Scalability of Parallel Genetic Algorithm for Two-mode Clustering

Briti Deb
Institute of Computer Science, University of Tartu,
J.Liivi 2, Tartu, Estonia

Satish Narayana Srirama
Institute of Computer Science, University of Tartu,
J.Liivi 2, Tartu, Estonia

ABSTRACT
Data matrix having the same set of entity in the rows and cloumns is known as one-mode data matrix, and traditional one-mode clustering algorithms can be used to cluster the rows (or columns) separately. With the popularity of use of two-mode data matrices where the rows and columns have different sets of entities, the need for simultaneous clustering of rows and columns popularly known as two-mode clustering increased. Additionally, the emergence of large data sets and the prediction of Moore's law slow-down have created the challenge of clustering scalability. In this paper, we address the problem of scalability of organizing an unlabelled two-mode dataset into clusters utilizing multicore processor. We propose a parallel genetic algorithm (GA) heuristics based two-mode clustering algorithm, which is an adaptation of the classical Cuthill-McKee Matrix Bandwidth Minimization (MBM) algorithm. The classical MBM method aims at reducing the bandwidth of a sparse symmetric matrix, which we adapted to make it suitable for non-symmetric real-valued matrix. Preliminary results indicate that our algorithm is scalable on multicore processor compared to serial implementation. Future work will include more extensive experiments and evaluations of the system.

General Terms
Multicore

Keywords
Scalability; two-mode clustering; parallel genetic algorithm; matrix reordering

1. INTRODUCTION
With the advent of big data, there has been an increase in demand for more computing resources for scalable analysis of the data. Recent study [1] indicates concern that the current biological tools need to improve in functionality to address issues of scalability. Furthermore, as the Moore's law prediction that the number of transistors on integrated circuits doubles approximately every 18 months is expected to slow down, thus slowing down the processing speed of computers, there has been an increase in efforts to increase the number of processing cores and develop algorithms to utilize multicore [2]. Though research is been conducted on developing various high-performance computing systems, there has been a lack of good programming framework for multicore architectures [2]. This has created the challenge of obtaining speedup in multicore architecture. Here we propose a parallel genetic algorithm (GA) based two-mode clustering algorithm to utilize multicore architecture, and show that its scalable on multicore processor compared to serial implementation.

Activities such as measurements, test methods, clinical trials, and the like, produce large amounts of data, which needs to be stored, analyzed, and curated. A popular way to store data is in the form of a matrix $X = (x_{ij})$. A matrix having the same set of entity in the *i* rows and *j* columns is called a one-mode matrix [3], and the traditional clustering techniques such as connectivity based clustering (e.g., hierarchical clustering), centroid-based clustering (e.g., k-means), distribution-based clustering (e.g., gaussian mixture model), and density-based clustering (e.g., DBSCAN), to name a few, can be applied to group the objects (*E*) in the rows (*R*), yielding row clusters $E \subseteq R$. Similarly, the objects in the columns (*C*) can also be grouped into column clusters. This implies that if the rows and columns need to be grouped, they will have to be grouped sequentially, not simultaneously. This poses a problem to simultaneously cluster the rows and columns of a two-mode matrix [3], who have two different sets of entities in the rows and cloumns. An example of this type of data matrix is the data generated from experiments on DNA, RNA, and protein microarrays, where the rows may correspond to genes while the columns correspond to conditions, and researchers might want to know the quantity of expression of gene(s) under the influence of particular condition(s). It requires a different approach than the traditional one-mode clustering algorithms in order to yield simultaneous clustering of the rows and columns $E \subseteq R x C$ or $E = F x G$ where $F \subseteq R$ and $G \subseteq C$ and *F* and *G* may not be contiguous. There are two main reasons [4] why a different approach to simultaneously cluster rows and columns in two-mode matrices is needed instead of the traditional one-mode clustering algorithms. First, simultaneous clustering method optimizes an objective function which cannot be reduced to a simple combination of constituent row and column objective functions. Second: The association between the rows and columns as implied in the data can only be captured by simultaneous clustering of the rows and columns.

We take the two-mode clustering problem as an optimization problem, and address it by adapting the Cuthill-McKee matrix bandwidth minimization (MBM) algorithm [5] solved by genetic algorithm [6]. The bandwidth of a sparse matrix is the maximum distance between two non-zero elements in any row. The problem consists in finding the desired permutation of the rows and columns of a matrix, so that by reducing the matrix bandwidth, the coherent non-zero elements are placed as close as possible to the main diagonal. This results to visually interpretable two-mode clusters. Each row and column are exclusively placed to one of the biclusters. In large matrices, there exists a large number of possible permutations of rows and columns (due to large search space), finding all of which is highly time intensive. We addressed this problem by

using genetic algorithm [6], which employs optimization strategies inspired by natural evolution and transform them for mathematical optimization to find the global optimum. In GA, three genetic operators, namely, selection, crossover, and mutation, are considered to evolve a population of candidate solutions to form new candidate solution. Selection selects a part of the population according to some fitness function. Crossover combines different parts of population to form new population. Mutation randomely changes parts of population to form new population. Finding an optimal solution using GA is a popular choice when the data matrix is very large and the landscapes are not smooth or not unimodal.

2. LITERATURE SURVEY

A common task in data analysis is clustering, which involves grouping a set of objects in such a way that objects in a group are more similar than objects in different groups. The difficulty in precisely defining the notion of a "cluster" [7] motivated the development of different clustering algorithms, such as hierarchical clustering, k-means, gaussian mixture model, DBSCAN, to name a few. The efficacy of different clustering algorithms depends on the particular data set, thereby eliminating any objectively "correct" clustering algorithm, as noted by [7] that "clustering is in the eye of the beholder".

There has been an increased availability of larger data sets (also known as big data). Many of such large data sets are two-mode data, which has the form of a real-valued matrix $X = (x_{ij})$ where the value of x_{ij} represent the relation between the i^{th} row and the j^{th} column. To analyze such data sets, it is often necessary to identify subsets of rows with certain coherence properties in a subsets of the columns, thus creating two-mode clusters. Several methods were developed to address this problem, such as bi-clustering, co-clustering [8] [9], matrix re-ordering/seriation [10], graph clustering [11], and graph partitioning [12]. Although such methods have been useful to certain extent in addressing the problem of two-mode clustering, the problem of addressing the scalability issue in the age of big data still remain a challenge [1]. Here we attempt to address the scalability issue of two-mode clustering algorithm by applying parallel genetic algorithm [6] in Cuthill-McKee matrix bandwidth optimization problem [5].

Two-mode clustering has application in several fields, such as metabolomics data analysis [6], multiple trait data stemming [13], community detection in social networks [14], marketing applications [15], graph co-clustering for ontology mapping [16], and many other applications mentioned in [17] such as dimensionality reduction (also known as subspace clustering) in large databases, and collaborative filtering for recommendation system / target marketing (where the data set rows can be customers and columns can be movies).

3. PROPOSED ALGORITHM

3.1 The Two-mode Clustering Problem

We adapt the Cuthill-McKee bandwidth minimization [5] problem towards formulating our two-mode clustering problem. Given a two-mode (unlabelled) data matrix $X = \{x_{i,j}\}$ where $i \subseteq [1..m]$ and $j \subseteq [1..n]$, the objective is to obtain, a partition of row and column space into non-overlapping index sets, $R = \{r_1, ..., r_p\}$ and $C = \{c_1, ..., c_q\}$ where $r_i \subseteq [1..m]$, $c_j \subseteq [1..n]$, p and q are row and column partition set cardinalities respectively, such that coherent values on both rows and columns comes as close as possible to the main diagonal, creating subsets (clusters) that reflects the underlying structure of the data.

If the rows and columns of a two-mode matrix are represented as nodes of a graph, and the non-zero elements of the matrix are represented by edges, then the MBM problem can also be described in terms of graph partitioning, by minimizing the bandwidth of the graph. The bandwidth of a vertex $B_f(v) = \max\{|f(v) - f(u)| : u \in N(v)\}$ where N(v) is the set of vertices adjacent to v, and the bandwidth of a graph $G = B_f(G) = \min\{ \max\{B_f(v) : v \in V\} \}$. In other words, the MBM problem is equivalent to the graph node re-labeling problem such that the length of the longest edge is minimized when the vertices are ordered on a line. The two-mode clustering method closely resembles to seriation, graph node labeling, and graph clustering, which, in graph theory parlance, essentially means minimizing the edges running between the graph partitions.

3.2 Parallel GA-based Two-mode Clustering Algorithm

The function of parallelism in our algorithm is based on the principle that when multiple processing cores simultaneously execute the same task with different inputs, faster results can be obtained. The data and code are sent to each of the cores, and the GA is run in each of the cores with different values of input parameters. As the algorithm depends on the random initial population, it becomes necessary to experiment with different random populations. In sequential algorithm, this is usually done by executing all the epochs in a single core. The parallel GA is designed based on a master/worker architecture, where the master send tasks to the workers (cores), who execute the tasks, and returns the results to the master, which aggregates the results, and select the the minimum bandwidth. Seed value is used to prevent the workers from replicating each others results.

The classical MBM algorithm was designed for symmetric, one-mode, sparse matrix. We propose the following algorithms to make it suitable for non-symmetric, two-mode, real valued matrix. As the fitness function in classical MBM is applicable to any scalar fields, we can use the same fitness function on real valued entries. To adapt the classical MBM for non-symmetric two-mode matrix, we obtained the row and column wise covariance matrices (algorithm 1 step-1 and step-3 respectively), followed by the dot product in step-5.

3.3 Algorithm for two-mode clustering

Algorithm 1. Two-mode Clustering

1: Obtain row order covariance matrix (P) by $X \cdot X^T$
2: Apply parallel GA-based bandwidth reduction algorithm on P to construct P_B
3: Obtain column order covariance matrix (Q) by $X^T \cdot X$
4: Apply parallel GA-based bandwidth reduction algorithm on Q to construct Q_B
5: Obtain two-mode cluster by $P_B \cdot X \cdot Q_B$

3.4 Parallel GA-based Bandwidth Reduction Algorithm

Algorithm 2. Parallel GA-based Bandwidth Reduction

1: Input: data matrix X = ($x_{i,j}$)
2: seed value
3: maxEpochs E
4: numberOfCores C
5: Population = randomInitialPopulation()
6: currentEpoch = 0
7: previousBandwidth = ∞
8: bestBandwidth = getBestBandwidth(Population)
9: Output: Permuted set of rows and columns
10: spawn C workers
11: for each i ∈ C do
12: for each j ∈ E/C
13: while (currentEpoch ≤ maxEpochs) and (bestBandwidth < previousBandwidth) do
14: previousBandwidth = getBestBandwidth(Population)
15: newPopulation = recombine(Population)
16: mutate(newPopulation)
17: prune(newPopulation)
18: Population = select(Population, newPopulation)
19: bestBandwidth = getBestBandwidth(Population)
20: currentEpoch = currentEpoch + 1
21: end while
22: end for
23: end for
24: Master aggregate the results and selects the permuted set of rows and columns which gives the minimum bandwidth.

4. EXPERIMENTAL ANALYSIS

Experiment has been performed to calculate speedup. Speedup is defined as the ratio of the execution time of the sequential algorithm and the execution time of the parallel algorithm with multiple cores. When the speedup is equal to the number of cores then the algorithm has a perfect scalability. We used the GA package for R [18], following the Simple Network of Workstations (SNOW) functionality of parallelism. The experiment was run on an AMD Phenom (tm) II X6 1100T processor 3.3 GHz computer with 8 GB of memory running on Windows 7 Professional. We applied our algorithm on the gene expression data [19] having 798 genes and 17 conditions.

The experiments were run with a population of 150 over 150 generations with 0.2 probability of mutation, and default values for other parameters. The consistency of the results were checked by taking the average value after running the GA for five times. The average execution time taken on one, two, three, four, five, and six cores respectively were 50010, 20625, 15015, 12520, 8700, and 7705 seconds. The speedup of our algorithm increased from 1 to 2.42, 3.33, 4.0, 5.74, and 6.49 for one, two, three, four, five, and six cores respectively (Fig. 1), which indicates the potential for further research on this approach. We also found that the GA converges to a near optimal solution in the initial generations, with marginal improvements in the later stages. Although it can be argued that global optimum is not guaranteed in GA, the probability of attaining the global optimum increases with the increase in the number of generations. However, with the increase in the number of generations, the amount of improvement in the objective function value decreases, thereby requiring a trade-off between the accuracy required and the number of generations. The problem of higher cost of computation due to the larger number of generations for optimal results has been, to certain extent, addressed by our approach of utilizing multiple cores. Overall, by looking at the increasing speedup from 1 to 6 cores, we believe that the proposed approach deserve further study towards a viable alternative to obtain two-mode clustering with speedup and ease of implementation.

The permutation approach used here to obtain two-mode cluster is free from several limitations of classical clustering approaches. Unlike the centroid-based clustering approach, it is not required in our approach to determine the number of clusters before clustering. Unlike the distribution-based clustering approach where the user has to choose a model to fit the data and optimize the model, which opens up the possibility of loss of generalization and overfitting, our permutation based approach is free from choosing any particular data model, and thus less possibility of overfitting.

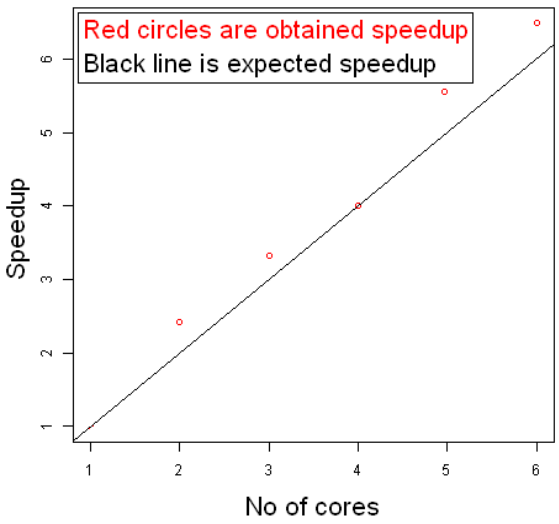

Figure 1: Performance on different number of cores.

5. CONCLUSIONS AND FUTURE DIRECTIONS

As the Moore's law which states that chip performance would double every 18 months is expected to slow down, and increasingly more cores being available in processors, there is a need for a good programming framework to take advantage of the mutiple cores and obtain speeding up big data analytics. We have demonstrated a simple approach where users can obtain linear speedup by increasing the number of cores. We addressed the scalability problem for two-mode clustering on multicore processor and showed that our approach can provide speedup compared to serial implementation. We adapted the classical MBM algorithm and used parallel GA as an optimizer, and obtained speedup increase from 1 to 2.42, 3.33, 4.0, 5.74, and 6.49 for one, two, three, four, five, and six cores respectively. This approach is free from several limitations of the classical clustering approaches. In future, we plan to further study the scalability of our approach on larger number of cores and conduct more extensive experiments on larger data sets and evaluate the system.

6. ACKNOWLEDGEMENTS

This research is supported by the European Regional Development Fund through the EXCS, Estonian Science Foundation grant PUT 360, Target Funding theme SF0180008s12 and European Social Fund for Doctoral Studies and Internationalisation Programme DoRa.

7. REFERENCES

[1] Suderman, M., & Hallett, M. 2007. Tools for visually exploring biological networks. Bioinformatics, 23(20), 2651-2659.

[2] Chu, Cheng-Tao, et al. Map-reduce for machine learning on multicore. NIPS. Vol. 6. (2006).

[3] Borgatti, Stephen P., and Martin G. Everett. Network analysis of 2-mode data. Social networks 19.3 (1997): 243-269.

[4] Van Mechelen, Iven, Hans-Hermann Bock, and Paul De Boeck. Two-mode clustering methods: a structured overview. Statistical methods in medical research 13.5 (2004): 363-394.

[5] E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices In Proc. 24th Nat. Conf. ACM, pages 157–172, (1969).

[6] Hageman, J. A., van den Berg, R. A., Westerhuis, J. A., van der Werf, M. J., & Smilde, A. K. (2008). Genetic algorithm based two-mode clustering of metabolomics data. Metabolomics, 4(2), 141-149.

[7] Estivill-Castro, Vladimir (June 2002). Why so many clustering algorithms — A Position Paper. ACM SIGKDD Explorations Newsletter 4 (1): 65–75.

[8] Cheng, Y., & Church, G. M. (2000, August). Biclustering of expression data. In ISMB, Vol. 8, pp. 93-103.

[9] Dhillon, I. S. 2001, August. Co-clustering documents and words using bipartite spectral graph partitioning. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 269-274. ACM.

[10] Liiv, I. 2010. Seriation and matrix reordering methods: An historical overview. Statistical analysis and data mining, 3(2), 70-91.

[11] Schaeffer, S. E. 2007. Graph clustering. Computer Science Review, 1(1), 27-64.

[12] Esposito, A., Catalano, M. S., Malucelli, F., & Tarricone, L. 1998. A new matrix bandwidth reduction algorithm. Operations Research Letters, 23(3), 99-107.

[13] Hageman, J. A., Malosetti, M., & Van Eeuwijk, F. A. 2012. Two-mode clustering of genotype by trait and genotype by environment data. Euphytica, 183(3), 349-359.

[14] Zhang, P., Wang, J., Li, X., Li, M., Di, Z., & Fan, Y. 2008. Clustering coefficient and community structure of bipartite networks. Physica A: Statistical Mechanics and its Applications, 387(27), 6869-6875.

[15] Arabie, P., Schleutermann, S., Daws, J., and Hubert, L. 1988, Marketing Applications of Sequencing and Partitioning of Nonsymmetric and/or Two-Mode Matrices, in Data, Expert Knowledge and Decisions, Eds. W. Gaul, and M. Schader, Berlin: Springer-Verlag, 215-224.

[16] Fonseca, Y. C. F. A bipartite graph co-clustering approach to ontology mapping, Proceedings of the Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data. Colocated with the Second International Semantic Web Conference (ISWC-03), CEUR-WS.org, 2003.

[17] Madeira, S. C., & Oliveira, A. L. 2004. Biclustering algorithms for biological data analysis: a survey. Computational Biology and Bioinformatics, IEEE/ACM Transactions on, 1(1), 24-45. Chicago

[18] Scrucca, L. 2012. GA: a package for genetic algorithms in R. Journal of Statistical Software, Volume 53, Issue 4.

[19] Yeast expression matrix, URL accessed on 02-01-2014: http://arep.med.harvard.edu/biclustering/yeast.matrix