Abstract: In Real Life Sequential Database are Dynamic in Nature. There Should be Change in Nature. So after Every Update a Frequent Pattern are not Frequent they may be Change and in Frequent. So what happen we have to mine all the data within a certain period of time from the Scratch? so it’s not a good idea to scratches all the data .For that there should be one solution that is called the Incremental Approach of Mining. The Main aim behind the use of Incremental Mining Approach is what to Save the Time Period of the owner who is using this.

It should mine the set of frequent patterns in significantly less time than a non-incremental mining algorithm [2].what should be actually done is to gain the Incremental Approach The original large and pre-large sequences with their counts from preceding runs are retained for later use in maintenance. As new transactions are added, the proposed approach first transforms them into new customer sequences and merges them with the corresponding old sequences existing in the original database. The newly merged customer sequences are then scanned to generate candidate 1-sequences with occurrence increments. These candidate sequences are compared to the large and pre-large 1-sequences which were previously retained.
1. Introduction:

Given a set of data sequences, Sequential Pattern Mining is to discover subsequences i.e. ordered events that are frequent in the sense that the percentage of data sequences containing them exceeds a user-specified minimum support [2].

E.g. “Customers who buy a PC are likely to return latter to buy a Laser Printer. There are Number of Sequence Mining Algorithm. Like GSM, AprioryAll, SPAM, FREESPAM etc. but this All Algorithm mine only Sequence Database or in terms of Static Database. They are not supposed to mine dynamic Database[1].

Note that the ultimate aim of using an incremental mining algorithm instead of non-incremental one is to gain efficiency with respect to time. Otherwise a non-incremental mining algorithm can also serve the purpose of mining very easily. So for incremental mining algorithm the time taken by the algorithm to mine complete set of frequent patterns must be considered [2].

2. Incremental Mining Algorithm:

2.1. ISM:

The ISM algorithm, proposed by [4], is actually an extension of SPADE, which aims at considering the update by means of the negative border and a rewriting of the database. SPADE is all about the static Database. But if want to do in terms of the dynamic in nature than new algorithm that is introduce by Agarwal and Srikant that is the Incremental Sequence Mining(ISM)[1].

How that algorithm is work is like as below:

The first step of ISM aims at Deleting that kind of the sequence using the prune step the become infrequent from the set of frequent sequences after the update. Once this step is done than it’s added to the negative border of the particular Lattice which is generated using the Appropriate Sequences. After that in the second step what happened the Sequences which is added one by one into Lattices. Which will have a support of sequences and will be added to the negative border [4].

After the End of these two steps the ISM algorithm can make update the complete set of database. And the ISM algorithm can give the new set of the Frequent Sub Sequence.
So there should be the disadvantage of the Increment Sequence Mining Algorithm is that it is not mine the complete set of sequence pattern so after that new algorithm which is extension of this is invented which is ISE and quite differ than ISM algorithm.

2.2. ISE:

In [7] the main consequence of adding new customers is to verify the support of the frequent sequences in $L^{\text{DATABASE}}$.

During the first pass on database, we count the support of individual items and are provided with $1$-candExt standing for the set of items occurring at least once in Database. Considering the set of items embedded in Database we determine which items of database are frequent in U. This set is called $M^{\text{DB}}_1$.

At the completion of this pass, if there are additional customers or additional Id will be generated, we prune out frequent sequences in $M^{\text{DB}}$ that no longer verify the minimum support.

The frequent 1-sequences in database to create new candidates. This candidate creating works by joining $M^{\text{DB}}_1$ with $M^{\text{DB}}_1$ and yields the set of candidate 2-sequences. We scan database and obtain the 2-sequences embedded in database. Such a set is called 2-candidates. This phase is somehow different from the GSP approach since we do not consider the support constraint to generate the sub candidate sequences. A sub candidate 2-sequence is in 2 if and only if it occurs at least once in Database. The main reason is that we do not want to provide the set of all 2-sequences, but rather to obtain the set of potential extensions of items embedded in Database. In other case we can say that, if a candidate 2-sequence does not occur in Database it cannot possibly be an extension of an original frequent sequence of DATABASE, and thus cannot give a frequent sequence for U and that will added. In the same way, if a candidate 2-sequence occurs in Database, this sequence might be an extension of previous sequences in DATABASE.

After that we can find the next step to add the generated sequence of first step into the U.

An additional operation is performed on the frequent items discovered in Database. The main idea is to retrieve in DATABASE the frequent sub-sequences of $M^{\text{DB}}$ preceding items of Database, according to their order in time.
In order efficiently to find the frequent sub-sequences preceding an item, we create for each frequent sub-sequence an array that has as many elements as the number of frequent items in Database.

When scanning U, for each data sequence and for each frequent sub-sequence, we check whether it is contained in the data sequence. In such a case, the support of each item following the subsequence is incremented.

During the scan to find 2-freqExt, we also obtain the set of frequent sub-sequences preceding items of Database. From this set, by appending the items of Database to the frequent sub-sequences we obtain a new set of frequent sequences. This set is called freqSeed.

2.3. Incspan:

After the ISE algorithm new Algorithm is approached and that is the IncSpan Algorithm that is somehow based on the frequent Sequence FreeSpan Algorithm. Though FreeSpan Algorithm is about the Static Database that is change in IncSpan Algorithm.

An incremental mining algorithm called IncSpan based on an existing algorithm called PrefixSpan. To gain efficiency the concept of semi-frequent patterns is introduced. Semi frequent patterns are the patterns which are not frequent but whose support is greater than the product of minimum support and a user specified factor $\mu$ $(0 < \mu < 1)$ i.e. patterns that are not frequent but are almost frequent. Note that these almost frequent patterns are most likely to be frequent in the updated database.

3. Comparision of Three Algo:

<table>
<thead>
<tr>
<th>Method</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISM and ISE[6]</td>
<td>incremental mining of sequential patterns in large Databases , It performs much better than when data is updated. It's Drawback is that</td>
</tr>
<tr>
<td>IncSpan[7]</td>
<td>store very large num of sequence</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Based on Non Incremental PrefixSpan and Incremental ISM, IncSpan has a major drawback of not able to mine the complete set of frequent sequential patterns</td>
<td></td>
</tr>
</tbody>
</table>

Table 1

4. Advantages Of Incremental Approach:

i. It avoids re-computing large sequences that have already been discovered.

ii. It focuses on newly added customer sequences, which are transformed from newly added transactions, thus greatly reducing the number of candidate sequences.

iii. It uses a simple check to further filter candidate sequences in newly added customer sequences.

iv. It effectively handles the case, in which sequences are small in an original database.

5. Conclusion:

Mining of Sequential database from the Large Sequence is very Difficult. Early the AprioryAll, GSP, SPAM all that kind of Algorithm are used to Mine the Sequence Pattern.
But That All Algorithms Mined only the Static Database. as AprioryALL[1] is Mining Sequence Patterns over Transactional Database , the Main Disadvantage of AprioriAll is that there are many passes over the database and to many sub candidates generated, which are very time consuming. GSP [2] is Time constraints, Sliding time windows, taxonomies, GSP is better than AprioriAll and generates less candidates than AprioriAll. But it Scan Database multiple times. So there is need of more efficient Algorithm. FreeSpan[9] is General idea is to use frequent items to recursively project sequence databases into a smaller projected databases, and Grow the Speed of Projected Database. ISM and ISE[6] is incremental mining of sequential patterns in large Databases, It performs much better than when data is updated. Its Drawback is that store very large num of sequence. IncSpan [7] is Based on Non Incremental PrefixSpan and Incremental ISM, IncSpan has a major drawback of not able to mine the complete set of frequent sequential patterns.
References:

1. R. Agrawal; R. Srikant;, “Mining sequential patterns,” International Conference on Data Engineering, pp. 3–14, 1995
2. R. Srikant; R. Agrawal;, “Mining of sequence patterns: Generalizations and performance improvements,”
6. Florent Massauglia; Pascal Poncelet; Maguelonne Teisseire;, “Incremental mining of sequential patterns in large databases,” Data & Knowledge Engineering, pp. 97-121, 2003
7. Hong Cheng, Xifeng Yan, Jiawei Han:” IncSpan: Incremental Mining of Sequential Patterns in Large Database “by U.S. National Science Foundation NSF

Books:

1. By Jiawei Han And Micheline Kamber, Data Mining Concept and Techniques, Copyright 2006, Second Edition, pp 5-9
2. By Jiawei Han And Micheline Kamber, Data Mining Concept and Techniques, Copyright 2006, Second Edition, pp 498-507.