

Memory-Efficient Clustering Algorithms for Microarray Gene Expression Data

Kazuyuki Numata¹

numata@ims.u-tokyo.ac.jp

Hideo Bannai²

bannai@i.kyushu-u.ac.jp

Yoshinori Tamada³

tamada@ims.u-tokyo.ac.jp

Michiel de Hoon⁴

mdehoon@c2b2.columbia.edu

Seiya Imoto¹

imoto@ims.u-tokyo.ac.jp

Satoru Miyano¹

miyano@ims.u-tokyo.ac.jp

¹ Human Genome Center, Institute of Medical Science, University of Tokyo, 4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan

² Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University, 6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan

³ Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji, Kyoto 611-0011, Japan

⁴ Center for Computational Biology and Bioinformatics, Columbia University, 1130 St Nicholas Avenue, New York, NY 10032, USA

Keywords: memory-efficient algorithm, single-linkage clustering, microarray data

1 Introduction

The development of microarray technology produces vast amount of genome-wide gene expression data under different experimental conditions such as gene disruptions, shocks, cancer tissues and so on. Clustering analysis of microarray gene expression data is a standard method for microarray analysis, which has made significant impacts on biomedical research [3]. While there are a number of clustering softwares for microarray gene expression data such as Cluster 3.0 [2], most of them use $O(n^2)$ memory space algorithms which can cause execution failure of clustering for large data due to insufficient memory space. Therefore, in order to handle large size data containing more than 100,000 objects such as whole genome arrays, a memory-efficient algorithm needs to be developed. We present a new single-linkage hierarchical clustering algorithm which requires only $O(n)$ memory space, where n is the number of objects. We show the practical usefulness of the proposed algorithm by comparison with Cluster 3.0.

2 Method and Results

Although there exist several memory-efficient algorithms for single-linkage clustering [4, 6], we have developed a conceptually simple $O(n)$ memory space algorithm. The derivation of our algorithm is as follows: First, we have the following proposition:

Proposition: Let $G = \{g_1, \dots, g_n\}$ be the set of objects to be clustered and let A and B be subsets of G satisfying $A \cup B = G$ and $A \cap B = \phi$. The minimum distant between these subsets is defined by

$$d_{A,B} = \min (d(a, b)) | a \in A, b \in B \quad (1)$$

where $d(a, b)$ is the distance between a and b . Then, $d_{A,B}$ must be used as a node of the resulting tree of the single-linkage clustering for G .

Based on this proposition, we obtain the following algorithm for single-linkage clustering:

Algorithm: [A memory-efficient single linkage clustering algorithm]

STEP 1: Select an element g from G arbitrarily.

Let $A = \{g\}$ and let $B = G \setminus \{g\}$.

STEP 2: While $A \neq G$ repeat following operation:

STEP 2-1: Find $g' \in B$ where

$$g' = \operatorname{argmin}_{g''} (d(a, g'') | a \in A, g'' \in B)$$

STEP 2-2: Add g' into A and remove g' from B .

STEP 2-3: Store $d_{A,B}$ and the pair (g^*, g') ,

where $g^* \in A$ and $d(g^*, g') = d_{A,B}$.

After the algorithm is run, we sort the gene pair list by distance. The sorted list can be easily transformed into resulting tree of the single-linkage clustering.

To evaluate the effectiveness of the proposed algorithm, we conduct computational experiments. We compare the proposed algorithm with Cluster 3.0 version 1.27 in terms of the computational time for various numbers of objects. Fig. 1 shows part of the results. These results prove that our method is superior to Cluster 3.0 in terms of computational speed, and that our method works even for large data sets. We also implemented CLINK [1], which is an memory-efficient complete-linkage hierarchical clustering algorithm with $O(n)$ memory space, into Cluster 3.0 and examined its effectiveness.

3 Discussions

From the computational experiments, we confirm that our algorithm is able to execute single-linkage clustering for microarray gene expression data whose amount is large. In addition, our algorithm requires only $O(n^2)$ time whereas Cluster 3.0 version 1.27 requires $O(n^3)$ time. It is certain that this time complexity reduction speed up the clustering. We expect to further speed up the algorithm by calculating the distances in parallel using multiple processors. We examined an memory-efficient complete-linkage method, i.e. CLINK, and confirmed its usefulness. Since the resulting tree for large size data is also large, it is difficult to extract effective information from such a tree. Therefore, the development of a visualization tool for large sized microarray clustering results is needed for further analysis [5].

References

- [1] Defays, D., An efficient algorithm for a complete link method, *The Computer Journal*, 20:346–366, 1977.
- [2] De Hoon, M.J.L., Imoto, S., Nolan, J., and Miyano, S., Open source clustering software, *Bioinformatics*, 20(9):1453–1454, 2004.
- [3] Eisen, M.B., Spellman, P.T., Brown, P.O., and Botstein, D., Cluster analysis and display of genome-wide expression patterns, *Proc. Natl. Acad. Sci. USA*, 95(25):14863–14868, 1998.
- [4] Olson, C.F., Parallel algorithms for hierarchical clustering, *Parallel Computing*, 21:1313–1325, 1995.
- [5] Saldanha, A.J., Java Treeview - extensible visualization of microarray data, *Bioinformatics*, 20(17):3246–3248, 2004.
- [6] Sibson, R., SLINK: an optimally efficient algorithm for the single-link cluster method, *The Computer Journal*, 16(1):30–34, 1973.

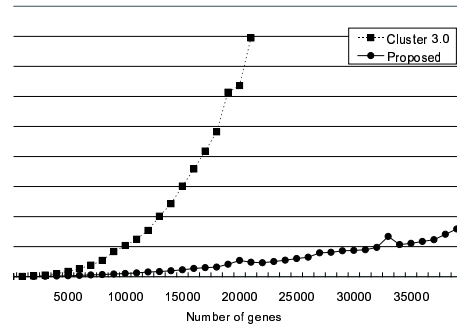


Figure 1: Comparison of calculation time between the proposed method and Cluster 3.0.