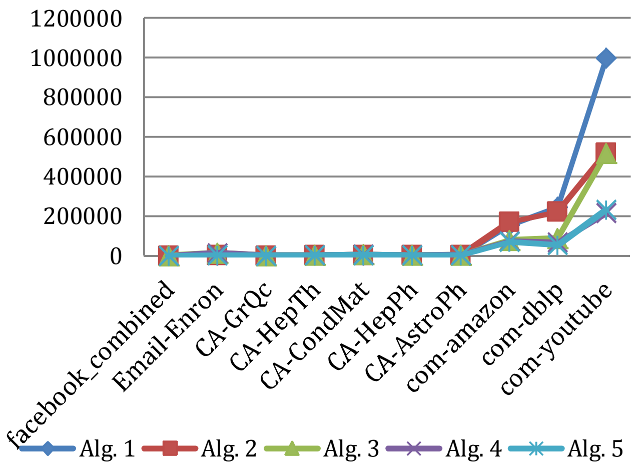

Figure 11. Dominating Set Size for RealData

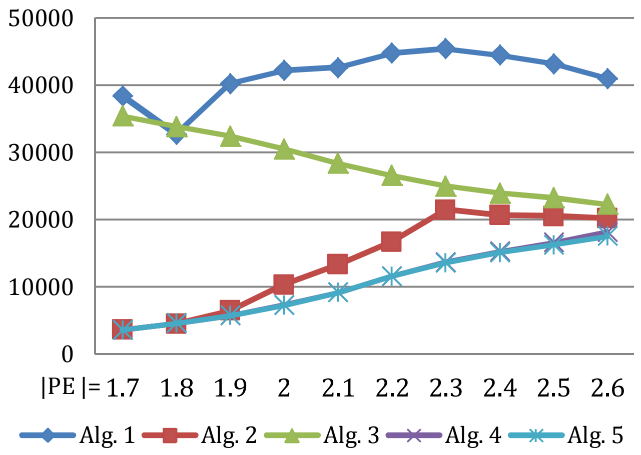

Figure 9. Dominating Set Size for ConfNetworks when |𝒩| = 50K.

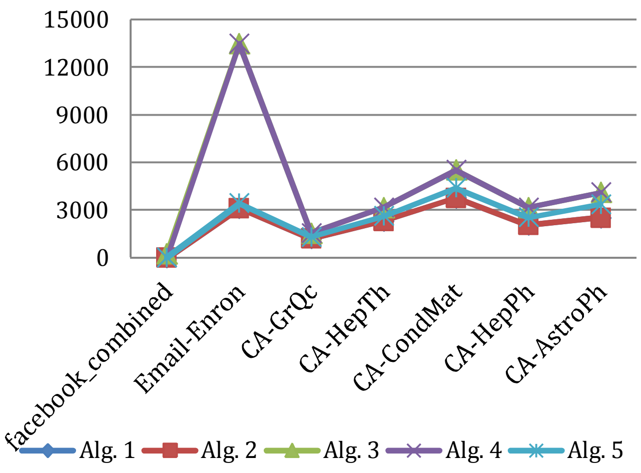

Figure 12. Dominating Set Size for RealData (smaller sets)

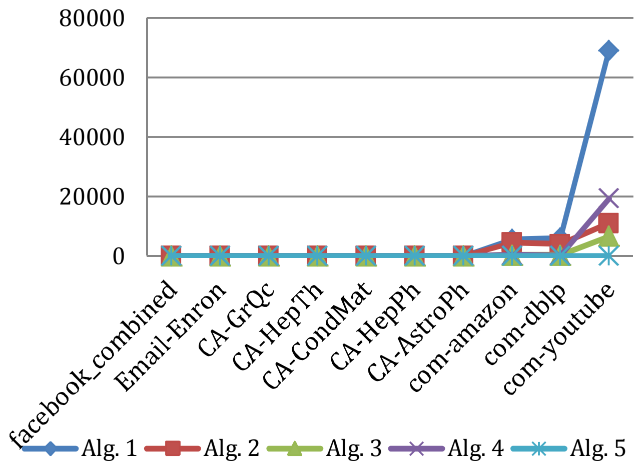

Figure 13. Running Time for RealData

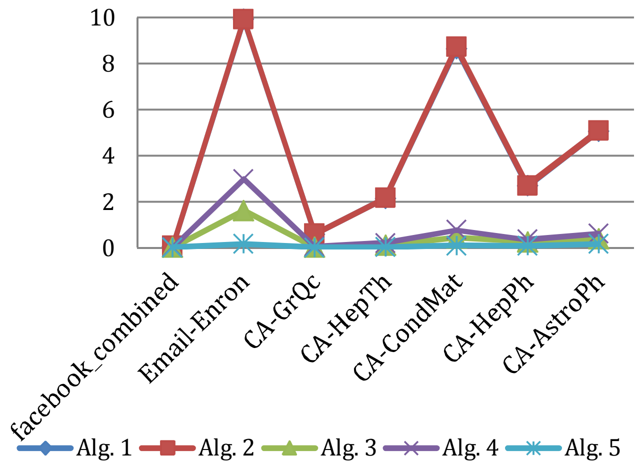

Figure 14. Running Time for RealData (smaller sets)

Discussion and Future Extensions

The experiments performed in this work show that the two new algorithms (and in particular Alg. 4) are viable options to determine a small dominating set for large networks. In many experiments, Alg. 4 outperforms the existing algorithms, while being very efficient in terms of running time.

This work is a preliminary work in the dominating sets for social network area. As part of our future work, we intend to investigate how the two new algorithms can be used for more specific dominating set problems, such as *partial dominating sets* (Formin et al. 2005), *total dominating set* (Cockayne at al. 1980), *independent dominating sets* (Cockayne at al. 1980), *connected dominating sets* (Wu and Li 1999), *d-hop dominating set* (Rieck et al. 2002), *positive influence dominating set* (Wang et al. 2009), and *k-tuple dominating set* (Klasing and Laforest 2004).

References

Berge C. 1962, Theory of Graphs and its Applications. Methuen, London.

Bollobás B., 2001, Random Graphs (2nd ed.). Cambridge University Press. ISBN 0-521-79722-5.

Britton T,; Deijfen M; and Martin-Lof A. 2006. Generating Simple Random Graphs with Prescribed Degree Distribution. *Journal of Statistical Physics*, Vol. 124, Issue 6, 1377-1397.

Catanzaro M.; Boguna M.; and Pastor-Satorras R., 2005, Generation of Uncorrelated Random Scale-Free Networks. *Physical Review*, E 71, 027103.

Cockayne E. J.; Dawes R. M.; and Hedetniemi S. T. (1980). Total Domination in Graphs. *Networks*, Vol. 10, No 3, pp. 211-219.

Eubank S.; Anil Kumar V. S.; Marathe M.; Srinivasan A.; and Wang N. 2004. Structural and Algorithmic Aspects of Massive Social Networks. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 718–727.

Fomin F. V.; Kratsch D.; and Woeginger G. J. (2005). Exact (Exponential) Algorithms for the Dominating Set Problem. In *Graph-Theoretic Concepts in Computer Science*, pp. 245-256 Springer Berlin Heidelberg.

Freeman L. C. 2011. The Development of Social Network Analysis — with an Emphasis on Recent Events. In *Sage Handbook of Social Network Analysis edited by J. Scott and P. Carrington*, 26–39.

Garey, M. R. and Johnson, D. S. 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, ISBN 0-7167-1045-5, page 190.

Klasing R. and Laforest C. (2004). Hardness Results and Approximation Algorithms of k-Tuple Domination in Graphs. *Information Processing Letters*, Vol. 89, No. 2, pp. 75-83.

Leskovec J. and Krevl A. 2014a. SNAP Datasets: Stanford Large Network Dataset Collection, Available at: *http://snap.stanford.edu/data*.

Leskovec J. and Sosic R. 2014b. SNAP: A general purpose network analysis and graph mining library in C++, Available at: *http://snap.stanford.edu/snap*.

Lovasz L. 1975. On the Ratio of Optimal Integral and Fractional Covers. Discrete Mathematics, Vol. 13, 383–390.

Mills B.; Clark J.; Peeples M.; Haas W. R.; Roberts J.; Hill J. B.; Huntley D.; Borok L.; Breiger R.; Clauset A.; and Shackley M. S. 2013. Transformation of Social Networks in the Late Pre-

Hispanic US Southwest. In *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 110, No. 15, April 9, 2013, 5785–5790.

Molloy M. and Reed B. 1995. A Critical Point for Random Graphs with a Given Degree Sequence. *Random Structures and Algorithms*, Vol. 6, 161-180.

Molnar F,; Sreenivasan S.; Szymanski K.; and Korniss G. 2013. Minimum Dominating Sets in Scale-Free Network Ensembles, Scientific Reports, No. 3, Nature Publishing Group.

Nacher J. C. and Akutsu T. 2012. Dominating Scale-free Networks with Variable Scaling Exponent: Heterogeneous Networks are not Difficult to Control. New Journal of Physics, Vol. 14.

Rieck M. Q.; Pai S.; and Dhar S. (2002). Distributed Routing Algorithms for Wireless Ad Hoc Networks using d-Hop Connected d-Hop Dominating Sets. In *Proc. 6th Int. Conf. High Performance Computing Asia Pacific*, Vol. 2, pp. 443-450.

Statista 2014. Available online at *http://www.statista.com/topics/1164/social-networks/.*

Wang F.; Camacho E.; and Xu K. (2009). Positive Influence Dominating Set in Online Social Networks. In *Combinatorial Optimization and Applications*, pp. 313-321, Springer Berlin Heidelberg.

Wu J. and Li H. (1999). On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks. In *Proceedings of the ACM 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pp. 7-14.