Bit Mask Search Algorithm for Trajectory Database Mining

P.Geetha
Research Scholar
Alagappa University
Karaikudi

E.Ramaraj, Ph.D
Professor
Dept. of Computer Science & Engineering
Alagappa University
Karaikudi

ABSTRACT

Mining great service entities in trajectory database indicates to the exposure of entities with huge service like acquisition. The extensive number of contender entities degrades the mining achievement in terms of execution time and space stipulation. The position may become worse when the database consists of endless lengthy transactions or lengthy huge utility entity sets. In this paper, we use two algorithms, namely Utility Pattern Growth (UP –Growth) for mining huge utility entities with a set of adequate approaches for pruning contender entities. The previous algorithms do not contribute any compaction or compression mechanism the density in bit vector regions. To raise the density in bit-vector sector the Bit search Mask Search (BM Search) starts with an array list. From root node, a BM Search representation for each frequent pattern is designed which gives an acceptable compression and compaction in bit search measure than UP Growth algorithm. This paper compared two algorithms such as UP Growth and BM Search. In the analysis of two algorithms BM Search produces best result compared than the other algorithms. An experimental result shows the comparison of two algorithms.

Keywords

Utility Pattern Growth, Bit Mask Search, Trajectory databases, Frequent Entity set.

1. INTRODUCTION

DATA MINING is the mechanism of extracting nontrivial, already unknown and probably convenient information from huge databases. Observing appropriate entities invisible in a database plays a central role in considerable data mining tasks, such as frequent pattern mining. Frequent pattern mining is a significant research topic that has been enforced to various kinds of databases, such as trajectory databases, transactional databases, streaming databases, and time series databases.

Data mining system should award users to enumerate hints to model the search for compelling entities. In this paper we proof experimentally that Bit Search Mask Search (BM Search) algorithm gives better result than UP Growth algorithm. It holds less time, less space and produces huge accurate results with diminished number of frequent trajectory transactions. BM Search contributes any compaction or compression mechanism to raise the density in bit vector sectors. In this structure, weights of entities, such as unit

benefits of entities in trajectory databases, are treated. With this approach, even if some entities arrive occasionally, they might still be found if they have huge benefit. However, in this scheme, the portions of entities are not treated yet. In view of this, utility mining appears as an essential case in data mining field. Mining huge utility entities from databases indicates to discovering the entities with huge benefits.

Here, the definition of entity utility is pleasingness, consequence, or desirability of an entity to users and trajectory device as Radio Frequency Identification (RFID). Utility of entity in a trajectory database consists of two conditions: 1. the essence of specific entities, which is labeled as extraneous utility, and 2. the essence of entities in transactions, which is termed as constitutional utility. Benefit of an entity is characterized as the product of its extraneous utility and its constitutional utility is no less than a user specified minimum utility threshold; otherwise it is called a low utility entity. Mining huge utility entities from databases in an essential effort has an extensive scope of applications for UP-Growth. An entity is termed frequent if its base is not less than a given absolute nominal support threshold value which is user defined one. Bit Stream Mask is a different approach in which the input fie is first transferred into numerical data. After this the transaction file is wrapped into an array for further processing. This approach raises the global competency of the apriori algorithm in terms of time and space complexity.

The remaining part of the paper is organized as follows:

Section II involves the works related to UP-Growth and BM Search. Section III involves a brief description of the existing methods –Utility Pattern Growth (UP Growth) and the problems involved in them. Section IV involves the description of the proposed method – Bit Mask Search (BM Search). Section V involves the performance evaluation and comparison of BM Search and existing techniques based on UP-Growth. The paper is concluded in Section VI.

2. RELATED WORK

This section deals with an efficient algorithm for mining great utility entities from trajectory databases. Tseng, et al proposed a paper based on trajectory databases[1]. The achievement of Utility Pattern (UP) Growth is correlated with the state of the art algorithms on frequent types of both real and synthetic data sets. Boaddh, et al proposed a paper of Bit Mask(BM).

Search for mining frequent entities [2]. This paper proves experimentally that Bit Mask (BM) Search algorithms gives better result compare than UP Growth algorithm. Bit Mask (BM) Search begins with an array list. A BM search representation for individual entity is established which gives a sufficient compression and compaction in Bit Search Mask Search by using this approach. Tseng, et al proposed an efficient algorithm for high utility itemset mining [3]. In this paper, they proposed an efficient algorithm, namely UP-Growth (Utility Pattern Growth), for mining high utility entities with a set of techniques for pruning candidate entities. The information of high utility entities is preserved in a special data structure named UP-tree (Utility Pattern Tree) such that the candidate entities can be developed efficiently with only two scans of the database. Abaya proposed an association rule mining based on Apriori algorithm in minimizing candidate generation[4]. Wang, et al proposed a model based approach based on accelerating probabilistic frequent itemset mining[5]. Hong, et al suggested a paper for effective utility mining with the measure of average utility [6]. Frequent entities mining only examines the frequency of occurrence of the entities but does not reflect any other aspects, such as benefit. Dr.E.Ramaraj proposed an general survey on multi-dimensional and quantitative association rule mining algorithms[7]. Discovery of hidden patterns and relationships often goes unexplained. State of the art data mining techniques can help to overcome this situation. Ahme, et al proposed an efficient candidate pruning technique to mine high utility patterns[8]. Conventional frequent pattern mining methods consider an equal benefit for all items and only binary occurrences of the items in transactions.

Yildiz, Ergenc suggested the comparison of two association rule mining algorithms without candidate generation[9]. In this paper, they compare matrix apriori and FP-Growt0h algorithms. Data mining defined as finding hidden information from huge data sources has become a popular way to discover strategic knowledge. Fan zhang, et al proposed a paper for accelerating frequent itemset mining on graphics processing units[10]. The goal of Frequent Itemset Mining (FIM) is to catch frequently appearing subsets within a database of sets. The objective of FIM is to consider a database of itemsets and classify all item subsets that appear more frequently than a given, user specified threshold. Schlege, et al proposed a paper for Memory Efficient Frequent Itemset Mining[11]. Efficient discovery of frequent itemsets in extensive datasets is a key fundamental of many data mining functions. Ustundag, et al proposed a paper for Fuzz rule based system for the economic analysis of RFID investments[12]. Radio Frequency Identification is a mechanism that incorporates the use of electromagnetic or electrostatic coupling in the Radio Frequency (RF) portion of the electromagnetic spectrum uniquely identifies an object. Kim, et al suggested a paper for Mining frequent itemsets with normalized weight in continuous data streams[13]. The problem of frequent itemsets. mining is finding the complete set of itemsets satisfying a minimum support in the transaction database. Roh, et al proposed a paper pattern matching queries over trajectories on road networks[14]. This paper subjects pattern scale trajectory data, say, from gripping vehicles. In this paper, they proposed to represent a trajectory as a sequence of road segments in road network. Liu, et al suggested a paper for mining frequent trajectory patterns for activity monitoring using radio frequency tag arrays[15]. The RFID technology provides an economically attractive solution due to the low cost of RF tags and readers.

3. EXISTING METHODS

Two existing methods for trajectory databases are considered. One is Utility Pattern (UP) Growth.

3.1 Utility Pattern Growth (UP Growth)

In view of this, utility mining appears as an extensive case in data mining field. Mining huge utility entities from datasets indicates to discovering the entities with huge benefits. Here, the meaning of entity utility is attractiveness, effectiveness, or advisability of an item to users. Utility of entities in a trajectory database consists of two aspects.

- The concern of specific entities, which is termed as extraneous utility.
- The concern of entities in transactions, which is termed as constitutional utility. Utility of an entity is defined as the product of its extraneous utility and its constitutional utility.

$$U_{TH} = (U_{MAX} + U_{MIN})\frac{25}{100} \qquad (1)$$

Where, - Utility Threshold

U_{max} Maximum Utility Count

U_{min} Maximum Utility Count

An entity is labeled as a high utility entity set if its utility is no less than a user-specified minimum utility threshold; otherwise, it is termed as a low-utility entity set. Mining high utility entity sets from databases is an important function has a wide range of applications. ALGORITHM I is for Trajectory UP Growth. The following equation (1) is used to calculate the utility threshold value.

3.1.1 Description for ALGORITHM

In the first algorithm first we initialize all the values like uList, uc = 0, du = 0. Where, uList – userList, ucList – utility count List, uc – Utility Count, dc- distinct user Count. And, get all the distinct users. After getting the distinct users assign a utility count for each user. Subsequently, set threshold using the equation (1). Transactions less than threshold are considered as Low Utility Count (LUT). Afterwards, remove the Low Utility Count. Apply steps from 1 to 5 for RTT and return FTTI

ALGORITHM I – Trajectory UP –Growth

```
Input:  Trajectory Transactions (TT)
Output: Frequent Trajectory Transactions (FTT)
Begin
1.  Initialize the values
2.  For all Transactions //Getting distinct users
      a. If(uList.contains(user))
              i. doNothing();
      b. else
              i. uList.add(user);
             ii. dc++;
3.  End For
4.  For all Transactions //utility count for each user
      a. For each user in uList
          i.      uc++;
      b. End For
      c. ucList.add(uc);
      d. uc = 0;
5.  End For
6.  Set Threshold using the equation(1)
7.  Transactions less than threshold are considered as
    Low Utility Transactions (LUT)
8.  Remove LUT
9.  TT – LUT = Reorganized Trajectory Transactions
    (RTT)
10. Apply steps 1 to 5 for RTT
11. Return FTT
                         End
```

4. BIT MASK SEARCH

Mask is an input that is used for bitwise operation. Using mask, multiple bits in a byte can be set either on, off or invented from on to off in a single bitwise operation, this is called as bit masking. An entity set is labeled persistent if its support is not less than a given absolute minimal support threshold value which is user defined one.

Bit Stream Mask is a different approach in which the input file is first transformed into numerical data. After this the transaction file is compressed into an array for further processing. This approach increases the overall performance of the apriori or Bit Mask Search (BM Search) algorithm in terms of time and space complexity

Captions should be Times New Roman 9-point bold. They should be numbered (e.g., “Table 1” or “Figure 2”), please note that the word for Table and Figure are spelled out. Figure’s captions should be centered beneath the image or picture, and Table captions should be centered above the table body.

Apriori is a seminal algorithm for mining frequent entity sets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses prior learning of frequent entity set properties.

Afterwards, identify distinct object in the overall database. Subsequently, converting strings to numbers and assigning the numeric value for each data. Then return the mask list. Finally, update TT with masked data.

The Radio Frequency Identification (RFID) technology is used to provide an inexpensive and relatively accurate approach to activity monitoring. The trajectory data mining technique is used to detect and analyze frequent trajectory patterns. A BM Search representation for each frequent entity is created which gives a sufficient compression and compaction in bit search.

4.1 Description for ALGORITHM II

Algorithm II is Bit Masking Algorithm to assign unique numeric values to all field values in the trajectory database. First, initialize the values for dFRList, dFRcountList, dFRcount = 0, maskList, temp = 0. Where, nof – number of fields, dFRList – distinctFieldRecordList. Afterwards, identify distinct object in the overall database. Subsequently, converting strings to numbers and assigning the numeric value for each data. Then return the mask list. Finally, update TT with masked data

```
Input:  Trajectory Transactions (TT), nof
  Output: Masked TT
  Begin
  1. Initialize dFRList, dFRcountList, dFRcount = 0,
     maskList, temp = 0
  2. For each Transaction
     a. For each field
        i. If(dFRList.contains(object))
           1. doNothing();
        ii. Else
               1. dFRList.add(object);
               2. dFRcount++;
        iii. dFRcountList.add(dFRcount);
        iv. dFRcount = 0;
     b. End For
  3. End For
  4. For each object in dFRList
     a. Temp ++;
     b. maskList.add(temp);
  5. End For
  6. Return maskList
  7. Update TT with masked data
End
```

4.2 Description for ALGORITHM III

Algorithm III is for BM Search to identify frequent utility of trajectory devices using the masking data. First, we initialize the values for tdCount = 0, userTdCount = 0. Where, td-Trajectory Device, tdCount – total td count, utdCountList – user Td count list, pList – pattern List, pCount – patternCount, ivp – invalid pattern

For example in Access pattern – denied access patterns are invalid patterns. Later, get all distinct td and get all distinct patterns. After that, delete TT from MaskedTT and assign FTT to MaskedTT. Finally, return the FTT value

ALGORITHM III – BM Search

```
Input:  MaskedTT
Output: FTT
Begin
  1. Initialize tdCount = 0, userTdCount = 0;
  2. For all MaskedTT // Getting distinct td
     a. If(tdList.contains(td))
        i.   doNothing();
     b. Else
        i.   tdList.add(td)
        ii.  tdCount++;
  3. End For
  4. For each user
     a. For all userTransactions
        i.   If(userTdList.contains(td))
                1. doNothing();
        ii.  Else
                1. userTdList(user).add(td);
                2. userTdCount++;
        iii. utdCountList.add(userTdCount);
        iv.  userTdCount = 0;
     b. End For
  5. End For
  6. For each utdCountList
     a. If(utdCountList(td) == tdCount)
```

```
            i.      doNothing();
        b.Else
            i.      Delete TT from MaskedTT
    7.  End For
    8.  For all MaskedTT // Getting distinct pattern
        a.If(pList.contains(pattern))
            i.      doNothing();
        b.Else
            i.      pList.add(pattern);
            ii.     pCount++;
    9.  End For
    10. For each user
        a.For all userTransactions
            i.      If(TT.contains(ivp))
                        1.  Delete TT from Masked TT
        b.End For
    11. End For
    12. After Deletion, MaskedTT = FTT
    13. Return FTT
End
```

5. PERFORMANCE ANALYSIS

5.1 Time Consumption

The Time consumption for UP-Growth and BM Search is analyzed and compared in Fig.1.

Time consumption for BM Search is lower compared than the UP-Growth algorithm. Here, the x-axis represents time in milliseconds and the y-axis the algorithms.

5.2 Processing Memory Consumption

The Memory Consumption for UP-Growth and BM Search algorithms are compared and analyzed in Fig.2

Memory consumption for BM Search is diminished compare than the UP-Growth algorithm. Here, the x-axis represents an algorithm and y-axis represents the memory in bytes.

5.3 CPU Usage

The usage of CPU for UP-Growth and BM Search algorithms are compared and analyzed in Fig.3. The CPU usage is decreased measured than the UP-Growth algorithm. Decisively, BM Search algorithm is outstanding. Here, the x-axis represents the algorithm and y-axis represents the CPU usage in percentage

5.4 Accuracy Evaluation

The efficiency assessment of UP-Growth and BM Search algorithms are analyzed and measured in Fig.4.

The accuracy of BM Search algorithm is exceptional. The accuracy evaluation of BM Search is superior compared than the UP-Growth algorithm. The accuracy evaluation of BM Search algorithm is improved correlated than the UP-Growthalgorithm. Here, the x-axis represents the algorithms and y-axis represents the accuracy in percentage

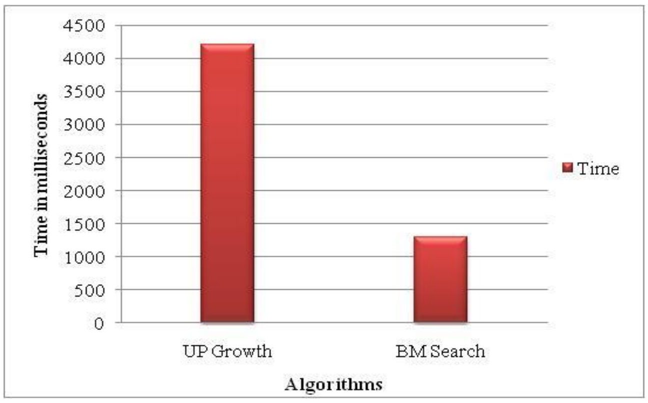

Fig.1 (a) Time Consumption for UP-Growth and BM Search algorithms

5.5 Data Reduction

The devaluation of data for UP-Growth and BM Search algorithms are compared and analyzed in Fig.5. The data reduction for BM Search is high compared than the UP-Growth algorithms. Data reduction for BM Search algorithm is exceptional compared than the UP-Growth algorithm. Here, the x-axis represents the algorithms and y-axis represents the data reduction in percentage. Completely, the BM Search algorithm is leading.

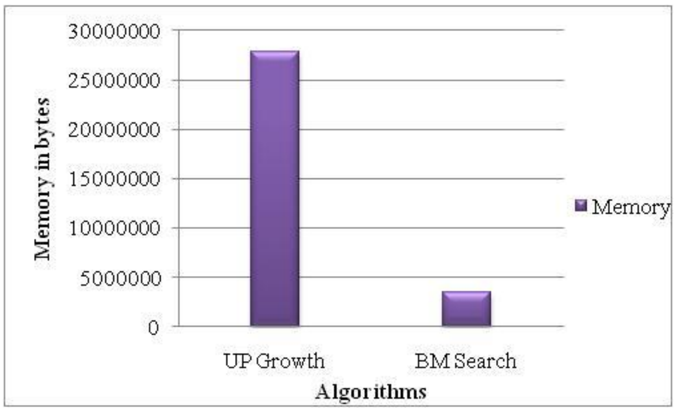

Fig.2 (a) Memory consumption for UP-Growth and BM Search algorithms

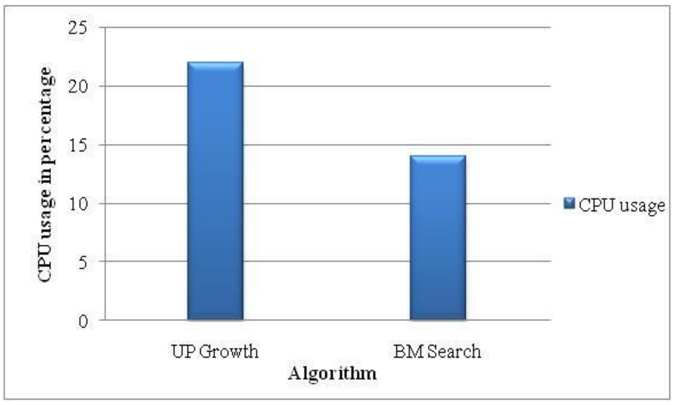

Fig.3(a) CPU Usage for UP-Growth and BM Search algorithms

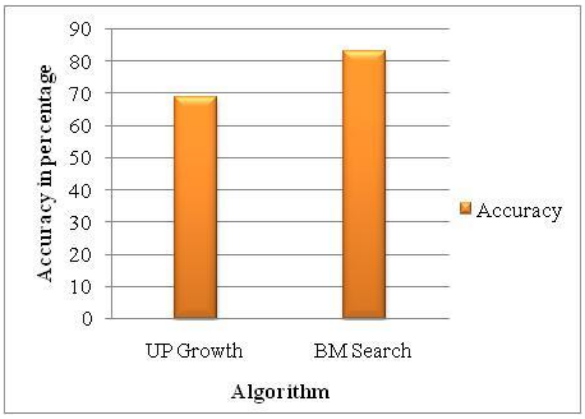

Fig.4 (a) Accuracy Evaluation of UP-Growth and BM Search Algorithms

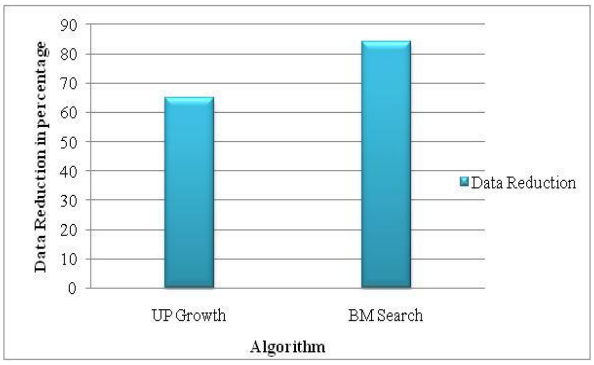

Fig.5(a) Data Reduction for UP-Growth and BM Search Algorithms

6. CONCLUSION

In this paper, the two algorithms were proposed namely, UP-Growth (Utility Pattern Growth) and BM Search (Bit Mask Search). The BM Search algorithm is more appropriate and best in terms of Time consumption, Memory consumption, CPU usage, Accuracy Evaluation, and Data reduction. The BM Search algorithm provides better results compared with the UP-Growth algorithm. An experimental results shows the performance of UP-Growth and BM Search algorithms. After the overall performance of the two algorithms, the BM Search algorithm is preferred. A new Bit Search Mask Search (BM Search) algorithm is to mine frequent entity sets. Quantitative proof that BM Search is superior to UP-Growth algorithm, because it diminishes the memory space for finding the frequent entity sets, it increments the overall efficiency of the bit search in terms of time complexity. BM Search gives better result especially when transactions are large.

7. REFERENCES

[1] Tseng, V, et al., Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases,IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 2013, vol. 25, pp. 1 - 15.

[2] Boaddh J, et al., Empirical Evaluation of Bit Mask Search for Mining Frequent Item Sets, International Journal of Engineering and Innovative Technology (IJEIT), 2012, vol. 2, no. 6, pp. 1-15.

[3] Tseng VS, et al., "UP-Growth: an efficient algorithm for high utility itemset mining," in Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, 2010, pp. 253-262.

[4] Abaya SA, Association Rule Mining based on Apriori Algorithm in Minimizing Candidate Generation, International Journal of Scientific & Engineering Research Volume, 2012, vol. 3, pp. 1-4.

[5] Wang L, et al., Accelerating probabilistic frequent itemset mining: a model-based approach, in Proceedings of the 19th ACM international conference on Information and knowledge management, 2010, pp. 429-438.

[6] Hong TP, et al., Effective utility mining with the measure of average utility, Expert Systems with Applications, 2011, vol. 38, no. 7, pp. 8259-8265.

[7] Dr. Ramaraj E, A General Survey on Multidimensional And Quantitative Association Rule Mining Algorithms, International Journal of Engineering Research and Applications, 2013, vol. 3, no. 4, pp. 1442-1448.

[8] Ahmed CF, et al., HUC-Prune: an efficient candidate pruning technique to mine high utility patterns, Applied Intelligence, 2011, vol. 34, no. 2, pp. 181-198.

[9] Yıldız B and Ergenç B, Comparison of two association rule mining algorithms without candidate generation, in Proceedings 10th IASTED international conference on artificial intelligence and applications, AIA, 2010, pp. 450-457.

[10] Zhang F, et al., Accelerating frequent itemset mining on graphics processing units, The Journal of Supercomputing, 2013, pp. 1-24.

[11] Schlegel B, et al., Memory-efficient frequent-itemset mining, in Proceedings of the 14th International Conference on Extending Database Technology, 2011, pp. 461-472.

[12] Ustundag A, et al., Fuzzy rule-based system for the economic analysis of RFID investments, Expert Systems with Applications, 2010, vol. 37, no. 7, pp. 5300-5306.

[13] Kim Y, et al., Mining Frequent Itemsets with Normalized Weight in Continuous Data Streams, JIPS, 2010, vol. 6, no. 1, pp. 79-90.

[14] Roh GP, et al., Supporting pattern-matching queries over trajectories on road networks, Knowledge and Data Engineering, IEEE Transactions on, 2011, vol. 23, no. 11, pp. 1753-1758.

[15] Liu Y, et al., Mining frequent trajectory patterns for activity monitoring using radio frequency tag arrays, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, 2012, vol. 23, pp. 1-12.