International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6, August, 2019

Multi-Core Processing Cloud Eclat Growth

V.Priya, S.Murugan

Abstract: ***Data mining is a lively process used in many leading technologies of this information era. Eclat growth is one of the best performance data mining algorithms. This work is indented to create a suave interface for Eclat growth algorithm to run in multi-core processor-based cloud computing environments. Recent improvements in processor manufacturing technology make it possible to create multi-core high performance Central Processing Units (CPUs) and Graphics Processing Units (GPUs). Many cloud services are already providing accessibility to these high-power processor virtual machines. The process of blending these technologies with Eclat Growth is proposed here in the name of "Multi-core Processing Cloud Eclat Growth" (MPCEG) to achieve higher processing speeds without compromising the standard data mining metrics such as Accuracy, Precision, Recall and F1-Score. New procedures for Cloud Parallel Processing, GPU Utilization, Annihilation of floating point arithmetic errors by fixed point replacement in GPUs and Hierarchical offloading aggregation are introduced in the construction process of proposed MPCEG.***

Keywords: Cloud parallel processing, Data Mining, Eclat Growth, Fixed point arithmetic, Graphics Processing Units, Hierarchical offloading, Multi-core processing

I. INTRODUCTION

Data mining algorithms are using everywhere these days. Almost all search engine providers and E-Mail portal organizations are collecting data from their users for marketing and other purposes [1]. The evaluation of Artificial Intelligence techniques and improvements in Machine Learning algorithms significantly improved the data science and modern marketing strategies. Data mining algorithms are the key to the modern machine learning algorithms. There are a number of refined computational methods are introduced during the last decade to handle ample of data in real-time. The revolution in communication systems made an emergent increase in the usage mobile computing devices. The increased number of mobile computing devices causes a notable raise in the swarming digital data quantity. Therefore the conventional desktop based data processing is fading now and the emerging cloud computing environments were spread in a very impressive manner. Present computers are also powered by multicore processors along with huge memory and storage capacity. Initially GPUs are introduced to process image transformations and rendering.

Revised Manuscript Received on October 30, 2019.
* Correspondence Author

V.Priya*, Assistant Professor B.Sc Computer Science from University of Madras in 1998 and M.Sc Computer Science from Nehru Memorial College, Puthanampatti,Bharathidasan University in 2001 and M.Phil Computer Science from Bharathidasan University in 2004.

S.Murugan, Associate Professor in Nehru Memorial College M.Sc degree in Applied Mathematics from Anna University in 1984 and M.Phil degree in Computer Science from National Institute of Technology formerly known as Regional Engineering College, Trichirappalli in 1994.

The CPUs are offloading their excessive image processing tasks to the GPUs to improve the user experience by reducing the response time. But advanced GPUs are good enough to perform massive
Mathematical calculations swiftly by their thousands of powerful energy-efficient parallel micro-core architecture [2].
This property of GPU is very useful for modern machine learning and big data analysis.The cloud platform already taking advantages of these GPUs an
d present virtual machines are ascending with CPUs combined with GPUs. Companies such as Amazon and Google are providing cost effective parallel cloud services. Amazon Light sail, EC2 [3], Microsoft Azure and Cloud4C are using multicore CPU-GPU combination servers to provide Virtual Machines for their clients. Conventional data mining algorithms are designed to run in single core processor-based computer systems. Though the performances of conventional data mining algorithms are improving with high speed processors, the performance can be significantly improved by modifying them. The proposed method is created to develop the tune-up utilities for Eclat Growth algorithm with some enhancements. Using GPUs in data mining has a vulnerability of GPU Floating point Paranoia [4][5] – a precision problem in real numbers. The proposed system is aspired also to eliminate this problem by introducing fixed point arithmetic in Eclat Growth GPU processing. The primary idea of this work is to use the Eclat Growth data mining procedure in powerful combination of Multi-core CPUs, Massive micro-core GPUs and cloud computing environment.

II. EXISTING METHODS

There are some notable efforts taken to optimize the data mining process in cloud environment. They are Adaptive-Miner: An efficient distributed association rule mining algorithm on Spark (AMIN) [6], Item-Centric mining of frequent patterns from big uncertain data (ICMIN) [7], Parallel Mining of Frequent Itemsets in Hadoop Cluster Having Heterogeneous Nodes (PMFIHHC) [8], Cost-effective Big Data Mining in the Cloud: A case Study with K-means (CEKBDM) [9] and A New Cloud Computing Architecture for the Classification of Remote Sensing Data (NCCARSD) [10] are discussed in this work to perform a comparative study.

A. An Efficient distributed association rule mining algorithm on Spark

AMIN is designed to operate in Cloud computing environments. The authors of this work justified the cost efficiency of the cloud computing technology as the reason for selecting Amazon Spark as the execution platform.

AMIN is designed to handle huge data efficiently with the help of MapReduce Parallel Computing framework which was designed by Google. Different map reducing techniques such as Flink, Hadoop and Spark are analyzed in this work. Important algorithms used in AMIN are Frequent Singletone generation and frequent itemsets generation. AMIN's performance is compared with Yet another Frequent Itemset Mining YAFIM [11] and R-Apriori [12] by using standard datasets. The merits and limitations of AMIN are given in Table I.

B. Item-Centric Mining of frequent pattern from big uncertain data

The authors of ICMIN introduce a new algorithm MR-UV-Eclat for Spark-based vertical data mining. MR-UV-Eclat algorithm can handle uncertain data and big uncertain data with existential probability values. There are two main phases designed in ICMIN. The first phase is used to create the first level tree projection under the NULL node. The first level nodes will be mapped further. This phase is responsible for the vertical data conversion. All constructed nodes in the first phase are allowed to expand in the second phase. This tree expansion is used to link the current itemset with all items in the list. Then the map-reduction is carried out in this phase for given support count. The performance of ICMIN is compared with MR-U-Apriori [13] in terms of processing time. The key-points of ICMIN are given in Table I.

C. Parallel Mining of Frequent Itemsets in Hadoop Cluster Having Heterogeneous Nodes

Hadoop MapReduce Model is the base of PMFIHHC. In this work the authors provide a clear analysis of some existing algorithms such as Modified Apriori Algorithm, MREclat Algorithm, Dist-Eclat and BigFIM algorithm, PARMA algorithm, MRPre Post algorithm and Cluster BigFIM algorithm. The performance issue analysis is performed on the existing methods and PMFIHHC is introduced to overcome the identified issues. PMFIHHC reduces the complication of MapReduce process in heterogeneous computing environments. A new threshold-based data placement algorithm is provided in PMFIHHC for computational offloading among different types computational nodes. The MapReduce job is also offloaded to several mappers in this model to compute local one-itemsets and global one-itemsets. The performance is measured in Hadoop Cluster with Heterogeneous nodes [14] based on the response time of the existing and proposed methods. The advantages and limitations of this method are given in Table I.

D. Cost-effective Big Data Mining in the Cloud: A case Study with K-means

K-means algorithm is used in CEKBDM. The total Euclidean distance calculation between each data in clustering problem is a NP-hard [15] category. This problem is solved in CEKBDM with the help of local search solution of Lloyd [16]. The Euclidean distances between each data and cluster centroid is minimized in CEKBDM iteratively. The performance CEKBDM have analyzed by processing Gaussian Dataset and Road Network Dataset. Accuracy is taken as the prime concern in CEKBDM and datasets are analyzed with different K values along with computational time. The primary merits and limitations of CEKBDM are given in Table I.

E. A New Cloud Computing Architecture for the Classification of Remote Sensing Data

NCCARSD proposes a new Inter-cloud data mining architecture for supervised classification fields. This architecture consists of Project Definition Layer, Classification Layer and Distribution Layer. NCCARSD is applicable in Hadoop Distributed File System (HDFS) which is optimized for high performance in processing larger data files. The MapReduce process contains three sequential phases - they are map phase, shuffle phase and reduce phase [17]. NCCARSD handles data in streaming mode from the auxiliary repository from cloud by getting the data URL. In this work, the processing speedup ratio along with different number of nodes are analyzed using WEKA tool for different datasets ranging from 2 to 20 GB. The merits and limitations of NCCARSD are given in Table I.

Table I: Key points, Merits and Limitations of existing works

Author	Work	Method	Metrics	Merits	Limitations
Sanjay Rathee et.al.	Adaptive-Miner	MapReduce Parallel Computing	Processin g Time	Processing Speed	Higher Memory consumption
Peter Braun et.al.	Item-centric mining of frequent patterns	MR-UV-Eclat	Processin g Time	Constant Processing time for different minimum support counts	Compromised Accuracy and Precision
Dhanashre e Shirke et.al.	Parallel Mining of Frequent Itemsets	Threshold based data placement computational offloading	Response Time	Processing Speed	Compromised Accuracy and Precision
Qiang He et.al.	Cost-effective Big Data Mining in the Cloud	Lloyd local search solution	Accuracy	Higher Accuracy	Higher Processing time
Victor Andres Ayma Quirita et.al.	New Cloud Computing Architecture for Classification	Multi-layered Map Recuce process	Processin g Speed	Processing Speed	Requires large training data

III. RELATED WORKS

Proposed MPCEG majorly depends on the latest technical evolutions of GPU based computing and Cloud Parallel Processing environments. The related works in these technologies relevant to the proposed method are consecrated here for clear explanation.

A. *Eclat Growth*

Eclat-Growth algorithm is proposed by Zhiyong Ma et al. in 2016 [18]. Eclat-Growth is designed based on increased search strategy. This procedure uses increased two-dimensional pattern tree and the TID-sets in vertical data format table added to the pattern tree rows. The existing pattern tree itemsets are combined with newly added itemsets to generate frequent itemsets. Breadth-first-search approach is used to perform the combining process. The frequent 2-itemsets and candidate itemsets are generated in the beginning. Then higher order candidate and frequent itemsets until entire frequent itemset of the pattern table are combined with the newly added itemset. The breadth-first-search procedure is used to clip the candidate itemsets easily and redundant candidate itemsets are clipped. Eclat-growth adopts Boolean Array setting and retrieval by indexes of transactions) instead of intersecting the TID-sets to calculate the support degree of two itemsets. This alternative process is also used to reduce the computational complexity.

B. *GPU based data mining*

The data mining procedures can be enhanced for processing speed by using CPU-based multi-threaded methods, distributed methods and Parallel processing in GPU methods. Many of the conventional data mining procedures such as Apriori, Eclat and FP-Growth are single threaded CPU based methods. The processing speed of these methods can be improved by introducing multiple threads in a multicore processor – which means running the procedure in a single machine with multiple parallel instances where each instance is performed in a separate core of the processor [19]. Another way is distributed, in which the data will be distributed among several different computers [20] and the processed data will be accumulated to get the final result.

Latest computing devices are powered by multicore processors and high-speed graphics processing units. GPUs are introduced for image rendering and video streaming processes in early days. But latest improvements in Operating Systems and computer hardware integration, it is possible to use the GPUs for almost all kind of computational processes. One of the main advantages of using GPUs instead of CPUs is the operation speed. A GPU is designed with thousands of micro-core computational device arrays. The GPUs works based on the Single Instruction Multiple Threads (SIMT) principle with amalgamated memory unit. These micro-core processing devices arrays can handle large amount of data in parallel. Instead of processing data one by one in a sequential order, the GPUs tend to process a block of data at-a-stretch. This property of GPUs can be used to increase the processing speed of some data mining processes [21]. Making the sequential data mining procedures into parallel processing friendly is the crucial part in using GPUs. The parallelization in amalgamated memory execution has Partitioning, Assignment and Execution modules [22]. The portioning module is responsible for splitting the data into sub-data packets for available GPU cores. The Assignment module is used to map the divided sub-data packets to corresponding GPU cores. The Execution module is used to trig the process initialization on the GPU cores in parallel. The optimization of the GPU kernel of the work “Parallelization of large vector similarity computations in a hybrid CPU+GPU environment” [22] is adopted in the proposed work. The GPU based frequent itemset extraction is given in Figure 1.

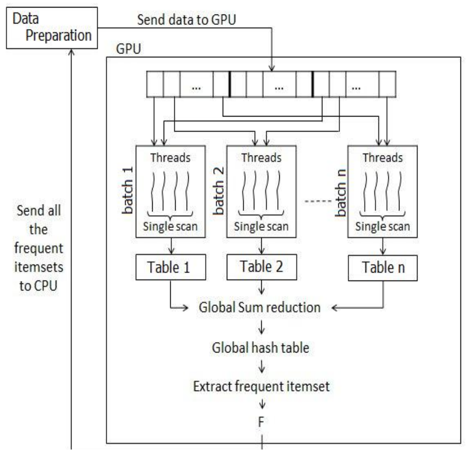


Fig 1. GPU based frequent itemset extraction

C. *Cloud Parallel Processing*

Cloud environments are capable of running parallel task-based applications through parallel cloud functions. This cloud facility is also optimized with energy awareness to optimize energy conservation during the parallel tasks execution process [23]. The number of Virtual Machines in a cloud environment has some limitations stated by the cloud service provider. Recent cloud architecture carries High-Availability Clusters (HA Clusters). HA Cluster consists a set of computers with server application handling capacity ensure the maximum uptime. A typical HA cluster can have 32 hosts and each host can have 40 Virtual Machines. This limit is extended in premium services in which a HA Cluster will have 8 hosts and each host can get 100 Virtual Machines. The maximum number of Virtual Machines in a cluster is 1280. IBM Service Delivery Manager (ISDM) is used in the hypervisor for a standard HA cloud infrastructure to enable Distributed Resource Sharing (DRS) [24]. Here the cloud system is provided as Infrastructure as a Service (IaaS) which provides Virtual Machines. The Virtual Machines are configures based on the requirements to execute the demanded applications. The HA Cluster model is illustrated in Figure 2.

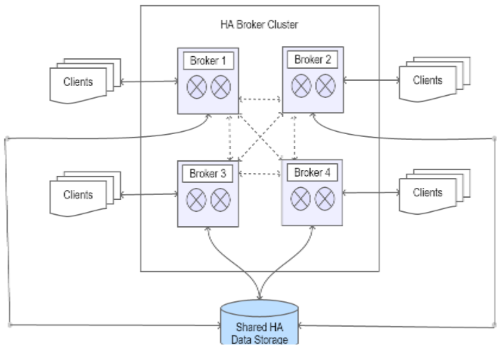


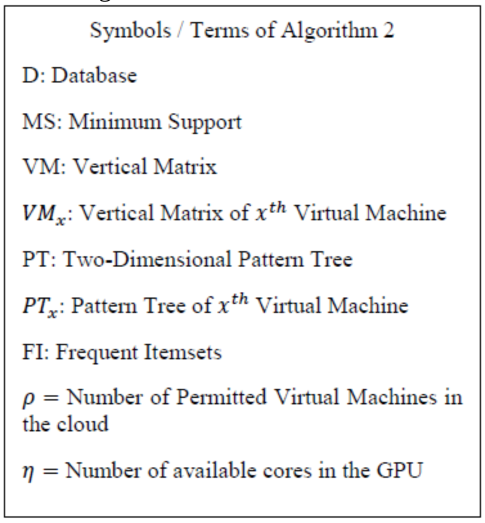
Fig 2. High-Availability Cluster

IV. PROPOSED METHOD

GPU Eclat-growth (G-Eclat-growth), Floating-point to Fixed-point Arithmetic Conversion (FL2FP) and Hierarchical Offloading - Aggregation (HOA) are the main modules of this proposed system “Multi-core Processing Cloud Eclat Growth”.

A. *G-Eclat-growth*

The standard Eclat algorithm is a recursive one with tidy steps given as Algorithm 1.



Symbols / Terms of Algorithm 2

D: Database

MS: Minimum Support

VM: Vertical Matrix

VM_x: Vertical Matrix of x^{th} Virtual Machine

PT: Two-Dimensional Pattern Tree

PT_x: Pattern Tree of x^{th} Virtual Machine

FI: Frequent Itemsets

ρ = Number of Permitted Virtual Machines in the cloud

η = Number of available cores in the GPU

Algorithm 1: Eclat

```
Step 1. Initialize i=1
Step 2. Do
Find all frequent i-itemsets by scanning the database.
Find all candidate $i+1$-temsets from frequent $i$-itemsets list.
        Increment i by value 1
  Step 3. Repeat Step 2 until no candidate itemset is generated for $i-1$-itemsets.
```

The G-Eclat-growth is derived from the standard Eclat algorithm for GPU based parallel cloud optimization given below as Algorithm 2.

Algorithm 2: G-Eclat Growth

```
For x = 1 to $\rho$ do
    $VM_x$ = CreateVMfromDatabase(D)
    Initialize $PT_x$ with $\emptyset$ values
     For i = 1 to length($VM_x$) do
     Parallel_GPU_Execute($\eta$) Begin
         If(length($VM_x$[i].TID_sets) $\geq$ MS then
       AddItemSetToPatternTree($VM_x$[i], $PT_x$, MS)
     Parallel_GPU_Execute End
    End for
End for
VM = Merge[$VM_1, VM_2 \cdots VM_\rho$]
PT = Merge[$PT_1, PT_2 \cdots PT_\rho$]
FI=GetAllFrequentItemsetsFromPatternTree(PT)
```

B. *Floating-Point To Fixed-Point Arithmetic Conversion*

GPUs are known for their rapid parallel execution speed where accuracy in floating point is compromised. This phenomenon is known as GPU floating paranoia. To eliminate the precision problems in GPU, A fixed-point to floating-point arithmetic interface is introduced in this work. IEEE 754 standard based Floating points are used in the latest CPUs and GPUs. IEEE 754 standard floating-point numbers are having three components. They are the sign, exponent and mantissa which are common for both 32-bit and 64-bit architectures.

The number of bits used to represent the components differs between 32-bit and 64-bit representations. The number of bits for allocated for these components are given in Table II.

Table II: Floating-point components memory allocation

Precision	Sign	Exponent	Fraction
Single (32-bit)	1[31]	8[30-23]	23[22-00]
Double (64-bit)	1[63]	11[62-52]	52[51-00]

The sign component is assigned with a single bit memory that can hold either 0 or 1 to represent positive sign and negative sign respectively. The exponent component in single precision gets 8 bits memory with the bias value of 127 and in double precision gets 11 bits memory with the bias value of 1023. The Mantissa component is used to represent the precision bits of a number. The latest optimization of IEEE 754 standard is available with base 2. The single and double precision IEEE 754 standard floating-point memory allocations are illustrated in Figure 3.

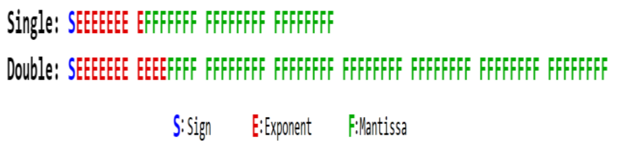

Single: SEEEEEEE EFFFFFFF FFFFFFFF FFFFFFFF
Double: SEEEEEEE EEEEFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
S: Sign
E: Exponent
F: Mantissa

Fig 3. IEEE 754 standard floating-point memory allocation

Special Values such as NaN (Not a Number), +∞ (Positive Infinity) and -∞ (Negative Infinity) are handled substantially in IEEE 754 standard. The operations with special values are listed in Table III.

Table III: Operations with Special Values

S. No.	Process	Result
1	$\frac{n}{\pm\infty}$	0
2	$\pm\infty \times \pm\infty$	$\pm\infty$
3	$\frac{\pm n \neq 0}{\pm 0}$	$\pm\infty$
4	$\pm finite \times \pm\infty$	$\pm\infty$
5	$\infty + \infty$ $\infty - \infty$	$+\infty$
6	$-\infty - \infty$ $-\infty + -\infty$	$-\infty$
7	$\infty - \infty$ $-\infty + \infty$	NaN
8	$\frac{0}{0}$	NaN
9	$\frac{\infty}{\infty}$	NaN
10	$\infty \times 0$	NaN
11	NaN==NaN	FALSE

The normalizing procedures used in GPUs cause accuracy issues, thus all the floating-point number are converted into fixed-point number in MPCEG. The gain says in this conversion are signed-unsigned handling, memory overflow and Invalid results. The signed-unsigned problem occurs when the result of an operation is negative and the host variable is unsigned type. Memory overflow occurs while processing huge operands. The invalid results refer conditional such as divide-by-zero.

MPCEG typecasts all data types into signed one to overcome the signed-unsigned handling issues. Memory overflow is prevented by checking the size of the input operands before fed them to process and discarding huge size data from calculations to trigger the overflow indication message and to save processing time. The memory overflow condition uncommon in could environments due to the sufficient memory resources. The invalid results are handled in the same way as IEEE 754 standard. Pointers are used to preserve the precision of input operands. The decimal places are neutralized using these pointers before converting them into fixed-point numbers. Same pointers are used to place the decimal point in the results based on the operation performed. A floating-point operation and its equivalent fixed-point operation are explained in Figure 4.

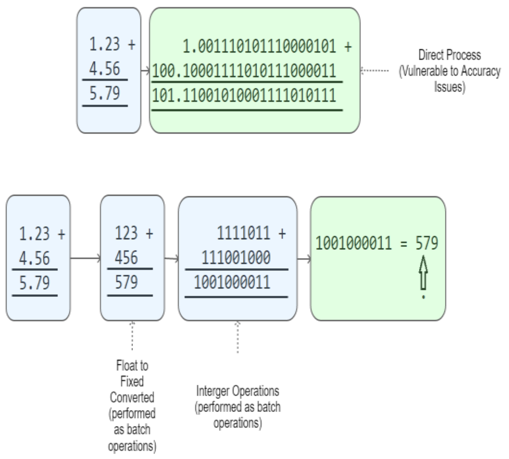


Fig 4: Example Floating-point operation and its fixed-point equivalent

MPCEG Floating-point to Fixed-Point Conversion algorithm is given below

Algorithm 3: MPCEG Floating-Point to Fixed-Point Conversion

```
Let $\gamma$ be the maximum permitted data size
Let $\delta_a$ and $\delta_b$ are the decimal location pointers of operand a and b
Let $\eta$ be the number available cores in GPU
Let $O$ be the operator
For $x = 1$ to $x \leq \gamma$ do
    Parallel_GPU_Execute($\eta$) Begin
        $\delta_a = GetDecimalPosition(a)$
        $\delta_b = GetDecimalPosition(b)$
        $Normalize(a, b, max\ (\delta_a, \delta_b))$
        $c = ApplyBatch(a, b, O);$
    Parallel_GPU_Execute End
End for
$FixDecimal(c, \delta_a, \delta_b, O)$
```

This algorithm is applied in parallel to the GPU cores as explained in Figure 1.

C. *Hierarchical Offloading Aggregation (HOA)*

Offloading the job and aggregating the completed tasks is an important phase in Cloud computing. In this work, the work dilution process is designed in to two stages. The first one is dealing with the work allocation between the virtual machines. The second one is thinning of tasks into sub-levels to assign them into CPU and GPU processing units. The CPU is dedicated to do this process. Gathering and accumulating the results from GPU cores is the main task of the CPUs in the Virtual Machines, whereas the GPU units performs the parallel calculations simultaneously in their massive number of cores. The data and control flow in HOA process is explained in Figure 5.

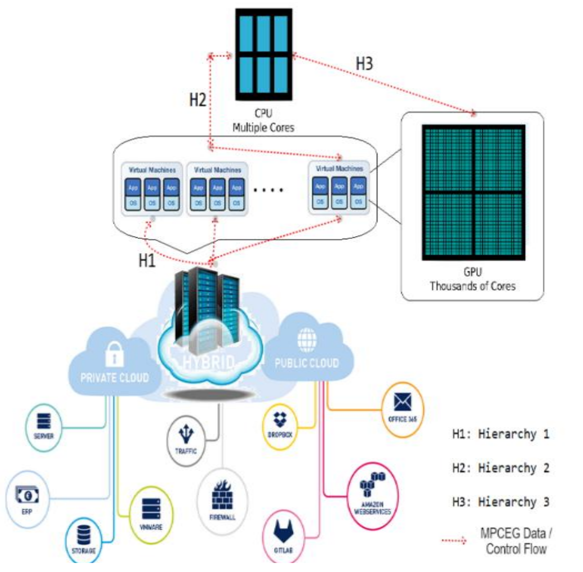


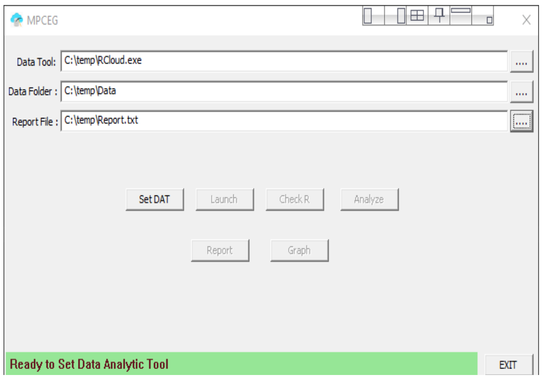
Fig 5: HOA hierarchical data / control flow

Data are collected for processing in Hybrid Cloud. The first level of hierarchy H1 starts here. Entire work is parsed and offloaded to different Virtual Machines in the server. Parsed jobs are distributed to different Processors cores in hierarchical level H2. Each core of the CPU then splits the jobs into different tasks and allocates them to the GPU cores. The aggregation process is performed in straight reverse process. Processed results are collected from the GPU cores by the CPU cores. Then the Virtual Machine accumulates the data from CPUs. The hybrid cloud server ensembles the entire process.

V. EXPERIMENTAL SETUP

The experimental setup is created based on the HP ProLiant DL160 Gen9 Intel Xeon E5-2620v4 12 core cloud server [25]. This server is equipped with Intel Xeon 2.1 GHz 8 core processor with 20MB L1 cache memory, 16GB DDR4-2400 RAM in 16 memory slots operates with 240V 1.8 KW power. To evaluate the existing methods and proposed method, this server is leased for 6 months along with its dedicated cloud services.

Since the existing methods are implemented in different programming languages, a standard Common Language Runtime (CLR) [26] environment is created to measure the performance metrics of the procedures. Visual Studio 2013 IDE [27][28] is used to create the CLR and Visual C++ programming language is used to write scripts to utilize the cloud services. A Legacy User Interface is designed to upload data and to distribute them in cloud environment. The data analytical tool R is accessed in the Cloud server through the legacy user interface. All algorithms are triggered to run in the server one by one and their performances are logged for generating the report file and comparison graphs. The user interface is showed Figure 6. The R-Cloud [29] is a community of github social coding provides the R-Cloud connectivity tool which is given in Figure 7.



Data Tool: C:\temp\RCloud.exe
Data Folder : C:\temp\Data
Report File : C:\temp\Report.txt
Ready to Set Data Analytic Tool

Fig 6: MPCEG User Interface

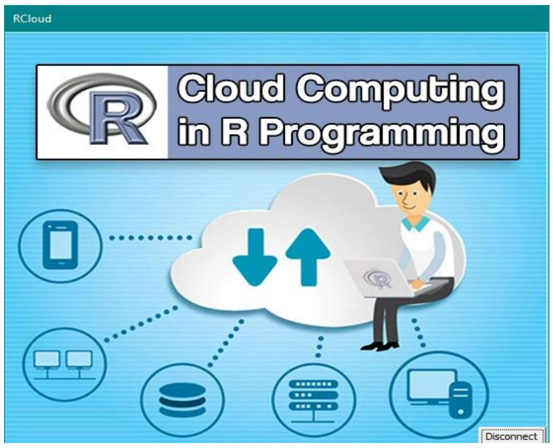


Fig 7. R-Cloud Connectivity Tool

VI. RESULTS AND ANALYSIS

The benchmark datasets [30][31] T10I4D100K, Kosarak, T40I10D100K, Mushroom, Chess and Accidents are used to evaluate the performance of the different procedures discussed in this work. Parameters like Processing Time, Accuracy, Precision, Recall and memory consumption are measured in the cloud server to prepare a detailed report and comparison graphs. The records from the above-mentioned datasets are treated as 100% data given for the methods in comparison. A total number record in the datasets accumulation is 7707525. The metrics are measured after processing 1541505 number of records and the results are tabulated for every 20% of data.

A. *Accuracy*

Accuracy is the main objective of data mining. A data mining procedure is considered as a best method if it provides more accuracy.

Accuracy is calculated using the standard formula $\frac{(TP+TN)}{(TP+TN+FP+FN)}$. Based on the observed values, proposed MPCEG has secured 90.69% of accuracy than the nearest achiever NCCARSD which scored 86.59% of accuracy. Calculated Accuracy values are provided in Table IV and a comparison graph is plotted in Figure 8.

Table IV: Accuracy (%)

Accuracy (%)						
Data (%)	**AMIN**	**ICMIN**	**PMFIHHC**	**CEKBDM**	**NCCARSD**	**MPCEG**
20	86.19	81.95	80.71	88.13	86.38	91.15
40	85.64	82.44	79.24	89.54	86.99	90.95
60	85.41	80.61	79.98	88.35	86.06	90.26
80	85.32	82.24	81.83	87.75	87.18	90.44
100	84.92	81.75	80.06	88.88	86.35	90.67

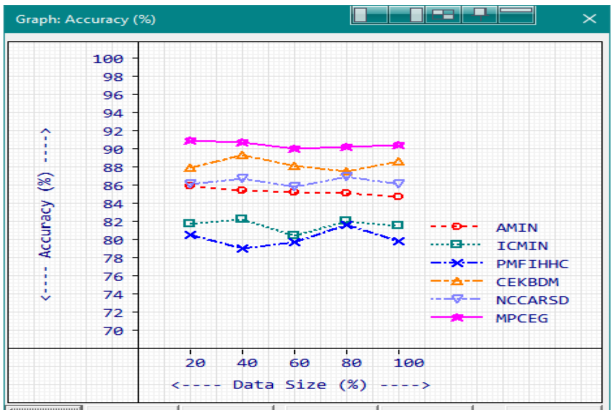


Fig 8. Accuracy Graph

B. *Precision*

Precision is also called as positive predicted value, which refers the specificity of a particular data mining procedure. Precision is calculated using the formula$\left(\frac{TP}{TP+FP}\right)$. The higher values of prediction refer the higher quality of a data mining procedure. Prediction is represented in terms of percent (%) in general. Based on the execution results of existing and proposed methods, the value of precision is calculated and given in Table V. The highest precision average of 90.14% is achieved by MPCEG method. The comparison graph for precision values is given in Figure 9.

Table V: Precision (%)

Precision (%)						
Data (%)	**AMIN**	**ICMIN**	**PMFIHHC**	**CEKBDM**	**NCCARSD**	**MPCEG**
20	85.64	81.67	80.71	88.74	87.27	90.8
40	84.84	82.48	79.44	89.08	85.54	90.01
60	86.6	80.74	80.89	89.35	86	90.64
80	85.37	82.49	81.94	88.06	87.69	89.31
100	85.71	81.83	80.26	88.38	86.31	89.93

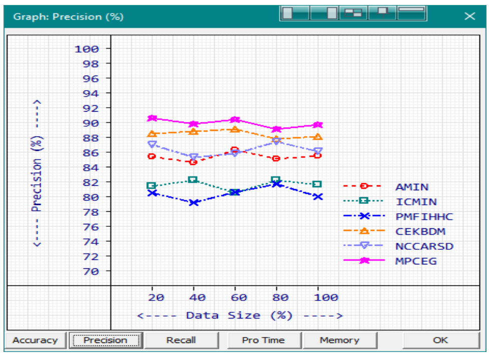


Fig 9. Precision Graph

By observing the values, it is learned that the existing method PMFIHHC which has the higher execution speed than the ICMIN, scored the precision values of 80.71%, 79.44%, 80.89%, 81.94% and 80.26% with the precision average of 80.65%. Proposed MPCEG scored the precision values of 90.8%, 90.01%, 90.64%, 89.31% and 89.93% in lesser processing time. Than all the other existing methods.

C. *Recall*

The term Recall and Sensitivity are interchangeable in data mining. Recall is calculated using the general formula$\left(\frac{TP}{TP+FN}\right)$. This refers the unit of relevant instances successfully retrieved from the overall relevant documents. A good data mining procedure should have higher sensitivity towards the input data. The Recall values of data mining procedures are given in Table VI. The recall values for existing and proposed methods are plotted as the graph for ease of understanding – given in Figure 10.

Table VI: Recall (%)

Recall (%)						
Data (%)	**AMIN**	**ICMIN**	**PMFIHHC**	**CEKBDM**	**NCCARSD**	**MPCEG**
20	86.6	82.13	80.71	87.67	85.75	91.43
40	86.23	82.41	79.12	89.9	88.09	91.73
60	84.59	80.54	79.44	87.59	86.09	89.95
80	85.28	82.08	81.76	87.52	86.8	91.38
100	84.39	81.69	79.94	89.27	86.38	91.28

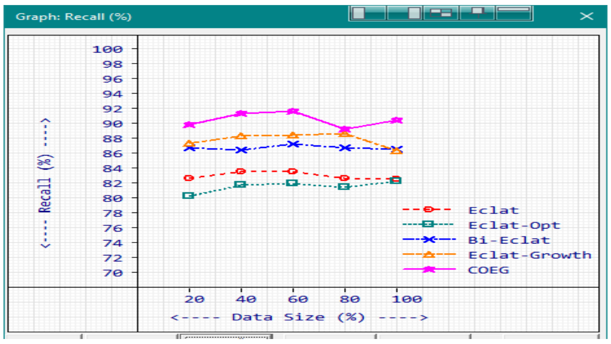


Fig 10. Recall values Comparison Graph

D. *Processing Time*

Processing time is one of the vital arguments in data mining procedures. The execution time of existing and proposed methods is measured here in millisecond units
for higher precision. The entire transaction records are split into 20% data chunks and the processing time is measured after every data chunk is processed. Therefore, 5 processing times are noted for every data mining procedure discussed here. The observed processing times are produced in Table VII.

Table VII: Processing Time (mS)

Processing Time (mS)						
Data (%)	**AMIN**	**ICMIN**	**PMFIHHC**	**CEKBDM**	**NCCARSD**	**MPCEG**
20	5585	4982	4739	7690	4621	3876
40	11180	10046	9566	15450	9328	7762
60	16851	15008	14286	23144	13908	11651
80	22417	19940	18969	30775	18522	15536
100	28004	24933	23737	38515	23204	19406

Based on the results, it is observed that the proposed method MPCEG accomplished the mining task in 19406 mS which is lesser than the processing time 23204 mS of NCCARSD. MPCEG completed the data mining process first in all 5 different stages of the entire mining process. The comparison graph for processing time is given in Figure 11.

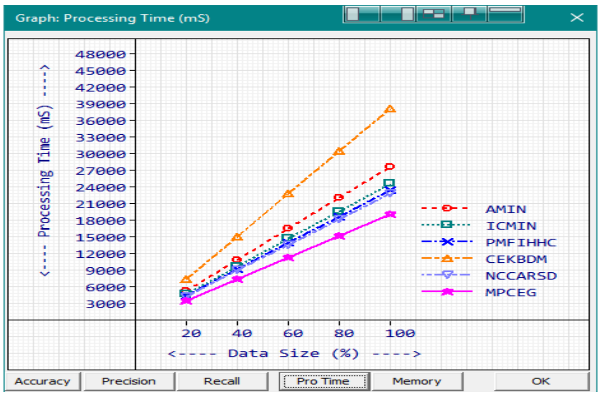


Fig 11. Processing Time (mS)

E. *Memory*

Memory is one of the important elements in computational resources. Every computational task occupies some amount of memory in Random Access Memory (RAM) which is considered as primary memory of a computer system. Adequate free memory in RAM is required to run a computer system optimally. Overloading the RAM will cause virtual memory utilization in which the process depends on the secondary memory. While comparing primary memory, secondary memory accessing is very slow, thus it leads to prolonged processing time. Therefore, it is important that not to overload a single system to the core to get the better performance. Proposed MPCEG uses a well-designed offloading mechanism splits the data mining process into several small tasks and distributes them to the appropriate execution CPU – GPU cores. The memory utilization is measured throughout the execution of data mining procedures and the result is given in Table VIII.

Table VIII: Memory (kB)

Memory (kB)						
Data (%)	**AMIN**	**ICMIN**	**PMFIHHC**	**CEKBDM**	**NCCARSD**	**MPCEG**
20	824	587	539	803	663	403
40	1649	1180	1084	1613	1329	813
60	2481	1772	1620	2412	1994	1216
80	3304	2356	2167	3221	2660	1618
100	4128	2944	2708	4026	3320	2020

Based on the experiment results, it is observed that the proposed method uses the minimum memory of 403 kB for handling 1541505 number of transaction data and it consumes 2020 kB of memory to handle 7707525 number of transaction data. The nearest performing method is the PMFIHHC with the memory consumption of 539 kB to 2708 kB. The memory utilization is plotted as graph for comparison and presented as Figure 12.

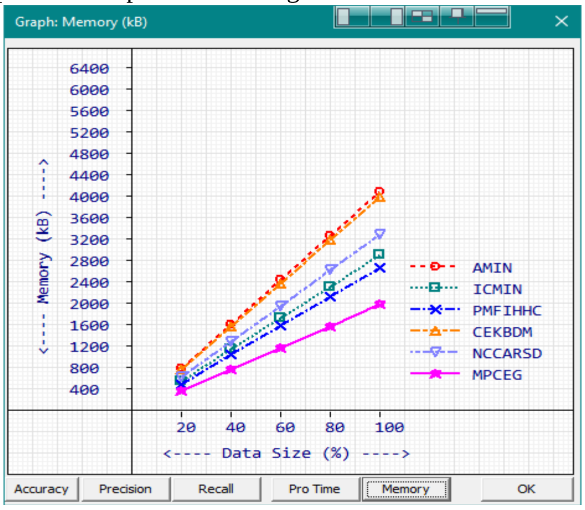


Fig 12. Memory Consumption (kB)

VII. CONCLUSION

Renowned enhancements are introduced in computing hardware architectures frequently these days. Cloud is the most modern economical technology used by the Information Technology companies. Multi-core Processing Cloud Eclat Growth is introduced and tested in this work which performs frequent itemset mining procedure in more efficient way by expending the usage of massive number of GPU cores. The data mining speed is increased with memory efficiency without compromising the essential elements of data mining such as accuracy, precision and recall.

REFERENCES

1. Wolfram Höpken, Tobias Eberle, Matthias Fuchs and Maria Lexhagen, "Search Engine Traffic as Input for Predicting Tourist Arrivals" in Information and Communication Technologies in Tourism 2018, Springer - 2017, pp. 381-393
2. R. L. Davidson and C. P. Bridges, "Error Resilient GPU Accelerated Image Processing for Space Applications", IEEE Transactions on Parallel and Distributed Systems (Volume: 29 , Issue: 9) IEEE 2018, pp.1990-2003
3. Bert David, "AWS: Amazon Web Services Tutorial for Beginners" in ACM Digital Library, dl.acm.org – 2018
4. Geof Sawaya, Michael Bentley, Ian Briggs, Ganesh Gopalakrishnan and Dong H. Ahn, "FLiT: Cross-platform floating-point result-consistency tester and workload" in 2017 IEEE International Symposium on Workload Characterization (IISWC), IEEE 2017, pp. 229-238

5. Karl E. and Anselmo Lastra, "GPU Floating-Point Paranoia" in Proceedings of GP2, cs.unc.edu- 2004
6. [6] Sanjay Rathee and Arti Kashyap, "Adaptive-Miner: an efficient distributed association rule mining algorithm on Spark" in Journal of Big Data, Springer 2018, pp. 1-17
7. PeterBraun, Alfredo Cuzzocrea, Carson K.Leung, Adam G.M.Pazdor and JoglasSouzaa,"Item-centric mining of frequent patterns from big uncertain data" in Knowledge-Based and Intelligent Information & Engineering Systems Volume 126, Elsevier 2018, pp. 1875-1884
8. Dhanashree Shirke and Deepti Varshney, "Parallel Mining of Frequent Itemsets in Hadoop Cluster Having Heterogeneous Nodes" in International Journal of Advance Research in Computer Science and Management Studies Volume 5 Issue 7, IJARCSMS - 2017, pp. 130-136
9. Qiang He, Xiaodong Zhu, Dongwei Li, Shuliang Wang, Jun Shen and Yun Yang, "Cost-Effective Big Data Mining in the Cloud: A Case Study with K-means" in IEEE 10th International Conference on Cloud Computing (CLOUD), IEEE 2017, pp.74-81
10. Victor Andres Ayma Quirita, Gilson Alexandre Ostwald Pedro da Costa, Patrick Nigri Happ, Raul Queiroz Feitosa, Rodrigo da Silva Ferreira, Dário Augusto Borges Oliveira and Antonio Plaza, "A New Cloud Computing Architecture for the Classification of Remote Sensing Data" in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (Volume: 10 , Issue: 2), IEEE 2016, pp. 409-416
11. Qiu H, Gu R, Yuan C and Huang Y. Yafim, "a parallel frequent itemset mining algorithm with spark" in IEEE international parallel distributed processing symposium workshops, IEEE 2014. pp. 1664–1671
12. Rathee S, Kaul M and Kashyap A. R-Apriori,"An efficient apriori based algorithm on spark" in Proceedings of the 8th workshop on Ph.D. workshop in information and knowledge management (PIKM 15), ACM 2015. pp. 27–34
13. Chun-Kit Chu, Ben Kao and Edward Hung "Frequent itemsets from uncertain data" in Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer 2007, pp. 47-58
14. M.Zaharia, A.Konwinski, A.Joseph, Y.zatz and I.Stoica, "Improving mapreduce performance in heterogeneous environments" in 8th USENIX Symposium on Operating Systems Design and Implementation, OSDI - 2008, pp. 29-42
15. D. Aloise, A. Deshpande, P. Hansen and P. Popat, "NP-Hardness of Euclidean Sum-of-Squares Clustering" in Machine Learning (volume-75, no. 2), Springer 2009, pp. 245-248
16. S. Lloyd, "Least Squares Quantization in PCM" in IEEE Transactions on Information Theory (Volume-28, no. 2), IEEE 1982, pp. 129-137
17. Alex Holmes, "Hadoop in Practice" book published by Manning Publications Co. Greenwich, CT, USA ©2012 ISBN:1617290238 9781617290237, ACM Digital Library 2012
18. Zhiyong Ma, Juncheng Yang, Taixia Zhang and Fan Liu, "An Improved Eclat Algorithm for Mining Association Rules Based on Increased Search Strategy" in International Journal of Database Theory and Application Vol.9, No.5, IJDTA - 2016, pp. 251-266
19. Michael Gowanlock, David M. Blair and Victor Pankratius, "Exploiting Variant-Based Parallelism for Data Mining of Space Weather Phenomena" in IEEE International Parallel and Distributed Processing Symposium (IPDPS), IEEE 2016, pp. 760-769
20. Feng Zhang, Min Liu, Feng Gui, Weiming Shen, Abdallah Shami and Yunlong Ma, "A distributed frequent itemset mining algorithm using Spark for Big Data analytics" in Cluster Computing (Volume 18, Issue 4), Springer 2015, pp. 1493–1501
21. Yuan-Shao Huang, Kun-Ming Yu, Li-Wei Zhou, Ching-Hsien Hsu and Sheng-Hui Liu, "Accelerating Parallel Frequent Itemset Mining on Graphics Processors with Sorting" in Network and Parallel Computing, Springer 2013, pp. 245-256
22. Paweł Czarnul, "Parallelization of large vector similarity computations in a hybrid CPU+GPU environment" in The Journal of Supercomputing (Volume 74, Issue 2), Sprinnger 2018, pp 768–786
23. Fredy Juarez, Jorge Ejarque and Rosa M.Badia, "Dynamic energy-aware scheduling for parallel task-based application in cloud computing" in Future Generation Computer Systems (Volume 78, Part 1), Elsevier 2018, pp.257-271
24. Silviu Răileanu, Florin Anton and Theodor Borangiu, "High Availability Cloud Manufacturing System Integrating Distributed MES Agents" in International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing (Service Orientation in Holonic and Multi-Agent Manufacturing), Springer 2017, pp 11-23
25. https://serverental.com/product/hpe-proliant-dl160-gen9-server-sale/
26. https://docs.microsoft.com/en-us/dotnet/standard/clr
27. Mickey Gousset, Martin Hinshelwood, Brian A. Randell, Brian Keller, Martin Woodward, "Professional Application Lifecycle Management with Visual Studio 2013", WILEY Publications
28. Sven Amann, Sebastian Proksch, Sarah Nadi, Mira Mezini, "A Study of Visual Studio Usage in Practice", Software Analysis, Evolution, and Reengineering (SANER), IEEE 2016
29. https://rcloud.social/index.html
30. Giang Nguyen, Tuong Le, Bay Vo, Bac Le, "A New Approach for Mining Top-Rank-k Erasable Itemsets", Asian Conference on Intelligent Information and Database Systems, Springer 2014
31. Junrui Yang, Cai Yang, Yanjun Wei, "Frequent Pattern Mining Algorithm for Uncertain Data Streams Based on Sliding Window", Intelligent Human-Machine Systems and Cybernetics (IHMSC), IEEE 2016

AUTHORS PROFILE

V.Priya received her B.Sc Computer Science from University of Madras in 1998 and M.Sc Computer Science from Nehru Memorial College, Puthanampatti,Bharathidasan University in 2001 and M.Phil Computer Science from Bharathidasan University in 2004. She is now pursuing her Ph.D degree in computer Science in Bharathidasan University where she is currently working as an Assistant Professor. She has 18 years of teaching experience in the field of Computer Science. Now she is Pursuing Ph.D Degree in Computer Science from Bharathidasan University. Her research interest include Network Security Data Mining and Cloud Computing.

S.Murugan received his M.Sc degree in Applied Mathematics from Anna University in 1984 and M.Phil degree in Computer Science from National Institute of Technology formerly known as Regional Engineering College, Trichirappalli in 1994. He is associated with the department of Computer, Nehru Memorial College (Autonomous), affiliated to Bharathidasan University since 1986 where he is currently working as an Associate Professor in Nehru Memorial College(Autonomous). He has 32 years of teaching experience in the field of Computer Science. He has completed his Ph.D degree in Computer Science from Bharathiyar University in 2015 with Data Mining specialization. His research interest includes Data and Web Mining and also published many research articles in the national and international journals.