Proceedings of the 5th WSEAS Int. Conf. on SIMULATION, MODELING AND OPTIMIZATION, Corfu, Greece, August 17-19, 2005 (pp231-236)

Clustering the Internet Topology at the AS-level

BILL ANDREOPOULOS[1], AIJUN AN[1], XIAOGANG WANG[2]
[1]Department of Computer Science and Engineering, York University
[2]Department of Mathematics and Statistics, York University
4700 Keele Street, Toronto, Ontario M3J 1P3
CANADA
http://www.cs.yorku.ca/~billa/MULIC/

Abstract: Clustering algorithms presented in the literature are rarely designed for the layered structure of the Internet topology and its unknown number of clusters. A recent approach in the research community is to collect IP path data and to map the IPs to known Autonomous Systems (ASs) forming the Internet backbones, resulting in large data sets of AS links. Clustering data sets of AS links permits inferring business relationships between ASs. We propose the MULICsoft algorithm for this purpose. MULICsoft clusters categorical data sets where the objects to be clustered are ASs and each categorical attribute value (CA) represents a link between ASs. Each CA has a 'weight' in the range of 0.0 to 1.0, that is inversely related to the number of unknown ASs in a link between ASs. MULICsoft produces as many clusters as it can find in the data set and each cluster consists of layers. The clustering results reflect the Internet topology at the AS-level, permitting inferring relationships between ASs.

Key-Words: Clustering, Multiple Layer, Categorical, Internet, AS, Autonomous System.

1 Introduction

Because of the large size of the Internet, researchers often restrict analysis of the Internet to the AS-level. An AS is a connected group of IP networks that adhere to a single and clearly defined routing policy. Routers typically receive multiple path advertisements for establishing communication between two Internet hosts and usually select the best path as the one that has the smallest number of ASs [3, 4].

Links between ASs represent communication between ASs in the form of packet transmission. Clustering the Internet topology at the AS-level using data sets of AS links has several benefits, such as identifying business relationships between ASs. Knowledge of the AS business relationships leads to understanding the relative importance of ASs. This helps providers and policy makers who define the routing policies to save money [4].

Clustering aims to partition a set of objects into groups, so that objects with similar characteristics are grouped together and different groups contain objects with dissimilar characteristics [1, 2]. Clustering can be applied to Internet data to identify prominent groups of the Internet map and help to understand the Internet structure. We cluster the ASs in the data set that is provided by CAIDA, representing links between ASs [3]. The goal is to identify groups of ASs with similar connectivity to the rest of the Internet. A cluster represents ASs that communicate with similar sets of ASs through sending and receiving packets.

The main contribution of this work is to propose a novel clustering algorithm that is designed for the structure of the Internet topology. We present the MULICsoft categorical clustering algorithm that is based on MULIC [2]. We have shown that MULIC results are of higher quality than those of other categorical clustering algorithms, such as k-Modes, ROCK, AutoClass and CLOPE in [2]. Some characteristics of MULICsoft and MULIC are: *a.* The number of clusters does not have to be specified by the user. Instead, it produces as many clusters as it can find in the data set. *b.* Each cluster consists of layers formed gradually, by relaxing the similarity criterion for inserting objects in layers.

This paper is organized as follows. Section 2 describes the AS data set. Section 3 presents the MULICsoft clustering algorithm. Section 4 discusses the experimental results. Section 5 discusses the properties of the AS graph. Finally, section 6 concludes the paper.

2 Description of Data Sets

An AS is a collection of IP networks under control of a single entity, typically an Internet service provider (ISP) or a very large organization. We apply the MULICsoft clustering algorithm to the AS data set provided by the Cooperative Association for Internet Data Analysis (CAIDA). This data set is derived from IP addresses collected using a TCP utility called traceroute that are mapped to ASs [3, 4]. This data set was generated on June 23, 2004. The data represents packets that have traversed a path as links between ASs. We chose this data as it is more likely to faithfully correspond to IP topology than traditional Border Gateway Protocol (BGP) Table Dump data [3].

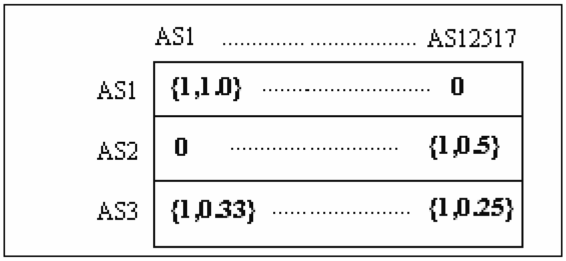
Each object in the data set corresponds to an AS on the Internet. Each AS has CAs of values 'zero' or 'one' representing the links between ASs, i.e. packets sent between ASs. The data set contains a large number of 'zeros' with 'ones' occurring sparsely. There are 12,517 objects representing ASs in the data set [3]. The links in the AS data set might be:

Direct links between ASs: A direct link between ASs exists if an IP was found in the source AS's address space and it was directly followed by an IP in the destination AS's address space.

Indirect links between ASs: An indirect link between ASs exists if an IP was found in the source AS's address space, followed by a number of IPs with an unknown prefix, ended by an IP in the destination AS's address space [3].

We use the number of unknown IP addresses *k* in a link between ASs to determine the 'weight' on the corresponding CA. A weight is determined by evaluating *1/(k+1)*. Thus, a *k* value of 0 is a weight of 1.0, a *k* value of 1 is a weight of 0.5, a *k* value of 3 is a weight of 0.25 and so on. The weights represent the confidence regarding the accuracy of the links. Indirect links are less accurate than direct links, since several intermediate ASs exist but are not shown in the data. The weights are meant to affect the clustering process by placing less importance on links with low confidence to be accurate. We use a special similarity formula that incorporates the weights, described in section 3.2.

Figure 1 shows the matrix representation of the data set. A cell with a CA of 'one' represents a link between the corresponding ASs and 'zero' represents no link. Each link has a weight 0.0-1.0.



	AS1	....	AS12517
AS1	{1,1.0}	....	0
AS2	0	....	{1,0.5}
AS3	{1,0.33}	....	{1,0.25}

Fig. 1. Cells representing AS links have values {CA: zero or one, weight: 0.0-1.0}.

3 MULICsoft Clustering

MULICsoft is an extension of the k-Modes clustering algorithm for categorical data sets [2, 5]. The k-Modes clustering algorithm requires the user to specify the number of clusters to be produced and the algorithm builds and refines the specified number of clusters. Each cluster has a mode associated with it. Assuming that the objects in the data set are described by *m* categorical attributes, the mode of a cluster is a vector $Q=\{q_1, q_2, \ldots, q_m\}$ where q_i is the most frequent value for the *i*th attribute in the given cluster.

MULICsoft makes substantial changes to the k-Modes algorithm. The purpose of the MULICsoft clustering algorithm is to maximize the similarity between the object and the mode of the cluster in which the object is inserted:

$$\mathit{similarity}(o_i, \mathrm{mode}_i) \qquad (1)$$

where o_i is the *i*th object in the data set and mode_i is the mode of the *i*th object's cluster. The similarity metric is described in Section 3.2. It is a variation of the k-Modes similarity that returns the number of identical CAs between an object and a mode, adjusted by their weights [5]. Maximizing (1) ensures that all objects are as similar to their clusters' modes as possible when clustered.

The MULICsoft algorithm has the following characteristics. First, the number of clusters is not specified by the user. Clusters are created, removed or merged during the clustering process, as the need arises. Second, it is possible for all objects to be assigned to clusters of size two or greater by the end of the process. However, outliers are assigned to separate clusters of size one. Third, clusters are layered.

Figure 2 shows the main part of the MULICsoft clustering algorithm. The algorithm starts by reading all objects from the input file and storing them in *S*. The first object is inserted in a new cluster, the object becomes the mode of the cluster and the object is removed from *S*. Then, it continues iterating over all objects that have not been assigned to clusters yet, to find the closest cluster. In all iterations, the closest cluster for each unclassified object is the cluster with the highest similarity between the cluster's mode and the object, as computed by the similarity metric [1, 2].

```
Input:  (1)  a set $S$ of objects;
Parameters:  (1)  $\delta\varphi$ : the increment for $\varphi$;
             (2)  threshold for $\varphi$ : the maximum number
                  of values that can differ between an
                  object and the mode of its cluster;
Default parameter values: (1)  $\delta\varphi = 1$;
                          (2)  threshold = the number of
                               categorical attributes $m$;
Output: a set of clusters;
Method:
  1.  Insert the first object into a new cluster, use the
      object as the mode of the cluster, and remove the
      object from $S$;
  2.  Initialize $\varphi$ to 1;
  3.  Loop through the following until S is empty or $\varphi$
      is greater than the specified threshold
      a.  For each object $o$ in $S$
            i.   Find $o$'s closest cluster $c$ by using the
                 similarity metric to compare $o$ with the
                 modes of all existing cluster(s);
            ii.  If the number of different values
                 between $o$ and $c$'s mode is larger than
                 $\varphi$, insert $o$ into a new cluster
            iii. Otherwise, insert $o$ into $c$ and update
                 $c$'s mode;
            iv.  Remove object $o$ from $S$;
      b.  For each cluster $c$, if there is only one object in
          $c$, remove $c$ and put the object back in $S$;
      c.  If in this loop no objects were placed in a
          cluster with size > 1, increment $\varphi$ by $\delta\varphi$.
```

Fig. 2. The MULICsoft clustering algorithm.

The variable *φ* is maintained to indicate how strong the similarity has to be between an object and the closest cluster's mode for the object to be inserted in the cluster – initially *φ* equals 1, meaning that the similarity has to be very strong between an object and the closest cluster's mode. If the number of different CAs between the object and the closest cluster's mode is greater than *φ* then the object is inserted in a new cluster on its own, else, the object is inserted in the closest cluster and the mode is updated.

At the end of each iteration, all objects assigned to clusters of size one have their clusters removed so that the objects will be re-clustered at the next iteration. This ensures that the clusters that persist through the process are only those containing at least 2 objects for which the required similarity can be found. Objects assigned to clusters with size greater than one are removed from the set of unclassified objects *S*, so those objects will not be re-clustered.

At the end of each iteration, if no objects have been inserted in clusters of size greater than one, then the variable *φ* is incremented by *δφ*. Thus, at the next iteration the criterion for inserting objects in clusters will be more flexible. The iterative process stops when all objects are classified in clusters of size greater than one, or *φ* exceeds a user-specified *threshold*. If the *threshold* equals its default value of the number of attributes *m*, the process stops when all objects are assigned to clusters of size greater than one.

The MULICsoft algorithm can eventually classify all objects in clusters, even if the closest cluster to an object is not that similar, because *φ* can continue increasing until all objects are classified. Even in the extreme cases, where an object *o* with *m* attributes has only zero or one value similar to the mode of the closest cluster, it can still be classified when *φ = m or φ = m-1*, respectively.

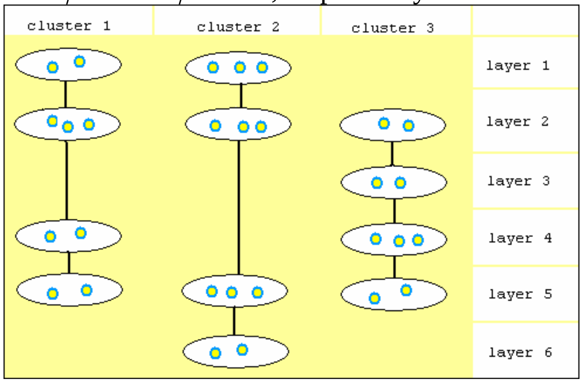


Fig. 3. MULICsoft results. Each cluster consists of one or more different layers representing different similarities of the objects attached to the cluster.

Figure 3 illustrates what the results of MULICsoft look like. Each cluster consists of many different "layers" of objects. The layer of an object represents how strong the object's similarity was to the mode of the cluster when the object was assigned to the cluster. The cluster's layer in which an object is inserted depends on the value of *φ*. Lower layers have a lower coherence - meaning a lower average similarity between all pairs of objects in the layer - and correspond to higher values of *φ*. MULICsoft starts by inserting as many objects as possible in top layers – such as layer 1 - and then moves to lower layers, creating them as *φ* increases.

3.1 MULICsoft Characteristics for Internet Data Clustering

MULICsoft includes characteristics specific for Internet data clustering, allowing the incorporation of both categorical attribute values (CAs) and weights in the clustering process.

All CAs of an object have "weights" in the range of 0.0 to 1.0 associated with them, which represent the number of unknown ASs in an indirect link between ASs. The weights are extracted as described in Section 2. We represent the weights of an object *o* as a vector *w_o*.

A position of the mode of a cluster is set to 'one' if there is at least one object in the cluster that has a CA of 'one' in the corresponding position, or has a weight greater than 0.0 at the corresponding position. We do not use the most frequent value for each position of the mode, because with our software data set most or all values of the mode would be set to 'zero'.

When calculating the similarity between a mode and an object, pairs of 'zero' attribute values between mode and object are ignored.

Special similarity metrics are used to compute the similarity between a mode and an object.

3.2 Similarity Metrics for Comparison of Objects to Modes

A similarity metric is used to find the closest cluster to an object, by computing the similarity between the cluster's mode and the object. MULICsoft handles indirect links between ASs by associating weights with CAs and incorporating these weights in the clustering process through a special similarity metric that considers CAs and weights. The similarity metric uses the weight vector *w_o*. The function *σ* returns 1 if an object *o* and a mode *μ* have identical values of 'one' at a position, and returns 0 otherwise. The similarity metric amplifies the weights of objects as follows:

$$similarity(o,\mu) = \sum_{i=1}^{m} \frac{x-(4\times w\_o_i)}{5-(4\times w\_o_i)} \times \sigma(o_i,\mu_i)$$

$$\sigma(o_i,\mu_i) = \begin{cases} 1 & (o_i = \mu_i = 1); \\ 0 & otherwise \end{cases}$$

The parameter *x* takes an integer value greater than 5. This similarity metric places more importance on high weights (1.0) than low weights (0.0). The intuition for this formula is that for each pair of CAs with identical values of 'one' between *o* and *μ* the contribution to the similarity result should be at least 1.0, for the lowest weight of 0.0. The maximum contribution, for the highest weight of 1.0, depends on the integer value of *x*. For example, for x=6 the contribution to the similarity result ranges from 1.2 for a low weight of 0.1 to 2.0 for a high weight of 1.0. For x=9 the contribution to the similarity result ranges from 1.8 for a weight of 0.1 to 5.0 for a weight of 1.0.

4 Experimental Evaluation

We use MULICsoft to cluster the AS data that is provided by CAIDA, representing ASs that form the Internet backbones and their links [3]. The clustering can identify groups of ASs that are known to represent prominent groups of the Internet. We use a union of WHOIS databases from ARIN, RIPE, APNIC and LACNIC. We assume two ASs belong to the same organization if, in the WHOIS database, they have the same organization names, or names different only in the last digits, e.g. 'ATT-12' and 'ATT-14' or very similar names, e.g. 'UUNET Canada' and 'UUNET Germany' [4].

The algorithm produces 681 clusters. Each cluster consists of layers ranging from 1 to 101. The size of each cluster ranges from 3 to 43 objects, i.e. ASs. The experiments use a linear increase of *φ* by setting it to an initial value of 1 and increasing it by a constant value *δφ* of 50, after each loop where no object was classified in a cluster of size greater than one. We set *threshold* equal to its default value of the number of attributes *m*, so that no objects are treated as outliers and all ASs are clustered. We do not merge the clusters after the clustering process.

Table 1 shows for some representative clusters, the most prominent AS organizations in each layer. What is most noteworthy about this table is that across the top layer 1 of several clusters the same AS organization such as UUNET or Bell remains prominent, but at lower layers different ASs are more prominent. Furthermore, the ASs in a cluster tend to have common geographical locations, for example, ASs of location 'France' tend to be assigned to separate clusters from ASs of 'UK'.

Table 1. The most prominent AS organizations in the layers of some clusters. *δφ=50* and *x=9*.

cluster	layers		
	1	51	101
1	UUNET	UUNET France	INRIA Fr.
2	UUNET	UUNET UK	BBC UK
3	AT&T Bell	Level 3	Level 3
4	AT&T Bell	AT&T Argentina	Telef.Arg.
5	Sprint	Sprint Canada	Stats. Can.
6	AT&T Bell	AT&T Israel	LuckyNet
7	UUNET	MacLaren Canada	ING Can.

Table 2 shows a comparison of the quality of the results of MULICsoft with the results of k-Modes and AutoClass [5]. For evaluating the quality of results we use HA-values and the metric (2) below. *a* represents the number of pairs of ASs that belong in the same organization and are in the same cluster, *b* represents the number of pairs of ASs that belong in the same organization and are in different clusters, *c* represents the number of pairs of ASs that do not belong in the same organization and are in the same cluster and *d* represents the number of pairs of ASs that do not belong in the same organization and are in different clusters:

$$\text{Quality of results} = \frac{a+d}{a+b+c+d} \qquad (2)$$

Table 2. Quality of results for AS data clustering. *δφ=50* and *x=9*.

Clustering algorithm	Quality of results	# of clusters
MULICsoft	59%	681
k-Modes	40%	700
AutoClass	37%	579

The results indicate that MULICsoft can identify clusters of common AS organizations more accurately than traditional algorithms. A MULICsoft cluster tends to contain many ASs from the same organization.

The reason why a high value of *δφ* of 50 is used is that sufficient ASs should be clustered at each iteration so that the modes of the clusters are given the opportunity to change, as opposed to remaining static.

The reason why x=9 produces the best results is that this similarity metric amplifies significantly the effect of the high weights on the clustering process. We tried setting *x* to even higher values, such as 10, 12 and 15, but the quality no longer improved. Thus, a high value for the parameter *x* improves the results until a specific point. As *x* decreases to 7 and 6, the results are of good quality.

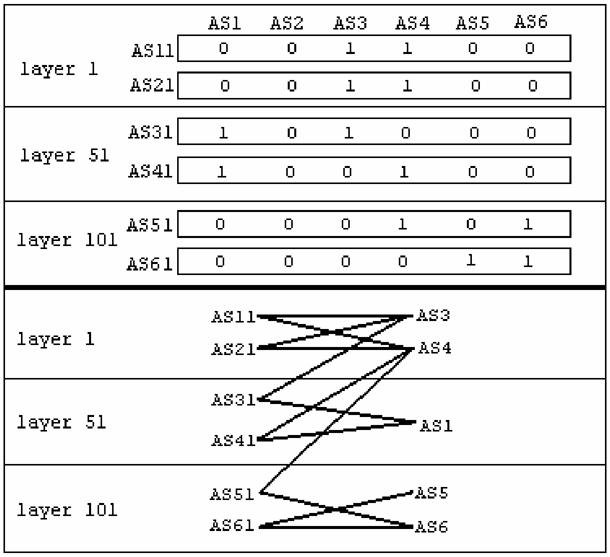
5 Dicussion of AS Graph Properties

We have shown that the clustering results are of a reasonably high quality and this allows drawing conclusions about the structure of the Internet topology that is suggested by the results. Our first observation is that in a cluster there exist one or more ASs with which communication occurs more frequently than would be expected by chance.

Several important properties of the clusters are:

a. The most significant ASs for a cluster often occur in the top layer 1 of a cluster. As layers decrease in a cluster the most significant ASs for the overall cluster do not seem to persist. Different ASs are introduced in bottom layers of clusters. Table 1 shows clusters 1,2,7 with an overall significant AS belonging to 'UUNET'. Layer 1 has 100% of significant ASs belonging to 'UUNET'. Layers 51 and 101 have 65% and 35% of significant ASs belonging to 'UUNET', respectively.
b. The most similar clusters that are merged to one another after clustering tend to contain overlapping, although not totally identical, sets of significant ASs. For a set of 5 merged clusters all of which contain significant ASs belonging to 'UUNET', almost 65% of significant ASs in the clusters belong to 'UUNET' and 35% belong to other domains.
c. Some significant ASs belonging to 'UUNET' are shared between different clusters that are not merged after the process. For example, 'UUNET' is found in approximately 25% of the top layer 1 of all clusters that are not merged. However, 'UUNET' is found as a significant AS in 23% of all clusters.

A cluster of the results can be viewed as a graph, where the nodes correspond to ASs and the edges correspond to links between ASs. A graph conforming to the properties above is referred to as an *outward decreasing density* (ODD) graph. An ODD graph has a set of 'central' nodes with a dense set of undirected edges between them, as well as a set of 'peripheral' nodes. The nodes are organized in 'layers' such that the central nodes are considered to belong to layer 1 and the peripheral nodes to layers 2 to *n*. The layers 2 to *n* with the peripheral nodes have sparsely occurring edges to the central nodes in layer 1.



		AS1	AS2	AS3	AS4	AS5	AS6
layer 1	AS11	0	0	1	1	0	0
	AS21	0	0	1	1	0	0
layer 51	AS31	1	0	1	0	0	0
	AS41	1	0	0	1	0	0
layer 101	AS51	0	0	0	1	0	1
	AS61	0	0	0	0	1	1

Fig. 4. A typical MULICsoft cluster with 3 layers and its representation as an ODD graph. ***density_layer(1)*=4/4, *density_layer(51)*=4/7, *density_layer(101)*=4/11.**

Specifically, let *density_layer(i)* represent the density of the edges occurring between the nodes in layer *i* and the nodes in layers 1 to *i*, defined as:

density_layer(i) = number of edges between nodes in layer i and nodes in layers 1 to i / number of nodes in layers 1 to i.

Then, an ODD graph has the following property:

density_layer(i) > density_layer(i+1)

The MULICsoft tool produces clusters that have an ODD graph structure. Figure 4 illustrates a typical cluster resulting from MULICsoft and its representation as an ODD graph.

We propose modeling the Internet topology at the AS-level as an ODD graph, since this can help the administrators who define the routing policies to save money. For example: *a.* Central AS nodes of communication can be identified. *b.* Peripheral AS nodes with many edges to other clusters can be separated in different clusters, instead of burdening the central cluster node. Graph search algorithms can identify peripheral AS nodes that would better be separated in different clusters.

6 Conclusion and Future Work

MULICsoft offers several advantages for clustering the Internet topology at the AS-level. It focuses from the start on identifying the clusters with 'tightest' connectivity. Each cluster consists of layers of varying connectivity. As the process continues, MULICsoft relaxes its criterion for assigning objects to clusters, forming layers that have less connectivity. MULICsoft does not require the number of clusters to be specified before the process and returns as many clusters as it can find. Similar clusters can be merged after clustering. We are developing an improved method for merging clusters that will hopefully improve the results [2].

Incorporating weights in the clustering process allows using significance metrics to identify the most significant ASs in a cluster [1]. In the future we will apply MULICsoft to more Internet data sets and we will propose graph search algorithms for mining the clustering results to infer relationships.

References:

[1] B. Andreopoulos, A. An and X. Wang. (2005) Clustering Mixed Numerical and Low Quality Categorical Data: Significance Metrics on a Yeast Example. ACM SIGMOD IQIS 2005, Balt, MD.

[2] B. Andreopoulos, A. An and X. Wang. (2004) MULIC: Multi-Layer Increasing Coherence Clustering of Categorical Data Sets. Technical Report # CS-2004-07. Department of Computer Science and Engineering, York University.

[3] Cooperative Association for Internet Data Analysis (CAIDA). http://www.caida.org/

[4] X. Dimitropoulos, D. Krioukov, B. Huffaker, KC Claffy, G. Riley. Inferring AS Relationships: Dead End or Lively Beginning? Efficient Algorithms workshop, 2005.

[5] Huang Z. (1998) Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. Data Mining and Knowledge Discovery 2(3): 283-304.