Streaming Networks Sampling using *top-K* Networks

Rui Sarmento[2], Mário Cordeiro[1] and João Gama[2]
[1]*LIAAD-INESC TEC, Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal*
[2]*Faculty of Economics and Business, University of Porto, Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal*

Keywords: Large Scale Social Networks Sampling, Data Streams, Telecommunication Networks, *top-K* Networks.

Abstract: The combination of *top-K* network representation of the data stream with community detection is a novel approach to streaming networks sampling. Keeping an always up-to-date sample of the full network, the advantage of this method, compared to previous, is that it preserves larger communities and original network distribution. Empirically, it will also be shown that these techniques, in conjunction with community detection, provide effective ways to perform sampling and analysis of large scale streaming networks with power law distributions.

1 INTRODUCTION

Large Scale Social Networks (LSSN) sampling has emerged as a hot research topic during the recent years. Approaches using full network data revealed to be ineffective, not only due to its computational constraints, but also because of the inherent difficulties to analyze huge networks and draw conclusions by its results observation. In social network analysis, the goal is to get more information from the data, with the least dissociation possible from the nodes of the network.

In this paper, we propose a new method for large-scale network sampling. The method performs an online sampling from a graph stream. The proposed algorithm selects, in real-time, the *k*-most active nodes on the network using the *Space-Saving* algorithm (Metwally et al., 2005). We show that the proposed sampling preserves the same distribution of the original network. We empirically demonstrate that the proposed sampling method can be used to represent global community structure of large networks in a summarized fashion. The results are empirically obtained by simulation of data streaming from databases and with a common commodity computer.

The paper is organized as follows. Section 2 presents the related work regarding methods for large-scale networks sampling. Section 3 introduces the algorithm for *top*-K Networks sampling. In Section 4 we use the proposed method in a large scale social networks dataset from the a telecommunications industry, showing the effectiveness of the proposed method. The last Section highlights the major contributions, and discuss further work to enhance the method.

2 RELATED WORK

2.1 Sampling Large Static Networks

Random sampling and *snowball sampling* are two of the most used strategies to perform sampling on static networks.

Hu and Lau (2013) present a survey on static graph sampling methods and a throughout theoretical overview. This work in progress is continuously updated and is an important reference for researchers in this field.

In *snowball sampling* (Goodman, 1961) a starting node is chosen. After getting the start node, its 1st, 2nd, to *n* order connections are gathered until the network reaches the chosen size for analysis. This approach, while easy to implement, has known problems: it is biased toward the part of the network sampled, and may miss other features. Nevertheless, it is one of the most common sampling approaches.

The *random sampling* (Granovetter, 1976), randomly selects a certain percentage of nodes and keeps all edges between them. As alternative, randomly selects a certain percentage of edges and keeps all nodes that are mentioned. The main problem with this approach is that edge sampling is biased towards high

Sarmento R., Cordeiro M. and Gama J..
Streaming Networks Sampling using top-K Networks.
DOI: 10.5220/0005341402280234
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 228-234
ISBN: 978-989-758-096-3

degree nodes, while node sampling might lose some structural characteristics of the network. Again, this is an easy method to implement.

The task, therefore, must be to sample a sub-graph in such a way that the sub-graph is representative of the original graph. A major question is what it means for a sample to be representative of the original network. Existing works consider such measures as similarity in degree distributions and clustering coefficients (Hübler et al., 2008; Leskovec and Faloutsos, 2006). Leskovec and Faloutsos (2006) present a large variety of graph sampling algorithms. They conclude that methods combining random node selection and some vicinity exploration give best network samples. They show that a 15% sample is usually enough, to match the properties of the original graph and that no list of network properties serving as basis for sampling evaluation will ever be perfect.

2.2 Sampling Large Streaming Networks

Several approaches have been proposed to gather information from streaming graphs. Typical Social Networks analysis problems like counting of triangles, degree measurements, page rank and community detection, among others, have been already implemented following a data stream approach. Network sampling of streaming graphs is still an area open for further research. Ahmed et al. (2012) presents a novel approach to graph streaming sampling. According to the authors, there was no previous contribution to streaming graphs sampling. The authors propose a novel sampling algorithm, PIES, based on edge sampling and partial induction by selecting the edges that connect sampled nodes.

Papagelis et al. (2013) introduces sampling-based algorithms that quickly obtains a near-uniform random sample of nodes in its neighbourhood, given a selected node in the social network. The authors also introduce and analyse variants of these basic sampling schemes, aiming the minimization of the total number of nodes in the visited network, by exploring correlations across samples.

Recently, Ahmed et al. (2014) propose a generic stream sampling framework for big-graph analytics, called Graph Sample and Hold (gSH). It samples from massive graphs sequentially in a single pass, one edge at a time, while maintaining a small state in memory.

Most of these approaches achieve node random sampling through graph streaming. Our objective is diverse. We aim to achieve sampling for specific nodes with high degree. Ahmed et al. (2014) provide means for doing such a sampling with their method focusing on edge sampling and uniform sampling of edges at random. Thus, the sampling method might lead to the selection of a large number of higher-degree nodes but it was not tested on resulting network communities, which is the aim of our work.

3 *top-K* NETWORKS

Scientific community has been trying to achieve efficient ways of doing data streams and graph summarization. The exact solution implies the knowledge of all the nodes and edges frequency, therefore this exact solution might be impossible to achieve in large-scale networks. The proposed method aims the summarization by filtering out less connected nodes. Thus, the proposed sampling approach is biased towards high frequent nodes in the stream. This differentiates the proposed method from previous attempts mentioned in the RELATED WORK section that focus on getting non-biased sampling methods.

3.1 *top-K* Itemsets

The problem of finding the most frequent items in a data stream S of size N is mainly how to discover the elements e_i whose relative frequency f_i is higher than a user specified support ϕN, with $0 \leq \phi \leq 1$ (Gama, 2010). Given the space requirements that exact algorithms addressing this problem would need (Charikar et al., 2002), several algorithms were already proposed to find the top-k frequent elements, being roughly classified into *counter-based* and *sketch-based* (Metwally et al., 2005). *Counter-based* techniques keep counters for each individual element in the monitored set, which is usually a lot smaller than the entire set of elements. When an element is identified as not currently being monitored, various algorithms take different actions to adapt the monitored set accordingly. *Sketch-based* techniques provide less rigid guarantees, but they do not monitor a subset of elements, providing frequency estimators for the entire set.

Simple *counter-based* algorithms, such as *Sticky Sampling* and *Lossy Counting*, were proposed in (Manku and Motwani, 2002), which process the stream in compressed size. Yet, they have the disadvantage of keeping a large amount of irrelevant counters. *Frequent* (Demaine et al., 2002) keeps only k counters for monitoring k elements, incrementing each element counter when it is observed, and decrementing all counters when an unmonitored element is observed. Zeroed-counted elements are replaced by new unmonitored elements. This strategy is similar to the one applied by *Space-Saving* (Metwally et al., 2005), which gives guarantees for the *top-m* most frequent elements. *Sketch-based* algorithms usually focus on families of hash functions which project the counters into a new space, keeping frequency estimators for all elements. The guarantees are less strict but all elements are monitored. The *CountSketch* algorithm (Charikar et al., 2002) solves the problem with a given success probability, estimating the frequency of the element by finding the median of its representative counters, which implies sorting the counters. Also, the *GroupTest* algorithm (Cormode and Muthukrishnan, 2005) employs expensive probabilistic calculations to keep the majority elements within a given probability of error. Despite the fact of being generally accurate, its space requirements are large and no information is given about frequencies or ranking.

3.2 Sampling Algorithm for *top-K* Networks

Algorithm 1 represents the proposed *top-K* method application using the *Space-Saving* algorithm. This type of application is based on landmark windows (Gama, 2010), it implies a crescent number of inspected events in the ever-growing time window. This landmark application is also useful in other contexts, e.g., when the network is relatively small and the user wants to check all events in it.

Basic landmark windows experiments proved to suffer from the problems we wished to avoid, like surpassing memory limits. This happens when the number of nodes and edges exceeds dozens of thousands of nodes. The *top-K* algorithm application, based on Landmark Window, enables an efficient approach for large-scale data. It focuses on the influential nodes and discards less connected nodes, which are the most frequent for power law distribution. The alternative option for sliding windows (Gama, 2010) would not be appropriate for the *top-K* approach, since it may remove less recent graph nodes. Those nodes might yet be included in the *top-K* list we wish to maintain.

In our scenario, *top-K* representation of data streams implies knowing the K elements of the simulated data stream from the database. Network nodes that have higher frequency of outgoing connections, incoming connections, or even specific connections between any node A and B, may be included in the graph, as well as their connections.

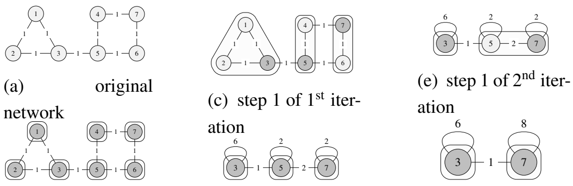
For this application, the user can insert as input a start date and hour and also the maximum number of *top-K* nodes to be represented (the K parameter) along with their connections. With the inserted start date and hour, the *top-K* application is expected to return the evolving network of the *top-K* nodes. Functions *getTopKNodes* and *updateTopNodesList* in **Algorithm 1** implement the *Space-Saving* algorithm. As the network evolves over time, new *top-K* nodes are added to the graph. Nodes that exit *top-K* list of numbers are removed from the *top-K* list and, thus, removed from the graph along with their connections.

Algorithm 1 : *top-K* Pseudo-Code for outgoing connections.

Input: $start, k\_param, tinc$ ▷ start timestamp, k parameter and time increment
Output: $edges$
$R \leftarrow \{\}$ ▷ data rows
$E \leftarrow \{\}$ ▷ edges currently in the graph
$R \leftarrow$ getRowsFromDB ($start$)
$new\_time \leftarrow start$
while ($R <> 0$) **do**
  for all $edge \in R$ **do**
    $before \leftarrow$ GETTOPKNODES(k_param)
    UPDATETOPNODESLIST($edge$) ▷ update node list counters
    $after \leftarrow$ GETTOPKNODES(k_param)
    $maintained \leftarrow before \bigcap after$
    $removed \leftarrow before \setminus maintained$
    for all $node \in after$ **do** ▷ add *top-K* edges
      if $node \subset edge$ **then**
        ADDEDGETOGRAPH($edge$)
        $E \leftarrow E \bigcup \{edge\}$
      end if
    end for
    for all $node \in removed$ **do** ▷ remove non *top-K* nodes and edges
      REMOVENODEFROMGRAPH($node$)
      for all $edge \in node$ **do**
        $E \leftarrow E \setminus \{edge\}$
      end for
    end for
  end for
  $new\_time \leftarrow new\_time + tinc$
  $R \leftarrow$ getRowsFromDB (new_time)
end while
$edges \leftarrow E$

3.3 Communities of *top-K* Nodes

The *top-K* communities in the scope of this work are detected considering only the *top-K* nodes and their 1st and 2nd order connections. Our method samples the original network with a method aiming to keep the characteristics and community structure of the original network. We apply *top-K* sampling to obtain the nodes that belong to the *top-K* group. To retrieve their



(a) original network
(b) initial communities
(c) step 1 of 1st iteration
(d) step 2 of 1st iteration
(e) step 1 of 2nd iteration
(f) step 2 of 2nd iteration

Figure 1: The original Louvain algorithm steps.

network we do a query to the database collecting all connections/edges representing the network with the neighbors of the *top-K* nodes. After having the sampled networks, the *Louvain Method* (Blondel et al., 2008) is applied to find the communities.

Figure 1 briefly explains how the *Louvain* algorithm works. In this figure the sequences describe the individual steps that the algorithm performs for detecting communities. It is non deterministic and performs a greedy optimization method to maximize the modularity of all the network partitions. A two-step optimization is performed for each iteration. In step 1, the algorithm seeks for small communities by locally optimizing the modularity. Only local changes of communities are allowed. In step 2, nodes belonging to the same community are aggregated in a single node representing that community in order to build a new aggregated network of communities. Steps are repeated iteratively until no increase of modularity is possible and a hierarchy of communities is produced. Figure 1(a) represents the initial network; Figure 1(b) represents initial individual node communities; Figure 1(c) represents local modularity optimization after first step; Figure 1(d) represents community aggregation results and the new initial communities; Figure 1(e) and Figure 1(f), are the two Louvain steps, where the local modularity optimization and community aggregation for the second iteration are presented; The algorithm stops at the 2nd iteration, once increasing modularity is no longer possible.

4 CASE STUDY

Telecommunication networks generate large amount of continuous data from users and network equipment. In this particular case study, we use Call Detail Records (CDR) log files, retrieved from equipment distributed geographically. CDR implicitly defines a network, where nodes are clients. An edge corresponds to a call between two clients. The stream of phone calls defines a network stream. Considering the large amount of calls occurred per second, we classify this particular dataset as large-scale network data. The network data has, on average 10 million calls per day. The phone numbers were changed to different identifiers to preserve users anonymity. A call between A and B phones is represented by an edge in the social network. Because some individuals receive and make more than one call, the full networks has an average of 6 million of unique users/nodes per day. The dataset contains anonymous data for 135 days. For each edge/call, timestamp information shows the date and hour of the beginning of the call. The number of calls made per second varies from around 10 at mid-night and reaches its peak at mid-day with 280.

Our goal with this case study was to test if we can use the proposed *top-K* method on large-scale telecommunications networks. We started by inspecting the distribution of the data. We then applied the method and expected the distribution to be maintained for the different *top-K* scenarios with different settings for the K parameter.

After this initial study we wanted to investigate if the larger communities obtained from *top-K* networks were representative of the original data, focusing on the larger communities. Moreover, we also needed to evaluate if the communities were coherent as the data streaming evolved over time.

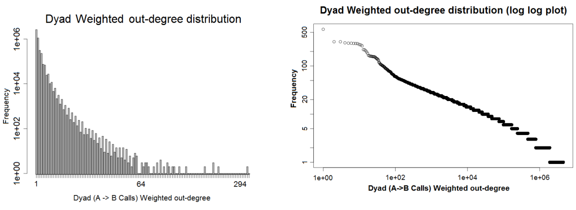
4.1 Data Distribution

To study the distribution of the available data, we aggregate the data in two different ways:

1. Count the number of calls, per day, from phone A to B i.e. A→B
2. Count the number of calls, per day, from each caller phone

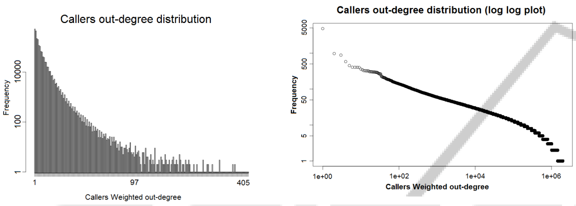
After the previous operation we observed the distribution of the aggregated data and there is some evidence these representations have a power law distribution (Barabasi, 2005) as can be seen in Figure 2(a) and Figure 3(a). These figures illustrate that, regarding a day period, it is expected a high amount of single calls between some A→B phones and a low amount of many calls between A→B phones. Moreover, we can expect a lower amount of highly active callers and a larger amount of low activity callers. We also plotted the distribution of the daily aggregated data with a log-log representation as seen in Figure 2(b) and Figure 3(b). These illustrations show a monomial approximation which indicates that both are derived from power law distributions.

We test the hypothesis that both distributions follow a power law using the method described in Gillespie (2014). We use the software available in the *poweRlaw* R package. The Figure 4 illustrates the hypothesis test for power law distribution presenting the



(a) (b)

Figure 2: A→B Calls Distribution (a) and log-log plot (b).



(a) (b)

Figure 3: Distribution of the Caller Calls (a) and log-log plot (b).

mean estimate of parameters x_{min}, α and the *p-value*, being x_{min} the lower bound of the power law distribution. Estimation parameter α is the scaling parameter ("Par 1" in Figure 4, Figure 7 and Figure 8) and α > 1. The dashed-lines give approximate 95% confidence intervals. The observed *p-value* when testing the null hypothesis H_0 that the original data is generated from a power law distribution is 0.1. Thus, H_0 cannot be rejected because the *p-value* is superior to 0.05. After proving that the data has power law distribution, there was evidence that the proposed *top-K* sampling method is a good approach for this dataset. The next section regards the distribution and characteristics of the *top-K* method application.

4.2 *top-K* Sampling Distributions and Characteristics

As the majority of data concerns isolated calls between two phones, our goal is to get a sampled version of the data that represents the network of most active users in the network. The *Space Saving* algorithm is applied with different settings and different k parameter, i.e. 10000, 50000 and 100000. The

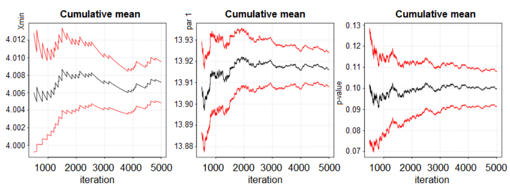


Figure 4: Original Network - Caller power law Distribution hypothesis Test.

respective *top-K* networks were then extracted from querying the database. Finally, the density and clustering coefficient of these networks were compared with the values of the original network (Figure 5 and Figure 6).

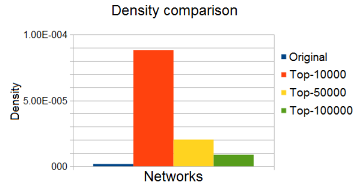


Figure 5: Density comparison between original network and Top-K *Space Saving* Sampling.

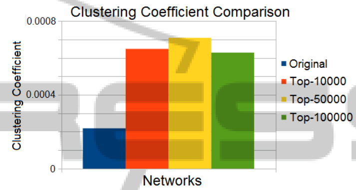


Figure 6: Clustering Coefficient comparison between original network and Top-K *Space Saving* Sampling.

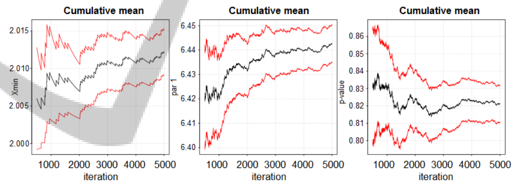


Figure 7: Top-10000 Network - Caller power law Distribution hypothesis Test.

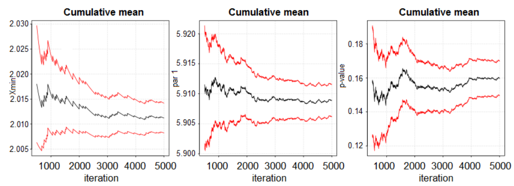


Figure 8: Top-50000 Network - Caller power law Distribution hypothesis Test.

The analysis of Figure 5 and Figure 6 leads to conclude that: i) the density of sampling generated networks lowers as the K parameter of *Space Saving* Sampling algorithm increases; ii) the clustering coefficient of sampling generated networks is more than two times the clustering coefficient of the original network, even though it still holds a low value; iii) as the K parameter of *Space Saving* sampling algorithm increases, the clustering coefficient does not seem to have a significant variation. Figure 7 represents the hypothesis test for power law distribution regarding the *top*-10000 network and for the most active callers. The observed *p-value* is 0.82. Thus, we cannot reject

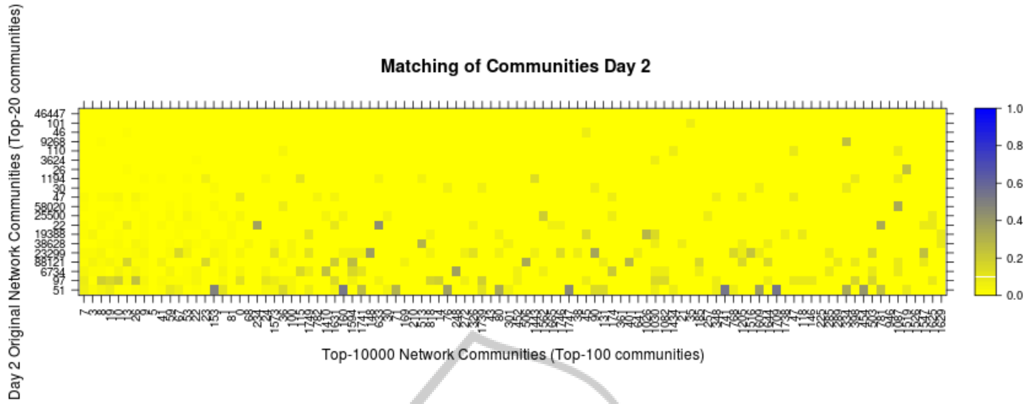


Figure 9: Community elements matching for same day period.

the hypothesis H_0 because the *p-value* is higher than 0.05.

Continuing the tests, Figure 8 represents the hypothesis test for power law distribution regarding the *top*-50000 network and for the 50000 most active callers. The observed *p-value* is 0.16. Therefore we cannot reject the hypothesis H_0. We also did the hypothesis test for power law distribution for the *top*-100000 network regarding 100000 most active caller numbers. Testing the null hypothesis H_0 that the *top*-100000 network for the callers is generated from a power law distribution the observed *p-value* is 0 so we cannot accept it because it is inferior to 0.05.

4.3 Original and Sampled *top-K* Communities Comparison

For the community detection task, both for the original network and the *top-K* networks, we selected the *Louvain Method* described in (Blondel et al., 2008). Figure 9 represents the matching between community elements taken from the *top*-10000 network and for the original network communities without sampling. This task was done for an entire day of data streaming. The matching of communities between both *Louvain Method* results is done by retrieving the percentage of matching elements between any *top-k* network community and the original network communities.

Further analysis of Figure 9 shows the matching of the 100 largest communities for the sampled network and the 20 largest original network's communities. The value of element matching varies with a color gradient between 0 (yellow) and 1 (blue). There is considerable matching of the *top*-10000 sampling communities and the 20 largest communities of the caller original network. These highly active callers and the communities they belong to are therefore represented in the *top-K* sampling as we expected.

Other days in the dataset were also analysed. The results are very similar and consistent throughout full day data comparisons and for the complete dataset of more than 100 days. In all comparisons it is visible that larger original dataset communities are matched by communities retrieved with the proposed *top-K* sampling method.

4.4 Communities of Consecutive Days Samples

Figure 10 represents the matching between community elements taken from the *top*-10000 network communities on consecutive days of the week. The matching in this case corresponds to the percentage of matching elements between any *top-k* network community of one day and all the *top-k* network communities of the following day data.

The matching of the 20 largest communities for consecutive days of daily sampled networks is intuitive with this representation. There is considerable matching of the *top*-10000 sampling communities on consecutive days. This leads to conclude that there is high stability of larger communities as time progresses throughout the week. Similar results were obtained with several combinations of consecutive days

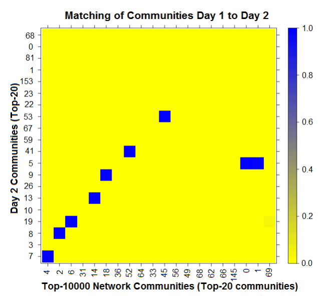


Figure 10: Consecutive days community elements matching.

over the 135 days of the available data. We also observed that there was some decreasing of matching elements when consecutive days represented transition from workdays to weekend days or vice versa. This is expected since the behaviour of major intervenients in the network favour higher activity in working days.

5 CONCLUSIONS

The *top-k* application is a suitable approach to our data that presents a power law distribution. This enables the focus on the influential individuals and discards isolated connections. The use of *Space-Saving* algorithm to sample *top-K* elements in a network is able to keep the original network's power law features. The *Louvain Method* enables the generation of representative communities with the most active elements in the network. This method for evolving networks sampling enables the use of a common commodity computer for massive network analysis. Future work will use Ahmed et al. method and compare it with our method for community detection. We also have the objective of testing the method with real-time data streaming systems.

ACKNOWLEDGMENTS

This work was supported by Sibila and Smartgrids research projects (NORTE-07-0124-FEDER-000056/59), financed by North Portugal Regional Operational Programme (ON.2 O Novo Norte), under the National Strategic Reference Framework (NSRF), through the Development Fund (ERDF), and by national funds, through Fundação para a Ciência e a Tecnologia (FCT), and by European Commission through the project MAESTRA (Grant number ICT-2013-612944); The financial support given by the project number 18450 through the "SI I&DT Individual" program by QREN and delivered to WeDo Business Assurance.

REFERENCES

Ahmed, N. K., Duffield, N., Neville, J., and Kompella, R. (2014). Graph sample and hold: A framework for big-graph analytics. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1446–1455, New York, NY, USA. ACM.

Ahmed, N. K., Neville, J., and Kompella, R. R. (2012). Space-efficient sampling from social activity streams. In Fan, W., Bifet, A., 0001, Q. Y., and Yu, P. S., editors, *BigMine*, pages 53–60. ACM.

Barabasi, A.-L. (2005). The origin of bursts and heavy tails in human dynamics. *Nature*, (435):207–211.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. arxiv.org. Paper which discusses the theory behind the BPLL/Louvain community detection algorithm.

Charikar, M., Chen, K., and Farach-Colton, M. (2002). Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, ICALP '02, pages 693–703, London, UK, UK. Springer-Verlag.

Cormode, G. and Muthukrishnan, S. (2005). What's hot and what's not: Tracking most frequent items dynamically. *ACM Trans. Database Syst.*, 30(1):249–278.

Demaine, E. D., López-Ortiz, A., and Munro, J. I. (2002). Frequency estimation of internet packet streams with limited space. In *Algorithms-ESA 2002*, pages 348–360. Springer.

Gama, J. (2010). *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 1st edition.

Gillespie, C. S. (2014). *Fitting heavy tailed distributions: the poweRlaw package*. R package version 0.20.5.

Goodman, L. A. (1961). Snowball Sampling. *The Annals of Mathematical Statistics*, 32(1).

Granovetter, M. (1976). Network sampling: Some first steps. *American Journal of Sociology*, 81:1267–1303.

Hübler, C., Kriegel, H.-P., Borgwardt, K. M., and Ghahramani, Z. (2008). Metropolis algorithms for representative subgraph sampling. In *ICDM*, pages 283–292. IEEE Computer Society.

Hu, P. and Lau, W. C. (2013). A survey and taxonomy of graph sampling. *CoRR*, abs/1308.5865.

Leskovec, J. and Faloutsos, C. (2006). Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 631–636, New York, NY, USA. ACM.

Manku, G. S. and Motwani, R. (2002). Approximate frequency counts over data streams. In *Proceedings of the 28th International Conference on Very Large Data Bases*.

Metwally, A., Agrawal, D., and El Abbadi, A. (2005). Efficient computation of frequent and top-k elements in data streams. In *Proceedings of the 10th International Conference on Database Theory*, ICDT'05, pages 398–412, Berlin, Heidelberg. Springer-Verlag.

Papagelis, M., Das, G., and Koudas, N. (2013). Sampling online social networks. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):662–676.