The 3rd International Conference on Hybrid Intelligent Systems (HIS'03), December 14-17, 2003, Melbourne, Australia.

Refining A Divisive Partitioning Algorithm for Unsupervised Clustering

Canasai KRUENGKRAI, Virach SORNLERTLAMVANICH, Hitoshi ISAHARA
Thai Computational Linguistics Laboratory
Communications Research Laboratory
112 Paholyothin Road, Klong 1, Klong Luang, Pathumthani 12120, Thailand
E-mail: {canasai,virach}@crl-asia.org, isahara@crl.go.jp

Abstract. The Principal Direction Divisive Partitioning (PDDP) algorithm is a fast and scalable clustering algorithm [3]. The basic idea is to recursively split the data set into sub-clusters based on principal direction vectors. However, the PDDP algorithm can yield poor results, especially when cluster structures are not well-separated from one another. Its stopping criterion is based on a heuristic that often tends to over-estimate the number of clusters. In this paper, we propose simple and efficient solutions to the problems by refining results from the splitting process, and applying the Bayesian Information Criterion (BIC) to estimate the true number of clusters. This motivates a novel algorithm for unsupervised clustering, which its experimental results on different data sets are very encouraging.

1 Introduction

Unsupervised clustering is one of the important techniques in scientific data analysis and data mining. The goal of clustering is to partition a set of data points into meaningful groups according to some predefined criteria. There are many application areas of clustering, including document clustering, gene expression analysis, and image segmentation. A wide variety of algorithms for unsupervised clustering problems have been intensively studied. For example, the classical k-means algorithm and its extensions group the input data into k clusters such that all the points in each cluster are more similar to one another than to those in the other clusters. One major drawback of the k-means algorithm is that the user must supply the number of clusters. Other approaches, such as agglomerative algorithms, have quadratic (or higher order) computational complexity and do not scale up [6].

Recently, Boley [3] has developed a fast and scalable clustering algorithm called the Principal Direction Divisive Partitioning (PDDP) algorithm. It was firstly developed for the document clustering task, and has been applied to other application domains, such as vision-based texture analysis, and movie rating [2]. The PDDP algorithm has several interesting properties. It employs the concept of the principal component analysis, and takes advantage of sparseness of the input data. It also generates a hierarchal tree of clusters that inherently produces a simple taxonomic ontology. However, the PDDP algorithm can yield poor results, especially when cluster structures are not well-separated from one another. Furthermore, its stopping criterion is based on a heuristic that often tends to over-estimate the number of clusters.

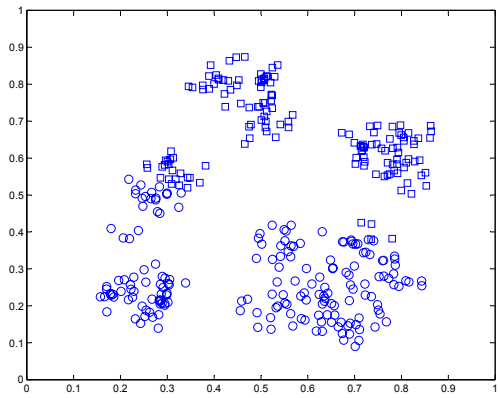

Figure 1: The first iteration of the PDDP algorithm.

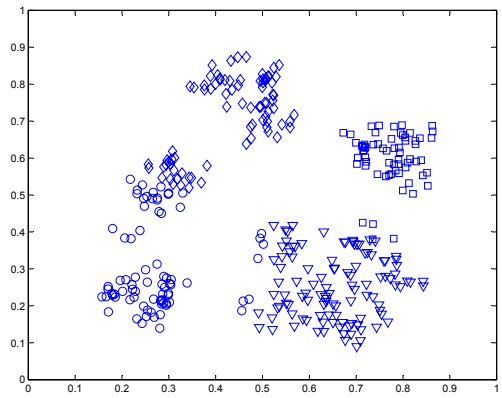

Figure 2: The third iteration of the PDDP algorithm.

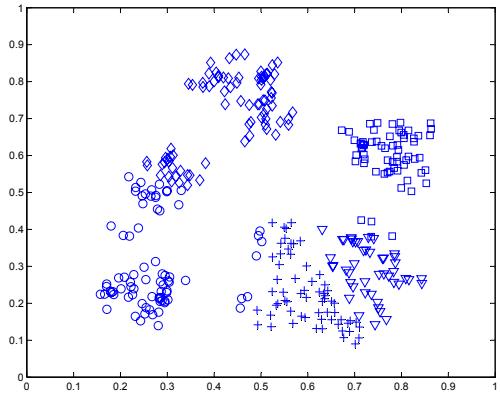

Figure 3: The fourth iteration of the PDDP algorithm.

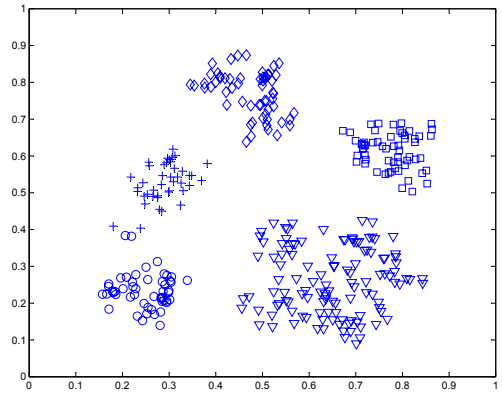

Figure 4: The fourth iteration of our algorithm.

Let us describe with an empirical example. We ran the PDDP algorithm on a data set taken from [9].[1] The data set consists of 334 points drawn in 2 dimensions. The actual class labels are not given, but we can observe that it is composed of five compact clusters of data points. Figure 1 shows the clustering result from the PDDP algorithm after the first iteration. Based on the principal direction vectors, it splits the data into two sub-clusters. We can see that the PDDP algorithm starts with significantly wrong partitioning on the middle left-hand cluster.

Figure 2 and 3 demonstrate the clustering results after the third and fourth iterations. The bottom right-hand cluster is divided into two large sub-clusters and two small sub-clusters, although it should be retained as a single cluster. From these results, we observe that only using the principal direction vectors for splitting clusters can produce poor solutions in some cases. We may need to adjust the centroids and their members resulting from each splitting process. Also, we require some information to suggest whether we should split the cluster further or should keep it as it is. This motivates our work to alleviate these problems and obtain better results. Figure 4 shows the clustering result using our new algorithm, which will be described in more detail later.

In this paper, we propose simple and efficient solutions to the problems by refining results from the splitting process, and applying a model selection technique called the Bayesian Information Criterion (BIC) to estimate the true number of clusters. In order to adjust the

[1]The data set is available at `http://www.jihe.net/datasets.htm`

centroids and their members in each splitting process, we locally run the 2-means algorithm, the k-means algorithm with the number of clusters $k = 2$, on the each data region. The BIC is used to measure the improvement of the cluster structure between the root cluster and its two children clusters. This can help to estimate the number of underlying clusters in the data set. Experimental results show that our new algorithm compares favorably to the original PDDP algorithm.

The rest of this paper is organized as follows. In Section 2, we describe the unsupervised clustering framework, containing some important background relevant to our work. In the context of this section, we briefly review how the original PDDP algorithm performs, and then summarize the BIC. In Section 3, we describe our new algorithm in detail. Section 4 explains the data sets and evaluation methods used in our experiments, and shows experimental results. Finally, we conclude in Section 5 with some directions of future work.

2 Unsupervised Clustering Framework

2.1 Principal Direction Divisive Partitioning

In contrast to the hierarchical agglomerative clustering that performs bottom-up clustering by merging the pair of closest points (or clusters) using some distant function, the divisive partitioning algorithm does the opposite by partitioning the data into sub-clusters in the top-down manner. For the PDDP algorithm, its partitioning scheme is based on the principal direction vectors that correspond to the first left singular vector of the singular value decompositions (SVD) for the cluster. Let $\mathcal{C}$ be a root cluster, where $\mathcal{C} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{C}|}\}$. After obtaining the principal direction vector of $\mathcal{C}$ denoted by $\mathbf{u}_\mathcal{C}$, we can split the cluster into two children clusters named the left child $\mathcal{L}$ and right child $\mathcal{R}$ by the following discriminant function:

$$f_\mathcal{C}(\mathbf{x}_i) = \mathbf{u}_\mathcal{C}^T(\mathbf{x}_i - \boldsymbol{\mu}_\mathcal{C}) \,, \tag{1}$$

and

$$\mathbf{x}_i \in \begin{cases} \mathcal{L}, \text{ if } f_\mathcal{C}(\mathbf{x}_i) \leq 0 \\ \mathcal{R}, \text{ if } f_\mathcal{C}(\mathbf{x}_i) > 0 \,. \end{cases} \tag{2}$$

where $\boldsymbol{\mu}_\mathcal{C}$ is the centroid vector corresponding to $\mathcal{C}$, which can be calculated as follows:

$$\boldsymbol{\mu}_\mathcal{C} = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} \mathbf{x}_i \,. \tag{3}$$

We can see that the important step of the PDDP algorithm is to find the vector $\mathbf{u}_\mathcal{C}$. Let $\mathbf{M}$ be the sample matrix, and $\tilde{\mathbf{M}} = \mathbf{M} - \boldsymbol{\mu}\boldsymbol{e}^T$, where $\boldsymbol{e} = (1, \ldots, 1)^T$. We need to compute the SVD of $\tilde{\mathbf{M}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, and get the first column of $\mathbf{U}$ to be the vector $\mathbf{u}_\mathcal{C}$ of the cluster. As suggested in [3][12], we can efficiently calculate the vector $\mathbf{u}_\mathcal{C}$ by using the Lanczos method. More details of solving the SVD with the Lanczos method can be found in [8].

The PDDP algorithm starts with all the data points in a large single cluster, and proceeds by recursively splitting the cluster into sub-clusters based on the discriminant function in Equation 1 and 2. It finally yields a binary tree, which leaf nodes represent the output clusters containing their members. In order to keep the binary tree balanced, it selects an un-split cluster to split by using the scatter value, measuring the average distance from the data points in the cluster to their mean. An alternative technique for selecting the cluster to split is based on the shape of the cluster [12].

2.2 Bayesian Information Criterion

Using model selection techniques has been applied in many clustering algorithms. For example, the x-means algorithm [11], which is an extension of the k-means algorithm, also employs the BIC to estimate the number of clusters. An equivalent technique called the Minimum Description Length (MDL) principal is applied in [10].

The problem of model selection is to choose the best one among a set of candidate models. Let $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a set of input data $\mathcal{D}$, where each $\mathbf{x}_i \in \Re^d$, and $\mathcal{D}$ can be partitioned into disjoint subset $\mathcal{C}_1, \ldots, \mathcal{C}_k$. The BIC of the model $\mathcal{M}_i$ is defined as:

$$BIC(\mathcal{M}_i) = \hat{l}_i(\mathcal{D}) - \frac{p_i}{2} \cdot \log n \,, \tag{4}$$

where $\hat{l}_i(\mathcal{D})$ is the log-likelihood of the data according to the model $\mathcal{M}_i$, and p_i is the number of independent parameters. The BIC contains two components, where the first term measures how well the parameterized model predicts the data, and the second term penalizes the complexity of the model [4].

The probability that a data point $\mathbf{x}_i$ belongs to a cluster $\mathcal{C}_j$ can be defined as the product of the probability of observing $\mathcal{C}_j$ and the multivariate normal density function of $\mathbf{x}_i$:

$$\hat{P}(\mathbf{x}_i) = \frac{n_j}{n} \cdot \frac{1}{\sqrt{2\pi}\hat{\sigma}^d} \exp\Big(-\frac{1}{2\hat{\sigma}^2} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2 \Big) \,, \tag{5}$$

where n_j is the number of points in the cluster $\mathcal{C}_j$, and $\hat{\sigma}^2$ is the maximum likelihood estimate (MLE) of the variance defined by:

$$\hat{\sigma}^2 = \frac{1}{n-k} \sum_i (\mathbf{x}_i - \boldsymbol{\mu}_j)^2 \,. \tag{6}$$

Thus the maximum log-likelihood of the data in cluster $\mathcal{C}_j$ can be calculated as:

$$\begin{aligned} \hat{l}(\mathcal{C}_j) &= \log \prod_{i \in \mathcal{C}_j} \hat{P}(\mathbf{x}_i) \\ &= \sum_{i \in \mathcal{C}_j} \Big(\log \frac{1}{\sqrt{2\pi}\hat{\sigma}^d} - \frac{1}{2\hat{\sigma}^2} \|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2 + \log \frac{n_i}{n} \Big) \\ &= -\frac{n_j}{2} \log(2\pi) - \frac{n_j \cdot d}{2} \log(\hat{\sigma}^2) - \frac{n_j - k}{2} + n_j \log n_j - n_j \log n \,. \end{aligned} \tag{7}$$

Finally, we can write the BIC as follows:

$$BIC(\mathcal{M}_i) = \sum_{j=1}^{k} \hat{l}(\mathcal{C}_j) - \frac{p_i}{2} \cdot \log n \,. \tag{8}$$

Given a set of candidate models, the model with the highest BIC score, $\text{argmax}_i BIC(\mathcal{M}_i)$, is selected. We use the BIC to measure the improvement of the cluster in both the local and global structure. We calculate the BIC locally when the PDDP algorithm performs the splitting test in each cluster. If the BIC score of the new cluster structure is less than the current BIC score, we do not split the cluster. The BIC is calculated globally to measure the entire structure improvement after breaking the cluster into two children clusters.

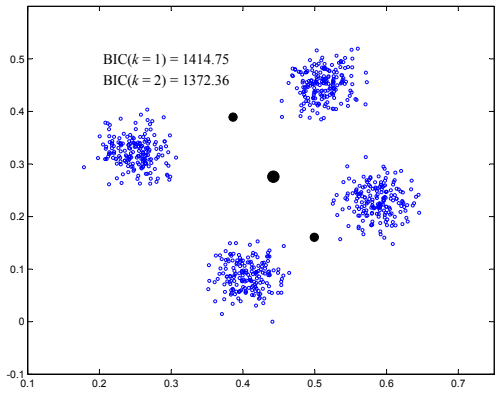

Figure 5: The BIC scores of the root cluster and its children clusters.

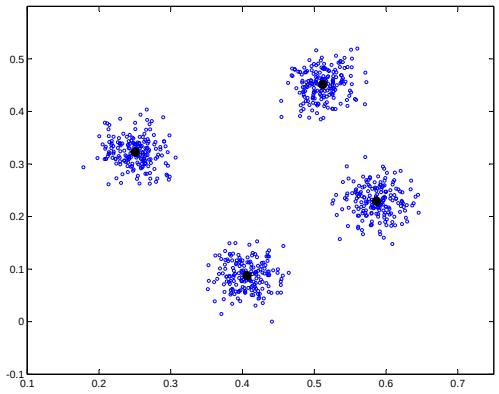

Figure 6: Four output centroids after partitioning further.

3 The *r*PDDP algorithm

In this section, we describe our new algorithm named the *r*PDDP (refinement PDDP) algorithm. As mentioned earlier, the first problem of the PDDP algorithm is that it may produce poor clustering results when cluster structures are not well-separated from one another, and the relative principal direction vectors are not informative. However, it is possible to adjust the cluster structure by re-allocating the centroid and their member positions. Thus, we run the 2-means algorithm on the local region containing two children centroids until convergence. This idea is close to the concept of the bisecting k-means algorithm [13]. However, our initial centroids are based on the principal direction vectors rather than using random initialization. In other words, we can think of the centroids of two children clusters as the initial centroids for the 2-means algorithm.

The second problem of the PDDP algorithm is that it uses the stop splitting criterion based on the user requirement or the change of the overall scatter values that often tends to over-estimate the number of clusters. When we need to apply the algorithm to new problem domains having little knowledge about the data, using these heuristics are inefficient to discover or predict the latent cluster structures. Here we adopt the BIC to determine whether we should split the cluster into two sub-clusters, or retain the current cluster structure. The BIC is also used to measure the improvement of the entire cluster structure after the splitting process. However, the BIC is not always useful in some cases.

Figure 5 shows an example, where the center point is the root centroid and the relatively small points are its children centroids. We can see that the BIC score does not improve, although this cluster structure should be partitioned further into two and four sub-clusters. From our preliminary experiments, we observe that the BIC is not useful for *null-centroids*. The root centroid is considered to be the null-centroid if it has these characteristics: very few members belong to the root centroid compared with its children centroids, and the root centroid often lies in the space with no data points in the vicinity. Therefore, we should definitely split the cluster without using the BIC. In our current work, we just measure the Euclidean distances among the data points and the candidate centroids to determine the null-centroid. Figure 6 shows the clustering result after partitioning further.

These above refinement strategies are combined in each splitting process of the PDDP run. The computational time is reasonable, since the initial centroids from the PDDP algorithm are better than random ones. It performs a moderate number of 2-means iterations. Figure 7

Input: A data set representing by a matrix $\mathbf{M} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$.

1. Initialize a binary tree $\mathcal{T}$ with a single root node, and set $k = 0$.
2. Loop while the global cluster structure improves or $k < k_{\max}$.
   - 2.1 Select the leaf node $\mathcal{C}$ with the largest scatter value.
   - 2.2 Compute the centroid $\boldsymbol{\mu}_{\mathcal{C}}$ and the principal direction vector $\mathbf{u}_{\mathcal{C}}$.
   - 2.3 For $\mathbf{x}_i \in \mathcal{C}$, assign $\mathbf{x}_i$ to left child $\mathcal{L}$ or right child $\mathcal{R}$ according to Equation 1 and 2.
   - 2.4 If $\mathcal{C}$ has the null-centroid or $BIC(2\text{-means}(\mathcal{L}, \mathcal{R})) > BIC(\mathcal{C})$ then Set $\mathcal{T} = \mathcal{T} \cup \{\mathcal{L}, \mathcal{R}\}$ and $k = k + 1$.

Output: A binary tree $\mathcal{T}$ forming a partitioning of the entire data set.

Figure 7: The outline of the rPDDP algorithm for unsupervised clustering.

shows the outline of the *r*PDDP algorithm.

4 Experimental Results

To study the performance of the *r*PDDP algorithm in unsupervised clustering, we performed empirical experiments on both synthetic and real data sets. On the data set given class labels, we could compare the clustering results against the true class labels directly. We measured the clustering results on the data set that has no class labels using the distortion (or the sum-of-squared-error criterion [5]), which smaller distortion values indicate better clustering results.

4.1 Data Sets

The synthetic data consist of two data sets used in [7]. The first data set, `2D2K`, contains 500 points of 2 Gaussian centroids in 2 dimensions. The second data set, `8D5K`, contains 1000 points from 5 multivariate Gaussian distributions (200 points each) in 8 dimensions. The data sets are available at `www.lans.ece.utexas.edu/~strehl/data.html`.

The Iris data set is the standard benchmark in the pattern recognition literature [1]. It consists of 150 instances of three types of flowers having four features: sepal length, sepal width, petal length, and petal width. One of the clusters is linearly separable from the other two. The remaining two clusters have significantly overlapping. Since each element is categorized, we can compare clustering results with the true class labels.

4.2 Results

We compare the *r*PDDP algorithm with the original PDDP algorithm. However, using the change in the overall scatter values as the stopping criterion in the original PDDP algorithm does not seem to converge to the true number of clusters. We also apply the BIC to be the stopping criterion for the PDDP algorithm.

Figure 8 shows the clustering result using the *r*PDDP algorithm on the `2D2K` data set. Since the structure of this data set is simple, both the *r*PDDP and PDDP algorithms generate

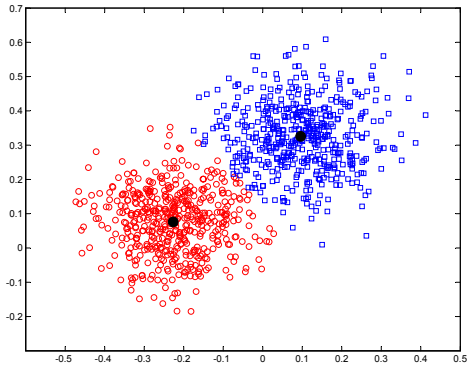

Figure 8: The clustering result of the *r*PDDP algorithm on the `2D2K` data set.

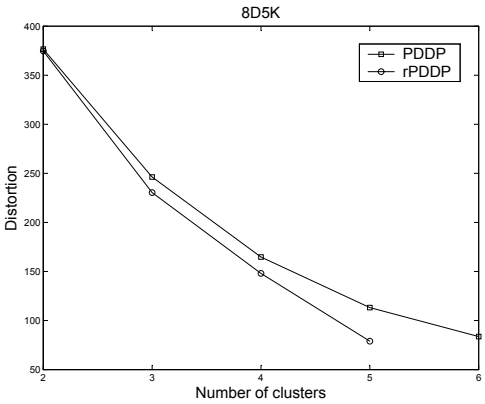

Figure 9: Distortion values of the PDDP and *r*PDDP algorithms on the `8d5k` data set.

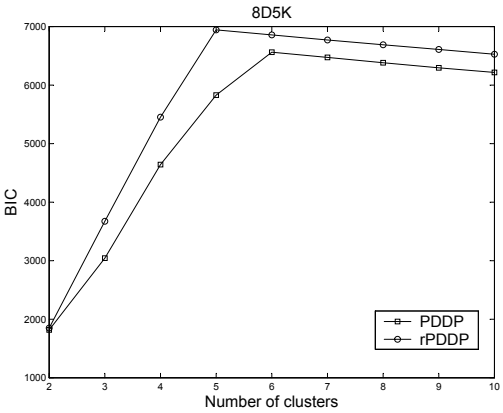

Figure 10: BIC scores of the PDDP and *r*PDDP algorithms on the `8d5k` data set.

slightly different results. While the PDDP algorithm achieves 21.87 distortion and 1685.86 BIC, the *r*PDDP algorithm reaches 19.04 distortion and 1826.26 BIC. Both algorithms converge, resulting only 2 clusters that are equivalent to the actual number of clusters in this data set.

Figure 9 and 10 demonstrate the distortion values and the BIC scores on the `8D5K` data set. Interestingly, on this data set, both algorithms using the BIC can efficiently estimate the true number of clusters. While the PDDP algorithm generates 6 clusters, the *r*PDDP algorithm converges, producing 5 clusters that are the true number of clusters. We can see from the curve that the BIC score of the *r*PDDP algorithm does not improve after 5 clusters. Based on the distortion values, we can observe that the *r*PDDP algorithm constantly outperforms the PDDP algorithm.

Table 1 shows the clustering result using the *r*PDDP algorithm on the Iris data set. The PDDP algorithm also generates the same result. There are 8 wrongly clustered elements. On this data set, both algorithms converge, producing 4 output clusters.

5 Conclusion and Future Work

We have presented refinement strategies for the PDDP algorithm. When the principal direction vectors are not informative due to some data distributions, the PDDP algorithm can give

cluster	setosa	versicolor	virginica
1	50	0	0
2	0	43	1
3	0	2	30
4	0	5	19

Table 1: The clustering result of the *r*PDDP algorithm on the Iris data set.

poor clustering results. Our *r*PDDP algorithm solves this problem by running the 2-means algorithm locally to adjust the centroids and their members. We also apply the BIC to estimate the true number of clusters. Preliminary results on different data sets are very promising.

In future work, we intend to conduct more extensive experiments on other benchmark data sets. We are also interested in using the binary tree structure generated by the *r*PDDP algorithm as a simple taxonomic ontology. We believe that it can be valuable for many other tasks, such as semi-automatic ontology construction.

References

[1] C. L. Blake and C. J. Merz. UCI Repository of Machine Learning Databases, 1998.

[2] D. Boley and V. Borst. Unsupervised Clustering: A Fast Scalable Method for Large Datasets. CSE Report TR-99-029, University of Minnesota, 1999.

[3] D. Boley. Principal Direction Divisive Partitioning. Data Mining and Knowledge Discovery, 2(4):325-344, 1998.

[4] D. Chickering, D. Heckerman, and C. Meek. A Bayesian Approach to Learning Bayesian Networks with Local Structure. In Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence, pages 80–89. Morgan Kaufmann, 1997.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. Pattern Classification. John Wiley & Sons, New York, 2001.

[6] G. Forman and B. Zhang. Linear Speed-Up for a Parallel Non-Approximate Recasting of Center-Based Clustering Algorithms, including K-Means, K-Harmonic Means, and EM. Workshop on Distributed and Parallel Knowledge Discovery, KDD-2000, 2000.

[7] J. Ghosh, A. Strehl, and S. Merugu. A Consensus Framework for Integrating Distributed Clusterings Under Limited Knowledge Sharing. Proc. NSF Workshop on Next Generation Data Mining, pages 99–108, 2002.

[8] G. Golub, and C. V. Loan. Matrix Computations. The Johns Hopkins University Press, Baltimore, 1989.

[9] J. He, A.-H. Tan, C.-L. Tan, and S.-Y. Sung. On Quantitative Evaluation of Clustering Systems. In W. Wu and H. Xiong, editors, Information Retrieval and Clustering. Kluwer Academic Publishers, 2002.

[10] T. Nomoto and Y. Matsumoto. A New Approach to Unsupervised Text Summarization. Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01), pages 26–34, 2001.

[11] D. Pelleg and A. Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In Seventeenth International Conference on Machine Learning, 2000.

[12] S. Savaresi, D. L. Boley, S. Bittanti, G. Gazzaniga. Choosing the Cluster to Split in Bisecting Divisive Clustering Algorithms. CSE Report TR 00-055, University of Minnesota, 2000.

[13] M. Steinbach, G. Karypis, and V Kumar. A Comparison of Document Clustering Techniques. In KDD Workshop on Text Mining, 2000.