OMEGA: An Order-Preserving SubMatrix Mining, Indexing and Search Tool

Tao Jiang(✉), Zhanhuai Li, Qun Chen, Zhong Wang, Kaiwen Li, and Wei Pan

School of Computer Science and Technology,
Northwestern Polytechnical University, Xi'an 710072, China
jiangtao@mail.nwpu.edu.cn

Abstract. Order-Preserving SubMatrix (OPSM) has been accepted as a significant tool in modelling biologically meaningful subspace cluster, to discover the general tendency of gene expressions across a subset of conditions. Existing OPSM processing tools focus on giving a or some batch mining techniques, and are time-consuming and do not consider to support OPSM queries. To address the problems, the paper presents and implements a prototype system for OPSM queries, which is called OMEGA (Order-preserving subMatrix mining, indExinG and seArch tool for biologists). It uses Butterfly Network based BSP model to mine OPSMs in parallel. Further, it builds index based on prefix-tree associated with two header tables for gene expression data or OPSM mining results. Then, it processes exact and fuzzy queries based on keywords. Meanwhile, the vital query results are saved for later use. It is demonstrated that OMEGA can improve the effectiveness of OPSM batch mining and queries.

Keywords: Order-Preserving SubMatrix · Indexing and search · Tool

1 Introduction

DNA microarray enables simultaneously monitoring of the expression level of tens of thousands of genes over hundreds of experiments. Gene expression data on DNA microarrays can be viewed as an $n \times m$ matrix with n genes (rows) and m experiments (columns), in which each entry denotes the expression level of a given gene under a given experiment. Existing clustering methods do not work well for gene expression data, due to that most genes are tightly coexpression only under a subset of experiments, and are not necessarily expression at the same or similar expression level. Thus, it makes *Order-Preserving SubMatrix* (OPSM)

This work was supported in part by the National Basic Research Program 973 of China (No. 2012CB316203), the Natural Science Foundation of China (Nos. 61033007, 61272121, 61332006, 61472321), the National High Technology Research and Development Program 863 of China (No. 2013AA01A215), the Graduate Starting Seed Fund of Northwestern Polytechnical University (No. Z2012128).

[1,2], a special model of pattern-based clustering, as the popular tool to find meaningful clusters. In essence, an OPSM is a subset of rows and columns in a data matrix where all the rows induce the same linear ordering of the columns, e.g., rows g_2, g_3 and g_6 have an increasing expression level on columns 2, 7, 5, and 1. And OPSM cluster model focuses on the relative order of columns rather than the actual values. As the high-rate increasing of the numbers and sizes of gene expression datasets, there is an increasing need for the fast mining techniques to handle the massive gene datasets. Further, OPSM mining results are accumulated and not efficiently utilized, thus it is urgent to build a tool for biologists to find supporting rows or columns based on keyword queries, which plays an important role in inferring gene coregulated networks.

Traditional OPSM processing tools such as *BicAT* [1] and *GPX* [2] are developed for single machine, and cannot work well on parallel distributed platform. For example, how to reduce the communication time, workload of bandwidth and percent of duplicate results. And they are mainly focusing on providing batch OPSM mining techniques, and have very limited consideration on supporting OPSM search, even if *GPX* uses a graphical interface to drill down or roll up.

To solve the problems, we present and implement a prototype system for OPSM queries called *OMEGA* (Order-preserving subMatrix mining, indExinG and seArch tool). The major features of *OMEGA* can be described as follows:

(1) *OMEGA* provides a Butterfly Network based parallel OPSM mining framework [4]. Existing batch OPSM mining methods only employ a machine, but *OMEGA* utilizes multi-machines to mine OPSMs.

(2) *OEMGA* supports OPSM indexing and search. It builds index based on prefix-tree with two header tables, then processes queries based on keywords [3], meanwhile saves the vital query results for later use. It is demonstrated that *OMEGA* can improve the effectiveness of OPSM batch mining and queries.

2 The Architecture and Key Technologies of OMEGA

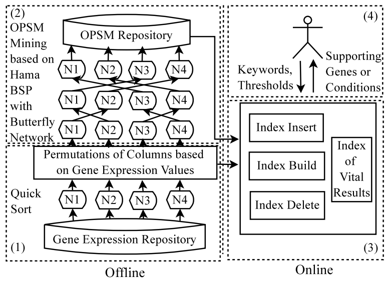
The architecture of *OMEGA* can be divided into four major shown in Fig. 1(a).

(1) Permutation of Columns. This module permutates columns based on expression values, under which row sequences are in ascending/descending order.

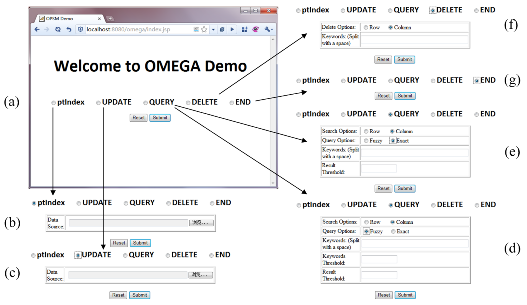
OPSM model focuses on the relative order of columns rather than actual values. By sorting row vectors and replacing entries with their corresponding column labels, data matrix can be transformed into a sequence database, and OPSM mining is reduced to a special case of sequential pattern mining problem with distinctive properties. Actually, we can use any sort method to permutate the columns of each row, and quick sort algorithm is employed in the paper.

(2) Parallel Mining OPSMs. This module employs Butterfly Network based BSP model for OPSM mining.

The number of nodes of Butterfly Network is N ($N = 2^n$, where n is the maximum number of super-steps). For simplicity, the names of nodes are denoted by integers which are from 0 to $2^n - 1$. In the ith super-step ($i \geq 1$), each


(2) OPSM Mining based on Hama BSP with Butterfly Network
OPSM Repository
N1 N2 N3 N4
Permutations of Columns based on Gene Expression Values
Quick Sort
Gene Expression Repository
(1)
Offline
(4)
Keywords, Thresholds
Supporting Genes or Conditions
Index Insert
Index Build
Index Delete
Index of Vital Results
(3)
Online

(a) System Architecture


Welcome to OMEGA Demo
ptIndex UPDATE QUERY DELETE END
Reset Submit
Data Source:
Delete Options: Row Column
Keywords: (Split with a space)
Search Options: Row Column
Query Options: Fuzzy Exact
Keywords Threshold
Result Threshold
(a) (b) (c) (d) (e) (f) (g)

(b) The User Interface of OMEGA

Fig. 1. System Architecture and User Interface

node firstly does local computation, then N nodes are divided into $(log_2 N)/2^{i-1}$ groups, where $1 \leq i \leq n$, i.e., each group has 2^i members which have continuous integers, further the members in each group are divided into 2 partitions, the members in first half partition communicate or transfer data with the nodes in the last half partition with 2^{i-1} steps, and vice versa, finally the nodes go into barrier synchronization. Once there are no data to transfer or the number of super-steps is equal to $log_2 N$, the computational work of the nodes on Hama will be stopped. For more details about the method, please refer to work [4].

(3) Indexing of Datasets. The module uses prefix-tree with two header tables to index the permutation of columns or OPSM mining results.

A compact index can be designed based on three observations described in work [3]. We use dataset in Table 1 to show how to build index in Fig. 2.

Table 1. Dataset

Row No.	Column No.
1,2,5	VI,III,I,VIII,XVI
3,6,9	VI,III,I,II,XIII
7,10,11	VI,II,III
4,8,12	III,II,XVI
4,6	VI,III,I,VIII,XVI

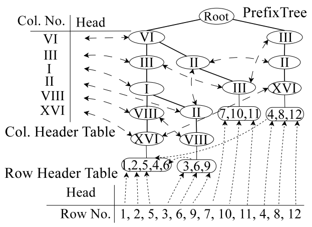


Fig. 2. pIndex

First, one may create the root of a tree, labelled with "null". Then, scan OPSM dataset. The scan of 1st OPSM leads to the construction of 1st branch of the tree: <VI, III, I, VIII, XVI>. Notice that we keep item order in an OPSM. And add a leaf node (1, 2, 5). For the 2nd OPSM, since its item list <VI, III, I, II, XIII> shares a common prefix <VI, III, I> with existing path <VI, III, I, VIII, XVI>. One new node (II) is created and linked as a child of (I), and another new node (XIII) is created and linked as the child of (II). The leaf node records row No. <3, 6, 9>. For other OPSMs, it uses the same method to build.

A column header table in Fig. 2 is built in which the order is conducted based on the occurrences of items from left to right and from top to bottom, and each item points to its occurrence in the tree via a column head of node-link. Nodes with the same column No. are linked in sequence via such bidirection node-links.

A row header table in Fig. 2 is built in which the order is conducted based on the occurrences of row Nos from left to right, and the tree nodes which have the same row No. will be saved in one hash set. For the sake of clarity, nodes with the same row No. are linked in sequence via a row head of node-link.

(4) OPSM Queries. Due to space limit, we only show one type of query.

For fuzzy queries on conditions FQ_c, first rotate the 1st element of keywords. Then, locate column keywords with column header table, from located node to tree root. If the number of keywords in the same branches is above length threshold τ, get gene names in the leaves. And test whether the number of gene names above size threshold δ, if true, add keyword set (gene name) as key (value) into final results. Otherwise, test until each keyword as first element one time.

When getting query result, and if query time is above a threshold, we save it in memory. If the memory is scarcity, we remove the longest unused result. When it does queries, we first test whether the memory saves the query results, if true, we return query result soon. For more details, please refer to work [3].

3 Demonstration

The gene expression repository used for demonstration consists of 6 documents (http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi). The offline part (permutation and mining) is conducted on Hama 0.4.0, and the online part (indexing and queries) is running on a PC with a 1.86GHz CPU, 2.9GB RAM, Ubuntu 14.04 system, and Firefox 28.0 web browser.

We use a web interface to demonstrate our system. Fig. 1(b) shows user interface of OMEGA. For more details, we provide a video to show the demonstration process (https://sites.google.com/site/jiangtaonwpu/). In the demonstration, we will show the permutation of columns, parallel OPSM mining, indexing of datasets, and searching examples under different configurations, such as different keyword and result thresholds.

References

1. Barkow, S., Bleuler, S., Prelić, A., Zimmermann, P., Zitzler, E.: BicAT: A biclustering analysis toolbox. Bioinformatics **22**(10), 1282–1283 (2006)
2. Jiang, D., Pei, J., Zhang, A.: GPX: interactive mining of gene expression data. In: Nascimento, M.A., Özsu, M.T., Kossmann, D., Miller, R.J., Blakeley, J.A., Schiefer, K.B. (eds.) VLDB, pp. 1249–1252. Morgan Kaufmann (2004)

3. Jiang, T., Li, Z., Chen, Q., Li, K., Wang, Z., Pan, W.: Towards order-preserving submatrix search and indexing. In: Renz, M., Shahabi, C., Zhou, X., Chemma, M.A. (eds.) DASFAA 2015. LNCS, vol. 9050, pp. 309–326. Springer, Heidelberg (2015)
4. Jiang, T., Li, Z., Chen, Q., Wang, Z., Pan, W., Wang, Z.: Parallel partitioning and mining gene expression data with butterfly network. In: Decker, H., Lhotská, L., Link, S., Basl, J., Tjoa, A.M. (eds.) DEXA 2013, Part I. LNCS, vol. 8055, pp. 129–144. Springer, Heidelberg (2013)