Sub-Matrix of Mining Order Structure in Gene Expression in The Order-Preserving

[1]KHAJA MOINUDDIN, [2]K.CHANDRA SHEKHAR
Associate Professor, Assistant Professor
Medak College of Engineering and Technology

Abstract- The Order-Preserving SubMatrices (OPSMs) capture consensus trends over columns shared by rows in a data matrix. the discovery of patterns in gene expression matrices, in which each element gives the expression level of a given gene in a given experiment. Most existing methods for pattern discovery in such matrices are based on clustering genes by comparing their expression levels in all experiments, or clustering experiments by comparing their expression levels for all genes. Our work goes beyond such global approaches by looking for local patterns that manifest themselves when we focus simultaneously on a subset *G* of the genes and a subset *T* of the experiments. Speci. cally, we look for *order-preserving submatrices* (OPSMs), in which the expression levels of all genes induce the same linear ordering of the experiments (we show that the OPSM search problem is NP-hard in the worst case). Such a pattern might arise, for example, if the experiments in *T* represent distinct stages in the progress of a disease or in a cellular process and the expression levels of all genes in *G* vary across the stages in the same way. We de. ne a probabilistic model in which an OPSM is hidden within an otherwise random matrix. Guided by this model, we develop an ef. cient algorithm for . nding the hidden OPSM in the random matrix. In data generated according to the model, the algorithm recovers the hidden OPSM with a very high success rate. Application of the methods to breast cancer data seem to reveal sign. cant local patterns.

I.Introduction

Matrices are a common data presentation in many applications, such as those involving data. the main approach taken for analyzing gene expression data is *clustering* (and variants thereof). Clustering methods have indeed proved successful in many contexts. There is a very large body of literature on clustering in general and on applying clustering techniques to gene expression data in particular. The following list of examples represents the viewpoint of the authors and is not comprehensive or representative. The pioneering paper (Eisen *et al.*, 1998) develops an initial approach to analyzing expression data, mostly adapting hierarchical clustering techniques for this purpose. These methods are successfully applied to yeast cell cycle data in Spellman *et al.* (1998). In Ben-Dor *et al.* (1999) and Sharan and Shamir (2000), more direct approaches to clustering are taken, using graph theoretic methods. Studies where combinations of clustering and classi. cation methods were applied are reported by Golub *et al.* (1999), Bittner *et al.* (2000), and Alizadeh *et al.* (2000). Surveys of analysis approaches appear in Bassett *et al.* (1999) and Gaasterland and Bekiranov (2000). A major drawback of clustering, however, is the dif. culty in identifying patterns that are common to only a part of the expression data matrix. Based on general understanding of cellular processes, we expect subsets of genes to be co regulated and co-expressed under certain experimental conditions, but to behave almost independently under other conditions. Discovery of such local expression patterns may be the key to uncovering many genetic pathways that are not apparent otherwise. It is, therefore, highly desirable to move beyond the clustering paradigm and to develop algorithmic approaches capable of discovering local patterns in microarray data. Ben-Dor *et al.* (2001) and Xing and Karp (2001) discuss approaches to unsupervised identification of patterns in expression data that distinguish two subclasses of tissues on the basis of a supporting set of genes. “Supporting” here means that high-success-rate classification can be performed based on this set of genes. Ben-Dor *et al.* (2001) describe subclasses that correlate with lymphoma prognosis, based on analyzing data reported (Alizadeh *et al.*, 2000). The present work extends the class discovery task to a progression of more than just two stages. The . rst to consider local patterns (*biclusters*) in gene expression data sets were Cheng and Church (2000). Their biclusters are based on uniformity criteria, and they use a greedy algorithm to . nd them. The *plaid model* (Lazzeroni and Owen, 2000) is a statistical model for gene expression and other data. The plaid model describes two-sided clusters where overlap is allowed. Like our model, their two-sided clusters are not necessarily supported on the entire set of either genes or tissues. The plaid model seeks submatrices that have almost uniform entries. It also affords the identi. cation of submatrices where, over a prescribed set of tissues, genes differ in their expression levels by an almost constant vector. Our work focuses on the uniformity of the relative order of the tissues rather than on the uniformity of the actual expression levels as in the plaid model. This approach is potentially more robust to the stochastic nature of the expression levels and to the variation caused by the measurement process. In this work, we address the identi. cation and statistical assessment of coexpressed patterns for large sets of genes. For example, in expression data that comes from a population of patients (such as in Bittner *et al.* [2000]), it is reasonable to expect that each individual is in a particular stage of the disease. There is a set of genes that are coexpressed with this progression, and we therefore expect the data to contain a set of genes and a set of patients such that the genes are identically ordered on this set of patients. The same situation occurs when considering data from nominally identical exposure to environmental effects, data from drug treatment, data representing some temporal progression, etc. In many cases, the data contains more than one such pattern. For example, in cancer data, patients can be staged according to the disease progression, as well as according to the extent of genetic abnormalities. These two orders on some subset of tissues are not necessarily correlated. Therefore, even in data where some nominal order is given a priori, we are seeking related or unrelated hidden orders and the sets of genes that support them.

II. Related work

The advent of high throughput data generation techniques have increased not only the number of objects collected in databases, but also the number of attributes describing these objects. The resultant datasets are often referred to as high dimensional. Clustering high dimensional data using traditional algorithms has suffered from the fact that many attributes may be irrelevant and can thus mask clusters located in some subspaces. Subspace clustering algorithms have recently been proposed to solve this problem. They search for clusters in subspaces formed by relevant attributes [1]. Among various subspace clustering models, one was designed to mine a set of objects which show identical attribute order, called *order preserving cluster* (OPC) [6]. We will give a formal description of this model in next section. This model originally attracts researchers’ interests because of its important utility in gene expression data analysis. Based on the understanding of cellular processes, it is a general belief that some subsets of genes may be co-expressed under certain experimental conditions, but behave independently under other conditions. The readout of a DNA chip containing n genes consists of n real numbers that represent the expression level of each gene, either as an absolute or as a relative quantity (with respect to some reference). When the readouts for m experiments (tissues) are combined, each gene yields a vector of m real numbers. To make our results independent of the scaling of the data, we consider only the relative ordering of the expression levels for each gene, as opposed to the exact values. This motivates us to consider the permutation induced on the m numbers by sorting them. Thus, we view the expressed data matrix, D, as an n-by-m matrix, where each row corresponds to a gene and each column to an experiment. The m entries in each row are a permutation of the numbers f1,…..,mg. The .i; j/ entry is the rank of the readout of gene i in tissue j , out of the m readouts of this gene. Typical values for n and m are in the ranges 500 · n · 15,000 and 10 · m · 150. We are seeking a biological progression that is represented as a “hidden” k-by-s submatrix G £ inside the data matrix D. The k genes from G are coexpressed in the s tissues from T . This means that the expression levels of all the genes in G move up and down together within the set T . Consider, for example, three genes g1; g2; g3 2 G and the three rows in D corresponding to g1; g2; g3, restricted to the columns in T D ft1,….., ts g. The s ranks in each row correspond to a partial permutation of f1,……,mg. By projecting the three partial permutations on the subset f1,…. sg, we get three identical permutations.

III. STOCHASTIC MODEL

We model the gene expression data set by a random data matrix D in which an unknown order-preserving sub-matrix G£T has been planted. The process of generating a data matrix with a planted order-preserving submatrix consists of three stochastic steps. First, we choose at random the indices for the planted rows and columns. Second, we choose a random ordering for the planted columns. Finally, we assign ranks at random to the data matrix in a way which is consistent with the planted submatrix. More formally, the parameters of the stochastic process are n (number of genes), m (number of experiments), s (size of T), and p (probability that a row i is in the subset G).

1. To determine the set of genes G, toss iid coins Xi for every i (i D 1, 2,….. , n), with probability p of coming up heads (Xi D 1). The set G is the set of indices i with Xi D 1, and the *expected size* of G equals p ¢ n. For the set of experiments, we choose a subset T ½ f1,…..,mg of size s uniformly at random.
2. Pick uniformly at random a linear ordering t1, t2,.. ; ts of the elements of T .
3. For every row i, assign the m entries in the i-th row of D independently by a random permutation of f1,……,mg.
4. For each row i with Xi D 1 (i 2 G), rearrange the ranks in the columns corresponding to T : The entry in column t1, D[i; t1], will be assigned the lowest rank among these s entries, the entry in column t2 will be assigned the second rank among the entries corresponding to T , and so on. The entry D[i; ts] will be assigned the highest rank among the entries of T .

IV. DISCOVERING MINING ORDER STRUCTURE IN GENE EXPRESSION

At the completion of these three steps, the data matrix D with the planted submatrix G£T is determined. Note that in addition to the set of planted rows, G, every nonplanted row has a probability of 1 s! to satisfy the same ordering constraints as the planted rows. Given D and T , those “spuriously planted” rows are indistinguishable from the “genuinely planted” rows. Thus, the algorithmic goal is, for a given D, to recover the set of planted columns T and their planted linear order ¼. The set of rows supporting this model (“genuinely planted” together with the “spuriously planted”) is then uniquely de. ned.

Subspace Clustering, Co-Clustering, and Bi-clustering

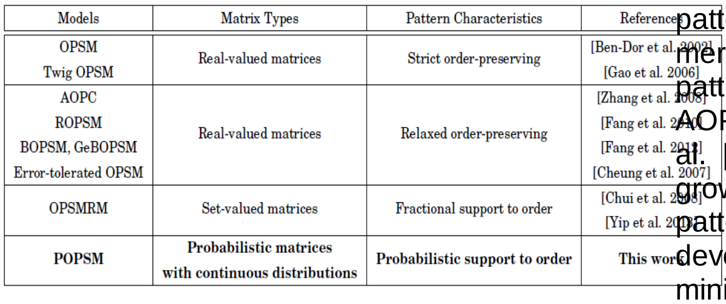
Studies about subspace clustering are motivated by the observation that the data points which are irrelevant in high dimensional space may be well clustered or correlated in lower dimensional subspaces

[Parsons et al. 2004; Kriegel et al. 2009]. These studies aim to identify a subset of attributes or features that form a subspace, and find the data points that form a cluster over the subspace. If high-dimensional data are organized as a matrix, the subspace features together with the clustered data points correspond to the submatrix patterns in the matrix. However, different from the order preserving relationship required by the OPSM model, the criteria for subspace clustering are usually the Euclidean distance between data points [Agrawal et al. 1998; Aggarwal et al. 1999; Moise and Sander 2008], the linearity correlation among data points [G¨unnemann et al. 2012], the density of clusters [Kailing et al. 2004], or the statistical significance of clusters [Moise and Sander 2008]. Another stream of submatrix pattern mining work aims to partition a matrix into grid-distributed disjoint sub-matrices such that the subset of rows and the subset of columns in each individual sub-matrix are expected to be highly correlated with each other. This category of problems is usually called co-clustering, and has mainly been studied in recommender systems and text mining [Daruru et al. 2009; Banerjee et al. 2004; Dhillon et al. 2003; Pan et al. 2008; Long et al. 2005; Ji et al. 2012]. The grid distribution structure of sub-matrix patterns required by the co-clustering methods avoids the explosion of the number of mined patterns, which however is rather restrictive. In order to achieve a grid structure that optimizes the overall pattern scores, the quality of a part of sub-matrix patterns usually has to be sacrificed. Cheng and Church [Cheng and Church 2000] first adopted the sub-matrix mining methods to analyze the biological gene expression data and called the problem bi-clustering. Bi-clustering methods aim to formulate different sub matrix models so as to capture the biological correlations among a subset of genes and a subset of conditions. Madeira et al. [Madeira and Oliveira 2004] classified existing sub matrix models into four categories: submatrix with constant values [Busygin et al. 2002], sub matrix with constant rows or constant columns [Getz et al. 2000; Pandey et al. 2009; Gupta et al. 2010], sub matrix with coherent values [Cho et al. 2004; Pontes et al. 2010], and sub matrix with coherent evolutions [Murali and Kasif 2003; Tanay et al. 2002; Gupta and Aggarwal 2010; Li et al. 2009; Madeira et al. 2010]. The OPSM model belongs to the fourth category according to this classification.

V. Mining Order-Preserving Sub-matrices

The Order-Preserving Submatrix (OPSM) model, proposed by Ben-Dor et al. [Ben-Dor et al. 2002], aims to capture the fact that the entry values of a set of rows exhibit the same trend under a set of columns. A comparative study conducted by Preli´c et al. [Preli´c et al. 2006] showed that, compared to five other coherent-value or coherent evolution sub matrix models [Cheng and Church 2000; Preli´c et al. 2006; Tanay et al. 2002; Murali and Kasif 2003; Ihmels et al. 2002], the OPSM model better captures the association of correlated genes and conditions and promotes the discovery of a larger fraction of biologically significant patterns. However, it is also recognized that the OPSM model may be too strict to be practical, since real gene expression data are noisy and the identical trend is usually hard to preserve [Ben-Dor et al. 2002]. To address this problem, various noise-tolerant OPSM models have been proposed in literature, such as the Approximate Order-Preserving Cluster (AOPC) model [Zhang et al. 2008], the Relaxed Order-Preserving SubMatrix (ROPSM) model [Fang et al. 2010], the Bucket Order-Preserving SubMatrix (BOPSM) model [Fang et al. 2012], the Generalized BOPSM (GeBOPSM) model [Fang et al. 2012], and the error-tolerated OPSM model [Cheung et al. 2007]. The AOPC model relaxes the condition that all the rows in an OPSM should induce the same linear order of columns, and only requires a pre-specified fraction of rows to induce the same linear order. The ROPSM model further relaxes the AOPC model, and allows all the rows in an ROPSM pattern to induce orders similar to the backbone order of the pattern. The BOPSM model tries to capture the consensus staged trend of a set of rows over a set of columns, and requires every row in a BOPSM pattern to induce an identical bucket order (i.e., an order of sets). The GeBOPSM model is a generalization of AOPC, ROPSM, and BOPSM. It allows the rows in a GeBOPSM pattern to induce bucket orders which are similar to the backbone order of the GeBOPSM pattern. The error-tolerated OPSM model still requires all the rows in a pattern to induce an identical order of a set of columns but allows the entry values of a row under two adjacent columns to violate the ordering relationship within a pre-specified error threshold. However, due to different scalings of expression values of different genes, it is difficult to set a proper absolute error threshold to guarantee that all the rows in a pattern still follow a consensus varying trend. While the OPSM model and several other noise-tolerant OPSM models are all defined based on real-valued data matrices, an alternative way to address the issue of noisy data is to keep a set of replicates for each entry in the matrix, and such data matrices can be presented as set-valued matrices [Hughes et al. 2000; Nguyen et al. 2010; Ideker et al. 2001]. In a set-valued matrix, it is actually assumed that the set of replicates in every entry are equally likely to be observed. If an entry stores a set of k replicates, all of them have an equal probability of 1 k . Therefore, a set-valued matrix can be regarded as a probabilistic matrix with discrete distribution. Based on the setvalued matrices, an OPSM with Repeated Measurement (OPSMRM) model [Chui et al. 2008; Yip et al. 2013] was introduced, where a fractional support measure is adopted to evaluate the extent to which a row supports a linear order. However, when the number of replicates is small, as the noisy and true replicates have equally large probabilities, the fractional support is easily affected by one or two noisy replicates. On the other hand, when the number of replicates grows, the cost of computing the fractional support increases sharply, which greatly degrades the performance of mining OPSMRM patterns. Continuous distributions such as normal distribution are known to be effective for smoothing out the influence of noise in scientific experiments, and thus are commonly adopted to infer the error model of scientific data. For example, the observational error in gene expression analysis, which may be caused by instrumental limits or measurement errors, is assumed to follow a normal distribution [Hughes et al. 2000; Nguyen et al. 2010; Chia and Karuturi 2010]. Thus, a gene expression matrix can be presented as a probabilistic matrix with normal distribution. Probabilistic matrices are also a natural representation of the data arising from many sensor applications such as RFID tracking systems [R´e et al. 2008]. For example, when a user is detected at a particular RFID detection site within a time range, such trace data can then be represented as a probabilistic matrix with uniform distribution. The POPSM model proposed in this paper is defined based on such probabilistic matrices with continuous distributions. We summarize the characteristics of the OPSM model and its variants in Table I. Ben-Dor et al. proved that mining OPSM patterns is an NP-complete problem [Ben- Dor et al. 2002]. Then, they proposed a model-based method which aims to mine the best OPSM in terms of the statistical significance, since patterns with high statistical significance are regarded more likely to be biologically significant. Their method keeps a limited number of partial models which are smaller OPSM patterns. Then, it expands the partial models into larger and thus more statistically significant patterns. Their method, however, is heuristic-based, and the significance of the mined OPSM patterns is very sensitive to the selection of the partial models. Trapp et al.

Table. 1 OPSM Related Models



Models	Matrix Types	Pattern Characteristics	References
OPSM Twig OPSM	Real-valued matrices	Strict order-preserving	[Ben-Dor et al. 2002] [Gao et al. 2006]
AOPC ROPSM BOPSM, GeBOPSM Error-tolerated OPSM	Real-valued matrices	Relaxed order-preserving	[Zhang et al. 2008] [Fang et al. 2010] [Fang et al. 2012] [Cheung et al. 2007]
OPSMRM	Set-valued matrices	Fractional support to order	[Chui et al. 2008] [Yip et al. 2013]
POPSM	**Probabilistic matrices with continuous distributions**	**Probabilistic support to order**	**This work**

[Trapp and Prokopyev 2010] and Humrich et al. [Humrich et al. 2011] both made use of integer programming techniques and proposed similar methods for mining the maximal OPSM pattern. While Ben-Dor et al., Trapp et al., and Humrich et al.'s work all aimed to mine a single optimal OPSM pattern, Liu et al. [Liu and Wang 2003] proposed a tree-based OPSM mining method, called OPC-Tree, to exhaustively mine all the OPSM patterns that satisfy some size thresholds. However, when the number of columns in the data matrix increases, the size of the tree grows extremely large, and the performance of pattern mining is greatly degraded. Cheung et al. [Cheung et al. 2007] adopted the sequential pattern mining method in [Agrawal and Srikant 1995; Srikant and Agrawal 1996], and developed an Apriori-based method to exhaustively mine OPSM patterns that satisfy some size thresholds. A post-processing solution was additionally designed to combine mined OPSM patterns into error-tolerated OPSM patterns [Cheung et al. 2007]. Gao et al. proposed a KiWi framework in [Gao et al. 2006; Gao et al. 2012] to mine twig OPSM patterns, which contain a large number of columns and very few rows. The framework expands a limited number of linear orders of columns in a breadth-first manner, and applies very strong conditions for pruning those linear orders that are less likely to grow into twig OPSM patterns. Their method is shown to be efficient but valid twig OPSM patterns may also get pruned. Zhang et al. [Zhang et al. 2008] adopted a similar pattern merging strategy as in [Cheung et al. 2007] to mine AOPC patterns. Taking a set of OPSM patterns as input, The AOPC mining method merges pairs of OPSM patterns into AOPC patterns in a greedy way until no more AOPC patterns can be generated. Fang et al. [Fang et al. 2010] proposed a pattern growth method, which expands seed OPSM patterns into ROPSM patterns. Later, they developed an Apriori-based method for mining BOPSM patterns and utilized the anti-monotonic property of the BOPSM model to control candidate generation [Fang et al. 2012]. However, the BOPSM model is defined based on the real-valued matrices and the rows in the BOPSM patterns are assumed to induce consensus bucket orders. Thus, the proof of the anti-monotonic property of the BOPSM model is different from that of the anti-monotonic property of the POPSM model. An efficient prefix-tree structure was designed to maintain the BOPSM patterns, which motivates us to employ the CandTree to compactly organize the POPSM patterns in this work. The OPSMRM mining method [Chui et al. 2008; Yip et al. 2013] also follows the Apriori-based framework, and consists of the following two steps. First, an Apriori based method is taken to exhaustively mine frequent orders of columns. An order is regarded frequent if the sum of fractional supports contributed by all the rows exceeds a certain threshold. In this step, the anti-monotonic property of the fractional support measure is made use of. However, as the fractional support is defined based on the set valued matrices, its computation method and the proof of the anti-monotonic property are different from our work that involves the notion of probabilistic support. Then, for each frequent order, the set of rows whose fractional supports to it satisfy another inclusion threshold are picked. The selected rows, together with the columns involved in the order, form an OPSMRM pattern. The inclusion threshold plays the similar role as the support threshold in our POPSM model. The OPSMRM mining process, however, has the following implications. In the first step, although the fractional support of every row with respect to an order is small, this order may still be regarded as frequent due to a large enough sum of the fractional supports contributed by all the rows. Then in the second step, a frequent order may fail to lead to a valid OPSMRM pattern if none of the rows has a large enough fractional support that satisfies the inclusion threshold. Another possibility is that, very few rows have large enough fractional supports with respect to the frequent order, and this order finally leads to a very small and hence statistically insignificant patterns.

VI. Sequential Pattern Mining

If a data matrix is transformed into a set of attribute (i.e., column label) sequences ordered by their values in every row, the matrix can be viewed as a transaction database with a collection of sequences. Accordingly, the OPSM mining problem is converted to a frequent sequential pattern mining problem [Agrawal and Srikant 1995; Srikant and Agrawal 1996]. However, due to some unique properties of the OPSM mining problem, using frequent sequential pattern mining methods for mining OPSM patterns is not satisfactory from the efficiency view point. First, each attribute appears at most once in each sequence. Second, as the data matrices in an OPSM mining application are usually very dense, the transformed sequences are also dense, in the sense that every attribute may appear in most of the sequences. As a result, the searching space of the depth-first sequential pattern mining methods would be extremely large, while few candidates can be pruned in the first few rounds of the breadth-first pattern mining methods. Liu et al. [Liu and Wang 2003] took into account the characteristics of the OPSM mining problem, and proposed an OPSM mining method called OPC-Tree, which improves the basic techniques of a well-known efficient frequent sequential pattern mining method called PrefixSpan [Pei et al. 2001; Pei et al. 2004]. The OPC-Tree was shown to outperform PrefixSpan in mining OPSMs. Agrawal et al. proposed an Apriori-based sequential mining method [Agrawal and Srikant 1995; Srikant and Agrawal 1996], based on which the BOPSM mining method [Fang et al. 2012] and the OPSMRM mining method [Chui et al. 2008] were respectively developed. The PROBAPRI method proposed in this work also adopts the Apriori-based framework, but we apply it to dealing with the probabilistic data with continuous distributions. Kum et al. [Kum et al. 2003] studied the problem of mining approximate frequent sequential patterns. They first clustered input sequences into disjoint sets, and then looked for consensus patterns within each cluster. However, unlike the support requirement for mining OPSM patterns, the consensus patterns were evaluated based on their global support, and thus a frequent consensus pattern is not necessarily supported by a large enough number of sequences. Aggarwal et al. [Aggarwal et al. 2009], Muzammal et al. [Muzammal and Raman 2011] and Zhao et al. [Zhao et al. 2012] studied the problem of mining frequent patterns or frequent sequential patterns in the context of uncertain data. However, the uncertain data are modeled as discrete values rather than continuous distributions.

VII. Evolution

we report the outcomes of these runs. For simulated data, all our datasets were 1,000-by-50 matrices. The number of planted columns s varied over the . ve values s D 3; 4; 5; 7; 10. The number of rows in G was determined by • ipping a p biased coin per row, where p varied over the four values p D 0:025; 0:05; 0:075; 0:1. Table 2 reports the probabilities that the algorithm recovers correctly the set of planted columns and their correct internal order. (All these probabilities are over . nite spaces, representing the proportion of certain cases out of the total number of cases.) Each entry of the table is based on one hundred random datasets. The running time of the algorithm (for one dataset) is 23 seconds (running in Matlab on a 500 MHz PC). In cases where the algorithm fails to recover the planted submatrix, we compared the size (number of rows) of the recovered submatrix to the size of the planted submatrix. For certain values of the parameters, the algorithm recovered submatrices that are larger than the planted one. Clearly, those cases should not be considered as failures of the algorithm, but rather as indication that for the simulation parameters applied there is no hope of recovering the planted submatrix. We report in Table 3 the failure rate of the algorithm, i.e., cases in which the search returned a smaller (less signi. cant) submatrix than the planted one. The success rate of the algorithm (proportion of cases where the algorithm produced a submatrix at least as large as the planted one) is obviously 1 minus the failure rate. It is typically very high and was below 0:5 for only one entry in the parameters' space. The maximal failure rate (0:57) occurred for s D 5 and p D 0:025. Our interpretation is that 25-by-5 OPSMs do not occur at random. However, the 25 (expected) planted rows do not give enough advantage to the correct partial model (1,1); it is *not* among the top ` D 100 pairs, and therefore the submatrix will not be recovered. Increasing ` would clearly improve results but will come at a cost of a higher running time. We have kept ` to low values to allow extensive simulations. Running the same parameter with ` D 500, we get a success rate of 0:7. For real data (that needs to be analyzed only once), we can afford much longer running time than the 23 seconds our algorithm currently utilizes on problems of this size, so it would be possible to set a much higher value.

VIII. Conclusion

In this paper we . nd submatrices G £ T in which the rows have a signi. cant tendency to be similarly ordered. It is perhaps too optimistic to expect submatrices to be identically ordered as the OPSM model requires, . rst because biological patterns are not always neat, and secondly because of error in the microarray measurements. One way to relax the requirement is to introduce an additional *corruption probability* ® which will in• uence the generation of the probabilistic model. As before, the model generates a hidden submatrix G £ T with a speci. ed ordering of its columns. But now each entry of the hidden submatrix is exempted from respecting the ordering with probability ® and, in each row, only the entries that have not been exempted are permuted according to the prescribed ordering. Another interesting relaxation is to require that, in each row in the submatrix G £ T , the ordering of the ranks is compatible with a given partial ordering of the columns. Of particular interest are *layered partial orderings*, in which T is partitioned into subsets T1, T2,….., Tr and, for i D 1; 2,……, r ¡ 1, every element of Ti
precedes every element of TiC1. Each of the sets Ti can be interpreted as a set of tissues representing a stage in a cellular process or in the progression of a disease. One of the major design goals of our algorithm was speed, especially as we tested it by running numerous simulations. For real biological data, which takes more time to generate than simulated data, we would be willing to sacrifice some speed for higher accuracy and reliability. One way to achieve this is to start with partial models of order .2; 1/, instead of .1; 1/ as was done here. This increases (by a factor of m) the number of models that are initially examined, but reduces the chances that the correct partial model of order

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Proc. VLDB '94, pages
487–499, September 1994.

[2] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and
implication rules for market basket data. In Joan Peckham, editor, SIGMOD 1997, Proceedings ACM
SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA,
pages 255–264. ACM Press, 05 1997.
[3] Bart Goethals and Mohammed J. Zaki. Advances in Frequent Itemset Mining Implementations: Report
on FIMI'03. SIGKDD Explor. Newsl., 6(1):109–117, 2004.
[4] Gosta Grahne and Jianfei Zhu. Efficiently using prefix-trees in mining frequent itemsets. In Proceedings
of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, November 2003.
[5] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In Proc.
SIGMOD '00, pages 1–12, 2000.
[6] J. Liu, Y. Pan, K. Wang, and J. Han. Mining Frequent Item Sets by Opportunistic Projection. In Proc.
2002 Int. Conf. on Knowledge Discovery in Databases (KDD'02), Edmonton, Canada, 2002.
[7] Claudio Lucchese, Salvatore Orlando, Paolo Palmerini, Raffaele Perego, and Fabrizio Silvestri. kdci:
a multi-strategy algorithm for mining frequent sets. In Proceedings of the IEEE ICDM Workshop on
Frequent Itemset Mining Implementations, November 2003.
[8] S. Orlando, P. Palmerini, R. Perego, and F. Silvestri. Adaptive and resource-aware mining of frequent
sets. In Proc. The 2002 IEEE International Conference on Data Mining (ICDM '02), pages 338–345,
2002.
[9] P. Palmerini, S. Orlando, and R. Perego. Statistical properties of transactional databases. In Proceedings
of the 2004 ACM symposium on Applied computing, pages 515–519. ACM Press, 2004.
[10] J. S. Park, M.-S. Chen, and P. S. Yu. An Effective Hash Based Algorithm for Mining Association Rules.
In Proc. of the 1995 ACM SIGMOD Int. Conf. on Management of Data, pages 175–186, 1995.
[11] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets
for association rules. In Proc. ICDT '99, pages 398–416, 1999.
[12] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Efficient mining of association rules
using closed itemset lattices. Information Systems, 24(1):25–46, 1999.
[13] J. Pei, J. Han, H. Lu, S. Nishio, and D. Tang, S. amd Yang. H-Mine: Hyper-Structure Mining of
Frequent Patterns in Large Databases. In Proc. The 2001 IEEE International Conference on Data
Mining (ICDM'01), San Jose, CA, USA, 2000.
[14] Jian Pei, Jiawei Han, and Runying Mao. Closet: An efficient algorithm for mining frequent closed
itemsets. In SIGMOD International Workshop on Data Mining and Knowledge Discovery, May 2000.
[15] Jian Pei, Jiawei Han, and Jianyong Wang. Closet+: Searching for the best strategies for mining frequent
closed itemsets. In SIGKDD '03, August 2003.
[16] Rafik Taouil, Nicolas Pasquier, Yves Bastide, Lotfi Lajhal, and Gerd Stumme. Mining freqent patterns
with counting inference. SIGKDD Explorations, 2(2):66–75, December 2000.
[17] Mohammed J. Zaki. Mining non-redundant association rules. Data Min. Knowl. Discov., 9(3):223–248,
2004.
[18] Mohammed J. Zaki and Karam Gouda. Fast vertical mining using diffsets. In Proceedings of the ninth

ACM SIGKDD international conference on Knowledge discovery and data mining, pages 326–335. ACM
Press, 2003.
[19] Mohammed J. Zaki and Ching-Jui Hsiao. Charm: An efficient algorithm for closed itemsets mining. In
2nd SIAM International Conference on Data Mining, April 2002.