Hypergraph Based Clustering in High-Dimensional Data Sets: A Summary of Results *

Eui-Hong (Sam) Han George Karypis Vipin Kumar
Bamshad Mobasher

Department of Computer Science and Engineering/Army HPC Research Center
University of Minnesota
{han,karypis,kumar,mobasher}@cs.umn.edu

Abstract

Clustering of data in a large dimension space is of a great interest in many data mining applications. In this paper, we propose a method for clustering of data in a high dimensional space based on a hypergraph model. In this method, the relationship present in the original data in high dimensional space are mapped into a hypergraph. A hyperedge represents a relationship (affinity) among subsets of data and the weight of the hyperedge reflects the strength of this affinity. A hypergraph partitioning algorithm is used to find a partitioning of the vertices such that the corresponding data items in each partition are highly related and the weight of the hyperedges cut by the partitioning is minimized. We present results of experiments on two different data sets: S&P500 stock data for the period of 1994-1996 and protein coding data. These experiments demonstrate that our approach is applicable and effective in high dimensional datasets.

1 Introduction

Clustering in data mining is a discovery process that groups a set of data such that the intracluster similarity is maximized and the intercluster similarity is minimized [CHY96]. These discovered clusters are used to explain the characteristics of the data distribution. For example, in many business applications, clustering can be used to characterize different customer groups and allow businesses to offer customized solutions, or to predict customer buying patterns based on the profiles of the cluster to which they belong.

Given a set of n data items with m variables, traditional clustering techniques [CS96, JD88] group the data based on some measure of similarity or distance between data points. Most of these clustering algorithms are

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

*This work was supported by NSF ASC-9634719, by Army Research Office contract DA/DAAH04-95-1-0538, by Army High Performance Computing Research Center cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred. Additional support was provided by the IBM Partnership Award, and by the IBM SUR equipment grant. Access to computing facilities was provided by AHPCRC, Minnesota Supercomputer Institute. See http://www.cs.umn.edu/~han for other related papers.

able to effectively cluster data when the dimensionality of the space (i.e., the number of variables) is relatively small. However, these schemes fail to produce meaningful clusters, if the number of variables is large.

Clustering of large dimensional data sets is of a great interest in many data mining applications. For example, in market basket analysis, a typical store sells thousands of different items to thousands of different customers. If we can cluster the items sold together, we can then use this knowledge to perform effective shelf-space organization as well as target sales promotions. Clustering of items from the sales transactions requires handling thousands of variables corresponding to customer transactions. Finding clusters of customers based on the sales transactions also presents a similar problem. In this case, the items sold in the store correspond to the variables in the clustering problem.

In this paper, we propose a method for clustering of data in a high dimensional space based on a hypergraph model. In a hypergraph model, each data item is represented as a vertex and related data items are connected with weighted hyperedges. A hyperedge represents a relationship (affinity) among subsets of data and the weight of the hyperedge reflects the strength of this affinity. Now a hypergraph partitioning algorithm is used to find a partitioning of the vertices such that the corresponding data items in each partition are highly related and the weight of the hyperedges cut by the partitioning is minimized.

To test the applicability and robustness of our scheme, we evaluated it on a wide variety of data sets [HKKM97a, MHB+97, HBG+98, HKKM97b]. We present a summary of results on two different data sets: S&P500 stock data for the period of 1994-1996 and protein coding data. These experiments demonstrate that our approach is applicable and effective in a wide range of domains. More specifically, our approach performed much better than traditional schemes for dimensionality reduction [Jac91, BDO95] in terms of quality of clusters and runtime.

The rest of this paper is organized as follows. Section 2 presents our clustering method based on hypergraph models. Section 3 presents the experimental results. Section 4 contains conclusion and directions for future work.

2 Hypergraph-Based Clustering

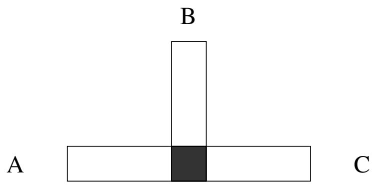
Our algorithm for clustering related items consists of the following two steps. During the first step, a weighted hypergraph H is constructed to represent the relations among different items, and during the second step, a hypergraph partitioning algorithm is used to find k partitions such that the items in each partition are highly related. In our current implementation, we use frequent item sets found by the association rule algorithm [AS94] as hyperedges.

2.1 Hypergraph Modeling

A hypergraph [Ber76] $H = (V, E)$ consists of a set of vertices (V) and a set of hyperedges (E). A hypergraph is an extension of a graph in the sense that each hyperedge can connect more than two vertices. In our model, the set of vertices V corresponds to the set of data items being clustered, and each hyperedge $e \in E$ corresponds to a set of related items. A key problem in modeling of data items as hypergraph is the determination of related items that can be grouped as hyperedges and determining weights of each such hyperedge.

The frequent item sets computed by an association rule algorithm such as Apriori are excellent candidates to find such related items. Note that these algorithms only find frequent item sets that have support greater than a specified threshold. The value of this threshold may have to be determined in a domain specific manner. The frequent item sets capture relationship of items of size greater than or equal to 2. Note that distance based relationships can only capture relationship among pairs of data points whereas the frequent items sets can capture relationship among larger sets of data points. This added modeling power is nicely captured in our hypergraph model.

Assignment of weights to the resulting hyperedges is more tricky. One obvious possibility is to use the support of each frequent item set as the weight of the corresponding hyperedge. Another possibility is to make the weight



support{A} = 0.25
support{B} = 0.25
support{C} = 0.25

support{A,B} = 0.1	P(A|B) = P(B|A) = 0.4
support{B,C} = 0.1	P(B|C) = P(C|B) = 0.4
support{A,C} = 0.1	P(A|C) = P(C|A) = 0.4
support{A,B,C} = 0.1	P(A|BC) = P(B|AC) = P(C|AB) = 1.0

Figure 1: Illustration of the power of confidence in capturing relationships among items.

as a function of the confidence of the underlying association rules. For size two hyperedges, both support and confidence provide similar information. In fact, if two items A and B are present in equal number of transactions (i.e., if the support of item set $\{$A$\}$ and item set $\{$B$\}$ are the same), then there is a direct correspondence between the support and the confidence of the rules between these two items (i.e., greater the support for $\{$A, B$\}$, more confidence for rules "$\{$A$\} \Longrightarrow \{$B$\}$" and "$\{$A$\} \Longrightarrow \{$B$\}$"). However, support carries much less meaning for hyperedges of size greater than two, as in general, the support of a large hyperedge will be much smaller than the support of smaller hyperedges. Furthermore, for these larger item sets, confidence of the underlying association rules can capture correlation among data items that is not captured by support. For example, consider three items A, B and C and their relationship as represented in Figure 1. The support of each of the pairs $\{$A, B$\}$, $\{$B, C$\}$ and $\{$A, C$\}$ is 0.1, and the support of $\{$A, B, C$\}$ is also 0.1. However, the three-way relationship among A, B and C is much stronger than those among pairs, as $P(C|AB)$, $P(B|AC)$ and $P(A|BC)$ are all 1.0. These conditional probabilities are captured in the confidence of corresponding association rules. This example also illustrates that relationships among item sets of size greater than 2 cannot always be captured by the pairwise relationships of its subsets irrespective of whether the relationship is modeled by the support or the confidence of the individual rules. Hence, a hypergraph is a more expressive model than a graph for this problem.

Another, more natural, possibility is to define weight as a function of the support and confidence of the rules that are made of a group of items in a frequent item set. Other options include correlation [BMS97], distance or similarity measure.

In our current implementation of the model, each frequent item-set is represented by a hyperedge $e \in E$ whose weight is equal to the average confidence of the association rules, called *essential* rules, that have all the items of the edge and has a singleton right hand side. We call them are *essential* rules, as they capture information unique to the given frequent item set. Any rule that has only a subset of all the items in the rule is already included in the rules of subset of this frequent item set. Furthermore, all the rules that have more than 1 item on the right hand size are also covered by the subset of the frequent item set. For example, if $\{$A,B,C$\}$ is a frequent item-set, then the hypergraph contains a hyperedge that connects A, B, and C. Consider a rule $\{$A$\}\Longrightarrow\{$B,C$\}$. Interpreted as an implication rule, this information is captured by $\{$A$\}\Longrightarrow\{$B$\}$ and $\{$A$\}\Longrightarrow\{$C$\}$. Consider the following essential rules (with confidences noted on the arrows) for the item set $\{$A,B,C$\}$: $\{$A,B$\}\overset{0.4}{\Longrightarrow}\{C\}$, $\{$A,C$\}\overset{0.6}{\Longrightarrow}\{B\}$, and $\{$B,C$\}\overset{0.8}{\Longrightarrow}\{A\}$. Then we assign weight of 0.6 ($\frac{0.4+0.6+0.8}{3} = 0.6$) to the hyperedge connecting A,B, and C. We will refer to this hypergraph as the association-rule hypergraph.

2.2 Finding Clusters of Items

Note that the frequent items sets already represent relationship among the items of a transaction. But these relationships are "fine grain". For example, consider the following three frequent item sets found from a database of stock transactions:

$$\{\text{Texas Inst}\uparrow, \text{Intel}\uparrow, \text{Micron Tech}\uparrow\}$$

$$\{\text{National Semiconduct}\uparrow, \text{Intel}\uparrow\}$$
$$\{\text{National Semiconduct}\uparrow, \text{Micron Tech}\uparrow\}$$

These item sets indicate that on many different days, stocks of Texas Instrument, Intel and Micron Technology moved up together, and on many days, stocks of Intel and National Semiconductor moved up together, etc. From these, it appears that Texas Instrument, Intel, Micron Technology and National Semiconductor are somehow related. But a frequent item set of these four stocks may have small support and may not be captured by the association rule computation algorithm.

However the hypergraph representation can be used to cluster together relatively large groups of related items by partitioning it into highly connected partitions. One way of achieving this is to use a hypergraph partitioning algorithm that partitions the hypergraph into two parts such that the weight of the hyperedges that are cut by the partitioning is minimized. Note that by minimizing the hyperedge-cut we essentially minimize the relations that are violated by splitting the items into two groups. Now each of these two parts can be further bisected recursively, until each partition is highly connected.

HMETIS [KAKS97, Kar98] is a multi-level hypergraph partitioning algorithm that has been shown to produce high quality bi-sections on a wide range of problems arising in scientific and VLSI applications. HMETIS minimizes the weighted hyperedge cut, and thus tends to create partitions in which the connectivity among the vertices in each partition is high, resulting in good clusters.

Once, the overall hypergraph has been partitioned into k parts, we eliminate bad clusters using the following cluster fitness criterion. Let e be a set of vertices representing a hyperedge and C be a set of vertices representing a partition. The fitness function that measures the goodness of partition C is defined as follow:

$$fitness(C) = \frac{\sum_{e \subseteq C} Weight(e)}{\sum_{|e \cap C| > 0} Weight(e)}$$

The fitness function measures the ratio of weights of edges that are within the partition and weights of edges involving any vertex of this partition. The high fitness value suggests that the partition has more weights for the edges connecting vertices within the partition. The partitions with fitness measure greater than a given threshold value are considered to be good clusters and retained as clusters found by our approach. In our experiments, we set this fitness threshold to 0.1. Note that this fitness criterion can easily be incorporated into a partitioning algorithm such that each partition is further bisected only if the fitness of the partition is below the given threshold value. Then all the partitions found can be considered good partitions.

Once good partitions are found, each good partition is examined to filter out vertices that are not highly connected to the rest of the vertices of the partition. The connectivity function of vertex v in C is defined as follow:

$$connectivity(v, C) = \frac{|\{e | e \subseteq C, v \in e\}|}{|\{e | e \subseteq C\}|}$$

The connectivity measures the percentage of edges that each vertex is associated with. High connectivity value suggests that the vertex has many edges connecting good proportion of the vertices in the partition. The vertices with connectivity measure greater than a give threshold value are considered to belong to the partition, and the remaining vertices are dropped from the partition. In our experiments, we set this connectivity threshold to 0.1.

3 Experimental Results

We tested the ability of our item-clustering algorithm to find groups of related items, on data-sets from many application areas. The results of these experiments are described in the following subsections. In all of our experiments, we used a locally implemented version of *Apriori* algorithm [AS94] to find the association rules and construct the association-rule hypergraph.

Discovered Clusters	Industry Group
APPLIED MATL↓, BAY NETWORK↓, 3 COM↓, CABLETRON SYS↓, CISCO↓, DSC COMM↓, HP↓, INTEL↓, LSI LOGIC↓, MICRON TECH↓, NATL SEMICONDUCT↓, ORACLE↓, SGI↓, SUN↓, TELLABS INC↓, TEXAS INST↓	Technology 1
APPLE COMP↓, AUTODESK↓, ADV MICRO DEVICE↓, ANDREW CORP↓, COMPUTER ASSOC↓, CIRC CITY STORES↓, COMPAQ↓, DEC↓, EMC CORP↓, GEN INSTRUMENT↓, MOTOROLA↓, MICROSOFT↓, SCIENTIFIC ATL↓	Technology 2
FANNIE MAE↓, FED HOME LOAN↓, MBNA CORP↓, MORGAN STANLEY↓	Financial
BAKER HUGHES↑, DRESSER INDS↑, HALLIBURTON HLD↑, LOUISIANA LAND↑, PHILLIPS PETRO↑, SCHLUMBERGER↑, UNOCAL↑	Oil
BARRICK GOLD↑, ECHO BAY MINES↑, HOMESTAKE MINING↑, NEWMONT MINING↑, PLACER DOME INC↑	Gold
ALCAN ALUMINUM↓, ASARCO INC↓, CYPRUS AMAX MIN↓, INLAND STEEL INC↓, INCO LTD↓, NUCOR CORP↓, PRAXAIR INC↓, REYNOLDS METALS↓, STONE CONTAINER↓, USX US STEEL↓	Metal

Table 1: Clustering of S&P 500 Stock Data

3.1 S&P 500 Stock Data

Our first data-set consists of the daily price movement of the stocks that belong to the S&P500 index. It is well known in the financial community, that stocks belonging to the same industry group tend to trade similarly. For example, a group of stocks from a particular industry tend to move up or down together depending on the market's belief about the health of this industry group. For this reason, we used this data set to verify the ability of our clustering algorithm to correctly cluster the various stocks according to their industry group.

The data set consists of a binary table of size 1000×716. Each row of this table corresponds to up or down movement indicator for one of the 500 stocks in S&P500 stocks, and each column corresponds to a trading day from Jan. 1994 to Oct. 1996. An entry of 1 in location (i, j) where i corresponds to up indicator of a stock means that the closing price of this stock on j-th day is significantly higher (2% or 1/2 point or more) than the day before. Similarly, an entry of 1 in location (i, j) where i corresponds to down indicator of a stock means that the closing price of this stock on j-th day is significantly lower (2% or 1/2 point or more) than the day before.

We clustered these stock indicators using our hypergraph-based method. We used a minimum support threshold of 3% which means that all stocks in a frequent item set must have moved together at least on 22 days. This lead to a hypergraph consisting of 440 vertices and 19602 hyperedges. Note that the number of vertices in the hypergraph is considerably smaller than the number of distinct items in the data-set. This is because some of the stocks do not move very frequently, hence the corresponding items do not have sufficient support. This hypergraph was then partitioned into 40 partitions. Out of these 40 partitions, only 20 of them satisfy the fitness function. Out of 20 clusters, 16 clusters were clean clusters as they contain stocks primarily from one industry group. Some of the clean clusters found from this data are shown in Table 1 and the complete list of clusters is available in [HKKM97b]. Looking at these clusters we can see that our item-clustering algorithm was very successful in grouping together stocks that belong to the same industry group. For example, our algorithm was able to find technology-, financial-, oil-, gold-, and metal-related stock-clusters. Also, it is interesting to see that our clustering algorithm partitioned the technology companies into two groups, and the first of them consists mostly of networking and semiconductor companies.

3.2 Protein Coding Database

Our next data set is from a problem from the domain of molecular biology. Molecular biologists study genes within the cells of organisms to determine the biological function of the proteins that those genes code for. Faced with a new protein, biologists have to perform a very laborious and painstaking experimental process to determine the function of the protein. To rapidly determine the function of many previously unknown genes, biologists generate short segments of protein-coding sequences (called expressed sequence tags, or ESTs) and match each

EST against the sequences of known proteins, using similarity matching algorithms [NRS+95]. The result of this matching is a table showing similarities between ESTs and known proteins. If the EST clusters can be found from this table such that ESTs within the same cluster are related to each other functionally, then biologists can match any new EST from new proteins against the EST clusters to find those EST clusters that match the new EST most closely. At this point, the biologists can focus on experimentally verifying the functions of the new EST represented by the matching EST clusters. Hence finding clusters of related ESTs is an important problem. Related work in this area can be found in [HHS92], which reports clustering of the sequence-level building blocks of proteins by finding transitive closure of the pairwise probabilistic similarity judgments.

To assess the utility of hypergraph-based clustering in this domain, we performed experiments with data provided by the authors of [NRS+95]. Our data set consists of a binary table of size 662×11986. Each row of this table corresponds to an EST, and each column corresponds to a protein. An entry of 1 in location (i, j) means that there is a significant match between EST i and protein j. We clustered the ESTs using our hypergraph-based method. In finding the frequent item sets we used a support of 0.02%, which essentially created frequent item sets of EST that are supported by at least three proteins. This led to a hypergraph with 407 vertices and 128,082 hyperedges. We used HMETIS to find 46 partitions, out of which 39 of them satisfied the fitness criteria. The total number of ESTs clustered was 389. These 39 EST clusters were then given to biologist to determine whether or not they are related. Their analysis showed that most of the EST clusters found by our algorithm correspond to ESTs that are related. In fact, 12 of the 39 clusters are very good, as each corresponds to a single protein family. These 12 clusters together contain 113 ESTs. Three clusters are bad (they contained random ESTs), and analysts could not determine the quality of 2 other clusters. Each of the remaining 22 clusters has subclusters corresponding to distinct protein families – 6 clusters have two subclusters, 7 clusters have three subclusters, 3 clusters have four subclusters, and 6 clusters have five subclusters. Furthermore, when we examined the connectivity of the sub-hypergraphs that correspond to some of these 22 clusters, we were able to see that further subdivision would have created single-protein clusters. This is particularly important, since it verified that the association-rule hypergraph is highly effective in modeling the relations between the various ESTs, and we are able to verify the quality of the clusters by looking at their connectivity. Also, our clustering algorithm took under five minutes to find the various clusters which includes the time required to find the association rules and form the hypergraph.

4 Conclusion and Directions for Future Work

In this paper, we have presented a method for clustering data in a high dimensional space based on a hypergraph model. Our experiments indicate that the hypergraph-based clustering holds great promise for clustering data in large dimensional spaces. Traditional clustering schemes such as *Autoclass* [CS96] and K-means [JD88] cannot be directly used in such large dimensionality data sets, as they tend to produce extremely poor results [HKKM97b]. These methods perform much better when the dimensionality of the data is reduced using methods such as Principal Component Analysis [Jac91]. However, as shown by our experiments in [HKKM97b], the hypergraph-based scheme produces clusters that are at least as good or better than those produced by *AutoClass* or K-means algorithm on the reduced dimensionality data sets.

One of the major advantages of our scheme over traditional clustering schemes is that it does not require dimensionality reduction, as it uses the hypergraph model to represent relations among the data items. This model allows us to effectively represent important relations among items in a sparse data structure on which computationally efficient partitioning algorithms can be used to find clusters of related items. Note that the sparsity of the hypergraph can be controlled by using an appropriate support threshold. An additional advantage of this scheme is its ability to control the quality of clusters according to the requirements of users and domains. With different levels of minimum support used in *Apriori* algorithm, the amount of relationship captured in the hypergraph model can be adjusted. Higher support gives better quality clusters containing smaller number of data items, whereas lower support results in clustering of larger number of items in poorer quality clusters. Furthermore, our hypergraph model allows us to correctly determine the quality of the clusters by looking at the internal connectivity of the nodes in each cluster. This fitness criterion in conjunction with the connectivity threshold discussed in Section 2 provide additional control over the quality of each cluster. Computationally, our scheme is linearly scalable with respect to the number of dimensions of data (as measured in terms of the number of binary variables) and items, provided the support threshold used in generating the association rules is sufficiently high.

Like other clustering schemes and dimensionality reduction schemes, our approach also suffers from the fact that right parameters are necessary to find good clusters. The appropriate support level for finding frequent item sets is largely depend on the application domain. Another limitation of our approach is that our scheme does not naturally handle continuous variables as they need to be discretized. However, such a discretization can lead to a distortion in the relations among items, especially in cases in which a higher value indicates a stronger relation. For this type of domains, in which continuous variables with higher values imply greater importance, we have developed a new algorithm called *Min-Apriori* [Han98] that operates directly on these continuous variables without discretizing them.

Our current clustering algorithm relies on the hypergraph partitioning algorithm HMETIS to find a good k-way partitioning. As discussed in Section 2, even though HMETIS produces high quality partitions, it has some limitations. Particularly, the number of partitions must be specified by the users as HMETIS does not know when to stop recursive bisection. However, we are working to incorporate the fitness criteria into the partitioning algorithm such that the partitioning algorithm determines the right number of partitions automatically. Another way of clustering is to do a bottom-up partitioning followed by a cluster refinement. In this approach, instead of using HMETIS to find the partitions in a top-down fashion, we can start growing the partitions bottom-up by repeatedly grouping highly connected vertices together. These bottom-up partitions can then be further refined using a k-way partitioning refinement algorithm as implemented in HMETIS.

Acknowledgments

We would like to thank Elizabeth Shoop and John Carlis for providing the protein coding data and verifying our results.

References

[AS94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*, pages 487–499, Santiago, Chile, 1994.

[BDO95] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573–595, 1995.

[Ber76] C. Berge. *Graphs and Hypergraphs*. American Elsevier, 1976.

[BMS97] S. Brin, R. Motwani, and C. Silversteim. Beyond market baskets: Generalizing association rules to correlations. In *Proc. of 1997 ACM-SIGMOD Int. Conf. on Management of Data*, Tucson, Arizona, 1997.

[CHY96] M.S. Chen, J. Han, and P.S. Yu. Data mining: An overview from database perspective. *IEEE Transactions on Knowledge and Data Eng.*, 8(6):866–883, December 1996.

[CS96] P. Cheeseman and J. Stutz. Baysian classification (autoclass): Theory and results. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI/MIT Press, 1996.

[Han98] E.H. Han. Knowledge discovery in a large dimensional space using hypergraph models. Technical Report Thesis In Progress, Department of Computer Science, University of Minnesota, M inneapolis, 1998.

[HBG+98] E.H. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. Webace: A web agent for document categorization and exploartion. In *Proc. of the 2nd International Conference on Autonomous Agents*, May 1998.

[HHS92] N. Harris, L. Hunter, and D. States. Mega-classification: Discovering motifs in massive datastreams. In *Proceedings of the Tenth International Conference on Artificial Intelligence (AAAI)*, 1992.

[HKKM97a] E.H. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs (position paper). In *Proc. of the Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 9–13, Tucson, Arizona, 1997.

[HKKM97b] E.H. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering in a high-dimensional space using hypergraph models. Technical Report TR-97-063, Department of Computer Science, University of Minnesota, Minneapolis, 1997.

[Jac91] J. E. Jackson. *A User's Guide To Principal Components*. John Wiley & Sons, 1991.

[JD88] A.K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[KAKS97] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in VLSI domain. In *Proceedings ACM/IEEE Design Automation Conference*, 1997.

[Kar98] G. Karypis. hMETIS 1.0.0. *http://www.cs.umn.edu/∼karypis/metis/hmetis/main.html*, 1998.

[MHB+97] J. Moore, E. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, and B. Mobasher. Web page categorization and feature selection using association rule and principal component clustering. In *7th Workshop on Information Technologies and Systems*, Dec. 1997.

[NRS+95] T. Newman, E.F. Retzel, E. Shoop, E. Chi, and C. Somerville. Arabidopsis thaliana expressed sequence tags: Generation, analysis and dissemination. In *Plant Genome III: International Conference on the Status of Plant Genome Research*, San Diego, CA, 1995.