

Chapter 2

Related Work

There are three areas of research highly related to our exploration in this dissertation, namely sequential pattern mining, multiple alignment, and approximate frequent pattern mining. We survey each in the following sections.

2.1 Support Model Based Sequential Pattern Mining

Since it was first introduced in [2], sequential pattern mining has been studied extensively. Conventional sequential pattern mining finds frequent subsequences in the database based on exact match. Much research has been done to efficiently find patterns, seq_p , such that $sup(seq_p) \geq min_sup$ ¹.

There are two classes of algorithms. The bottleneck in applying the support model occurs when counting the support of all possible frequent subsequences in database \mathcal{D} . Thus the two classes of algorithms differ in how to efficiently count support of potential patterns.

The apriori based breadth-first algorithms such as GSP and SPADE [2, 45, 55] pursue level-by-level candidate-generation-and-test pruning following the Apriori property: any super-pattern of an infrequent pattern cannot be frequent. In the first scan, they find the length-1 sequential patterns, i.e., single items frequent in the database. Then, length-2 candidates are assembled using length-1 sequential patterns. The sequence database is scanned the second time and length-2 sequential patterns are found. These methods scan the database multiple times until the longest patterns have been found. At each level, only potentially frequent candidates are generated and tested. This pruning saves much computational time compared to naively counting support of all possible subsequences in the database. Nonetheless, the candidate-generation-and-test is still very costly.

In contrast, the projection based depth-first algorithms such as PrefixSpan, FreeSpan, and SPAM [4, 34, 48] avoid costly candidate-generation-and-test by growing long patterns

¹the *support* of a sequence seq_p , denoted as $sup(seq_p)$, is the number of sequences in \mathcal{D} that are super-sequences of seq_p . The exact definition is given in chapter 3

from short ones. Once a sequential pattern is found, all sequences containing that pattern are collected as a projected database. And then local frequent items are found in the projected databases and used to extend the current pattern to longer ones. Another variant of the projection-based methods mine sequential patterns with vertical format [66]. Instead of recording sequences of items explicitly, they record item-lists, i.e., each item has a list of sequence-ids and positions where the item appears.

It is interesting to note that the depth first methods generally do better than the breadth first methods when the data can fit in memory. The advantage becomes more evident when the patterns are long [64].

There are several interesting extensions to sequential pattern mining. For example, various constraints have been explored to reduce the number of patterns and make the sequential pattern mining more effective [28, 49, 55, 65]. Last year, Yan published the first method for mining frequent closed subsequences using several efficient search space pruning methods [63]. In addition, some methods (e.g., [54]) mine confident rules in the form of “ $A \rightarrow B$ ”. Such rules can be generated by a post-processing step after the sequential patterns are found.

Recently, Guralnik and Karypis used sequential patterns as features to cluster sequential data [29]. They project the sequences onto a feature vector comprised of the sequential patterns, then use a k -means like clustering method on the vector to cluster the sequential data. Interestingly, their work concurs with this study that the similarity measure based on edit distance is a valid measure in distance based clustering methods for sequences. However, we use clustering to group similar sequences here in order to detect approximate sequential patterns. Their feature-based clustering method would be inappropriate for this purpose because the features are based on exact match.

2.2 Multiple Alignment

The support model based sequential pattern mining algorithms extend the work done in association rule mining. Association rule mining ignores the sequence dimension and simply finds the frequent patterns in the itemsets. Approaching sequential analysis from association rule mining might be natural. Association rules mine intra-itemset patterns in sets while sequential mining finds inter-itemset patterns in sequences.

However, there is one important difference that can make sequential data mining easier. Unlike sets, sequences have extra information that can come in very handy - the ordering. In fact there is a rich body of literature on string (simple ordered lists) analysis in computer science as well as computational biology.

The most current research in computational biology employs the multiple alignment technique to detect common underlying patterns (called consensus strings or motifs) in a group of simple sequences (strings) [30, 25, 57]. In computational biology, multiple alignment has

Table 2.1: Multiple alignment of the word *pattern*

seq1	P	A	T	T	T	E	R	N
seq2	P	A	{}	{}	T	E	R	M
seq3	P	{}	{}	T	T	{}	R	N
seq4	O	A	{}	T	T	E	R	B
seq5	P	{}	S	Y	Y	R	T	N
Underlying pattern	P	A	{}	T	T	E	R	N

Table 2.2: The profile for the multiple alignment of the word *pattern* given in Table 2.1

position	1	2	3	4	5	6	7	8
A		3						
B								1
E						3		
M								1
N								3
O	1							
P	4							
R							1	4
S			1					
T			1	3	4		1	
Y				1	1			
{}		2	3	1		1		
Consensus string	P	A	{}	T	T	E	R	N

been studied extensively in the last two decades to find consensus strings in DNA and RNA sequences.

In the simple edit distance problem, one is trying to find an alignment of two sequences such that the edit distance is minimum. In multiple alignment, the purpose is to find an alignment over N strings such that the total pairwise edit distance for all N strings is minimal. Aligned strings are defined to be the original string with null, {}, inserted where appropriate so that the lengths of all strings are equivalent. A good alignment is one in which similar characters are lined up in the same column. In such an alignment, the concatenation of the most common characters in each column would represent the underlying pattern in the group of strings.

For example, if we collect the various typos for a word *pattern*, and then align them, we should be able to uncover the correct spelling of the word. This is shown in Table 2.1. Even though non of the strings have the proper spelling we are able to find the correct spelling by multiple alignment.

Although, this is a straight extension of the well known edit distance problem, and can be solved with dynamic programming, the combinatorial nature of the problem makes the solution impractical for even small number of sequences [30]. In practice, people have employed the greedy approximation solution. That is, the solution is approximated by aligning two sequences first and then incrementally adding a sequence to the current alignment of $p - 1$

sequences until all sequences have been aligned. At each iteration, the goal is to find the best alignment of the added sequence to the existing alignment of $p - 1$ sequences. Consequently, the solution might not be optimal. The various methods differ in the order in which the sequences are added to the alignment. When the ordering is fair, the results are reasonably good [30].

For strings, a sequence of simple characters, the alignment of p sequences is usually represented using a profile. A profile is a matrix whose rows represent the characters, whose columns represent the position in the string, and whose entries represent the frequency or percentage of each character in each column. Table 2.2 is the profile for the multiple alignment of the word *pattern* given in Table 2.1. During multiple alignment at each iteration we are then aligning a sequence to a profile. This can be done nicely using dynamic programming as in the case of aligning two sequences. Only the recurrence is changed to use the sum of all pairwise characters in the column with a character in the inserted string. Although we cannot reconstruct the p sequences from the profile, we have all the information we need to calculate the distance between two characters for all p sequences. Therefore, profiles are an effective representation of the p aligned strings and allow for efficient multiple alignment.

In this dissertation, we generalized string multiple alignment to find patterns in ordered lists of sets. Although much of the concept could be borrowed, most of the details had to be reworked. First, we select an appropriate measure for distance between sequences of itemsets. Second, we propose a new representation, *weighted sequences*, to maintain the alignment information. The issue of representing p sequences is more difficult in this problem domain because the elements of the sequences are itemsets instead of simple characters. There is no simple depiction of p itemsets to use as a dimension in a matrix representation. Thus, effectively compressing p aligned sequences in this problem domain demands a different representation form. In this dissertation, we propose to use weighted sequences to compress a group of aligned sequences into one sequence. And last, unlike strings the selection of the proper items in each column is not obvious. Recall that for strings, we simply take the most common character in each column. For sequence of itemsets, users can use the *strength* cutoffs to control the level of detail included in the consensus patterns. These ideas are discussed in detail in chapter 4.

In [12], Chudova and Smyth used a Bayes error rate model under a Markov assumption to analyze different factors that influence string pattern mining in computational biology. Extending the theoretical model to mining sequences of sets could shed more light to the future research direction.

2.3 Approximate Frequent Pattern Mining

A fundamental problem of the conventional methods is that the exact match on patterns does not take into account the noise in the data. This causes two potential problems. In real data, long patterns tend to be noisy and may not meet the support level with exact matching [50]. Even with moderate noise, a frequent long pattern can be mislabeled as an infrequent pattern [64]. Not much work has been done on approximate pattern mining.

[62] was the first to do approximate matching on frequent itemsets. Although the methodology is quite different, in spirit *ApproxMAP* is most similar to [62] in that both try to uncover structure in large sparse databases by clustering based on similarity in the data and exploiting the high dimensionality of the data. The similarity measure is quite different because [62] works with itemsets while *ApproxMAP* works on sequences of itemsets.

[50] also presents an apriori-based algorithm to incorporate noise for frequent itemsets, but is not efficient enough. Although the two methods introduced in [62] and [50] are quite different in techniques, they both explored approximate matching among itemsets. Neither address approximate matching in sequences.

Recently, Yang et al. presented a probabilistic model to handle noise in mining strings [64]. A compatibility matrix is introduced to represent the probabilistic connection from observed items to the underlying true items. Consequently, partial occurrence of an item is allowed and a new measure, *match*, is used to replace the commonly used support measure to represent the accumulated amount of occurrences. However, it cannot be easily generalized to apply to sequences of itemsets targeted in this research, and it does not address the issue of generating huge number of patterns that share significant redundancy.

By lining up similar sequences and detecting the general trend, the multiple alignment model in this paper effectively finds consensus patterns that are approximately similar to many sequences and dramatically reduce the redundancy among the derived patterns.

2.4 Unique Aspects of Our Research

As discussed in the introduction, to the best of our knowledge, though there are some related work in sequential pattern mining, this is the first study on mining consensus patterns from sequence databases. It distinguishes itself from the previous studies in the following two aspects.

- It proposes the theme of approximate sequential pattern mining, which reduces the number of patterns substantially and provides much more accurate and informative insights into sequential data.
- It generalizes the multiple alignment techniques to handle sequences of itemsets. Mining sequences of itemsets extends the application domain substantially.