

CHAPTER ?

CLUSTERING OF WEB DOCUMENTS USING A GRAPH MODEL

Adam Schenker¹, Mark Last², Horst Bunke³, and Abraham Kandel¹

¹*Department of Computer Science and Engineering, University of South Florida,
4202 E. Fowler Ave. ENB 118
Tampa, FL, 33620, USA
E-mail: aschenke, kandel@csee.usf.edu*

²*Department of Information Systems Engineering, Ben-Gurion University of the Negev
Beer-Sheva 84105, Israel
E-mail: mlast@bgumail.bgu.ac.il*

³*Inst. Fur Informatik und angewandte Mathematik,
Department of Computer Science, University of Bern,
Neubrückestrasse 10, CH-3012 Bern, Switzerland
E-mail: bunke@iam.unibe.ch*

In this chapter we enhance the representation of web documents by utilizing graphs instead of vectors. In typical content-based representations of web documents based on the popular vector model, the structural (term adjacency and term location) information cannot be used for clustering. We have created a new framework for extending traditional numerical vector-based clustering algorithms to work with graphs. This approach is demonstrated by an extended version of the classical k-means clustering algorithm which uses the maximum common subgraph distance measure and the concept of median graphs in the place of the usual distance and centroid calculations, respectively. An interesting feature of our approach is that the determination of the maximum common subgraph for measuring graph similarity, which is an NP-Complete problem, becomes polynomial time with our graph representation. By applying this graph-based k-means algorithm to the graph model we demonstrate a superior performance when clustering a collection of web documents.

1. Introduction

In the field of machine learning, clustering has been a useful and active area of research for some time. In clustering, the goal is to separate a given group of data items (the *data set*) into groups (called *clusters*) such that items in the same

cluster are similar to each other and dissimilar to the items in other clusters. Unlike the supervised methods of classification, no labeled examples are provided for training. Clustering of web documents is an important problem for two major reasons. First, clustering a document collection into categories enables it to be more easily browsed and used. Automatic categorization is especially important for the World Wide Web with its huge number of dynamic (time varying) documents and diversity of topics; such features make it extremely difficult to classify pages manually as we might do with small document corpora related to a single field or topic. Second, clustering can improve the performance of search and retrieval on a document collection. Hierarchical clustering methods, for example, are used often for this purpose.¹

When representing documents for clustering, a vector model is typically used.² In this model, each possible term that can appear in a document becomes a feature dimension. The value assigned to each dimension of a document may indicate the number of times the corresponding term appears on it. This model is simple and allows the use of traditional clustering methods that deal with numerical feature vectors in a Euclidean feature space. However, it discards information such as the order in which the terms appear, where in the document the terms appear, how close the terms are to each other, and so forth. By keeping this kind of structural information we could possibly improve the performance of the clustering. The problem is that traditional clustering methods are often restricted to working on purely numeric feature vectors. This comes from the need to compute distances between data items or to calculate some representative of a cluster of items (*i.e.* a centroid or center of a cluster), both of which are easily accomplished in a Euclidean space. Thus either the original data needs to be converted to a vector of numeric values by discarding possibly useful structural information (which is what we are doing when using the vector model to represent documents) or we need to develop new, customized algorithms for the specific representation.

We deal with this problem by introducing an extension of a classical clustering method that allows us to work with graphs as fundamental data structures instead of being limited to vectors of numeric values. Our approach has two main benefits. First, it allows us to keep the inherent structure of the original documents by modeling each document as a graph, rather than having to arrive at numeric feature vectors that contain only term frequencies. Second, we do not need to develop new clustering algorithms completely from scratch: we can apply straightforward extensions to go from classical clustering algorithms that use numerical vectors to those that deal with graphs. In this chapter we will describe a k-means clustering algorithm that utilizes graphs instead of vectors

and illustrate its usefulness by applying it to the problem of clustering a collection of web documents. We will show how web documents can be modeled as graphs and then clustered using our method. Experimental results will be given and compared with previous results reported for the same web data set based on a traditional vector representation.

The chapter is organized as follows. In Sec. 2 we introduce the mathematical foundations we will use for clustering with graphs. In Sec. 3, we extend the classical k-means algorithm to use graphs instead of numerical vectors. In Sec. 4 we will describe a web page data set and its representation by the graph model. In Sec. 5 we present experimental results and a comparison with previous results from clustering the same web documents when using a vector model and classical k-means algorithms. Conclusions are given in Sec. 6.

2. Graphs: Formal Notation

Graphs are a mathematical formalism for dealing with structured entities and systems. In basic terms a graph consists of *vertices* (or *nodes*), which correspond to some objects or components. Graphs also contain *edges*, which indicate the relationships between the vertices. The first definition we have is that of the graph itself. Each data item (document) in the data set we are clustering will be represented by such a graph:

Definition 1. A *graph*^{3,4} G is formally defined by a 4-tuple (quadruple): $G=(V, E, \lambda, \mu)$, where V is a set of vertices (also called *nodes*), $E \subseteq V \times V$ is a set of edges connecting the vertices, $\lambda: V \rightarrow \Lambda_V$ is a function labeling the vertices, and $\mu: E \rightarrow \Lambda_E$ is a function labeling the edges (Λ_V and Λ_E being the sets of labels that can appear on the nodes and edges, respectively).

The next definition we have is that of a *subgraph*. One graph is a subgraph of another graph if it exists as a part of the larger graph:

Definition 2. A graph $G_1 = (V_1, E_1, \lambda_1, \mu_1)$ is a *subgraph*⁵ of a graph $G_2 = (V_2, E_2, \lambda_2, \mu_2)$, denoted $G_1 \subseteq G_2$, if $V_1 \subseteq V_2$, $E_1 \subseteq E_2 \cap (V_1 \times V_1)$, $\lambda_1(x) = \lambda_2(x) \ \forall x \in V_1$, and $\mu_1((x,y)) = \mu_2((x,y)) \ \forall (x,y) \in E_1$.

Next we have the important concept of the *maximum common subgraph*, or *mcs* for short, which is the largest subgraph a pair of graphs have in common:

Definition 3. A graph G is a *maximum common subgraph*⁵ (*mcs*) of graphs G_1 and G_2 , denoted $mcs(G_1, G_2)$, if: (1) $G \subseteq G_1$ (2) $G \subseteq G_2$ and (3) there is no other subgraph $G' \subseteq (G_1 \cap G_2)$ such that $|G'| > |G|$. (Here $|G|$ is intended to convey the “size” of the graph G ; usually it is taken to mean $|V|$, *i.e.* the number of vertices in the graph.)

Using these definitions, a method for computing the distance between two graphs using the maximum common subgraph has been proposed:

$$dist(G_1, G_2) = 1 - \frac{|mcs(G_1, G_2)|}{\max(|G_1|, |G_2|)} \quad (1)$$

where G_1 and G_2 are graphs, $mcs(G_1, G_2)$ is their maximum common subgraph, $\max(\dots)$ is the standard numerical maximum operation, and $|\dots|$ denotes the size of the graph as we mentioned in Definition 3.⁶ This distance measure has four important properties.³ First, it is restricted to producing a number in the interval $[0, 1]$. Second, the distance is 0 only when the two graphs are identical. Third, the distance between two graphs is symmetric. Fourth, it obeys the triangle inequality, which ensures the distance measure behaves in an intuitive way. For example, if we have two dissimilar objects (*i.e.* there is a large distance between them) the triangle inequality implies that a third object which is similar (*i.e.* has a small distance) to one of those objects must be dissimilar to the other.

Methods for computing the *mcs* are presented in the literature.^{7,8} In the general case the computation of *mcs* is NP-Complete, but as we will see later in the chapter, for our graph representation the computation of *mcs* is polynomial time due to the existence of unique node labels in the considered application. Other distance measures which are also based on the maximum common subgraph have been suggested. For example, Wallis *et al.* have introduced a different metric which is not as heavily influenced by the size of the larger graph.⁹ Fernández and Valiente combine the maximum common subgraph and the minimum common supergraph in their proposed distance measure.¹⁰ However, Eq. 1 is the “classic” version and the one we will use in our implementation and experiments. As yet there are no reported findings to indicate which distance measure is most appropriate for various applications, and this is a topic we will investigate in future research. However, the distance measure of Eq. 1 has the advantage that it requires the least number of computations when compared to the other two distance measures we mentioned above.

Finally we need to introduce the concept of the *median of a set of graphs*. We define this formally as:

Definition 4. The *median of a set of n graphs*,¹¹ $S = \{G_1, G_2, \dots, G_n\}$, is a graph G such that G has the smallest average distance to all elements in S :

$$G = \arg \min_{G \in S} \frac{1}{n} \sum_{i=1}^n \text{dist}(G, G_i) \quad (2)$$

Here S is the set of n graphs (and thus $|S|=n$) and G is the median. The median is defined to be a graph in set S . Thus the median of a set of graphs is the graph from that set which has the minimum average distance to all the other graphs in the set. The distance $\text{dist}(\dots)$ is computed from Eq. 1 above. There also exist the concepts of the generalized median and weighted mean, where we don't require that G be a member of S .^{11,12} However, the related computational procedures are much more demanding and we do not consider them in the context of this chapter. Note that the implementation of Eq. 2 requires only $O(n^2)$ graph distance computations and then finding the minimum among those distances.

3. The Extended k-Means Clustering Algorithm

With our formal notation now in hand, we are ready to describe our framework for extending classical clustering methods which rely on Euclidean distance. The extension is surprisingly simple. First, any distance calculations between data items is accomplished with a graph-theoretical distance measure, such as that of Eq. 1. Second, since it is necessary to compute the distance between data items and cluster centers, it follows that the cluster centers (centroids) must also be graphs if we are to use a method such as that in Eq. 1. Therefore, we compute the representative "centroid" of a cluster as the median graph of the set of graphs in that cluster (Eq. 2). We will now show a specific example of this extension to illustrate the technique.

To avoid any confusion, we should briefly emphasize here the difference between our method and the family of "traditional" graph-theoretic clustering algorithms.^{1,13} In the typical graph clustering case, all the data to be clustered is represented as a single graph where the vertices are the data items and the edge weights indicate the similarity between items. This graph is then partitioned to create groups of connected components (clusters). In our method, each data item to be clustered is represented by a graph. These graphs are then clustered using some clustering algorithm (in this case, k-means) utilizing the distance and

median computations previously defined in lieu of the traditional Euclidean distance and centroid calculations.

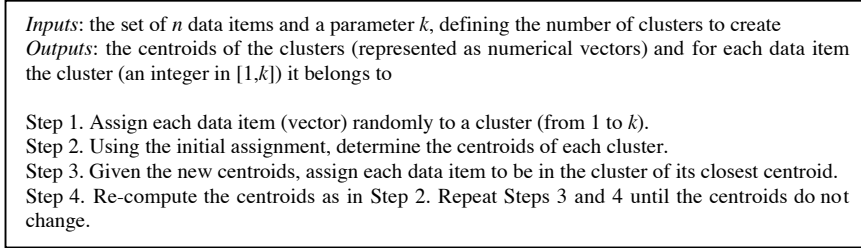


Fig. 1. The basic k-means clustering algorithm.

The k-means clustering algorithm is a simple and straightforward method for clustering data.¹⁴ The basic algorithm is given in Fig. 1. This method is applicable to purely numerical data when using Euclidean distance and centroid calculations. The usual paradigm is to represent each data item, which consists of m numeric values, as a vector in the space \mathbb{R}^m . In this case the distances between two data items are computed using the Euclidean distance in m dimensions and the centroids are computed to be the mean of the data in the cluster. However, now that we have a distance measure for graphs (Eq. 1) and a method of determining a representative of a set of graphs (the median, Eq. 2) we can apply the same method to data sets whose elements are graphs rather than vectors. The k-means algorithm extended to operate on graphs is given in Fig. 2.

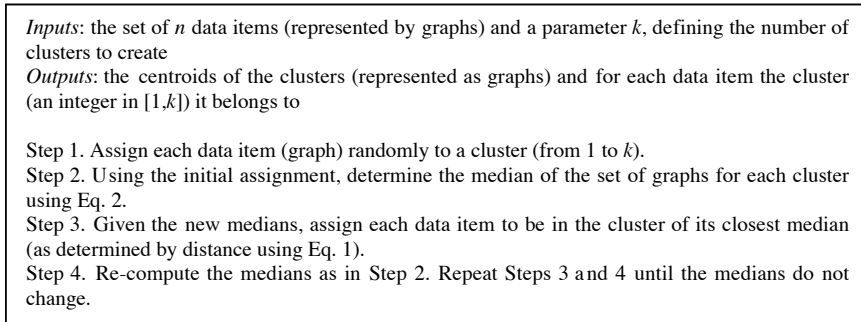


Fig 2. The k-means algorithm for using graphs.

4. Clustering of Web Documents using the Graph Model

In order to demonstrate the performance and possible benefits of the graph-based approach, we have applied the extended k-means algorithm to the clustering of a collection of web documents. Some research into performing clustering of web pages is reported in the literature.^{15–18} Similarity of web pages represented by graphs has been discussed in a recent work by Lopresti and Wilfong.¹⁹ Their approach differs from ours in that they extract numerical features from the graphs (such as node degree and vertex frequency) to determine page similarity rather than comparing the actual graphs; they also use a graph representation based on the syntactical structure of the HTML parse tree rather than the textual content of the pages. However, the work we are most interested in for evaluation purposes is that of Strehl *et al.*²⁰ In that paper, the authors compared the performance of different clustering methods on web page data sets. This paper is especially important to the current work, since it presents baseline results for a variety of standard clustering methods including the classical k-means using different similarity measures.

The data set we will be using is the Yahoo “K” series,^{*} which was one of the data sets used by Strehl *et al.* in their experiments.²⁰ This data set consists of 2,340 Yahoo news pages downloaded from www.yahoo.com in their original HTML format. Each page is assigned to one of 20 categories based on its content, such as “technology”, “sports” or “health”. Although a pre-processed version of the data set is also available in the form of a term–document matrix and a list of stemmed words, we are using the original documents in order to capture their inherent structural information using graphs. We represent each web document as a graph using the following method:

- Each term (word) appearing in the web document, except for stop words (see below), becomes a vertex in the graph representing that document. This is accomplished by labeling each node (using the node labeling function \square , see Definition 1) with the term it represents. Note that we create only a single vertex for each word even if a word appears more than once in the text. Thus each vertex in the graph represents a unique word and is labeled with a unique term not used to label any other node.
- If word a immediately precedes word b somewhere in a “section” s of the web document (see below), then there is a directed edge from the vertex corresponding to a to the vertex corresponding to b with an edge

^{*} This data set is available at: <ftp://ftp.cs.umn.edu/dept/users/boley/PDDPdata>

label s . We take into account certain punctuation (such as a period) and do not create an edge when these are present between two words.

- We have defined three “sections” for the web pages. First, we have the section *title*, which contains the text in the document’s TITLE tag and any provided keywords (meta-data). Second we have the section *link*, which is text appearing in clickable links on the page. Third we have the section *text*, which comprises any of the readable text in the document (this includes link text but not title and keyword text).
- We perform removal of stop words, such as “the”, “and”, “of”, etc. which are generally not useful in conveying information by removing the corresponding nodes and their incident edges. We also perform simple stemming by checking for common alternate forms of words, such as the plural form.
- We remove the most infrequently occurring words on each page, leaving at most m nodes per graph (m being a user provided parameter). This is similar to a dimensionality reduction process for vector representations.

This form of knowledge representation is a type of *semantic network*, where nodes in the graph are objects and labeled edges indicate the relationships between objects.²¹ The conceptual graph is a type of semantic network sometimes used in information retrieval.²² With conceptual graphs, terms or phrases related to documents appear as nodes. The types of relations (edge labels) include synonym, part–whole, antonym, and so forth. Conceptual graphs are used to indicate meaning-oriented relationships between concepts, whereas our method indicates structural relationships that exist between terms in a web document.

We give a simple example of our graph representation of a web document in Fig. 3. The ovals indicate nodes and their corresponding term labels. The edges are labeled according to title (TI), link (L), or text (TX). The document represented by the example has the title “YAHOO NEWS”, a link whose text reads “MORE NEWS”, and text containing “REUTERS NEWS SERVICE REPORTS”. This novel method of document representation is somewhat similar to that of *directed acyclic word graphs*²³ (or *DAWGs*); however, our nodes represent words rather than letters, our model allows for cycles in the graphs, and the edges are labeled.

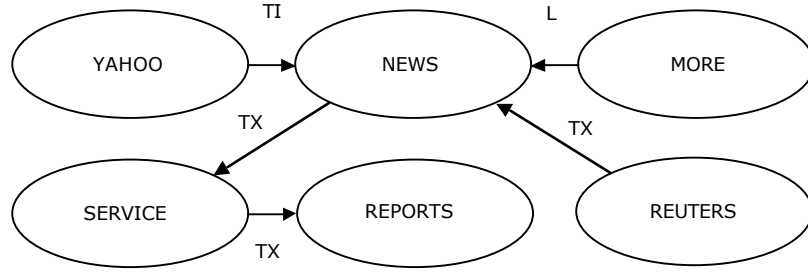


Fig. 3. An example graph representation of a web document.

When determining the size of a graph representing a web document (Definition 3) we use the following method:

Definition 5. The size of a graph $G=(V, E, \square, \sqsupset)$, denoted $|G|$, is defined as $|G|=|V|+|E|$.

Recall that the typical definition is simply $|G|=|V|$. However, for this application it is detrimental to ignore the contribution of the edges, which indicate the number of phrases identified in the text. Further, it is possible to have more than one edge between two nodes since we are labeling the edges according to the document section in which the terms are adjacent separately.

Before moving on to the experiments, we mention an interesting feature this model of representing documents has on the time complexity of determining the distance between two graphs (Eq. 1). In the distance calculation we are using the maximum common subgraph; the determination of this in the general case is known to be an NP-Complete problem.²⁴ However, our graphs for this application have the following property:

$$\square x,y \square V, \square(x)=\square(y) \text{ if and only if } x=y \quad (3)$$

In other words, each node in a graph has a unique label assigned to it, namely the term it represents. No two nodes in a graph will have the same label. Thus the maximum common subgraph $G_m=(V_m, E_m, \square_m, \sqsupset_m)$ of a pair of graphs G_1 and G_2 can be created using the following method:

- Step 1. Create the set of vertices by $V_m=\{x|x \square V_1 \text{ and } x \square V_2 \text{ and } \square_1(x)=\square_2(x)\}$
- Step 2. Create the set of edges by $E_m=\{(x,y)|x,y \square V_m \text{ and } \square_1((x,y))=\square_2((x,y))\}$

The first step states the set of vertices in the maximum common subgraph is just the intersection of the set of terms of both graphs. Each term in the intersection becomes a node in the maximum common subgraph. The second step creates the edges by examining the set of nodes created from the previous step. We examine all pairs of nodes in the set; if both nodes contain an edge between them in both original graphs and share a common label, then we add the edge to the maximum common subgraph. Note that this is different from the concept of induced maximum common subgraph, where nodes are added only if they are connected by an edge in both original graphs. If there is a common subset of nodes but different edge configurations in the original graphs, we still add the nodes using our method. We also note that in document clustering, the nodes, which represent terms, are much more important than the edges, which only indicate the relationships between the terms (*i.e.*, *followed by*). We see that the complexity of this method is $O(|V_1| \cdot |V_2|)$ for the first step and $O(|V_{mcs}|^2)$ for the second step. Thus it is $O(|V_1| \cdot |V_2| + |V_{mcs}|^2) \approx O(|V|^2 + |V_{mcs}|^2) = O(|V|^2)$ overall if we substitute $V = \max(|V_1|, |V_2|)$.

5. Experimental Results

In order to compare clustering methods with differing distance measures, Strehl *et al.* proposed the use of an information-theoretic measure of clustering performance.²⁰ This measurement is given as:

$$\square^M = \frac{1}{n} \prod_{l=1}^k \prod_{h=1}^g n_l^{(h)} \frac{\log \frac{n_l^{(h)} \cdot n}{n_l^{(h)} \cdot n}}{\log(k \cdot g)} \quad (4)$$

where n is the number of data items, k is the desired number of clusters, g is the actual number of categories, and $n_l^{(j)}$ is the number of items in cluster l classified to be category j . The above measure is, in fact, *mutual information*²⁵ normalized by the sum of its maximum values ($\log k$ and $\log g$) and it represents the overall degree of agreement between the clustering and the categorization. In an attempt to adhere to the methodology of the original experiments, which used the vector model approach, we have selected a sample of 800 documents from the total collection of 2,340 and have fixed the desired number of clusters to be $k=40$ (two times the number of categories), which is the same number of clusters used in the original experiment. Strehl *et al.* used this number of clusters “since this seemed to be the more natural number of clusters as indicated by preliminary

runs and visualisation.”²⁰ The results for our method using different numbers of maximum nodes per graph and the original results from Strehl *et al.* for vector-based k-means and a random baseline assignment are given in Table 1 (higher mutual information is better); results from our method are shown in bold. Each row gives the average of 10 experiments using the same 800 item data sample. The variation in results between runs comes from the random initialization in the first step of the k-means algorithm. We used t-tests to evaluate the statistical significance of our results as compared with the best reported vector-based k-means method (Extended Jaccard Similarity). Confidences less than 0.950 are marked with a “-”. The same performance data is plotted graphically in Fig. 4. In Fig. 5 we show the execution times for performing a single clustering of the document collection when using 5, 50, 100, and 150 nodes per graph. These results were obtained on a 733 MHz single processor Power Macintosh G4 with 384 megabytes of physical memory running Mac OS X. The clustering took 7.13 minutes at 5 nodes per graph and 288.18 minutes for 150 nodes per graph. Unfortunately, no execution time data is available for comparison from the original experiments in Strehl *et al.*

Table 1. Results of our experiments compared with results from Strehl *et al.*²⁰

Method	Max. Nodes/Graph	\square^M (average)	t-test
Graphs	150	0.2218	1.000
Graphs	120	0.2142	1.000
Graphs	90	0.2074	0.999
Graphs	75	0.2045	0.999
Graphs	60	0.1865	-
Extended Jaccard Similarity	-	0.184	-
Pearson Correlation	-	0.178	-
Cosine Measure	-	0.178	-
Graphs	45	0.1758	-
Graphs	30	0.1617	-
Graphs	15	0.1540	-
Graphs	5	0.1326	-
Random (baseline)	-	0.066	-
Euclidean	-	0.046	-

From Fig. 4 we see that the mutual information generally tends to increase as we allow larger and larger graphs. This makes sense since the larger graphs incorporate more information. On the figure we indicated the values of mutual information from the original experiments for three out of the five methods from Table 1. Euclidean is the classical k-means with a Euclidean distance measure. Random baseline is simply a random assignment of data to clusters; it is used to

provide a baseline for comparison. We would expect any algorithm to perform better than Random, but we see the Euclidean k-means did not. Finally, Jaccard is k-means using the Extended Jaccard Similarity.^{2,20} It was the best performing of all the k-means methods reported in the original experiment so we have omitted cosine similarity and Pearson correlation on the chart for clarity.

It is not a surprising result to see Euclidean distance perform poorly when using the vector model for representing documents, as it does not have the property of vector length invariance. Because of this, documents with similar term frequency proportions but differences in overall total frequency have large distances between them even though they are supposed to be considered similar. For example, if we were interested in the topic “data mining”, a document where the terms “data” and “mining” each appeared 10 times and a document where both terms each appeared 1,000 times are considered to be identical when we have the length invariance property (*i.e.* their distance is 0). It is only the relative proportion between the terms that is of interest when determining the document’s content, since there are often large variances in total term frequency even for documents related to the same topic. Here both documents contain an equal proportion of the terms “data” and “mining”. If the term “mining” occurred much more frequently than “data”, we would expect the document to be related to a different topic (*e.g.*, “gold mining”). Under Euclidean distance these two documents would have a large distance (*i.e.* be considered dissimilar) due to the fact that the difference in total frequency (10 vs. 1,000) is large. This is why distance measures with the length invariance property (such as the cosine measure, which measures the cosine of the angle between two feature vectors) are often used in these types of applications in lieu of standard Euclidean distance.

We see that even with only 5 nodes per graph our method outperforms both Euclidean k-means and the random baseline; as we increased the number of nodes per graph the performance approaches that of the other k-means methods until it exceeded even the best k-means method reported at 75 nodes per graph or more. For comparison, the original experiment used a term–document matrix where each vector had 2,903 dimensions. We note a general increasing trend in performance as we allow for larger graphs, which would be consistent with the increase in information that occurs as we introduce new terms (nodes) and phrases (edges) in the graphs. However, the performance improvement is not always strictly proportional with the increase in graph size. For example, the improvement from 60 to 75 is greater than the improvement from 75 to 90 even though we are adding 15 new nodes in each case. This may be due to the fact that the extra nodes added when we increase the graph size, while they are

frequently occurring terms, may not always provide information that is useful for discriminating between the documents and in actuality may hinder performance by introducing extraneous data. A future improvement may be to find better methods of selecting the nodes to be used in each graph rather than relying strictly on term frequency.

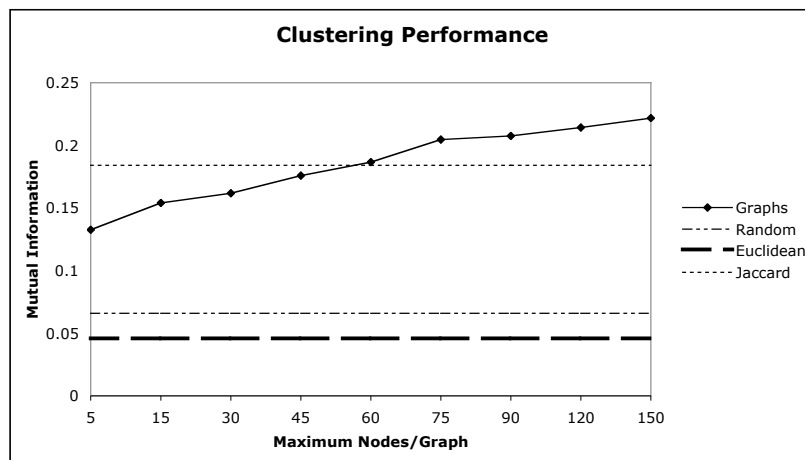


Fig. 4. Mutual Information as a function of the maximum number of vertices per graph.

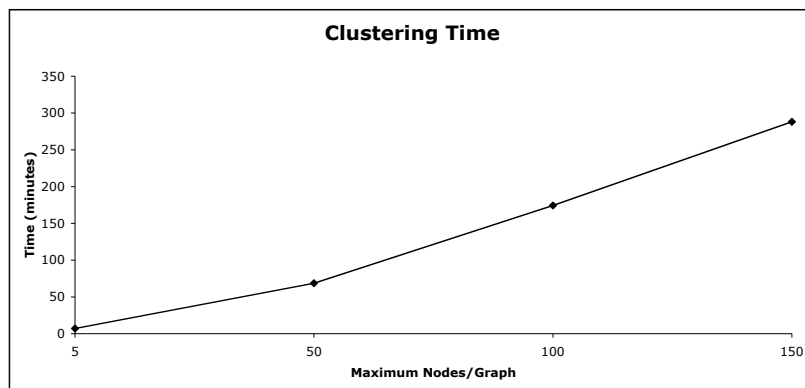


Fig. 5. Clustering time as a function of the maximum number of vertices per graph.

6. Conclusions

In this chapter we showed how it is possible to cluster web documents using a graph representation rather than a vector representation. A graph representation allows us to retain structural information such as where terms are located in a document and the order in which terms appear — information which is mostly discarded when using the typical vector model approach. Given a graph model of web documents, we can apply traditional clustering techniques such as k-means by performing an extension from Euclidean distance and centroid calculations to graph distance and median graphs, respectively. To demonstrate the performance of the extended k-means method with our graph representation of web documents, we performed experiments on a web document collection and compared with previous results of clustering using k-means when utilizing a vector model for the same documents. We have discovered the following from our experiments:

- Our method outperformed the baseline random assignment method and the vector-based k-means method using Euclidean distance, even in the case of maximum dimensionality reduction using 5 nodes per graph.
- As the maximum number of nodes allowed per graph became larger, the performance of our method generally increased. This reflects an increase in the amount of information in the graphs as we add nodes and edges.
- Our method outperformed all the k-means clustering methods (Euclidean distance, cosine measure, Pearson correlation, and Jaccard similarity) described in Strehl *et al.*²⁰ when we allowed 75 nodes per graph or more. We believe this reflects the information retained by the graph representation which is not present when using the vector model approach.

We have many avenues to explore for future work. We have shown one graph distance measure here, but others have been proposed. We will perform experiments with other graph distance measures and compare clustering performance. We can also attempt to create a more elaborate graph representation for web documents. For example, we can recognize more document sections, connect words that appear in the same sentence or paragraph, and so on. Such representations could capture even more information, possibly leading to better performance. It is also possible to apply our technique to structured text, such as XML documents and software

programs, and we intend to investigate clustering collections of source code using our method. We also wish to extend other clustering algorithms to work with graphs, such as hierarchical agglomerative clustering and fuzzy c-means.

Acknowledgments

This work was supported in part by the National Institute for Systems Test and Productivity at the University of South Florida under U.S. Space and Naval Warfare Systems Command grant number N00039-01-1-2248.

References

1. A. K. Jain, M. N. Murty and P. J. Flynn, "Data clustering: a review", *ACM Computing Surveys* Vol. 31, No. 3, 1999, pp. 264–323.
2. G. Salton, *Automatic Text Processing: the Transformation, Analysis, and Retrieval of Information by Computer*, Addison–Wesley, Reading, 1989.
3. H. Bunke, X. Jiang, and A. Kandel, "On the minimum common supergraph of two graphs", *Computing*, Vol. 65, 2000, pp. 13–25.
4. J. T. L. Wang, K. Zhang, and G.–W. Chirn, "Algorithms for approximate graph matching", *Information Sciences*, Vol. 82, 1995, pp. 45–74.
5. H. Bunke, "On a relation between graph edit distance and maximum common subgraph", *Pattern Recognition Letters*, Vol. 18, 1997, pp. 689–694.
6. H. Bunke and K. Shearer, "A graph distance metric based on the maximal common subgraph", *Pattern Recognition Letters*, Vol. 19, 1998, pp. 255–259.
7. G. Levi, "A note on the derivation of maximal common subgraphs of two directed or undirected graphs", *Calcolo*, Vol. 9, 1972, pp. 341–354.
8. J. J. McGregor, "Backtrack search algorithms and the maximal common subgraph problem", *Software Practice and Experience*, Vol. 12, 1982, pp. 23–34.
9. W. D. Wallis, P. Shoubridge, M. Kraetz, and D. Ray, "Graph distances using graph union", *Pattern Recognition Letters*, Vol. 22, 2001, pp. 701–704.
10. M.–L. Fernández and G. Valiente, "A graph distance metric combining maximum common subgraph and minimum common supergraph", *Pattern Recognition Letters*, Vol. 22, 2001, pp. 753–758.
11. H. Bunke, S. Günter, and X. Jiang, "Towards bridging the gap between statistical and structural pattern recognition: two new concepts in graph matching" in *Advances in Pattern Recognition — ICAPR 2001*, eds. S. Singh, N. Murshed, and W. Kropatsch, Springer–Verlag, 2001.
12. H. Bunke and A. Kandel, "Mean and maximum common subgraph of two graphs", *Pattern Recognition Letters*, Vol. 21, 2000, pp. 163–168.
13. J. G. Augustson and J. Minker, "An analysis of some graph theoretical cluster techniques", *Journal of the Association of Computing Machinery*, Vol. 17, No. 4, 1970, pp. 571–588.
14. T. M. Mitchell, *Machine Learning*, McGraw–Hill, Boston, 1997.
15. D. Boley, M. Gini, R. Gross, E. H. Han, K. Hastings, G. Karypis, B. Mobasher, and J. Moore, "Partitioning-based clustering for web document categorization", *Decision Support Systems*, Vol. 27, 1999, pp. 329–341.

16. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "Extracting large-scale knowledge bases from the web", *Proceedings of the 25th International Conference on Very Large Databases*, 1999, pp. 639–649.
17. S. A. Macskassy, A. Banerjee, B. D. Davison, and H. Hirsh, "Human performance on clustering web pages: a preliminary study", *Proceedings of The 4th International Conference on Knowledge Discovery and Data Mining*, 1998, pp. 264–268.
18. O. Zamir and O. Etzioni, "Web document clustering: a feasibility demonstration", *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998, pp. 46–54.
19. D. Lopresti and G. Wilfong, "Applications of graph probing to web document analysis", *Proceedings of the 1st International Workshop on Web Document Analysis (WDA2001)*, 2001, pp. 51–54.
20. A. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering", *AAAI-2000: Workshop of Artificial Intelligence for Web Search*, 2000, pp. 58–64.
21. S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice–Hall, Upper Saddle River, 1995.
22. X. Lu, "Document retrieval: a structural approach", *Information Processing and Management*, Vol. 26, No. 2, 1990, pp. 209–218.
23. M. Crochemore and R. V erin, "Direct construction of compact directed acyclic word graphs" in *CPM97*, eds. A. Apostolico and J. Hein, Springer–Verlag, 1997.
24. B. T. Messmer and H. Bunke, "A new algorithm for error-tolerant subgraph isomorphism detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 5, 1998, pp. 493–504.
25. T. M. Cover and J.A. Thomas, *Elements of Information Theory*, Wiley, 1991.