Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 17 (2013) 80 – 87

Procedia Computer Science

Information Technology and Quantitative Management (ITQM2013)

Pattern Discovery in DNS Query Traffic

Weizhang Ruan[a], Ying Liu[b,c,*], Renliang Zhao[b]

[a]School of Information and Telecommunication Engineering, Beijing University of Posts and Telecommunications
10 West TuCheng Road, Beijing 100876, China
[b]*School of Computer and Control, Universiy of Chinese Academy of Sciecnes*
[c]*Fictitious Economy and Data Science Research Center, Chinese Academy of Sciences*
80 ZhongGuanCun East Road, Beijing 100190, China

Abstract

DNS provides a critical function in directing Internet traffic. Traditional rule-based anomaly or intrusion detection methods are not able to update the rules dynamically. Data mining based approaches can find various patterns in massive dynamic query traffic data. In this paper, a novel periodic trend mining method is proposed, as well as a periodic trend pattern based traffic prediction method. Clustering is adopted to partition numerous domain names into separate groups by the characteristics of their query traffic time series. Experimental results on a real-word DNS log indicate data mining based approaches are promising in the domain of DNS service.

Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management

Keywords: Pattern discvery, Data mining, DNS, Clustering, Prediction, Anomaly detection;

1. Introduction

Domain Name System (DNS) is a hierarchical distributed naming system for computers, services, or any resource connected to the Internet. A Domain Name Service resolves queries for URLs into IP addresses for the purpose of locating computer services and devices worldwide. By providing a worldwide, distributed keyword-based redirection service, DNS is an essential component of the functionality of the Internet.

With the ever increasing network flow and complexity of network topology, problems often happen in DNS service. For example, a large-scale network break-down happened in 6 provinces in China in 2009 due to an attack to the DNS server by a hacker. China has been one of the countries suffering from enormous network attacks in the world. Security has become a crucial problem in DNS service. Thus, it is an important task for

* Corresponding author. Tel.: 86-10-82163605.
E-mail address: ruanweizhang@gmail.com, yingliu@ucas.ac.cn, zhaorenliang1@126.com.

Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management
doi:10.1016/j.procs.2013.05.012

DNS service providers to detect and report anomalies or exceptions as early as possible, and reduce the loss resulted from the unexpected events. Another important task is to provide high quality service to Web users.

Traditional methods in DNS security are rule-based methods. DNS experts have to identify the characteristics or features of any abnormal behavior from historical data offline, and then explicitly provide them to the monitoring system in the form of rules. However, such rule-based methods have two serious weaknesses: 1) the rules are not easy to update since efforts of domain experts are required. However, the patterns of abnormal behaviors in the network are evolving dramatically, and thereby the effectiveness of the detection system will be significantly reduced; 2） the size of historical data set collected by DNS system is so huge that beyond the ability of human being to analyze. For example, the number of query records captured by DNS log at a top level domain server is over 40 billion in a single month. Automatic quantitative analysis techniques on massive DNS data are in real demand.

Data mining is a kind of technique that can discover interesting, meaningful and understandable patterns hidden in massive datasets. The patterns discovered by data mining can be utilized in decision-making in many domains. To our best knowledge, data mining has not been widely used in DNS query traffic analysis yet. Therefore, in this paper, we explore to solve problems in DNS service by applying various data mining methods. Our contributions are listed as follows：

1) In order to predict the traffic volume at a domain name and prevent attacks by hackers, we propose a *query traffic trend prediction method*. It discovers the periodically occurred query traffic trend patterns from the most recent DNS log. If the current query traffic of a domain name does not match with the predicted trend, an anomaly alarm will be delivered to the system instantly.
2) In order to have a deep understanding of the characteristics of the query traffic of different domain names, we partition the query traffic time series from all the domain names into distinct clusters by adopting a clustering algorithm. The representative query traffic time series of each cluster is referred as the query traffic pattern. Such results provide us a chance to further investigate the browsing patterns of the Web users or identify the common characteristics of various anomalous queries.
3) The effectiveness of our proposed methods is examined by a real-world DNS log dataset and the experimental results are presented.

The rest of this paper is organized as follows. Section 2 overviews some related work. In Section 3, we present our proposed query traffic trend prediction method. In Section 4, we briefly introduce SNN clustering algorithm and our clustering-based analysis. Section 5 summaries our current work and point out the future work.

2. Related Work

Since query traffic flow is an accurate reflection of DNS service, anomaly detection in query traffic has been paid more and more attention. For example, Jung et al. [1] proposed a novel method to detect anomaly in SMTP Client by DNS query traffic. Ishibashi et al. [2] proposed a method to discover junk mail senders by studying ISP DNS. But in some circumstance, DNS itself can be part of the attack in Internet like DDoS [3] and DNS cache poisoning [4].

Ji et al. [5] proposed a K-means clustering based algorithm to cluster the temporal behaviors of IP addresses and domain names. It partitions the domain names into four clusters. Rather than comparing the traffic volume between happened at different domain names, [5] performs clustering on a set of derived variables, such as the total number of DNS requests, the number of distinct IPs, average time interval between two DNS requests, etc. Although it produces interesting results, it needs prior knowledge of the number of clusters, k, which is not easily obtained beforehand.

Wang et al. [6] proposed a mathematical method to detect nation-wide large-scale attacks on the Internet. A covariance matrix is built to record the covariance between the query volumes happened at two different provinces at different time stamps. Average covariance matrix indicates a normal situation. If the current

covariance matrix deviates from the average covariance significantly, an abnormal event may be going on. This method is suitable for nation-wide attacks but fails in the detection of attacks towards a specific domain name.

Xu et al. [7] improved RIPPER algorithm to detect Botnet, which is often used for malicious activities (e.g., DDos, spam, phishing etc.). It outperforms the traditional algorithms, such as features matching or statistical methods, in discovering more less-visited domain names.

3. Query Traffic Trend Prediction and Anomaly Detection

Anomaly detection in DNS query traffic is significant for DNS service providers. Malfunction of DNS server or network may result in abnormal query traffic, as well as abnormal behaviours or illegal behaviours of Web users. Thus, DNS query traffic indicates the health of the DNS and the network. As far as our knowledge concerns, most existing algorithms [1, 6, 7] are rule-based or black-list based approaches, where the rules are not able to be adjusted dynamically.

In this section, we propose a novel problem, *periodic query traffic trend mining*, and a method to discover all the periodic trend patterns in a sequence database. We assume that if a pattern occurs frequently in the recent history, it may occur repeatedly with high probability unless an abnormal event happens. Since patterns with time interval tend to be haphazard and unreliable, we are only interested in patterns with no interval. Then we predict the traffic volume at the next moment by referring the recent periodically occurred query traffic patterns.

3.1. Problem statement

Periodic trend mining problem can be stated as follows:

- $I = \{i_1, i_2 \ldots, i_m\}$ is a set of items;
- $T = \{t_1, t_2 \ldots, t_p\}$ is a set of timing stamps;
- An event set, E, is a 2-tuple, (i_j, t_k) , where item i_j $(1 \le j \le m)$ happened at t_k $(1 \le k \le p)$;
- $D = \{S_1, S_2, \ldots, S_n\}$ is a time series database where each time series $S_i \in D$ consists of a sequence of events;
- An event set E is a *periodic trend* if $sup(E) \geqslant min\_sup$, and the timing stamps in E are consecutive, where *min_sup* is the user pre-specified minimum support threshold;
- The task of periodic trend mining is to find the complete set of periodic trends, $U = \{E| sup(E) \geq min\_sup \}$.

Let's use the query traffic in Table 1 as an example. Let $I = \{0, 1, 2\}$, and $T = \{0, 1, 2, \ldots, 23\}$, denoting 24 hours of a day. Assume the minimum support threshold is 3, then ((0:0), (1:1), (1:2)) is a periodic trend pattern because it occurred in three sequences, S_2, S_3, S_4, consecutively occurred from 0 o'clock to 2 o'clock. Since ((1:2), (2:3)) only occurred in sequence S_1 and S_2, it is not a periodic trend pattern although ((2:2), (1:3)) occurred in S_{28}, where the two items, 1 and 2 did not occur in the right order. ((0:0), (1:2), (2:3)) is not a periodic pattern because it violates the definition that the timing stamps must be consecutive.

Table 1.Query traffic variation database of *Yahoo.cn*

ID	Traffic query traffic variations（24 hours）			
S_1	(0:0)	(0:1)	(1:2)	(2:3) ……
S_2	(0:0)	(1:1)	(1:2)	(2:3) ……
S_3	(0:0)	(1:1)	(1:2)	(0:3) ……
S_4	(0:0)	(1:1)	(1:2)	(0:3) ……
	……			
S_{28}	(0:0)	(1:1)	(2:2)	(1:3) ……

3.2. Periodic trend mining

If the frequency of an event set, E, beyond a user pre-specified minimum support threshold, we call E a periodic trend pattern. Each event is an item with an associated timing stamp. A periodic pattern indicates that

```
Input: Time series database D, minimum support threshold, min_sup.
Output: U, periodic trends in D.

1     U1 = find_periodic 1-event-sets (D);
2     for (k = 2; Uk-1 ≠ ∅; k++) {
3          Ck = candidate_gen (Uk-1, min_sup);
4          for each time series S ∈ D
5               for each c ∈ Ck
6                    if  c ⊆ S then
7                        c.count++;
8          Uk = {c ∈ Ck | c.count≥ min_sup}
9     }
10    return U = ∪k Uk;

procedure candidate_gen (Uk-1: periodic (k-1)-event-sets; min_sup: minimum support
threshold)
1     for each event-set p1 ∈ Uk-1
2          for each event-set p2 ∈ Uk-1
3               if (p1 1] = p2 1]) ∧ (p1 2] = p2 2]) ∧ … ∧ (p1 k-2] = p2 k-2])) then {
4                    c = p1 ⊳⊲ p2 in increasing order by time;
5                    if timing points of c[k-1] and c k] are not consecutive then
6                         delete c;
7                    else if has_infrequent_subsets (c, Uk-1) then
8                         delete c;
9                             else add c to Ck;
10              }
11     return Ck;

procedure has_infrequent_subsets (c: candidate k-event-sets; Uk-1: periodic (k-1)-
event-sets)
1  for each (k-1)-event-set s of c
2       if s ∉ Uk-1 then
3            return TURE;
4  return FALSE;
```

Fig. 1. Pseudo code of periodic trend mining algorithm.

the sequence occurs frequently/repeatedly at the same time. Periodic trend mining is to find all the periodic trends. The periodic trend mining problem follows *downward closure property* or *Apriori property* [8], that is, if a trend E is not a periodic (frequent) pattern, its superset cannot be periodic (frequent) at all. Thus, we propose an iterative level-wise approach based on candidate generation.

Figure 1 shows the pseudo code of our proposed periodic trend mining algorithm. U denotes the collection of all the periodic trend patterns, and events within any $p \in U$ are in increasing order in time. It follows the *generation-and-test* methodology like Apriori [8]. In the main loop (from line 2 to 9), it generates the k-event-set candidates and then finds out the real periodic k-event-sets by scanning the database. With the support of *Downward Closure Property*, procedure *candidate_gen* ( ) generates a k-event-set candidate by joining two periodic (k-1)-event-sets which share the common (k-2) prefix. Note that the timing points of all the event in each event-set must be consecutively increasing.

3.3. Query traffic trend prediction and anomaly detection

We predict the traffic volume of a given domain name at the incoming moment by referring the recent periodic patterns. The prediction method can be roughly outlined in three steps:
1) Mine the periodic query traffic trend patterns from the recent query traffic data at a given domain name;
2) Obtain strong patterns from the periodic trend patterns. Here we adopt the concept, *confidence*, from association rules mining problem [8]. Assuming $E=A \cup B$ is a periodic (frequent) trend, the confidence of E is defined as the conditional probability of B given A, that is, $confidence(E)= confidence(A \cup B) = P(B|A)$. If a periodic trend pattern E satisfies the minimum confidence threshold, *min_conf*, we call E a strong trend pattern.
3) Predict the incoming query traffic by referring the strong patterns. If the actual traffic volume deviates from the prediction significantly, an anomaly warning will be delivered to the central monitoring system immediately.

3.4. Experimental results

We use a real-world DNS log dataset in our experiments. A DNS log record is generated whenever a DNS request is issued by a client. For example, when a user browses a domain name by its URL, a DNS log record will be created and recorded in the log. Attributes collected by DNS log include requesting time, source IP, destination URL, query type, etc. The size of a DNS log is huge. For example, around 40 billion records are captured at a top level domain server every month. It is even larger at a root domain server. The DNS log dataset we use in our paper is captured by CN Top Level Domain Server in February 2012, containing 40 billion records.

An expert at China Internet Network Information Center suggested that an alert of an anomalous variation is more useful and valuable to its system, rather than the raw number of visits, therefore, we pre-process the log data as follows:

1) Find all the distinct domain names occurred in the log;
2) Count the number of visits to each domain name by hour by day;
3) Create a separate dataset for each domain name, each row containing a sequence ID and the number of visits happened hourly;
4) Transform the number of visits into variation. '*0*' denotes *decline* in query traffic volume from the last hour, '*1*'denotes *rise*, and '*2*'denotes *steady*.

We select relatively popular domain names for our experiment, whose average query frequency per hour is over 2000. 676 data sets are created, one set per domain name. Table 1 shows part of the dataset for *Yahoo.cn*, where each row represents the query traffic variations hourly in a day.

Experiments are performed on all the domain names and numerous strong periodic trend patterns are mined. Due to the space limitation, we only present the results from *Yahoo.cn* as an example. 37 strong periodic trend patterns are mined from the query traffic variation database of *Yahoo.cn* when *min_sup* and *min_conf* are set at 35% and 70%, respectively. Table 2 presents some of the strong periodic trend patterns. From Table 2, we can make predictions with high confidence in some cases. For example, if the query traffic has been rising continuously for two hours since 21:00pm, the probability the traffic will continue rising from 23:00 to 24:00pm is 100%. So, if the query traffic volume declines some time between 23:00 to 24:00pm, we have high confidence to believe that an anomalous event is happening towards *Yahoo.cn*.

Table 2. Strong periodic trend patterns discovered at *Yahoo.cn*

Pattern length	Start time	Periodic Trend Patterns	Support	Confidence
3	13	[1. 1. 1.]	13/28	If the query traffic at *Yahoo.cn* has been rising for two hours since 13:00pm, the probability the traffic will continue rising from 15:00 to 16:00pm is 92.8%.
3	17	[0. 0. 0.]	11/28	If the query traffic has been declining for two hours since 17:00pm, the probability the traffic will continue declining from 19:00 to 20:00pm is 84.6%.
3	21	[1. 1. 1.]	13/28	If the query traffic has been rising for two hours since 21:00pm, the probability the traffic will continue rising from 23:00 to 24:00pm is 100%.
17	0	[0. 0. 0. 0. 0. 0. 0.]	11/28	If the query traffic has been declining for three hours since 00:00am, the probability the traffic will continue declining from 3:00 to 7:00am is 73.3%.
…	…	…	…	…

4. Clustering-based Query Traffic Characteristic Analysis

The number of Domain names on the Internet is so enormous that beyond the ability of human being to analyse their query traffic characteristics one by one. Thus, it is necessary to partition the domain names into groups where domain names with similar query traffic are in the same group, and domain names with dissimilar characteristics are in other groups. Experts can thereby perform analysis in each group, such as extracting the common characteristics of the Web users, or identifying the common characteristics of anomalous behaviors, which is useful for DNS service providers to prevent attacks, and valuable for National Security Bureau.

4.1. Shared-Nearest-Neighbour clustering

Clustering is one of the most widely used data mining techniques. It partitions the data into clusters so that objects within a cluster have high similarity in comparison to one another but are dissimilar to objects in other clusters. Dissimilarities are assessed based on the attribute/dimension values describing the objects. Distance is often used as the similarity measure. K-means is the most popular clustering algorithm with no doubt. K-means clustering based has been studied in DNS Query traffic in [5]. Although it produces interesting results, it needs prior knowledge of the number of clusters, *k*, which is not easily obtained beforehand. Another serious weakness of K-means is that it is not suitable for high dimensional data.

In this paper, we are interested in finding out the characteristics of query traffic time series at different domain names, which is high in dimensionality in nature. Therefore, we adopt a more recent clustering algorithm, Share-Nearest-Neighbor clustering (SNN). In SNN, similarity of two objects is measured by the number of neighbors that two objects share. Density of an object is defined as the sum of the similarities of an object's nearest neighbors. SNN eliminates noise (low density objects) and builds clusters by associating non-noise objects with representative or core objects (high density objects). The neighbor-based similarity measure allows SNN to find clusters for high dimensional objects, and the use of core/representative objects allows SNN to find arbitrarily shaped clusters. In addition, SNN automatically determines the number of clusters. Experimental results on high dimensional time series data have been reported [9].

4.2. Data pre-processing

We use the DNS log captured by CN Top Level Domain Server in February 2012 in our experiment. The temporal nature of DNS query traffic series poses a number of challenges. For instance, DNS query traffic series is noisy; such data displays temporal autocorrelation, i.e., traffic volume close in time tend to be highly

correlated, or similar. To handle the issues of 'seasonality' (hourly cycles) and temporal autocorrelation, we pre-processed the data to remove seasonality by hourly *Z-score* transformation. The Z-score transformation also reduces temporal autocorrelation significantly. Since the time series spanning 28 days (24 numbers each year), we also detrended the data before our analysis. Our data pre-processing method can be outlined as follows:

1) Find all the distinct domain names occurred in the log;
2) Count the number of visits to each domain name by hour by day;
3) Select the most popular 676 domain names whose average number of visits per hour beyond 2000;
4) Use hourly *Z-score* transformation, which takes the set of numbers for a given hour, calculates the mean and standard deviation of that set of numbers, and then 'standardizes' the data by subtracting of the corresponding hourly mean and then dividing by the hourly standard deviation;
5) Detrend the query traffic time series for each domain name.

Eventually, we obtained a table consisting of 676 rows (domain names) and 672 columns (28 days x 24 hours), where each number denotes the transformed number of visits to a given domain name in a given hour.

4.3. Experimental results

We implement SNN exactly the same as in [9]. Here the link strength threshold for representative object and noise are set at 3800 and 200, and the strong link threshold is set at 10. Seven clusters are discovered and the corresponding clustering representatives are presented in Figure 2, from 2(a) to 2(g), where each curve reflects the hourly query traffic volume. 50 out of 676 domain names are discarded as noise. In Figure 2, we can see that the difference between the query traffic trends of different clusters is evident, indicating that the behaviors of the users visiting the domain names in different clusters have quite different characteristics, especially cluster 7 in Figure 2(g). Although our current clustering result is preliminary, it is promising to segment abnormal temporal behaviors from others by SNN clustering, and thereby extract the characteristics of anomalous DNS queries.

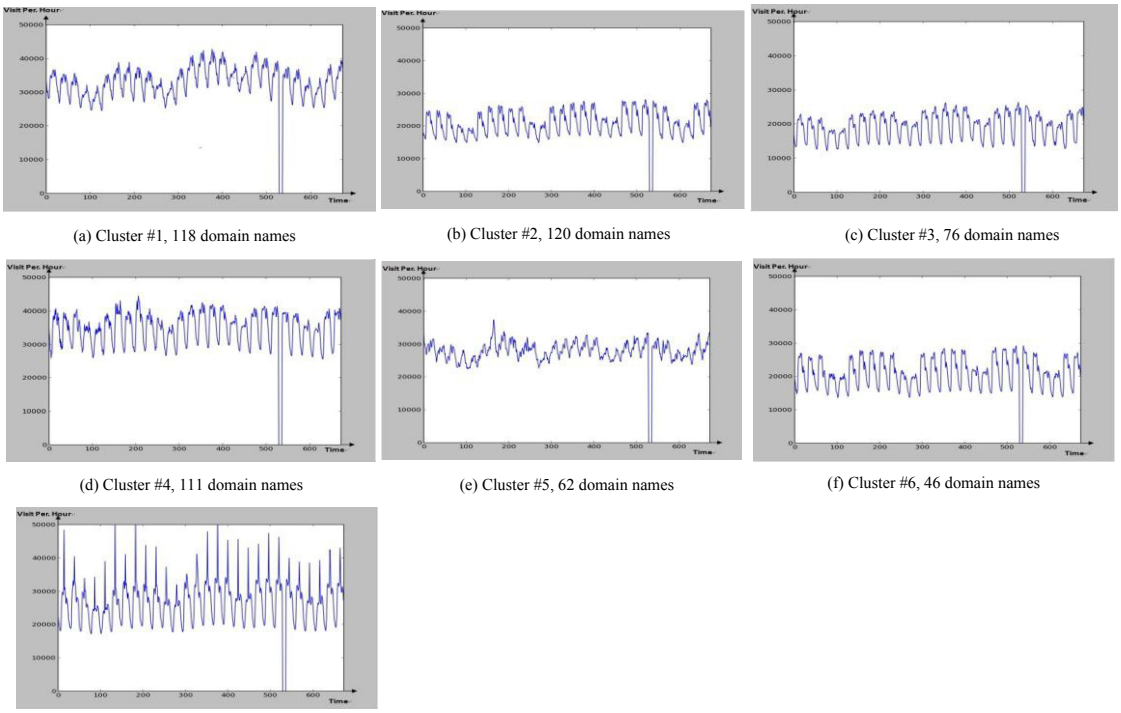
5. Conclusion & Future Work

In this paper, we focused on discovering useful patterns from the DNS query traffic data to assist DNS service providers to detect anomalies and get to learn the behavior patterns of the Web users of different domain names. Since DNS log data is enormous in size and evolving fast, data mining approaches which can discover patterns dynamically are adopted in this work. Firstly, we proposed a novel problem, *periodic trend mining,* and then a prediction method is proposed to predict the incoming traffic volume and detect anomaly. Secondly, we partitioned the query traffic time series data into separate clusters by using SNN clustering algorithm, and thereby further insights of each cluster are studied. The effectiveness of our methods has been demonstrated by the experimental results on a real-world DNS query traffic log dataset. Interesting patterns are observed which are useful for the DNS service providers and the National Security Bureau.

A few more experiments need to be performed in the recent future. For example, vary the length of the query traffic sequence, which may bring more interesting clusters; applying our methods on the data of all the domain names may provide us more interesting results.

Following research will be conducted in the near future:

1) Explore methods to discover the propagation routes of query traffic towards some particular domain names;
2) Analyze the characteristics or features of each cluster, and the contents at the corresponding websites, as well as the characteristics of the users behind;
3) Our current work is focusing on the 'destination' of each DNS request. How to mine useful pattern from the IP addresses of the source requires investigation, especially with geographical information of the IP addresses;
4) Derive more variables from raw variables to find more valuable patterns;
5) Explore methods to mine query traffic patterns under the influence of unexpected incidents.



(a) Cluster #1, 118 domain names (b) Cluster #2, 120 domain names (c) Cluster #3, 76 domain names

(d) Cluster #4, 111 domain names (e) Cluster #5, 62 domain names (f) Cluster #6, 46 domain names

Figure 2. Cluster representatives of DNS query traffic data.

Acknowledgements

Project is supported by National Natural Foundation of China #70921061 and DNSLAB at China Internet Network Information Center.

References

[1] Jung, J., Sit, E. An Empirical Study of Spam Traffic and the Use of DNS Black Lists. *Proc. of the 4th ACM SIGCOMM Conference on Internet Measurement*, 2004, pp. 370-375.

[2] Ishibashi， K., Toyono T., Toyama, K., et al. Detecting Mass-mailing Worm Infected Hosts by Mining DNS Traffic Data. W*orkshop on Mining Network Data at ACM SIGCOMM*, USA, 2005, pp. 159-164.

[3] Klein, A. BIND 9 DNS cache poisoning. *http://www.trusteer.com/docs/BIND_9_DNS_Cache_Poisonding.pdf*.

[4] US-CERT. The Continuing Denial of Service Threat Posed by DNS Recursion. *http://www.us-cert.gov/readingroom/DNS-recursion033006.pdf*.

[5] Cheng, J., Li, X., Yuan, J., et al. K-means Based Analysis of DNS Query Patterns. *Journal of Tsinghua University*, 2010.

[6] Wang, Z., Li, X., Yan, B. Abnormity Detection of DNS Query Traffic at CN Top Level Domain Server, Technical Report, 2010. *http://www.cdns.cn/about/vision-2.html.*

[7] Xu, H., Li, Z., Zhou, L. Botnet Detection Methods in DNS Traffic. *Journal of Xiamen University*, 2007.

[8] Agrawal, R., Srikant, R. Fast Algorithms for Mining Association Rules. *Proc. of the 20th International Conference on Very Large Data Bases*, 1994, pp. 487-499.

[9] Ertoz, L., Steinbach, M., Kumar, V. A New Shared Nearest Neighbor clustering Algorithm and its Applications. *Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining*, 2002.