Vol.2, No.4 ISSN Number (online): 2454-9614

Accurate Discovery of Motifs in Sequence Datasets

P.Anil Kumar[#1], G.L.N.Jayaprada[*2], T.Surekha[*3]

[#]*M.Tech Student, Department of CS, Narasaraopeta engineering College, JNTU-KAKINADA university, Narasaraopeta, Andrapradesh, India*

[1] 1244anil@gmail.com

[*]*Associate Professor, Department of CSE,Narasaraopeta engineering College, JNTU-KAKINADA university, Narasaraopeta, Andrapradesh, India*

[2] njayaprada@yahoo.co.in

[3] tsurekha1234@gmail.com

***Abstract*—Most of existing sequence mining algorithms focuses on mining for subsequences. A large class of applications, such as biological DNA and protein motif mining requires efficient mining of "approximate" patterns that are contiguous. Very few existing algorithms that can be applied to find such contiguous approximate patterns. Such algorithms having drawbacks like poor scalability, lack of guarantees in finding the pattern, and difficulty in adapting to other applications.**

In this paper, we present a new algorithm called FLexible and Accurate Motif DEtector (FLAME). FLAME is a flexible suffix-tree-based algorithm that can be used to find frequent patterns with a variety of definitions of motif (pattern) models. It is also accurate, as it always finds the pattern if it exists. Using both real and synthetic data sets, we demonstrate that FLAME is fast, scalable, and outperforms existing algorithms on a variety of performance metrics. In addition, based on FLAME, we also address a more general problem, named extended structured motif extraction, which allows mining frequent combinations of motifs under relaxed constraints.

I. Introduction

In a number of sequential data mining applications, the goal is to discover frequently occurring patterns. The challenge in discovering such patterns is to allow for some noise in the matching process. This approximate subsequence mining problem is of particular importance in computational biology, where the challenge is to detect short sequences, usually of length 615, that occur frequently in a given set of DNA or protein sequences. These short sequences can provide clues regarding the locations of so called "regulatory regions," which are important repeated patterns along the biological sequence.

The repeated occurrences of these short sequences are not always identical, and some copies of these sequences may vary from others. These frequently patterns are called motifs in computational biology. In the rest of this paper, we use this term to describe frequently occurring approximate sequences.

Different applications require different similarity models to suit the kind of noise that they deal with. It is desirable for a motif mining algorithm to be able to deal with a variety of notions of similarity. In this paper, we present a powerful new model for approximate motif mining that fits several applications with varying notions of approximate similarity. We also present FLexible and Accurate Motif DEtector (FLAME)—novel motif mining algorithms which can efficiently find motifs that satisfy our model.The problem of finding frequently occurring (noncontiguous) subsequences in large sequence databases has been extensively studied in previous works. Traditionally, B is called a subsequence of A, if B can be constructed by projecting out some of the elements of sequence A. For instance, if A is the sequence "a,b,a,c,b,a,c," the sequence "a,b,b,c" is a subsequence constructed by choosing the first, second, fifth, and seventh elements from the original sequence and omitting the rest. While mining for frequent non-contiguous sub sequences have many issues.

II. RELATED WORK

Many surveys of literature on mining databases for frequent patterns have been done. Early work focused on mining association rules. The problem of mining for subsequences was introduced and has several applications, and many algorithms like SPADE, BIDE and several others have been proposed as improvements over existing one. The repeated occurrences of these short sequences are not always identical, and some copies of these sequences may differ from others.

The similarity metric that is used here could be complex—for example, when comparing proteins, a similarity matrix like PAM [1] or BLOSUM [2], may be used for comparing the "distance" between each symbol (protein) pair. Yang et al used a statistical sampling-based method with a compatibility matrix to find patterns in the presence of noise. However, they primarily focus on subsequence mining, while we focus on contiguous patterns. Early work focused on mining association rules. We note that the problem of motif mining is related to the problem of mining for frequent item sets [3].

Some subsequence mining algorithms allow certain constraints. Constraints which limit the maximum gap between two items in the subsequence make it possible to use these algorithms to mine for contiguous patterns. The problem of mining for subsequences was introduced in [4]. Furthermore, they tend to be inefficient even when used for exact substring mining. FLAME, on the other hand is extremely efficient even for approximate substrings.

The vast body of work in bioinformatics for finding patterns in long noisy DNA sequences can be divided into two classes—pattern-based and statistical based. The pattern based algorithms typically search through the space of potential patterns and find a motif that satisfies the minimum support. This method is primarily focused at finding pairs (or sets) of motifs that co-occur in the data set within a short distance of each other. Subsequence mining has several applications, and many algorithms like SPADE [5], BIDE [6], and CloSpan [7] (and several others) have been proposed.

This method only considers a simple mismatch-based definition of noise, and does not consider other more complex motif models such as a substitution matrix or a compatibility matrix. These optimizations make FLAME faster by an order of magnitude. Zhu et al proposed an algorithm for mining approximate substrings but it only accommodates the Hamming distance model. Similarly, Rajasekaran et al propose an algorithm for solving an instance of the motif mining problem where wildcard characters are allowed but it also uses the Hamming distance model. Other approaches are developed for the planted motif search problem, where it is known a priori that an instance of the motif is included in every sequence of the data set. These can also accommodate only the Hamming distance measure.

Yang et al. [8] use a statistical sampling-based method with a compatibility matrix to find patterns in the presence of noise. However, they primarily focus on subsequence mining, while we focus on contiguous patterns. YMF [9] is a simple algorithm that computes statistical significance of each motif. YMF scales very poorly with increasing complexity of motifs, and thus cannot be easily adapted to other applications Weeder [10] is suffix-tree-based algorithm that makes certain assumptions about the way the mismatches in an instance of the motif are distributed. This makes Weeder extremely fast, but it is not guaranteed to always find the motif.

Weeder too, cannot be adapted for other motif models. MITRA [11] is a mismatch tree-based algorithm which uses clever heuristics to prune the large space of possible motifs. MITRA is very resource intensive and requires large amounts of memory. The Random Projections algorithm of Buhler and Tompa [12] has recently been applied to time series data for motif mining, to search for frequent patterns in the data. All of these approaches run the risk of finishing and may not be able to find the right motif. There are several applications of motif mining in addition to those mentioned above. It is often the first step in discovering association rules in sequence data (“basic shapes” in [13] and “frequent patterns” in [14]). It can also be used to find good seeds for clustering sequence data sets [15]. Records of medical signals, like ECG or respiratory data from patients can also be mined to find signals that can indicate a potentially critical condition.

III. THE MODEL

Critical aspect of the motif mining problem is defining the model under which two or more sequences are considered to match (approximately). Developing such models faces an interesting challenge: On the one hand, we want a model that is robust enough to detect the occurrence of a pattern even in the presence of noise, and on the other hand, we do not want it to be so general that it matches unrelated subsequences. Since different applications may have different criteria for how to strike this balance, a natural approach is to develop a flexible model with a few intuitive parameters that can be set by the user based on the application characteristics. In this section, we present a new model for motifs that can be used for pattern mining in many different domains.

We call our motif model the L;M;s;k model after the four parameters that determine it. L is the length of the motif, M is a distance matrix that is used to compute the similarity between two strings, s is the maximum distance threshold within which two strings are considered similar, and finally, k is the minimum support required for a pattern to qualify as a motif.

The L;M;s;k model is very sensitive and permits the user a lot of flexibility in making the right trade-off between specificity and noise tolerance of a model. As we describe below, much of this power comes from the ability to use any matrix M as the distance matrix.

The matrix M allows us to define a distance penalty when a symbol X in the model matches a symbol Y in the data sequence. The penalty is specified by M(X,Y), an entry in the matrix. The total distance between the two strings is computed by summing the distance penalties of the corresponding symbols. That is, if A ¼ $a_1a_2a_3$ ...a_n and B ¼ $b_1b_2b_3$ ...b_n are two strings, then the distance between them. By using these strings we can determine required motifs according to the given input.

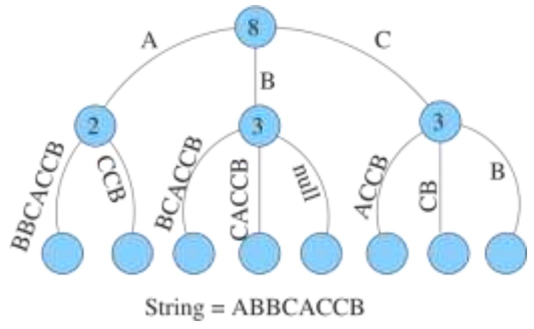
Formally speaking, a string S is L;M;s;k motif if there exist at least k strings T_1;...;T_k in the database . Every string S that satisfies the above is an L;M;s;k motif. Note that the string S need not actually appear in the database for it to qualify as a motif. Only the instances T_i need to be in the database.

Protein motif mining is an example of a domain which requires a matrix-based measure of similarity. Finding regions in protein sequences that appear frequently in different proteins is useful in inferring the functional sites in proteins. As in the case of DNA, the patterns in protein sequences do not repeat exactly. The instances of the pattern usually differ from the model in a few positions. To complicate things further, not all mismatches are equally bad. Some amino acids are very similar to each other, while some are very different. For instance, Alanine and Valine are both hydrophobic amino acids, while Glycine and Serine are both hydrophilic. The matrix can be used to award a small penalty for M(X,Y) when X and Y are similar (Alanine and Valine, for instance) and a larger penalty otherwise (say, Alanine and Glycine) . Popular substitution matrices such as PAM and BLOSUM can easily be used in our model. The matrix can be adapted to allow other kinds of models. In fact, the matrix approach lets us simulate any L_p-norm (Manhattan distance, Euclidean distance, etc.). If we wanted to match two sequences only if the corresponding values (in the two sequences) were within two units of each other. In general, any measure that can be computed in an incremental fashion by comparing the symbols in the corresponding positions can be simulated by constructing an appropriate distance penalty matrix. Note that our model does not allow insertion and deletion events. In general, suffix trees work best for matching when the lengths are the same. We now discuss two special cases of the L;M;s;k model that are commonly used in computational biology and other domains—the L;d;k and L;f;d;k models.

3.1 Special Case: The L;d;k Model The L;d;k model is a mismatch-based model commonly used in computational biology for finding DNA motifs. The distance measure between two strings is the Hamming distance, or merely the number of mismatches. In the L;d;k model, (d) denotes the maximum Hamming distance, and (L) and (k) are as per our model. It follows as a special case of our model by setting s ¼ d and using 1 for all non-diagonal entries of (M).

One of the applications of this model is in the field of computational biology. The L;d;k model and its derivatives have been considered a good fit for DNA regulatory motifs. Briefly, the related problem of using this model to find regulatory motifs in DNA is as follows: Biologists today are interested in understanding how different genes in the genome are regulated and the way they interact with each other. To this end, biologists often study genes that exhibit similar expression patterns to extract clues about the proteins that control their expression. It is believed that genes that are co regulated by the same protein (called a transcription factor) share some signal that allows the transcription factor to recognize the gene and turn it on. This signal is usually present in the region upstream of a gene (within a few thousand base pairs) called the promoter region. The signature is usually a short string of DNA 6-15 bases long. As is often the case in biology, these signatures are seldom identical, and differ in a few positions from one gene promoter region to another.

Finding this noisy signature that is common across all the genes is a very important step toward locating the binding site for the transcription factor. Modeling the set of promoter regions as our database, and the signature binding site as an L;d;k pattern, we can simply apply the FLAME algorithm to solve this problems. In most practical situations, we don't know the exact value of L, and therefore, we might have to try several values. In the case of DNA regulatory patterns, we know that the signature is usually between 6 to 15 bases long, and therefore we can try these lengths with varying number of mismatches.

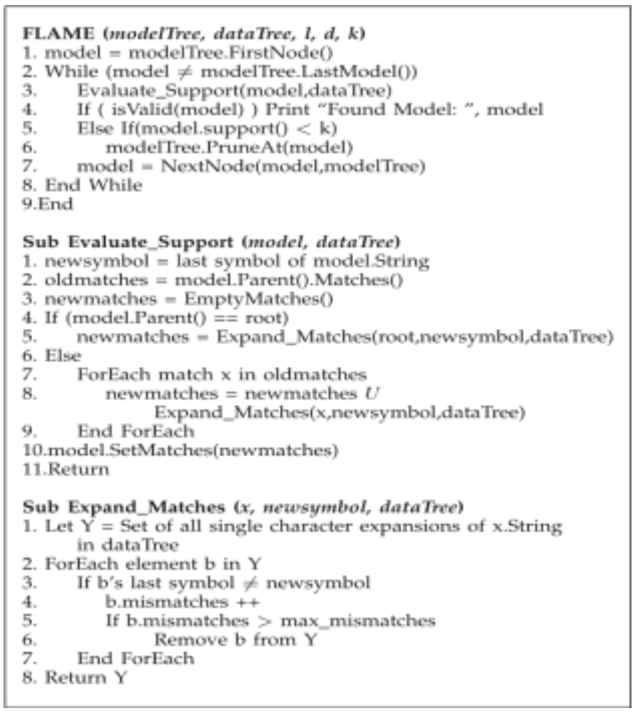


A suffix tree on the string ABBCACCB. The counts are indicated inside the node.

In general, this model is useful in applications where the noise has a positional bias as it allows us to be more specific in finding the right patterns while ignoring extraneous matches. Some DNA motif finding applications use models that are somewhat similar to the L;f;d;k model.

IV.EXISTING SYSTEM

The algorithm we developed for the extended structured motif extraction problem is based on the techniques described above. Given a p-structured model $M_1M_2 \ldots M_p$, in the first step of the algorithm, we run FLAME on the enhanced suffix tree for every simple $L_i;M_i;s_i;k$ model, that constitutes the p-structured model.



```
FLAME (modelTree, dataTree, l, d, k)
1. model = modelTree.FirstNode()
2. While (model ≠ modelTree.LastModel())
3.      Evaluate_Support(model,dataTree)
4.      If ( isValid(model) ) Print "Found Model: ", model
5.      Else If(model.support() < k)
6.          modelTree.PruneAt(model)
7.      model = NextNode(model,modelTree)
8. End While
9.End

Sub Evaluate_Support (model, dataTree)
1. newsymbol = last symbol of model.String
2. oldmatches = model.Parent().Matches()
3. newmatches = EmptyMatches()
4. If (model.Parent() == root)
5.      newmatches = Expand_Matches(root,newsymbol,dataTree)
6. Else
7.      ForEach match x in oldmatches
8.          newmatches = newmatches U
                  Expand_Matches(x,newsymbol,dataTree)
9.      End ForEach
10.model.SetMatches(newmatches)
11.Return

Sub Expand_Matches (x, newsymbol, dataTree)
1. Let Y = Set of all single character expansions of x.String
        in dataTree
2. ForEach element b in Y
3.      If b's last symbol ≠ newsymbol
4.          b.mismatches ++
5.          If b.mismatches > max_mismatches
6.                  Remove b from Y
7.      End ForEach
8. Return Y
```

The FLAME algorithm.

The algorithm we developed for the extended structured motif extraction problem is based on the techniques described above. Given a p-structured model $M_1M_2 \ldots M_p$, in the first step of the algorithm, we run FLAME on the enhanced suffix tree for every simple $L_i;M_i;s_i;k_i$ model, that constitutes the p-structured model. After every run, we get a set of qualifying motifs that match the model, the corresponding occurrences and the Boolean arrays. By using the techniques described, we compute for each run of FLAME all the Motif Existential arrays and the Model Existential array.At the end of this phase, we have p Model Existential Boolean arrays, one for each simple model component. The fact that this algorithm uses FLAME as a building block, makes our technique general and flexible. It is possible to use different types of models for each simple motif and different distance measures. For example, we could use a L;M;s;k model for the first motif and a L;f;d;k for the second or two L;M;s;k models with different substitution matrices.

V. PROPOSED STSTEM

SPRINT (Scalable Parallelizable Induction of Decision Tree) Algorithm. SPRINT is Decision- tree based Algorithm. Motif mining can be done in parallel, means we can mine more number of Patterns at a time. Divide the dataset among N share-nothing machines Categorical data: just divide it evenly numerical data: use a parallel sorting algorithm to sort the data. It stands for scalable parallelizable induction of decision tree algorithm. It was introduced by Shafer et al in 1996. It is fast, scalable decision tree classifier. It is not based on Hunt's algorithm in constructing the decision tree, rather it partitions the training data set recursively using breadth-first greedy technique until each partition belong to the same leaf node or class.

It can be implemented in both serial and parallel pattern for good data placement and load balancing. It uses two data structure: attribute list and histogram which is not memory resident making sprint suitable for large data sets, thus it removes all the data memory restrictions on data. It handles both continuous and categorical attributes..

VI. CONCLUSION

In this paper, we presented a powerful new model: L;M;s;k for motif mining in sequence databases. The L;M;s;k model subsumes several existing models and provides additional flexibility that makes it applicable in a wider variety of data mining applications. We also presented FLAME, a flexible and accurate algorithm that can find L;M;s;k motifs. Through a series of experiments on real and synthetic data sets, we demonstrate that FLAME is a versatile algorithm that can be used in several real motif mining tasks.

Having problems like no sorting technique to sort sequential patterns and difficult to adopt to other applications we approached new algorithm SPRINT. The goal was to develop a decision tree classification algorithm that was robust, scalable, and parallelizable.

REFERENCES

[1] M.O. Dayhoff, R.M. Schwartz, and B. Orcutt, "A Model for Evolutionary Changes in Proteins," Atlas of Protein Sequence and Structure, vol. 5, pp. 345-352, Nat'l Biomedical Research Foundation, 1978.

[2] S. Henikoff and J. Henikoff, "Amino Acid Substitution Matrices from Protein Blocks," Proc. Nat'l Academy of Sciences USA, vol. 89, no. 22, pp. 10915-10919, 1992.

[3] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 487-499, 1994.

[4] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th IEEE Int'l Conf. Data Eng. (ICDE), pp. 3-14, 1995.

[5] M.J. Zaki, “SPADE: An Efficient Algorithm for Mining Frequent Sequences,” Machine Learning, vol. 42, nos. 1/2, pp. 31-60, 2001.
[6] J. Wang and J. Han, “BIDE: Efficient Mining of Frequent Closed Sequences,” Proc. 20th IEEE Int’l Conf. Data Eng. (ICDE), pp. 79-90, 2004.
[7] X. Yan, J. Han, and R. Afshar, “CloSpan: Mining Closed Sequential Patterns in Large Datasets,” Proc. SIAM Int’l Conf. Data Mining (SDM), 2003.
[8] J. Yang, W. Wang, P.S. Yu, and J. Han, “Mining Long Sequential Patterns in a Noisy Environment,” Proc. ACM SIGMOD, pp. 406417, 2002.
[9] S. Sinha and M. Tompa, “YMF: A Program for Discovery of Novel Transcription Factor Binding Sites by Statistical Overrepresentation,” Nucleic Acids Research, vol. 31, no. 13, pp. 3586-3588, 2003.
[10] G. Pavesi, P. Mereghetti, G. Mauri, and G. Pesole, “Weeder Web: Discovery of Transcription Factor Binding Sites in a Set of Sequences From Co-Regulated Genes,” Nucleic Acids Research, vol. 32, pp. W199-W203, 2004.
[11] E. Eskin and P.A. Pevzner, “Finding Composite Regulatory Patterns in DNA Sequences,” Proc. 10th Int’l Conf. Intelligent Systems for Molecular Biology (ISMB), pp. S354-S363, 2002.
[12] J. Buhler and M. Tompa, “Finding Motifs Using Random Projections,” J. Computational Biology, vol. 9, no. 2, pp. 225-242 , 2002.
[13] G. Das, K.-I. Lin, H. Mannila, G. Renganathan, and P. Smyth, “Rule Discovery from Time Series,” Proc. Int’l Conf. Knowledge Discovery and Data Mining (KDD), pp. 16-22, 1998.
[14] S. Hoppner, “Discovery of Temporal Patterns—Learning Rules about the Qualitative Behaviour of Time Series,” Proc. Fifth European Conf. Principles and Practice of Knowledge Discovery in Databases, pp. 192-203, 2001.
[15] P. Patel, E. Keogh, J. Lin, and S. Lonardi, “Mining Motifs in Massive Time Series Databases,” Proc. IEEE Int’l Conf. Data Mining (ICDM), pp. 370-377, 2002.

Selected Paper from International Conference on Computing (NECICC-2k15)