RESEARCH ARTICLE OPEN ACCESS

PPCT-FIM: Prepost Computation Tree Based Frequent Itemset Mining

Mr. Phalke Sagar Balkrishna*, Prof. Rajpure Amol S**
Department of Computer Engineering
*(Dattakala Group of Institutions, Faculty of Engineering, Swami Chincholi, Daund, Pune
Email: sagarphalkeit@gmail.com)
Department of Computer Engineering
**(Dattakala Group of Institutions, Faculty of Engineering, Swami Chincholi, Daund, Pune
Email: amolrajpure9@gmail.com)

----------**********----------

Abstract:

Mining successive itemsets is a fundamental issue in information mining and assumes a significant job in numerous information mining applications. Lately, some itemset portrayals dependent on hub sets have been proposed, which have demonstrated to be proficient for mining incessant itemsets. In our project, we propose a PrePost Computation Tree based Frequent Itemset Mining (PPCT-FIM), calculation for mining continuous itemsets. To accomplish high productivity, PPCT-FIM finds visit itemsets utilizing a set-identification tree with a half and half pursuit system and legitimately lists visit itemsets without applicant age under some case. For assessing the presentation of PPCT-FIM, we have direct broad trials to contrast it against and existing driving calculations on an assortment of genuine and manufactured datasets. The trial results show that PPCT-FIM is fundamentally quicker than PFIM calculations

Keywords — **Data mining, Frequent itemset, Mining Massive Data, Pruning Rule, Incremental Update**

----------**********----------

I. INTRODUCTION

Frequent itemset mining is a significant activity that has been broadly concentrated in numerous down to earth applications, for example, information mining [1]-[3], programming bug location [4], spatiotemporal information examination and organic investigation [5]. Given an exchange table, wherein every exchange contains a lot of things, visit itemset mining restores all arrangements of things whose frequencies (additionally alluded to as help of the arrangement of things) in the table are over a given limit. Because of its down to earth significance, since initially proposed in [6], visit itemset mining has gotten broad considerations and numerous calculations are proposed [7]-[9]. The current continuous itemset mining calculations can be ordered into two gatherings: applicant age based calculations [10]-[14] and design development based calculations [15]-[17]. The competitor age based calculations initially produce applicant itemsets and these up-and-comers are approved against the exchange table to distinguish visit

itemsets. The counter monotone property is used in competitor age based calculations to prune search space. Be that as it may, the competitor age based calculations require various pass table sweeps and this will bring about a high I/O cost on huge information. The example development based calculations don't produce applicants expressly. They develop the uncommon tree-based information structures to keep the fundamental data about the incessant itemsets of the exchange table. By utilization of the developed information structures, the successive itemsets can be processed effectively. Notwithstanding, design development based calculations have the issue that the built information structures are perplexing and typically surpass the accessible memory on huge information. To summarize, the current calculations can't register visit itemsets on monstrous information effectively. In visit itemset mining, the quantity of the incessant itemsets ordinarily is touchy to the estimation of the help edge. In the event that the help edge is little, there will be an enormous number of continuous itemsets and it is hard for the clients to settle on proficient choices. Despite what might be expected, if the help limit is huge, it is conceivable that no continuous itemsets can be found or the fascinating itemsets might be missed. In this way, an appropriate help edge is critical for the down to earth visit itemset mining and the clients regularly need to perform visit itemset digging for a few times before the palatable help limit is resolved. The procedure regularly is intelligent. On gigantic information, the current calculations regularly need a long execution time to figure visit itemsets and this will influence clients' working proficiency genuinely [18].

The focal point of this work is to locate another effective calculation to register visit itemsets on monstrous information rapidly. One valuable stunt, which is embraced to accelerate the execution in the current calculations, is to reuse the work done in the tallying activity of the shorter itemsets for that of the more extended itemsets. In this work, we need to use this reuse thought to an a lot bigger degree. In common enormous information applications, with the expanding information volume and the plate I/O bottleneck, information for the most part is put away in read/attach just mode [19]. Along these lines, the general informational index can be separated into two sections: the a lot bigger old informational collection putting away the chronicled information, and the relative little new informational collection putting away the recently created information. In light of the portrayal over, this work devises another PFIM calculation (Pre computation-based Frequent Itemset Mining calculation) on huge information, which uses the pre-developed continuous itemsets on the old informational collection to restore the successive itemsets rapidly. Since the too little estimation of help edge will produce too many incessant itemsets, we expect in this work there exists a lower bound omega of the help limit determined by the clients in commonsense applications. Due to the genuine/affix just mode, given the old table TO, PFIM first pre-builds the continuous itemsets (allude to as semi visit itemsets in this work) whose supports are no not as much as omega.

The new exchanges are collected in the new table T1. Exploiting the pre-developed semi visit itemsets, given the predefined bolster edge, PFIM can process the regular itemsets on TO U T1 rapidly. During the time spent execution of PFIM, three pruning rules are formulated in this work to decrease the quantity of up-and-comer visit itemsets. A gradual update system is proposed in this work to rapidly refresh the semi visit itemsets when TO and T1 are combined. The broad examinations are directed on engineered and genuine informational indexes. The test results show that, PFIM outflanks the current calculations

fundamentally; it runs two sets of greatness quicker than the most recent calculation.

The input of PPCT-FIM is a transaction database and a threshold named minsup (a value between 0 and 100 %). A transaction database is a set of transactions. Each transaction is a set of items. For example, consider the following transaction database. It contains 5 exchanges (t1, t2,...t5) and 5 things (1, 2, 3, 4, and 5). For instance, the main exchange speaks to the arrangement of things 1, 3 and 4. Note that a thing isn't permitted showing up twice in a similar exchange and those things are thought to be arranged by lexicographical request in an exchange.

II. Related Work

In this work, we want to utilize this reuse idea to a much larger degree. In typical massive data applications, with the increasing data volume and the disk I/O bottleneck, data usually is stored in read/append-only mode [19]. Therefore, the overall data set can be divided into two parts: the much larger old data set storing the historical data, and the relative small new data set storing the newly generated data. Based on the description above, this work devises a new PFIM algorithm (Pre computation-based Frequent Itemset Mining algorithm) on massive data, which utilizes the pre-constructed frequent itemsets on the old data set to return the frequent itemsets quickly. Since the too small value of support threshold will generate too many frequent itemsets, we assume in this work that there exists a lower bound omega of the support threshold specified by the users in practical applications. Because of the real/append-only mode, given the old table TO, PFIM first pre-constructs the frequent itemsets (refer to as quasi-frequent itemsets in this work) whose supports are no less than omega. The new transactions are accumulated in the new table T1. Taking advantage of the pre-constructed quasi-frequent itemsets, given the specified support threshold, PFIM can compute the frequent itemsets on TO UT1 quickly.

The candidate-generation-based algorithms firstly generate the candidates of the frequent itemsets, then the candidates are validated against the transaction table, and the frequent itemsets are discovered. Apriori algorithm [11], [20] adopts a level-wise execution mode. It uses the downward closure property, i.e. any superset of an infrequent itemset must also be infrequent, to prune the search space. By a pass of scan on the transaction table, it first counts the item occurrences to find the frequent 1-itemsets F1. Subsequently, the frequent k-itemsets in Fk are used to generate the candidates CkC1 of the frequent (kC 1)-itemsets. Another pass of scan is needed to compute the supports of candidates inCkC1 to find the frequent (k C 1)-itemsets FkC1. This process iterates similarly until theFkC1 is empty. Apriori algorithm often needs multiple passes over table; it will incur a high I/O cost on massive data. Savasere et al. [12] propose Partition algorithm to generate frequent itemsets by reading the transaction table at most two times. The execution of Partition consists of two stages. In the first stage, Partition algorithm divides the table into a number of non-overlapping partitions in terms of the allocated memory, and the local frequent itemsets for each partition are computed. All the local frequent itemsets are merged at the end of first stage to generate the candidates of frequent itemsets. In the second phase, another pass over table is performed to acquire the support of the candidates and the global frequent itemsets can be discovered.

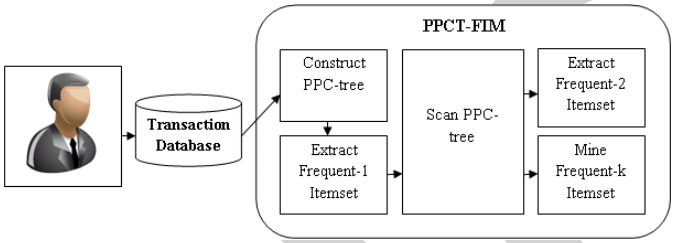
III. Architecture of Proposed System

1. PPC-Tree Construction

Given a database and a minimum support threshold, the PPC-tree construction is defined as follows.
PPC-tree is a tree structure:

(1) It consists of one root labeled as "null", and a set of item prefix subtrees as the children of the root.

(2) Each node in the item prefix subtree consists of five fields: item-name, count, childrenlist, pre-order, and post-order. Item-name registers which item this node represents. Count registers the number of transactions presented by the portion of the path reaching this node. Children-list registers all children of the node. Pre-order is the pre-order number of the node and post-order is the post-order number of the node.



2. Frequent-1 Itemset Extraction:

This module extracts frequent-1 itemset using PPC-Tree. Scan Transaction database once to find F1, the set of frequent-1 items using PPC-Tree. Followed by, Sort F1 in support descending orders as L1. Delete all infrequent items from T and then sort T as Tf according to the order of L1.

3. Scan PPC-Tree and Frequent-2 Itemset Extraction

This module extracts frequent-2 itemset using PPC-Tree with Frequent-1 Itemset. First, this module generates the Nodeset of each frequent item by scanning the PPC-tree. Furthermore, Build 2-itemset DN() to generate the DiffNodeset of each 2-itemset. Followed by, it computes the support of each 2-itemset

4. Mine Frequent-k Itemset

This module extracts frequent-k itemset using PPC-Tree with Frequent-2 Itemset. First, this module generates the Nodeset of each frequent item by scanning the PPC-tree. Furthermore,
This module extracts all frequent k-itemsets (k >= 3) by calling procedure
Constructing Pattern Tree() method to generate all frequent k-itemsets (k >= 3) extended
from frequent 2-itemsets.

IV. PROPOSED ALGORITHM

Input: Transaction Database (TD), Minimum Support (minsup)
Output: Frequent Itemsets

Step 1: Initializes F, which is used to store frequent itemsets, by setting it to be null.

Step 2 : Constructs the PPC-tree and finds F1, the set of all frequent 1- itemset, by calling procedure Construct PPC-tree().

Step 3: Generate the Nodeset of each frequent item by scanning the PPC-tree.

Step 4: Calls procedure Build 2-itemset DN() to generate the DiffNodeset of each 2-itemset.

Step 5: Computes the support of each 2-itemset.

Step 6: Check whether ixiy is frequent or not**.**

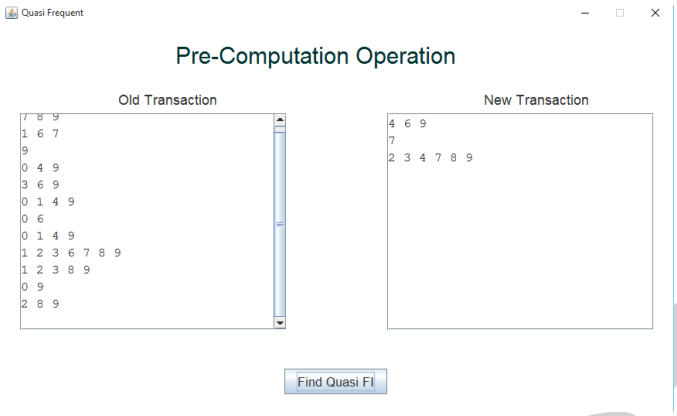
Step 7: Generate all frequent k-itemsets (k >= 3) by calling procedure Constructing Pattern Tree () to generate all frequent k itemsets (k > = 3) extended from frequent 2itemsets

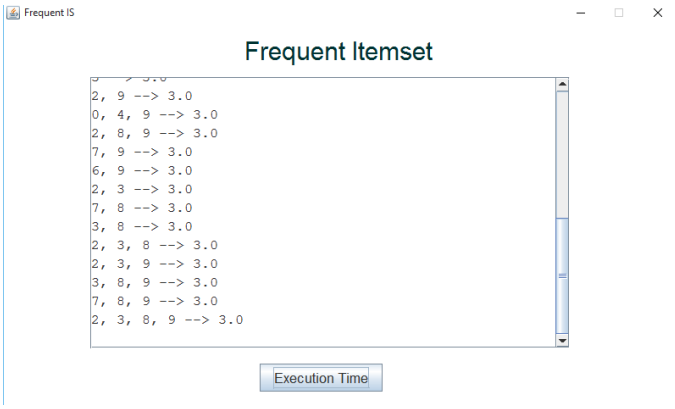
V. MATHEMATICAL MODEL

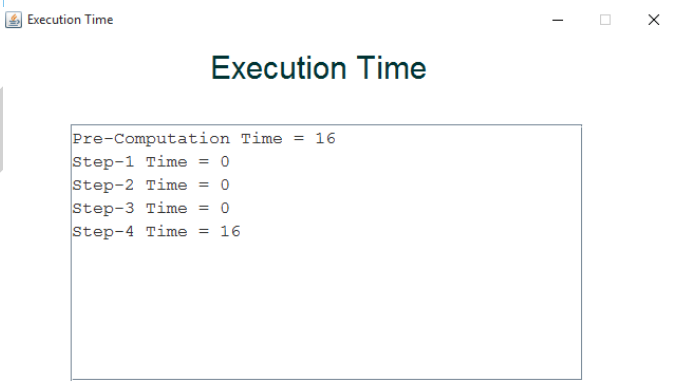
TD –> Transaction Database
minsup –> Minimum Support
F1 –> Frequent-1 Itemset
F2 –> Frequent-2 Itemset
Fk –> Frequent-k Itemset

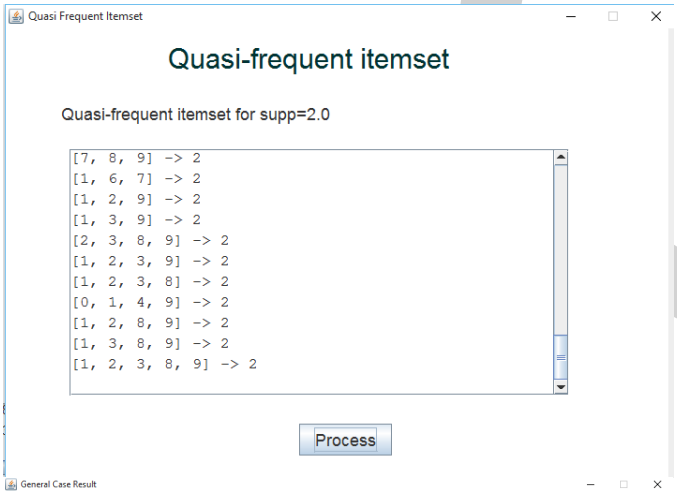
resultLen –> the size of the current itemset
nlLenSum –> Node list length of the current itemset

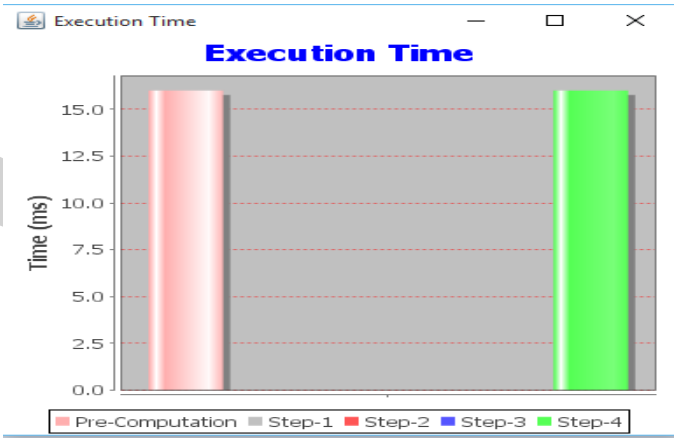
VI. SIMULATION RESULTS

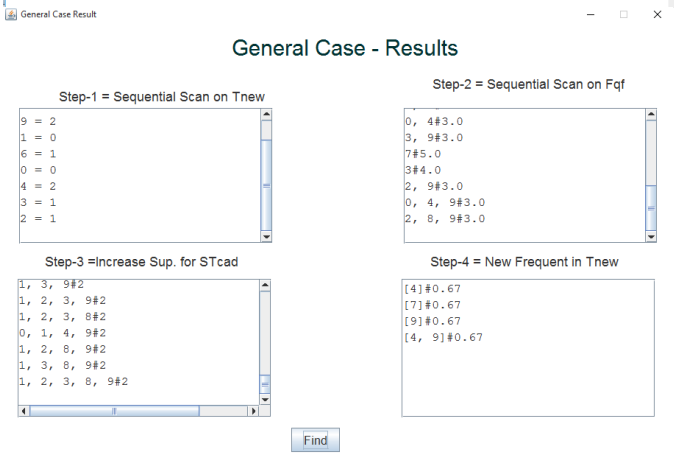












CONCLUSION AND FUTURE WORK

We present a novel structure called PPCT-FIM to facilitate the process of mining frequent itemsets. Based on PFIM, an algorithm named PPCT-FIM is proposed to fast find all frequent itemsets in databases. Compared with Nodeset, the key advantage of PPCT-FIM lies in that its size much smaller. This makes PPCT-FIM more suitable for

mining frequent itemsets. The extensive experiments show that PPCT-FIM is favorable. PPCT-FIM proves to be state-of-the-art since it always runs fastest on all datasets with different minimum supports and occupied less memory when compared with previous leading algorithms.

REFERENCES

[1] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute based encryption," Proc. IEEE Symposium on Security and Privacy (SP '07), pp. 321-334, 2007

[2] H. Chen, Y. Hu, P. Lee, and Y. Tang, "Nccloud: A network-coding-based storage system in a cloud-of-clouds," 2013.

[3] H. Cui, X. Yi, and S. Nepal, "Achieving scalable access control over encrypted data for edge computing networks," IEEE Access, vol. 6, pp. 30049–30059, 2018.

[4] K. Xue, W. Chen, W. Li, J. Hong, and P. Hong, "Combining data owner-side and cloud-side access control for encrypted cloud storage," IEEE Transactions on Information Forensics and Security, vol. 13, no. 8, pp. 2062–2074, 2018.

[5] C. Delerabl´ee, "Identity-based broadcast encryption with constant size ciphertexts and private keys," Proc. International Conf. on the Theory and Application of Cryptology and Information Security (ASIACRYPT '2007), pp. 200-215, 2007.

[6]. B. Lang, J. Wang, and Y. Liu, "Achieving flexible and self-contained data protection in cloud computing," IEEE Access, vol. 5, pp. 1510- 1523, 2017.

[7]. L. Liu, Y. Zhang, and X. Li, "KeyD: secure key-deduplication with identity-based broadcast encryption," IEEE Transactions on Cloud Computing, 2018, https://ieeexplore.ieee.org/document/8458136.

[8]. N. Paladi, C. Gehrmann, and A. Michalas, "Providing user security guarantees in public infrastructure clouds," IEEE Transactions on Cloud Computing, vol. 5, no. 3, pp. 405-419, 2017.

[9].T. G. Papaioannou, N. Bonvin, and K. Aberer, "Scalia: an adaptive scheme for efficient multi-cloud storage," in Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. IEEE Computer Society Press, 2012, p. 20.

[10]. Z. Yan, X. Li, M.Wang, and A. V. Vasilakos, "Flexible data access control based on trust and reputation in cloud computing," IEEE Transactions on Cloud Computing, vol. 5, no. 3, pp. 485-498, 2017.

[11]. H. He, R. Li, X. Dong, and Z. Zhang, "Secure, efficient and fine-grained data access control mechanism for P2P storage cloud," IEEE Transactions on Cloud Computing, vol. 2, no. 4, pp. 471-484, 2014.

[12]. Z. Qin, H. Xiong, S. Wu, and J. Batamuliza, "A survey of proxy reencryption for secure data sharing in cloud computing," IEEE Transactions on Services Computing, 2018, https://ieeexplore.ieee.org/docu ment/7448446.

[13]. J. Son, D. Kim, R. Hussain, and H. Oh, "Conditional proxy reencryption for secure big data group sharing in cloud environment," Proc. of 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 541–546, 2014

[14]. S. Choy, B. Wong, G. Simon, and C. Rosenberg, "A hybrid edge-cloud architecture for reducing on-demand gaming latency," Multimedia Systems, pp. 1–17, 2014.

[15]. L. Jiang, and D. Guo "Dynamic encrypted data sharing scheme based on conditional proxy broadcast re-encryption for cloud storage," IEEE Access, vol. 5, pp. 13336 – 13345, 2017.

[16]. K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, Y. Yu, and A. Yang, "A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," Future Generation Computer Systems, vol. 52, pp. 95-108, 2015.

[17.] Q. Huang, W. Yue, Y. He, and Y. Yang, "Secure identity-based data sharing
and profile matching for mobile healthcare social networks in cloud computing," IEEE Access, vol. 6, pp. 36584–36594, 2018.

[18]. V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," Proc. 13th ACM Conf. on Computer and Communications Security (CCS '06), pp.89- 98, 2006.

[19]. S. Wang, K. Liang, J. K. Liu, J. Chen, J. Yu, and W. Xie, "Attribute based
data sharing scheme revisited in cloud computing," IEEE Transactions on Information Forensics and Security, vol. 11, no. 8, pp. 1661–1673, 2016.

[20]. L. Guo, C. Zhang, H. Yue, and Y. Fang, "A privacy-preserving social assisted mobile content dissemination

scheme in DTNs," Proc. 32nd IEEE International Conf. on Computer Communications (INFOCOM '2013), pp. 2301-2309, 2013.

[21]. W. Teng, G. Yang, Y. Xiang, T. Zhang, and D. Wang, "Attribute based access control with constant-size ciphertext in cloud computing," IEEE Transactions on Cloud Computing, vol. 5, no. 4, pp. 617-627, 2017.

[22]. K. Seol, Y. Kim, E. Lee, Y. Seo, and D. Baik, "Privacy-preserving attribute- based access control model for XML-based electronic health record system," IEEE Access, vol. 6, pp. 9114-9128, 2018.

[23]. J. Weng, R. H. Deng, X. Ding, C. K. Chu, and J. Lai, "Conditional proxy reencryption secure against chosen-ciphertext attack," in Proc. of 4th International Symposium on Information, Computer, and Communications Security (ASIACCS '09), pp. 322-332, 2009.

[24]. P. Xu, T. Jiao, Q.Wu,W.Wang, and H. Jin, "Conditional identity based broadcast proxy re-encryption and its application to cloud email," IEEE Trans. on Computers, vol. 65, no. 1, pp. 66-79, 2016.

[25]. S. Jiang, T. Jiang, and L. Wang, "Secure and efficient cloud data deduplication with ownership management,", IEEE Transactions on Services Computing, https://ieeexplore.ieee.org