Information Management and Computer Science (IMCS)

DOI : http://doi.org/10.26480/imcs.01.2019.15.24

Online ISSN : 2616-5961

CODEN: IMCSBZ

PARALLEL AND DISTRIBUTED ASSOCIATION RULE MINING ALGORITHMS: A RECENT SURVEY

Sudarsan Biswas[1], Neepa Biswas[2], Kartick Chandra Mondal[2]

[1]Department of Information Technology RCC Institute of Information Technology Kolkata, India
[2]Department of Information Technology Jadavpur University Kolkata, India.
**Corresponding Author Email: biswas.sudarsan@gmail.com, biswas.neepa@gmail.com, kartickjgec@gmail.com*

ARTICLE DETAILS

Article History:

Received 27 July 2019
Accepted 29 August 2019
Available online 05 September 2019

ABSTRACT

Data investigation is an essential key factor now a days due to rapidly growing electronic technology. It generates a large number of transactional data logs from a range of sources devices. Parallel and distributed computing is a useful approach for enhancing the data mining process. The aim of this research is to present a systematic review of parallel association rule mining (PARM) and distributed association rule mining (DARM) approaches. We have observed that the parallelized nature of Apriori, Equivalence class, Hadoop (MapReduce), and Spark proves to be very efficient in PARM and DARM environment. We conclude that this comprehensive review, references cited in this article will convey foremost hypothetical issues and a guideline to the researcher an interesting research direction. The most important hypothetical issue and challenges include the large size of databases, dimensionality of data, indexing schemes of data in the database, data skewness, database location, load balancing strategies, methods of adaptability in incremental databases and orientation of the database.

KEYWORDS

D-sampling, Equivalence Class, MapReduce, Parallel Apriori, Spark.

1. INTRODUCTION

Databases information in this era is intrinsically distributed. This trend of sharing various source of data and generation of immense data volume is certainly showing the way to ridiculous communications costs. Association rule mining (ARM) explores the relationship for finding the meaningful correlation between those large databases. It has been paid more attention to both data mining users and database researchers in the last decade. ARM techniques represent a better choice as they are scalable, highly flexible and can efficiently manage data heterogeneity. It was formally proposed by Agrawal et al. [2] having two counter parts named as PARM and DARM [3]. DARM has become essential for huge and multi scenario database requiring resources, which are heterogeneous and distributed [28]. Recently due to the massive growth of data in organizations, extensive data processing is an essential point of Information Technology. It is not feasible to store and in memory processing of those massive amount of transactional data in organizational databases. ARM is innately disk I/O concentrated task. The I/O costs may be decreased using two approaches by minimizing database scan, or parallelization. This minimization is very essential because the database is generally very large also it is stored in secondary memory. This difficulty is exercised in DARM problem. The aim of this type of DARM problem is solved by parallelizing the disk I/O. In DARM, the database is partitioned between several sites which can perform autonomous parallel computations as well as communication with one site to other sites efficiently [70]. ARM practice may be recognized as distributed and centralized depending on the position of data. For centralized data processing, all data are kept on a single location. In distributed processing, data are placed in several locations. Data are separately accessed from each location [6]. In distributed processing, a collection of sensors, computers, and devices communicate within each other's.

Among the ARM techniques shown in Figure 1, PARM and DARM conveys the most significant role towards rule generation. Prediction and analysis of enormous data volume due to rapidly growing of electronic technologies is a challenging task in recent days. Those huge number of transactional data logs generated from several sensors, Internet relay chats, Network logs, Twitter, Facebook, Online banking, ATM transactions etc. in everyday life. Prediction of those data for finding meaningful patterns often became very much helpful in many recent fields included network attack prediction in intrusion detection system [47] [37], locate the causal relationship between drugs and their associated adverse drug reactions [19], detection of suspicious activities in web application [75], advertisement, market basket analysis [2], analysis of business risk , bioinformatics [65], epidemiology, sports, social networking, fluid dynamics, fraud detection [23], [61], crime prevention and prevention [63], telecommunication network, statistical disclosure risk assessment [45], cross marketing, crowd mining and cyber security [63], catalog design, weather prediction [48], recommendation system [21] etc. Also, ARM will examine customers buying behavior that assist the retailers in making advertising strategies, marketing policies, catalog design, store layout orientation, etc.

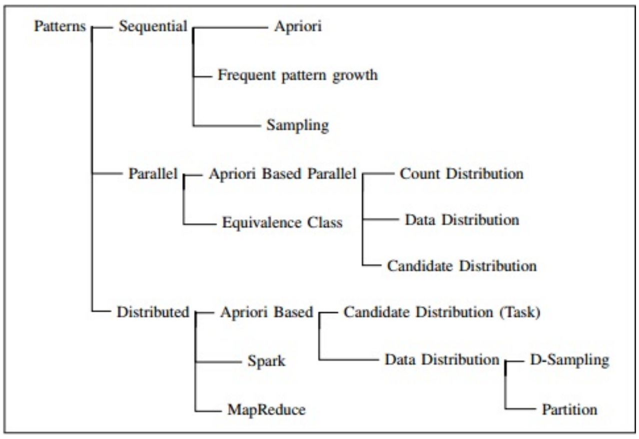

Figure 1: Taxonomy of Pattern Mining Algorithms

In a distributed computing environment, there are m sites *{$S_1, S_2, . . . ,S_m$}* with transactional database TD which is partitioned among m sites into *{$TD_1,TD_2, . . .,TD_n$}* respectively. Assuming the dimension of partitions are *TD_i* be *D_i*, where *i= {1, 2, 3 . . . m}*. In addition, local support count or global support count can be measured *A.sup* and *$A.sup_i$* of A at site *S_i* respectively [13]. Let user assigned minimum support constraint is s. An itemset A noted as large if A.sup ≥ s x *D* globally whereas itemset A is reported as large at site Si locally, if A.supi ≥ s x *D_i* [28], [69].

1.1 Characteristic of PARM

Mainly architecture of parallel computing can be divided into two branches: distributed memory system (share nothing architectures) and shared memory system [10], [93]. The parallel computing framework includes the following :
(i) Type of parallelism is required (task or data).
(ii) Hardware platform of the system.
(iii) Type of load balancing approaches are used.

1.2 Characteristic of DARM

Discover useful correlation among two nonempty frequent itemsets from distributed large databases.
(i) A distributed system is loosely coupled that may consist cluster of nodes situated in various places, connected via LAN or internet.
(ii) Message passing interface design is used for communication between multiple distributed sites that increase the scalability [6].

This overall paper is structured as follows. Section 2 presents the contribution to this paper. Section 3 gives a comprehensive survey of parallel pattern mining algorithms. Section 4 presents a comprehensive survey of distributed pattern mining algorithms. Section 5 presents a review and year wise comparative study among PARM and DARM algorithms. The conclusion of this survey is included in Section 6.

2. RARENESS OF THIS SURVEY

In this paper, we are presenting a comprehensive survey of most exercised PARM and DARM algorithms. This can be considered both as an introduction as well as a guideline to advances opportunities for research in ARM. We have given a systematic and rigorous assessment of significant developments in this field. Those are the following:
(i) Making the classification of pattern mining algorithms using information collected from the different literature shown in Figure 1.
(ii) An extensive scrutiny has been performed for both PARM and DARM type of research and development. We have discussed different algorithmic conceptualization, analytical performance characteristics, year wise research impact among the academic and research world and more.
(iii) In PARM the review of count distribution, data distribution, candidate distribution, equivalence class based parallel algorithms are reported in table 2, 3, 4, 5 respectively.
(iv) In DARM the review of candidate distribution, data distribution (Sampling, Partition) and Spark & MapReduce based distributed algorithms are shown in table 6, 7, 8, 9 respectively.
(v) Survey on PARM and DARM mining techniques and presents an algorithm wise comprehensive comparison between those algorithms using different key factor like algorithms names, database orientation, name of parent algorithm, the type of data structure or approaches are used, references, year of publication, citations of those literature is shown in table 10 and table 11 respectively.
(vi) Presents year wise comparative study between PARM and DARM research approaches and identification research growth form last two decades in this area are shown in Figure 2
(vii) Presents year wise comparative study between PARM and DARM algorithmic count to recognize the research challenges, important issues in this domain shown in Figure 3.

3. SEQUENTIAL PATTERN MINING ALGORITHMS

3.1 Apriori (Candidate Generation)

Finding rules from largely distributed data sources is a demanding task [80] where ARM is used to generate hidden rules and the relationship between data [2]. The model of association rules is represented as knowledge based on an unsupervised learning method. A formal definition of association rule *(A ⇒B)* where $A \subseteq T$, $B \subseteq T$ and *(A ∩ B)*= φ. Rules must convey a certain amount of support and confidence constraint. However, initially, the task of rule mining is to discover the itemsets that are generated frequently from transactional database. Where support (*A ⇒ B)* ≥ minimum support. Secondly, project all rules from frequent itemsets with confidence *(A ⇒B)* ≥ minimum confidence [58]. An item or itemset is a collection of patterns that belongs to *I= {i_1, i_2, i_3, . . ., i_n}*. An itemsets X with k distinct items is known as {k-itemsets} [2]. Support *(A ⇒ B)* = s%, is the frequency of item's occurrence {AB} from a transactional database. Confidence *(A ⇒B)* = c%, represent strength of the rules. A single minsup and minconf constraint are reported by the user to identify those rules. Agrawal et al. proposed an Apriori [4] is well exercise traditional sequential technique to select ARM from those transactional databases. Apriori principals:
(i) Itemset Generation: Any subset of frequent items must be frequent. Also, it will significantly minimize the item set search space. For example, if {X, Y, Z} are frequent then {X, Y} and {Y, Z} also be frequent.
(ii) Itemset Pruning: Any infrequent patterns superset will not be checked for itemset generation.

3.2 Frequent Pattern Growth

Frequent Pattern Growth (FP growth) approach was suggested by Han et al. [30] that generates a list of frequent patterns except generating any candidate similar to Apriori [4]. The weakness of Apriori has overcome by FP Growth using divide and conquer approach. It takes only two database scans for frequent pattern generation by downward closure property. During the first scan, it counts the number of occurrences for each itemsets. The second scan builds the initial FP tree that keeps the frequency details about the original database [18]. FP growth includes two stages for mining the frequent pattern.
(i) Initially construct compressed data structure known as FP tree. It is an improved version of bidirectional prefix tree structure that allows bottom up scanning. Every branch of the tree represents a frequent itemsets. Prefixes of the corresponding branches are usually represented by every overlapping itemsets.
(ii) Next, acquire the frequent itemsets directly from the FP Tree. Create a conditional pattern tree extracted from an initial suffix pattern. After that recursively create its conditional FP tree as per the equal order of magnitude.

Advantages of FP growth:
- Used compact data structure.
- No candidate generation and test process.
- Building of FP tree without pattern matching.
- Counting frequent pattern.
- Two scans over the database reduces repeated costly database scan rather than Apriori.

Disadvantages of FP growth:
- For large pattern, FP tree construction complexity is very high.
- It does not accommodate into main memory with bigger FP tree.
- Generation of FP tree is expensive.

3.3 Sampling

Sampling is a greedy approach to find out frequent item sets from transactional database suggested by Toivonen et al. [78]. It has taken a random sample from transactions in the database efficiently for selecting the ARM. Where sample size is a prime factor to provide the good approximation of frequent sets. Zaki et al. shown in [91] how random sampling of transactions is useful for finding ARM. However, they have considered the MethodA algorithm for identifying samples of the database. MethodA considerably speeds up the sampling process by well determining the number of records to overlook before chosen the new samples.

Advantage of Sampling:
- Single scan is required to generate rules very efficiently from a large database.
- Reducing disk, I/O cost by significantly decreasing the number of transactions to be measured.
- Sampling can enhance the rules generation process by higher than the order of its previous scale.

Disadvantage of Sampling:
- Second pass is required to find missing frequent sets.

4. PARALLEL PATTERN MINING ALGORITHMS

Two major approaches for exploiting parallelism within ARM algorithms are recognized as task parallelism and data parallelism [6].
1) Data parallelism: In this model, the training set is divided among each processor either horizontally or vertically segments. All processors create the same model synchronously where each processor works on a different portion of the database with essential message exchange.
2) Task parallelism: In this model, total works are distributed among the processors of a parallel machine with each processor performing a unique part of a learning model prior to synchronizing with the other processors to form a global model.

4.1 Apriori Based Parallel

Various algorithms have been formally developed in the parallel frameworks [1], [3], [13], [87], [90]. Agrawal et al. presents a different version of Apriori like count distribution (CD), data distribution (DD) and candidate distribution (Can.D) of sequential Apriori [4]. Those parallel approaches are used for label wise search for candidate generation that has O(m.c) communication cost for each phase. Where m and c is the number of data sites and size of candidates itemsets respectively. Also, it will take multiple scans of the database for itemset generation. For example, if the database has N number of transactions then it needs (N+1) scan. table 1 describes the taxonomy of parallel Apriori algorithms. Assessment of CD, DD and Can.D approach are discussed based on their parallelism type, number of database scan, database operational strategy, data structure mentioned as DS and their system architecture.

Table 1: Review of Parallel Apriori Algorithms

Parallelism	Assessment of Parallel Apriori				
	Algo	***Scan***	***Database***	***DS***	***Architecture***
Data	CD	N+1	Partition	Hash tree	[a] DSN
Task	DD	N+1	RRP	Hash tree	[a] DSN
Task	Can. D	N+1	Replicated	Hash tree	[a] DSN

[a]Distributed Shared nothing, RRP Round robin partition, .

4.1.1 Count Distribution (CD)

CD is a sequential Apriori [4] based candidates generation approach by redundant computations in parallel, else processors keep idle to avoid communication cost [3]. It has used DSN architecture for mining the rules from distributed databases. The main objective is to be parallelizing the computation either using a horizontal or vertical partition of the database. In horizontal partition contain the whole transaction in one partition whereas vertical partition contains the same transaction in a different partition. It makes database partitioned into equal sized blocks among all the processors (or different sites) [12]. Each site is identified the biased support of candidates itemsets from local partitions. All processors having its own private memory and disks. All sites are communicated using message passing interface (MPI) having n(n-1) message exchange at each pass. Each processor finds out global itemset frequency by interchanging local itemset frequency from other processors [14]. CD can generate disjoint candidate sets independently. Except for global minimization, each processor exercises traditional Apriori on locally loaded transactions. table 2 describes count distribution based PARM type of algorithms. Here we are considering all parallel algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the parallel algorithms are perceptible in this table.

Table 2: Algorithmic Review of PARM

Parallel Algorithms	Algorithms for CD		
	Parent Algo	Performance Criteria	Year
DHP [57]	CD	Hash table size	1995
PDM [58]	DHP	Count exchange	1995
NPA, SPA [74]	CD	Memory size	1996
CCPD [89]	CD	Adaptive hash table size	1996
HPA, HPA-LED [74]	CD	Hash join	1996
DIC [9]	CD	Reordering items	1997
APM [12]	DIC	Virtual Partition Pruning	1998
FPM [14]	CD	Entropy based partition	1998
APM-DIC [12]	APM	Impulsive candidate generation	1998
NPGM [73]	NPA	Candidate itemset partition	1998
APM-AIC [12]	APM	k clustering	1998
DAA [46]	FPM	Principal component analysis	2001
DCP [60]	CD	Heuristics pruning	2001
DCI [52]	DCP	K way itemset intersection	2001
PHP [55]	DHP	Hash table size	2001
kDCI [43]	DCP	FSC	2003
OPT-DIC [56]	DIC	Scan reduction	2009

Advantages of count distribution:

- Data parallelism algorithm that has superior performance among DD and Can.D [28].
- For minimizing communication cost, exchange only count do not exchange data tuples between the processors.
- FPM has shown better performance than CD always [14].

Disadvantages of count distribution:

- Total memory of the system does not utilize efficiently.
- Candidates replication are required.
- Synchronization are essential at each and every pass with high communication load.

4.1.2 Data Distribution (DD)

DD approach are exercised in DSN architecture where each processor has its own private memory and disks [3]. Each processor communicates through message passing interface [29]. Databases are partitioned between all the processors in same sized blocks. After every passes n(n-1) message exchange and synchronization required accordingly [89]. During iteration, each processor scans the complete database to identify global support like local and isolated partitions. It is considered as good utilization of the overall system memory and better load balancing strategies with minimizing idle time for processors [83]. Because the number of processors is large, even disagreement is a major difficulty and processors can be idle at the time of communication. table 3 describes data distribution based PARM type of algorithms. Here we are considering all parallel algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the parallel algorithms are perceptible in this table.

Table 3: Algorithmic Review of PARM

Algorithms	Algorithms for DD		
	Parent Algo	Performance Criteria	Year
PCCD [89]	DD	Hash tree balancing	1996
HD, IDD [29]	DD	Candidate hash tree	2000
PA [82]	DD	Size of trie tree	2006
WDPA [83]	PA	TID counts	2008
PARMA-P [94]	FP tree	Hash distribution	2009
dRAP independent [8]	Partition	ILP classifier	2012

[PA]Parallel Apriori.

Advantages of data distribution:

- Task parallelism algorithm distributes candidate item sets within the processors.
- Competent utilization of system memory.

Disadvantages of data distribution:

- Produced poor performance with respect to CD.
- Huge communication cost because of broadcasting of local database portion to other processors and perform some redundant work.
- Distribution of transactional task for each processor is not possible.

4.1.3 Candidate Distribution (Can.D)

Can.D approach is used in distributed shared nothing architecture [3]. All Processors are communicated using message passing interface, at initial passes n(n-1) no of message exchange required. During the first iteration, it partitions the data among each processor where every processor may produce disjoint candidates that is not dependent on other processors. Every processor gets an equal amount of task based on heuristics approach [74]. Synchronization is not required after each pass. Without exchanging local data, only global values are exchanged. Accordingly, data is acknowledged asynchronously between sites. Processors may not wait for the complete pruning information to arrive from all the processors. table 4 describes candidate distribution based PARM type of algorithms. Here we are considering all parallel algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the parallel algorithms are perceptible in the table.

Advantage of candidate distribution:

- All processors have balanced workload.

Table 4: Algorithmic Review of PARM

Algorithms	Algorithms for Can.D		
	Parent Algo	Performance Criteria	Year
HPA [74]	Can.D	Hash join	1996
HPGM, H-HPGM [73]	HPA	Candidate itemset partition	1998
^{HH}-TGD, ^{HH}-PGD, ^{HH}-FGD [73],	HH	Candidate itemset partition	1998

^{HH}H-HPGM.

Disadvantages of candidate distribution:

- Redistribution of databases includes additional cost.
- Repeated scan of local database partitions.
- Can.D performs worse than CD.

4.2 Equivalence Class (EC)

Equivalence class transformation (ECLAT) is a parallel ARM algorithm proposed by Zaki et al. [87]. It provides clusters oriented frequent itemsets from vertical transactional databases by equivalence class partitioning. All interconnection between processors accepts a user application that write the memory of remote nodes. That user application makes it convenient to rapid user level messages communication with minimum synchronization costs. The performance of this approach is better than CD algorithm [3]. table 5 describes equivalence class based PARM type of algorithms. Here we are considering all parallel algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the parallel algorithms are perceptible in this table.

Table 5: Algorithmic Review of PARM

Algorithms	**Algorithms for EC**		
	Parent Algo	***Performance Criteria***	***Year***
Eclat [87]	EC	Tid intersection	1997
Par Eclat [90]	Eclat	Bottom up search	1997
Max Eclat [90]	Eclat	Hybrid search	1997
Par Max Eclat [90]	Max Eclat	Hybrid search	1997
Clique [88]	Itemset lattice	Bottom up search	1997
Par Clique [90]	Clique	Bottom up	1997
Top Down [88]	Itemset lattice	Top down search	1997
Max Clique [88]	Clique	Hybrid search	1997
Par Max Clique [90]	Max Clique	Hybrid search	1997
Apr Clique [88]	Clique	Hash tree	1997
VIPER [72]	Snake	Run length encoding	2000
UV-Eclat, U-Eclat [86]	Eclat	Diffset	2003
dEclat [86]	Eclat	DFS	2003
hEclat [95]	Eclat	Hash table	2010
Eclat-opt [24]	Eclat	Tow layer hash table	2013
Bi-Eclat [84]	Eclat	BFS	2014
P-Eclat [26]	Eclat	Partial BFS	2016
Eclat-Growth [44]	Eclat	BSRI	2016
HashEclat [92]	Eclat	Min hash	2019

^{EC}Equivalence class.

These approaches shown in table 5 not only reduce I/O costs by producing only a smaller number of databases scan [90]. It has also minimized computation costs by using competent search strategies.

5. DISTRIBUTED PATTERN MINING ALGORITHMS

Generation of meaningful correlation from distributed large databases. Majority of DARM algorithms are associated with traditional sequential algorithms [4], [30], [68], [78]. Usually, two approaches used to dispense data for parallel processing can be recognized [10], [28].

5.1 Apriori Based DARM Algorithms

Generally, Taxonomy of parallel Apriori based DARM algorithms [16] includes the following:
a) **Candidate Distribution**: Task distribution.
b) **Data Distribution**: Sampling, Partition.

5.1.1 Candidate Distribution Based DARM

DARM has been coupled with Candidate Distribution (Can.D) and Data Distribution (DD). Whereas Can.D has mainly chosen the condition when data is evenly partitioned into different data sites. Each data site calculates support counts for the corresponding candidate itemsets and accumulates at a central site for selecting the large itemsets for succeeding pass. The most well known algorithms used in DARM are CD [3], FDM [11], ODAM [6], FPM [14]. table 6 describes candidate distribution based DARM type of algorithms. Here we are considering all Apriori based task distributed algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the distributed algorithms are perceptible in this table.

Table 6: Algorithmic Review of DARM

Algorithms	**Apriori Based Task**		
	Parent Algo	***Performance Criteria***	***Year***
DMA [13]	CD	Local pruning	1996
FDM [11]	CD	Local & Global pruning	1996
FDM-LB [11]	FDM	Local & Lower bound pruning	1996
FDM-LUB [11]	FDM	Local & Upper bound pruning	1996
FDM-LPP [11]	FDM	Polling site pruning	1996
DMCA [80]	FDM	Local pruning & itemset	2000
DDM [69]	FDM	Priority queue	2001
DDDM [69]	DDM	Local count reduction	2001
MDDM [69]	DDM	Priority queue	2001
PDDM [69]	DDM	R-optimal negotiation	2001
ODAM [6]	CD	Candidate itemset reduction	2004
PPDM [34]	FDM	RSA encryption	2004
D-HOTM [40]	CD	RRE	2005
DiHO [39]	CD	Level wise search	2005
DFDM [27]	FDM	Logistic operations	2006
BFDM [27]	DFDM	Logistic operations	2006
DDRM [77]	CD	Lattice based	2007
EDMA [31]	DMA	CMatrix	2008
AprTidRec [79]	CD	TidRec	2009
DDN [22]	CD	Nonderivable & derivable itemset	2009
LMatrix [66]	FDM	Compressed binary matrix	2010
EDFIM [1]	CD	Local & global pruning	2013
PPFDM [64]	FDM	Secure multi party protocol	2014
FDM-KC [76]	FDM	RSA encryption	2014
FDM–UK [76]	FDM	HMAC encryption	2014
PEMA [51]	CD	Size of partition	2015

$^{FDM-UK}$FDM-UNIFI-KC

Cheung et al. presented Fast Distributed Mining (FDM) that is CD based candidate's selection that minimize message exchange cost to O(cp.m) where cp=union of locally large itemsets [11]. It has observed that, globally large itemsets is also happened to be locally large at single or multiple sites. FDM doesn't extend well with skewed organization of data. The databases are divided between all processors in equal sized blocks. Generally, ARM algorithms only come across for rules that are globally large. In other words, FDM finds out rules those are locally as well as globally large. FDM has adapted CD to minimize the communication overhead [69]. In FDM the first phase of CD is divided into two rounds of communication. First round finds out locally large in its partition at each site. Second round globally count and sum up of those itemset that is common in all sites. Moreover, interesting associations among locally and globally large itemset are found as smaller set of candidate sets at individual pass. As a result, it cut down the number of messages exchanges. Subsequent to candidates itemset generation global pruning, local pruning is used to prune itemset from each site at each pass. FDM also suggest another optimization count polling.

Advantage of FDM:
- Better Performance than CD.
- Small candidate sets rather than CD [31] .
- Three optimizations are used count, local, global polling.
- Less number of candidates are used for counting than CD.

Disadvantage of FDM:
- CD has shown poor performance in terms of communication for large number of partitions.
- Each iteration polling mechanism need 2 passes of messages.
- This 2 pass scheme for calculating global supports and broadcasting frequent itemsets may reduce performance in parallel environment.

Ashrafi et al. presents ODAM [6], to identify optimized association rules from the distributed database on distributed shared nothing architecture. During first scan counting support of {1-itemsets} from all sites is similar to Apriori [4]. After first scan, it removes all infrequent itemsets and keeps into primary memory. Then broadcasts support count of other sites for finding the frequent {1- itemsets} globally. But it accumulates higher transactions within main memory. This approach has minimized average transactions length and also reduces itemset size apparently. During the new transaction inclusion, it checks the main memory for its presence. If presence increases the transaction counter or include that transaction into memory will increase the counter by one. It uses message exchange optimization and reduction techniques are client-server based. Due to distributed nature, it achieves better performance by reducing the number of message exchanges. Communication and synchronization are required to each and every pass for indirect support or direct counts exchange.

Advantage of ODAM:
- Exchanges fewer messages compared to CD and FDM.
- Minimize communication cost by 50% to 80% and 20% to 45% respect to CD and FDM respectively
- Instead of sending each support count to individuals' site, FDM sends directly to the polling site.
- Competent method for finding rules from different distributed sites.

Disadvantage of ODAM:
- Exchanged numerous messages during mining process due to efficient message optimization technique.

Cheung et al. presented FPM [14] to identify association rules on a distributed memory architecture that has taken count distribution approach. Database are partitioned between all the processors in same size blocks. It has included two different types of candidate pruning techniques i.e., global and distributed pruning adopted from FDM [11]. In each iteration, it performs single round of message interchange. It is very competent when data imbalance is large. Global pruning is highly efficient compared to distributed pruning.

Advantage of FPM:
- Better than CD in context to performance.
- Not uses count polling mechanism and subsequentl broadcasts local supports to all processors.
- Distributed pruning is efficient to handling high degree skewness of data.
- Global pruning is effective when mild skewness.
- Selected less number of candidates compared to other.
- High workload balance.

Disadvantage of FPM:
- Much responsive to workload balance rather than data skewness.
- FPM proves effective by partitioning initially database using balanced k-means clustering.
- It is a variation of CD.

However, Schuster et al. proposed DDM that has reduced communication cost to O(prob.c.n) [69]. Where, prob is the probability of itemset contains support larger then known threshold. Like FDM, candidate's generation in DDM is level wise approach and count from its database locally [11]. After that, the nodes execute a distributed decision protocol for exploring frequent or infrequent. FDM diverge from DDM in such a way that, a locally large itemset is not recognized as globally large itemset till it is demonstrated by messages exchange. This type of behavior has two hypotheses for candidate sets generation. In public hypothesis, the individual node reports global support of the itemset is same as the average local support available for it, or otherwise zero. In private hypothesis, individual node keeps track of its local support and it is available for those nodes that has not preserved their own local support for a candidate. GyHorodi presents a comparative study of DARM algorithms between CD and FDM in [28]. Otey et al. suggested an incremental technique ZIGZAG, a distributed asynchronous approach that provides less communication overhead to mining rules from dynamically distributed datasets [54].

5.1.2 Data Distribution Based DARM

DARM algorithms competence is mostly reliant on DD. Those are concentrated on maximizing parallelism by exchange of the data partitions. DD broadcast candidate itemsets such that individual site calculates a disjoint subset of item sets. Therefore, it can be feasible only for machines with high speed communications between sites. table 7 describes data distribution based DARM type of algorithms. Here we are considering all sampling based data distributed algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the distributed algorithms are perceptible in this table.

Table 7: Algorithmic Review Of DARM

Algorithms	Algorithm for Sampling Based DARM		
	Parent Algo	***Performance Criteria***	***Year***
Sampling [78]	Negative Border	Sample size	1996
D-Sampling [70]	Sampling	Size of tries	2003
SEE [15]	Sampling	Self similarity curve	2005
Par-Fp [18]	Sampling	Selective sampling	2005

5.1.2.1 D-Sampling

D-Sampling was proposed by Schuster et al. [71], a parallel approach of sampling algorithm [78]. It performs loading a sample into memory that used clusters of shared nothing architecture. All samples are represented by a trie structure known as lexicographic tree and examined entire subroutines repeatedly. Individual node of trie represents more structural information like parents, descendants, etc., that are connected with these nodes. The first level of trie created from samples and intersection is used to create new trie node from the TID list [15]. After creation of candidates set all individual partition is scanned exactly once to count the frequency of individual candidate in parallel. This approach uses M max algorithm to enhance a number of frequent itemsets by a given factor rather than reducing support constraint by a random value. However, creating a candidate's sets takes distributed samples using modified DDM [69].

Advantage of D-Sampling:
- Single database scan is required to itemset generation.
- Superior speed up and performance over the sampling [78].
- Combined memory is utilized to linearly enlarge the sample size.

Disadvantage of D-Sampling:
- Increasing communication overhead for large number of patterns.

5.1.2.2 Partition

Partition algorithm was suggested by Savasere et al. [68] that minimizes the database size. It can be used to extract rules in a large database by dividing the database among a set of sites. It implements the task in a distributed way. It has selected all frequent itemsets using two database scans by the level wise approach [8]. It has partitioned the entire database until small enough non overlapping partition that may be handled in main memory. The several processors are working on a different subset of database partitions [50]. The databases are either horizontal or vertical segmented. In the vertical segment, databases are divided based on column number whereas horizontal segment databases are divided based on row number. For *{*k-itemsets*}* generation like *{*1- itemsets*}*, *{*2-itemsets*}*, *{*3-itemsets*}*a different partition will be formed and comparing the minimum support constraint to find out the frequent itemsets from different partitions. This concept is very essential to enhance the execution speed to minimize disk I/O cost [17]. table 8 describes data distribution based DARM type of algorithms. Here we are considering all lattice partition based data distributed algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the distributed algorithms are perceptible in this table.

Table 8: Algorithmic Review Of DARM

Algorithms	Algorithm for Lattice Partition		
	Parent Algo	***Performance Criteria***	Year
SPTID SPINC, SPEAR [50]	Partition	Prefix trees	1995
PPAR, PEAR [50]	SPEAR	Size of messages	1995
AS-CPA [41]	Partition	Prior knowledge of sampling	1998
Apriori T [17]	Apriori	No. of message exchange	2006

5.2 Spark & MapReduce

MapReduce is a programming model for parallel processing of large scale data on Hadoop platforms. It has provided distributed data processing capabilities presented by Google [20]. It consists of two phases, mapper and reducer. First phase does the filtering job by splitting the input based on requisite output keys and values. Second phase takes values from the first phase, perform grouping based on key values and finally aggregate all output keys as well as values [49]. MapReduce can operate on GFS, NDFS or another distributed file system also. Apache Hadoop provides an ecosystem for open source distributed project development suitable for process and storage of large dataset [53]. Where MapReduce supervise the data processing task and HDFS (Hadoop distributed file system) does the data storage job. It can store structured, semi structured and unstructured types of data. Basically, hadoop framework is suitable for handling big data problems. Spark is another Scala based Apache framework targeting real time data analytics [85]. Sing spark in memory computation can be performed for rapid data processing over MapReduce. For large scale data processing, its performance is much faster than hadoop. Spark provides high level library support for Python, R, SQL, Java etc. which support flawless integration within any complex workflow [33]. Moreover, it promotes various service integration like GraphX, SQL, Dataframe, MLlib, Streaming etc.

A research effort has been enhanced of Apriori based and other sequential ARM algorithms by changing them into distributed versions using the Spark and MapReduce. Currently, several DARM algorithms are designed based on Hadoop or Spark framework shown in table 9. Here we are considering all spark and MapReduce based algorithms and their corresponding parent algorithms, performance criteria including their year of publication. Relationship hierarchy of all the distributed algorithms are perceptible in this table.

Table 9: Algorithmic Review of DARM

Algorithms	**Algorithms for Spark & MapReduce**		
	Parent Algo	***Performance Criteria***	***Year***
PFP [38]	MapReduce	Group dependent transactions	2008
PARMA [67]	MapReduce	Sampling&Aggregation	2012
AH [53]	MapReduce	Size of the node clusters	2013
BigFIM [49]	MapReduce	Block partitioning& Prefix tree size	2013
Dist-Eclat [49]	MapReduce	Round Robin& Prefix tree size	2013
MR-Apriori [42]	MapReduce	Candidate itemset partitions	2014
RuleMR [35]	MapReduce	Entropy minimization	2014
DFIMA [93]	Spark	Boolean vectors size	2015
PaMPa-HD [5]	MapReduce	Enumeration tree partitions	2015
DWS [36]	Spark	LRU partitions	2015
MRH-mine [25]	MapReduce	Queue size	2015
R-Apriori [63]	Spark	Bloom filter	2015
FiDoop [81]	MapReduce	Metric of workload balance	2016
YAFIM [33]	Spark	Buffer size	2016
AM [62]	Spark	Bloom filter	2018

^{DWS}Distributed Weka Spark, ^{AH}Apache Hadoop, ^{AM}Adaptive Miner

6. COMPARATIVE ANALYSIS OF PATTERN MINING ALGORITHMS

The main objective of this section is to represent a comparative study between parallel and distributed oriented approaches. An extensive scrutiny has been performed for both type of research and development. This section will discuss different algorithmic conceptualization, analytical performance characteristics, year wise research impact among the academic and research world and more. We are presenting the characteristic of all PARM type approaches discussed before into a summarized format in table 10. Here we are considering some essential parameters like name of the algorithms, database orientation, name of parent algorithm, the type of data structure or approaches are used, references, year of publication, citations of those literature respectively.

The major key issues are associated with the parallel rule mining framework includes:

(i) I/O minimization, load balancing among processors.

(ii) Reducing communication, synchronization cost between nodes.

(iii) Efficient choice database layout, search strategies, correct decomposition of data.

(iv) Minimizing duplication of task or overlapping.

Most of the literature has suggested I/O costs that can be minimized by different approaches are given below.

- Minimizing database scan.
- Parallelization the mining process to speed up the rule generation.
- Partitioning the database between individual processor and perform a distributed computing.

We are presenting the characteristic of all DARM type approaches discussed before into a summarized format in table 11. Here we are considering some essential

Table 10: Comparison Of PARM Algorithms

Algorithms	**Important features of PARM Algorithms**				
	DB	***Parent Algo***	***Approach/Features***	***Year***	***Impact***
DHP [57]	H	CD	bit vector	1995	380
PDM [58]	H	DHP	Hash table	1995	2371
HPA [74]	H	Can D	Message broadcast	1996	185
CCPD [89]	H	CD	Hash Tree Balancing	1996	188
NPA [74]	H	CD	Candidate distribution	1996	185
PCCD [89]	H	DD	Synchronization	1996	188
DIC [9]	H	CD	Trie with Counter	1997	2705
Eclat [87]	V	Partition	Equivalence class	1997	82
Par Eclat [90]	V	Eclat	Itemset clustering	1997	344
Max Eclat [88]	V	Eclat	Bottom up search	1997	1513
APM [12]	H	DIC	without synchronization	1998	53
FPM [14]	H	CD	Global&Distributed pruning	1998	92
HPGM [73]	H	HPA	Load skew minimization	1998	90
NPGM [73]	H	NPA	Minimize load skew	1998	90
HD,IDD [29]	H	DD	Hash tree	2000	252
VIPER [72]	V	Snake	Bit vectors	2000	357
DAA [46]	H	FPM	PCA	2001	15
DCP [60]	H	CD	Candidate itemset sorting	2001	135
DCI [52]	H	DCP	Itemset intersecting	2001	07
PHP [55]	H	DHP	Perfect hashing	2001	43
kDCI [43]	H	DCP	Prefix sharing	2003	34
U-Eclat [86]	V	Eclat	Automatic pruning	2003	698

^{PA} [82]	H	DD	Trie structure	2006	145
WDPA [83]	H	PA	Tid with metadata	2008	28
OPT-DIC [56]	H	DIC	Itemsets counting	2009	02
PARMA-P [94]	H	FP tree	Hash assignment strategy	2009	04
hEclat [95]	V	Eclat	Boolean matrix	2010	11
P-Mine [7]	H	FP Tree	HY Tree	2013	16
Eclat-opt [24]	V	Eclat	Suffix based	2013	05
Bi-Eclat [84]	V	Eclat	Sorted support	2014	15
P-Eclat [26]	V	Eclat	Partial BFS	2016	01
Eclat-Growth [44]	V	Eclat	Increased search strategy	2016	06
HashEclat [92]	V	Eclat	Min Hash	2019	-

^{H}Horizontal,^{V}Vertical,^{R}Random,^{Hy}Hybrid,^{ME}MessageExchange,^{PA}Parallel Apriori.

parameters like name of the algorithms, database orientation, name of parent algorithm, the type of data structure or approaches are used, references, year of publication, citations of those literature respectively.

Table 11: Comparison Of Distributed ARM Algorithms

Algorithms	**Important features of DARM**				
	DB	***Parent Algo***	***Approach/Features***	***Year***	***Impact***
DMA [13]	H	Apriori	ME optimization	1996	541
FDM [11]	H	CD	PVM,ME	1996	670
FDM-LP	H	FDM	Local pruning		
FDM-LUB	H	FDM	Local& Upper bound pruning		
FDM-LPP	H	FDM	Local& Polling site pruning		
FPM [14]	H	FDM,CD	Simple messaging schemes	1998	93
DMCA [80]	H	Apriori	DNF,Guided set, ME	2000	09
DDM [69]	H	FDM	Message Exchange	2001	155
PDDM	H	DDM			
DDDM	H	DDM			
MDDM	H	DDM			
H-mine [59]	H	Fp Growth	H Struct	2001	535
ZIGZAG [54]	H	Backtrack Tree	MFI search	2003	46
Apriori-T [16]	V	Apriori	T tree, ME	2003	42
ODAM [6]	H	DD	ME optimization	2004	137
PPDM [34]	H	FDM	Collision probability	2004	1171
D-HOTM [40]	Hy	Apriori	RRE, Hybrid fragmentation	2005	29
DiHO [39]	V	Apriori	Multi relational ARM, Trie	2005	03
D-sampling [71]	R	Sampling	Lexicographic tree	2005	109
BFDM [27]	H	FDM	Binary code mapping	2006	02
DDRM [77]	Hy	Apriori	WMPI, Lattice Partition	2007	03
EDMA [31]	H	Apriori	Compressed matrix	2008	13
PFP [38]	H	FP Growth	MapReduce	2008	453
DDN [22]	H	Apriori	MPI, DI, NDI	2009	05
AprTidRec [79]	H	Apriori	TidRec record structure	2009	03
LMatrix [66]	H	FP tree	FP array technique	2010	10
PARMA [67]	H	FP Growth	MapReduce	2012	117
EDFIM [1]	H	Apriori	Node&Global pruning, FPM	2013	01
BigFIM [49] Dist-Eclat [49]	$^{H,\ V\ H}$	Apriori, Eclat Eclat	BFS, MapReduce Diffsets, MapReduce	2013	168
AH [53]	H	Apriori	MapReduce	2013	20
PPFDM [64]	H	PPDM	Local Pruning, Broadcasting sup	2014	03
FDM-UNIFI-KC [76]	H	FDM	Private binary vectors,	2014	93
MR-Apriori [42]	H	Apriori	MapReduce	2014	31
RuleMR [35]	H	ID3, CN2	MapReduce	2014	05
PEMA [51]	Hy	Apriori	MAARM, ME	2015	06
MRH-mine [25]	H	H mine	MapReduce	2015	01
DFIMA [93]	H	Apriori	Matrix pruning, Spark	2015	33
DWS [36]	H	CloudWatch	Spark	2015	31
R-Apriori [63]	H	Apriori	Spark	2015	35
PaMPa-HD [5]	H	Carpenter	MapReduce	2015	05
FiDoop [81]	H	FIU tree	MapReduce	2016	48
YAFIM [33]	H	Faster IAPI	Spark	2016	08
AM [62]	H	Bloom filter	Spark	2018	03
MAD-ARM [32]	H	IDMA, AeMSAR	Mobile agent	2018	06

^{H}Horizontal,^{V}Vertical,RRandom,^{Hy}Hybrid, ^{ME}MessageExchange, ^{DWS}Distributed Weka Spark.

The major key issues are associated with the distributed rule mining framework includes:

(i) These approaches are essential to decrease communication costs.

(ii) May produce enormous communication overhead due to huge information forwarding by the data sites.

(iii) The majority of distributed algorithms may not enclose a competent optimization technique.

(iv) It generates low cost global association rules.

The graph in Figure 2 represents a comparative study between distributed and parallel oriented research approach. Research citation is counted for each five year slap for both cases. A year wise decreasing pattern is observed in both research field. After year 2000 research in distributed gains more impact than parallel approach. Year wise algorithmic count in both parallel and distributed field is represented in Figure 3. Here also it is observed that after 2000 distributed field gain more attention compared to parallel approach. Here for both graph representation, all the observations are thoroughly performed on Google Scholar up to year 2019.

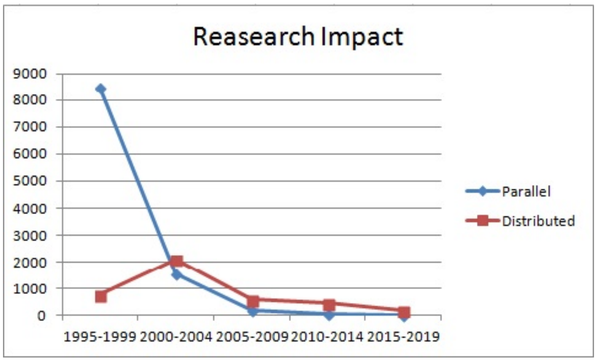

Figure 2: Year wise Research Impact of PARM and DARM

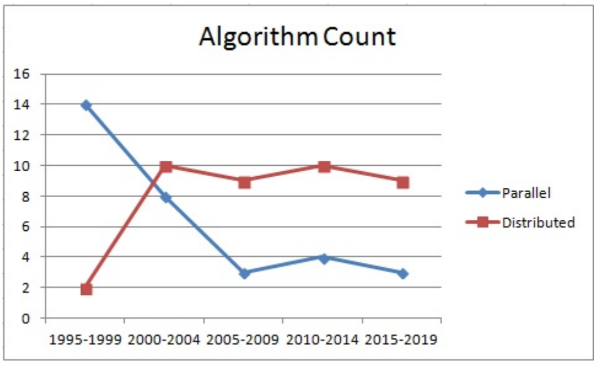

Figure 3: Year wise Algorithms Count of PARM and DARM

Security concerns are associated with privacy, sensitivity, correctness, integrity and false matches. Now a days, secured and privacy related PARM and DARM has become the most essential task practically. Where diverse business organizations attempt to distribute their transaction database with one another. In security domain, usual behavior is mostly frequent but abnormal or suspicious activity is less frequent. For example, consider in a shopping mall event activity analysis, the set of customer's activities is normally represented by frequent patterns, but suspicious activity may be considered as rare patterns. From this review work, various real life application areas are identified and reported. Some important application areas where the results may be useful are distributed network attack prediction, fraud detection, fluid dynamics, risk analysis, bioinformatics, weather analysis and many other fields.

7. CONCLUSION

The significant problems in data mining are finding of association rules from the large databases. This article mainly gives the idea about several algorithms correlated with PARM, DARM and classify them into interrelated techniques.

We have presented a comparative study between distributed and parallel oriented research approach that makes a pathway for future research directions in ARM. Research citation is counted for each five year slap for both cases. A year wise decreasing pattern is observed in both research field. After year 2000 research in DARM gains more impact than PARM approach. Year wise algorithmic count in both PARM and DARM field is represented. Here also it is observed that after 2000 distributed field gain more attention compared to PARM approach. PARM and DARM rule mining techniques required the consistent framework to mining the constructive knowledge from spread database sites. DARM has needed external communications during the entire mining process. The majority of distributed algorithms try to decrease communication cost that is an important issue. Also, we have presented the comparative study between the list of PARM algorithms that are very useful to load balancing strategy, MPI policy among the various sites would be the major demanding area of research in this domain.

The recent trend of research on this filed is paying more attention to advance the algorithm efficiency, high flexibility, scalability, and enhancement of the rule mining process. A research effort has been enhanced of Apriori based and other sequential ARM algorithms by changing them into distributed versions using the Spark and MapReduce. In future, we would like to explore the approaches that partition the set of frequent patterns into several groups and applying a dissimilar interestingness measure on each group. Future target also includes identifying rare pattern approaches to mine infrequent pattern under the parallel and distributed environment.

REFERENCES

[1] A. Adelpoor and M.S. Abadeh. A new dynamic distributed algorithm for frequent itemsets mining. International Journal of Computer Applications, 67(15), 2013.

[2] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In Acm sigmod record, volume 22, pages 207–216. ACM, 1993.

[3] R. Agrawal and J. C. Shafer. Parallel mining of association rules. IEEE Transactions on knowledge and Data Engineering, 8(6):962–969, 1996.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Proc. 20th int. conf. very large data bases, VLDB, volume 1215, pages 487–499, 1994.

[5] D. Apiletti, E. Baralis, T. Cerquitelli, P. Garza, P. Michiardi, and F. Pulvirenti. Pampa-hd: a parallel mapreduce based frequent pattern miner for high dimensional data. In Data Mining Workshop (ICDMW), 2015 IEEE International Conference on, pages 839–846. IEEE, 2015.

[6] M. Z. Ashrafi, D. Taniar, and K. Smith. Odam: An optimized distributed association rule mining algorithm. IEEE distributed systems online, 5(3), 2004.

[7] E. Baralis, T. Cerquitelli, S. Chiusano, and A. Grand. P-mine: Parallel itemset mining on large datasets. In Data Engineering Workshops (ICDEW), 2013 IEEE 29th International Conference on, pages 266–271. IEEE, 2013.

[8] J. Blatak and L. Popelinsky. Drap independent: A data distribution algorithm for mining first-order frequent patterns. Computing and Informatics, 26(3):345–366, 2012.

[9] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In ACM SIGMOD Record, volume 26, pages 255–264. ACM, 1997.

[10] J. Chattratichat, J. Darlington, M. Ghanem, Y. Guo, H. F. Huning, M. Kohler, J. Sutiwaraphun, H. W. To, and D. Yang. Large scale data mining: Challenges and responses. In KDD, pages 143–146, 1997.

[11] D. W. Cheung, J. Han, V. T. Ng, A. W Fu, and Y. Fu. A fast distributed algorithm for mining association rules. In Parallel and Distributed Information Systems, 1996., Fourth International Conference on, pages 31–42. IEEE, 1996.

[12] D. W Cheung, K. Hu, and S. Xia. Asynchronous parallel algorithm for mining association rules on a shared memory multi processor. In Proceedings of the tenth annual ACM symposium on Parallel algorithms and architectures, pages 279–288. ACM, 1998.

[13] D.W Cheung, V. T. Ng, A. W. Fu, and Y. Fu. Efficient mining of association rules in distributed databases. IEEE transactions on Knowledge and Data Engineering, 8(6):911–922, 1996.

[14] D.W. Cheung and Y. Xiao. Effect of data skewness in parallel mining of association rules. In Pacific Asia Conference on Knowledge Discovery and Data Mining, pages 48–60. Springer, 1998.

[15] K. Chuang, M. Chen, and W. Yang. Progressive sampling for association rules based on sampling error estimation. In Pacific Asia Conference on Knowledge Discovery and Data Mining, pages 505–515. Springer, 2005.

[16] F. Coenen, P. Leng, and S. Ahmed. T-trees, vertical partitioning and distributed association rule mining. In Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, pages 513–516. IEEE, 2003.

[17] P. Coenen, F.and Leng. Partitioning strategies for distributed association rule mining. The Knowledge Engineering Review, 21(01):25–47, 2006.

[18] S. Cong, J. Han, J. Hoeflinger, and D. Padua. A sampling based framework for parallel data mining. In Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming, pages 255–265. ACM, 2005.

[19] C. Creighton and S. Hanash. Mining gene expression databases for association rules. Bioinformatics, 19(1):79–86, 2003.

[20] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. Communications of the ACM, 51(1):107–113, 2008.

[21] X. Deng, C. Jin, Y. Higuchi, and C.J. Han. An efficient association rule mining method for personalized recommendation in mobile ecommerce. 12 2010.

[22] M. Deypir and M. Sadreddini. Distributed association rules mining using non derivable frequent patterns. Iranian Journal of Science and Technology, 33(B6):511, 2009.

[23] M.J. Elliot, A. Manning, K. Mayes, J. Gurd, and M. Bane. Suda: A program for detecting special uniques. 2005.

[24] P.E. Feng, Y. Liu, Q.Y. Qiu, and L.X. Li. Strategies of efficiency improvement for eclat algorithm. Journal of Zhejiang University. Engineering Science, 47(2):223–230, 2013.

[25] X. Feng, J. Zhao, and Z. Zhang. Mapreduce based h-mine algorithm. In Instrumentation and Measurement, Computer, Communication and Control (IMCCC), 2015 Fifth International Conference on, pages 1755–1760. IEEE, 2015.

[26] G. Gatuha and T. Jiang. Novel frequent pattern mining algorithm based on parallelization scheme. International Journal of Engineering Research in Africa, 23, 2016.

[27] C. Geng, N. Wei-wei, Z. Yu-quan, and S. Zhi-hui. A fast distributed algorithm for association rule mining based on binary coding mapping relation. Wuhan University Journal of Natural Sciences, 11(1):27–30, 2006.

[28] C. GyHorodi. A comparative study of distributed algorithms in mining association rules. In International Symposium on System Theory–XI Edition, volume 1, pages 339–345, 2003.

[29] E. Han, G. Karypis, and V. Kumar. Scalable parallel data mining for association rules. IEEE Transactions on Knowledge and Data Engineering, 12(3):337–352, 2000.

[30] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In ACM Sigmod Record, volume 29, pages 1–12. ACM, 2000.

[31] W. Jian and L. X. Ming. An efficient association rule mining algorithm in distributed databases. In Knowledge Discovery and Data Mining, 2008. WKDD 2008. First International Workshop on, pages 108–113. IEEE, 2008.

[32] Y. Joshi, S. G. Totad, R. Geeta, and P. P. Reddy. Mobile agent based frequent pattern mining for distributed databases. In Intelligent Computing and Information and Communication, pages 77–85. Springer, 2018.

[33] R. Joy and K. Sherly. Parallel frequent itemset mining with spark rdd framework for disease prediction. In Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on, pages 1–5. IEEE, 2016.

[34] M. Kantarcioglu and C. Clifton. Privacy preserving distributed mining of association rules on horizontally partitioned data. IEEE transactions on knowledge and data engineering, 16(9):1026–1037, 2004.

[35] V. Kolias, C. Kolias, I. Anagnostopoulos, and E. Kayafas. Rulemr: Classification rule discovery with mapreduce. In Big Data (Big Data), 2014 IEEE International Conference on, pages 20–28. IEEE, 2014.

[36] A. Koliopoulos, P. Yiapanis, F. Tekiner, G. Nenadic, and J. Keane. A parallel distributed weka framework for big data mining using spark. In Big Data (BigData Congress), 2015 IEEE International Congress on, pages 9–16. IEEE, 2015.

[37] H. Kong, C. Jong, and U. Ryang. Rare association rule mining for network intrusion detection. arXiv preprint arXiv:1610.04306, 2016.

[38] H. Li, Y. Wang, D. Zhang, M. Zhang, and E.Y Chang. Pfp: parallel fp-growth for query recommendation. In Proceedings of the 2008 ACM conference on Recommender systems, pages 107–114. ACM, 2008.

[39] S. Li and W. M. Pottenger. Diho:" a distributed higher order association rule miner. In the Proceedings of the 24th ACM SIGMOD International Conference on Management of Data. Baltimore, MD, June 2005.

[40] S. Li, T. Wu, and W. M. Pottenger. Distributed higher order association rule mining using information extracted from textual data. ACM SIGKDD Explorations Newsletter, 7(1):26–35, 2005.

[41] J.L. Lin and M.H. Dunham. Mining association rules: Anti skew algorithms. In Data Engineering, 1998. Proceedings., 14th International Conference on, pages 486–493. IEEE, 1998.

[42] X. Lin. Mr-apriori: Association rules algorithm based on mapreduce. In Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference on, pages 141–144. IEEE, 2014.

[43] C. Lucchese, P. Orlando, S.and Palmerini, R. Perego, and F. Silvestri. kdci: A multi strategy algorithm for mining frequent sets. In Proceedings of the IEEE ICDM Workshop of Frequent Itemset Mining Implementations (FIMI), Melbourne, Florida. Citeseer, 2003.

[44] Z. Ma, J. Yang, T. Zhang, and F. Liu. An improved eclat algorithm for mining association rules based on increased search strategy. International Journal of Database Theory and Application, 9(5):251–266, 2016.

[45] A. M. Manning, D. J. Haglin, and J. A Keane. A recursive search algorithm for statistical disclosure assessment. Data Mining and Knowledge Discovery, 16(2):165–196, 2008.

[46] A. M. Manning and J.A Keane. Data allocation algorithm for parallel association rule discovery. In Pacific Asia Conference on Knowledge Discovery and Data Mining, pages 413–420. Springer, 2001.

[47] V. Markam and L. S. M. Dubey. A general study of associations rule mining in intrusion detection system. International Journal of Emerging Technology and Advanced Engineering, 2(1):347–356, 2012.

[48] N. Mishra, H. Soni, S. Sharma, and AK. Upadhyay. A comprehensive survey of data mining techniques on time series data for rainfall prediction. Journal of ICT Research and Applications, 11(2):168–184, 2017.

[49] S. Moens, E. Aksehirli, and B. Goethals. Frequent itemset mining for big data. In BigData Conference, pages 111–118, 2013.

[50] A. Mueller. Fast sequential and parallel algorithms for association rule mining: a comparison. 1995.

[51] A. Ogunde, O. Folorunso, and A. Sodiya. A partition enhanced mining algorithm for distributed association rule mining systems. Egyptian Informatics Journal, 16(3):297–307, 2015.

[52] S. Orlando, P. Palmerini, and R. Perego. Dci: a hybrid algorithm for frequent set counting. 2001.

[53] S. Oruganti, Q. Ding, and N. Tabrizi. Exploring hadoop as a platform for distributed association rule mining. In FUTURE COMPUTING 2013 the Fifth International Conference on Future Computational Technologies and Applications, pages 62–67. Citeseer, 2013.

[54] M. E. Otey, C. Wang, S. Parthasarathy, A. Veloso, and W. Meira. Mining frequent item sets in distributed and dynamic databases. In Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, pages 617–620. IEEE, 2003.

[55] S.A. Ozel and H.A. Guvenir. An algorithm for mining association rules using perfect hashing and database pruning. In 10th Turkish Symposium on Artificial Intelligence and Neural Networks, pages 257–264. Citeseer, 2001.

[56] P. Paranjape and U. Deshpande. An optimistic messaging distributed algorithm for association rule mining. In India Conference (INDICON), 2009 Annual IEEE, pages 1–5. IEEE, 2009.

[57] J. S. Park, M.S. Chen, and P.S. Yu. Efficient parallel data mining for association rules. In Proceedings of the fourth international conference on Information and knowledge management, pages 31–36. ACM, 1995.

[58] J.S. Park, M.S. Chen, and P.S. Yu. An effective hash based algorithm for mining association rules, volume 24. ACM, 1995.

[59] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, and D. Yang. H-mine: Hyper structure mining of frequent patterns in large databases. In Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, pages 441–448. IEEE, 2001.

[60] S. Perego, R.and Orlando and P. Palmerini. Enhancing the apriori algorithm for frequent set counting. In International Conference on Data Warehousing and Knowledge Discovery, pages 71–82. Springer, 2001.

[61] C. Phua, V. Lee, K. Smith, and R. Gayler. A comprehensive survey of data mining based fraud detection research. arXiv preprint arXiv:1009.6119, 2010.

[62] S. Rathee and A. Kashyap. Adaptive miner: an efficient distributed association rule mining algorithm on spark. Journal of Big Data, 5(1):6, 2018.

[63] S. Rathee, M. Kaul, and A. Kashyap. R-apriori: an efficient apriori based algorithm on spark. In Proceedings of the 8th Workshop on Ph.
D. Workshop in Information and Knowledge Management, pages 27–34. Acm, 2015.

[64] M.R. Raut and H. Dakhore. Association rule mining in horizontally distributed databases. International Journal of Computer Science and Information Technologies, 5(6):7540–7544, 2014.

[65] K. Raza. Application of data mining in bioinformatics. arXiv preprint arXiv:1205.1125, 2012.

[66] J. A. Renjit and KL. Shunmuganathan. Mining the data from distributed database using an improved mining algorithm. arXiv preprint arXiv:1004.1677, 2010.

[67] M. Riondato, J. A. DeBrabant, R. Fonseca, and E. Upfal. Parma: a parallel randomized algorithm for approximate association rules mining in mapreduce. In Proceedings of the 21st ACM international conference on Information and knowledge management, pages 85–94. ACM, 2012.

[68] A. Savasere, E.R. Omiecinski, and S.B. Navathe. An efficient algorithm for mining association rules in large databases. Technical report, Georgia Institute of Technology, 1995.

[69] A. Schuster and R. Wolff. Communication efficient distributed mining of association rules. In ACM SIGMOD Record, volume 30, pages 473– 484. ACM, 2001.

[70] A. Schuster, R. Wolff, and D. Trock. A high performance distributed algorithm for mining association rules. In Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, pages 291–298. IEEE, 2003.

[71] A. Schuster, R. Wolff, and D. Trock. A high performance distributed algorithm for mining association rules. Knowledge and Information Systems, 7(4):458–475, 2005.

[72] P. Shenoy, J.R. Haritsa, G. Sudarshan, S.and Bhalotia, and D. Bawa, M.and Shah. Turbo charging vertical mining of large databases. In ACM SIGMOD Record, volume 29, pages 22–33. ACM, 2000.

[73] M. Shintani, T.and Kitsuregawa. Parallel mining algorithms for generalized association rules with classification hierarchy. In ACM SIGMOD Record, volume 27, pages 25–36. ACM, 1998.

[74] T. Shintani and M. Kitsuregawa. Hash based parallel algorithms for mining association rules. In Parallel and Distributed Information Systems, 1996., Fourth International Conference on, pages 19–30. IEEE, 1996.

[75] J. Srivastava, R. Cooley, M. Deshpande, and P. Tan. Web usage mining: Discovery and applications of usage patterns from web data. Acm Sigkdd Explorations Newsletter, 1(2):12–23, 2000.

[76] T. Tassa. Secure mining of association rules in horizontally distributed databases. IEEE Transactions on Knowledge and Data Engineering, 26(4):970–983, 2014.

[77] W. Thomas. Parallel mining of association rules using a lattice based approach. IEEE, 2007.

[78] H. Toivonen. Sampling large databases for association rules. In VLDB, volume 96, pages 134–145, 1996.

[79] A. Wang. Research on mining association rules in distributed system. In Business Intelligence and Financial Engineering, 2009. BIFE'09. International Conference on, pages 472–475. IEEE, 2009.

[80] C. Wang, H. Huang, and H. Li. A fast distributed mining algorithm for association rules with item constraints. In Systems, Man, and Cybernetics, 2000 IEEE International Conference on, volume 3, pages 1900–1905. IEEE, 2000.

[81] Y. Xun, J. Zhang, and X. Qin. Fidoop: Parallel mining of frequent itemsets using mapreduce. IEEE transactions on Systems, Man, and Cybernetics: systems, 46(3):313–325, 2016.

[82] Y. Ye and C. Chiang. A parallel apriori algorithm for frequent itemsets mining. In Software Engineering Research, Management and Applications, 2006. Fourth International Conference on, pages 87–94. IEEE, 2006.

[83] K. Yu and J. Zhou. A weighted load balancing parallel apriori algorithm for association rule mining. In Granular Computing, 2008. GrC 2008. IEEE International Conference on, pages 756–761. IEEE, 2008.

[84] X. Yu and H. Wang. Improvement of eclat algorithm based on support in frequent itemset mining. Journal of Computers, 9(9):2116–2124, 2014.

[85] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault tolerant abstraction for in memory cluster computing. In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation, pages 2–2. USENIX Association, 2012.

[86] M. J. Zaki and K. Gouda. Fast vertical mining using diffsets. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 326–335. ACM, 2003.

[87] M. J. Zaki, S. Parthasarathy, and W. Li. A localized algorithm for parallel association mining. In Proceedings of the ninth annual ACM symposium on Parallel algorithms and architectures, pages 321–330. ACM, 1997.

[88] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In KDD, volume 97, pages 283–286, 1997.

[89] M.J. Zaki, M. Ogihara, S. Parthasarathy, and W. Li. Parallel data mining for association rules on shared memory multi processors. In Supercomputing, 1996. Proceedings of the 1996 ACM/IEEE Conference on, pages 43–43. IEEE, 1996.

[90] M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. Parallel algorithms for discovery of association rules. Data mining and knowledge discovery, 1(4):343–373, 1997.

[91] Mohammed J. Zaki, S. Parthasarathy, W. Li, and M. Ogihara. Evaluation of sampling for data mining of association rules. In Research Issues in Data Engineering, 1997. Proceedings. Seventh International Workshop on, pages 42–50. IEEE, 1997.

[92] C. Zhang, P. Tian, X. Zhang, Q. Liao, Z. L. Jiang, and X. Wang. Hasheclat: an efficient frequent itemset algorithm. International Journal of Machine Learning and Cybernetics, pages 1–14, 2019.

[93] F. Zhang, M. Liu, F. Gui, W. Shen, A. Shami, and Y. Ma. A distributed frequent itemset mining algorithm using spark for big data analytics. Cluster Computing, 18(4):1493–1501, 2015.

[94] X. Zhang, Y. Zhu, and N. Hua. Privacy parallel algorithm for mining association rules and its application in hrm. In Computational Intelligence and Design, 2009. ISCID'09. Second International Symposium on, volume 2, pages 296–299. IEEE, 2009.

[95] X. Zhong-Yang, C. Pei-En, and Z. Yu-Fang. Improvement of eclat algorithm for association rules based on hash boolean matrix. Application Research of Computers, 4:1323–1325, 2010.