Improving Performance in Cloud using Multi-Job Scheduling based Group Discovery Algorithm

Nandhini A.
PG Scholar
Sri Ramakrishna Engineering College,
Tamil Nadu, India

Saravana Balaji B.
Assistant Professor
Sri Ramakrishna Engineering College,
Tamil Nadu, India

ABSTRACT:

Cloud computing is a large model change of computing system. It provides high scalability and flexibility among an assortment of on-demand services. To imporve the performance of the multi-cloud environment in distributed application might require less energy efficiency and minimal inter-node latency correspondingly. The major problem is that the energy efficiency of the cloud computing data center is less if the number of server is low, else it increases. To overcome the energy efficiency and network latency problem a novel energy-efficient particle swarm optimization representation for multi-job scheduling and Latency representation for the grouping of nodes with respect to network latency is proposed. Design a realistic particle swarm optimization algorithm for the cloud servers and construct an overall energy competence based on the purpose of the servers and calculation of fitness value for each cloud servers. Also, in order to speed up the convergent speed and improve the probing aptitude of our algorithm, a local search operative is introduced. Finally, the experiment demonstrates that the proposed algorithm is effectual and well-organized.

Keywords

Cloud Computing, Network Latency, Energy Efficiency, Particle Swarm Optimization (PSO) and Multi-Job Scheduling.

1. INTRODUCTION

Cloud computing becomes a one of important rising and an empowering technology in now a days, because which through us to consdier beyond what is possible. These technologies related services are offered by general 3-tier architecture. The complete architecture of the cloud technology performed based on the user request to service the requirments or perform scheduling based on user required task or job.In order to perform scheduling strategy it incorporates of the resource provider, the service.

Providers and the consumers are the Service provider requires particular new hardware resources or resource provider to perform service request given by the user. It becomes the one of the technology [1] for empowering well-situated, on-demand network admission to a public collection of specific cloud computing resources ,that can be quickly provisioned and delivered with less organization effort.

According to the current CEMS project from amazon's [2], amortization schedule for 3-years and amortization schedule for 15 -years are crosspodingly used for servers. In other words, the largest speculation to construct data centers for cloud computing is not simply to obtain millions of server equipment, but also to acquire the distribution and cooling infrastructure and to compensate the bill for energy utilization of every one of these facilities.

In order to solve these both network latency and energy efficiency difficulties a latency and particle swarm optimization based energy-efficient multi-job scheduling is proposed to supports both the problems. This effort mostly focuses on how to develop the energy efficiency and decrease the network latency of servers during suitable scheduling strategies. Taking complete concern of the association among the performance and energy utilization of servers, network latency for energy efficiency proposes a new particle swarm optimization based energy-efficient multi-job scheduling model based on the cloud environment, MapReduce and give its equivalent particle swarm optimization algorithm. Latency based is used model to grouping the nodes with smallest inter-node latency together. This algorithm fulfills all the necessities mentioned in the problem statement. The network latency algorithm is moderately able to group the nodes with the less obtainable latency information and make equally selected groups with negligible communication overhead with less energy efficiency in cloud environment. But, the requirement's in the existing methods that does not satisfied in one or more condition. But none of them fulfills all of the requirements. Proposed work fulfills all the requirements as mentioned above. The whole work of the paper is organized as below the section 2 study the network latency and energy efficiency problem related methods, section 3 latency and particle swarm optimization based energy-efficient multi-job scheduling for group discovery algorithm, section 4 experimental results were measured and finally section 5 concludes the result.

2. RELATED WORK

Clustering or grouping of the nodes in the cloud environment has a several characteristics is referred to as community detection. Community, group, or cluster detection comprise be completed for various areas and fields, together with internet [3], WWW [4], biological network [5], extract network [6], graph theory [7] and so on. Dissimilar algorithms have been proposed to modify to the requirements of the regulation. The majority of these algorithms are based on grouping of nodes on the foundation of association calculation for an exacting node.

Resource Aware Cloud computing framework (RAC2) [8] and the accomplishment of its algorithm is constituent of the Resource Awareness Cloud Scheduling (RACS) constituent. RACS component assist the scheduler in the creation more of well-organized scheduling decisions on the foundation of dissimilar resource characteristics/ criterion. Network latency becomes one of those criterions. With the assist of cloud resource manager, RACS has information regarding the numeral number of nodes obtainable from dissimilar cloud operator and community detection. Due to the advance development of cloud service provider technology, an organization has to choose variety of cloud operators. For carrying out tests rationale, have plugged in our scheduler section with ProActive scheduler [3]. ProActive is an open source cloud middle-ware, which enables the consumer to accomplish its tasks on a group or cloud infrastructure.

Anderson et al. [9] developed a Local spectral partitioning algorithm for partitioning the cloud operations. It is one of the types of graph partitioning algorithm and it use page rank vector to attain its partitioning procedure for operators. They expected to decrease the conductance in their algorithm. It can cluster the nodes on the beginning of connection assessment and generate equally restricted groups. It also does not necessitate an algorithm to run every time, whenever you like a procedure demands for a cluster.

Lang et al. [10] also presented a graph partitioing algorithm Flow-based Metis+MQI .The main aim of the algorithm is to increase the conductance of graph cuts. They have shared MQI with Metis to achieve a heuristic graph partitioner to discover the outcome. It can cluster the nodes on the beginning of connection value and construct equally restricted groups.

Garcia et al. [11] presented a Hierarchical compact algorithm and hierarchical star algorithm for measuring the group nodes collection; it is based on universal structure for agglomerative hierarchical clustering. In which they experienced and compared these algorithms. The structure is based on graphs. They attain the dissimilar hierarchical agglomerative clustering algorithms starting this structure by using dissimilar measures.

To decrease the energy utilization of data centers and recover energy efficiency, numerous scholars have completed a number of associated researches, such as literatures [12-16]. Decrease energy addicted by cooling scheme. For instance, you preserve to use Google's "free cooling" approach, heat of each servers from free cooling mode are removed by using low heat ambient air. Google states that still there is no specfic cooling equipment avaliable for data centers in Belgium.

According to the success of google engineers, the climatic condtions of the belgium support free cooling approximately year-round, with temperatures increasing higher than the suitable assortment for free cooling concerning seven days per year. Improve the energy efficiency [17] of servers are responsive with the aim of low energy consumption of a server is mostly outstanding to its idle status caused by low CPU consumption.

3. LATENCY AND PARTICLE SWARM OPTIMIZATION BASED ENERGY-EFFICIENT MULTI-JOB SCHEDULING FOR GROUP DISCOVERY ALGORITHM

In existing network latency grouping algorithm doesn't add the energy efficiency conception in all stages of group discovery algorithm, so it becomes more energy consumption with less network latency to overcome the problem, a proposed work tradeoff among energy consumption and network latency for group discovery algorithm. The basics of the proposed work are grouping the similar nodes with less inter-node latency and less energy utilization for every service provider (SP). In the initial stage of cloud scheduler, it makes the decision for scheduling based on the grouping information obtained from network latency and PSO based energy efficiency multi-job scheduling algorithm. In this resolution, it is reasonably proficient to cluster the nodes with the smallest obtainable latency and enegry infromation results from PSOBEMJS ( ). Latency and energy information is accessible only for those nodes, which comprise done several communications through other nodes. Thus the sensible result of which heuristically performs the combination decision on incomplete latency and energy information.

The algorithm runs for each and every time to find the less network latency and minumum energy efficiency also evaluated using multi job scheduling algorithm in same intiliazation phase. This latency and energy efficiency is compared to a threshold, PMU and group threshold value, for making the decision to group the similar nodes. The iteration of the algorithm is performed when it is equals to the number of nodes with number of times. The algorithm has three dissimilar phases; initialization, configuration and reconfiguration. Initialization phase is dependable to place the situation for the algorithm implementation. Configuration phase is dependable to find out the groups, it is the major core part of the phase. In detail it is the concrete group discovery phase. Final phase of the algorithm is Reconfiguration phase. It group the node which maynot be grouped with other nodes than the nodes grouped by configuration phase. Before going more into the details of these phases, first introduce the energy efficiency algorithm that are going to be used in group discovery algorithm.

Improving the energy effectiveness of cloud servers, the Power usage effectiveness (PUE) of data centers can be improved. Though, this energy utilization problem cannot be easily solved by using general load balancing between the servers consequently while to formulate all the servers CPU utilization as an alternative, there exists an best performance and energy peak for each server [18]. Energy utilization per task is inclined by the CPU consumption of servers other than, in order to make simpler the representation, simply consider the impact of CPU utilization. While the CPU utilization is small, power of each server is not mentioned effectively and therefore the energy efficiency for each task is high. In other hand while the CPU utilization is high as well as the energy utilization of each server is high, due to the contest for resources between tasks, it leads to degradation of performance and more execution time per task. In previous work the variation of energy for each task is measured by using "U-shaped" curvature with CPU utilization. So it can be understood that the servers accomplish the optimal energy efficiency when all servers consecutively at its best possible performance. Get this best optimal result of enery efficiency for each server, first give the major work related description of the energy-efficient multi-job scheduling difficulty and then build its matching particle swarm optimization (PSO) algorithm.

The scheduling procedure of the algorithm considers the N servers for each data center. Consider the present utilization of the CPU resources for each server k is denoted as $\mathrm{CS_k}$ and its best optmial values of CPU utilization are $\mathrm{CO_k}$, where $\mathrm{k} = 1,2,\ldots\ldots.\mathrm{N}$. If there are J jobs to perform for each cloud server $\mathrm{A} = \{\mathrm{A_1},\ldots\ldots.\mathrm{A_j}\}$ necessitate being process and the input server data for each job $\mathrm{A_q}$ is $\mathrm{D_q}$ where $\mathrm{q} = 1,2,\ldots\ldots.\mathrm{J}$. The input data file of each and every cloud server data $\mathrm{D_q}$ determination is separated into $\mathrm{m_q}$ equal splits with each size, consequently present be $\overline{\mathrm{m}} = \sum_{\mathrm{q}=1}^{\mathrm{F}} \mathrm{m_q}$ splits, which are denoted s $\mathrm{D} = \{\mathrm{D_1},\mathrm{D_2},\ldots\ldots..\mathrm{D_F}\} = \{\mathrm{d_1},\mathrm{d_2},\ldots\ldots..\mathrm{d_{\overline{m}}}\}$. It first required storing the result of each splits on N servers. To make sure the consistency of data, every split might select combination of three dissimilar servers for each and every storage node. Using a $\overline{\mathrm{m}} \times 3$ matrix P to correspond to the storage space position of each split, and the constituent p_{ij} indicates the storage space position of split d_i, somewhere integer $p_{ij} \in [1, N], i = 1,2, \ldots \ldots . \bar{m}$ denoted the nodes and $j = 1,2,3 \ldots \ldots$the corresponds to the MapReduce structure, know that every input data D_q determination to be procedure by map and reduce process, it is represented as m_q and r_q with less CPU utilization for map and reduce process, it is also represented as CM_q and CR_q.

The major difficulty of the process is how to assign the job to each and every split of data center for N server with less energy efficiency, for that purpose need to convert the map and reduce task data into vector form of representation that is $v = \sum_{q=1}^{F} m_q + \sum_{q=1}^{F} r_q$. The vector representation of data center is converted into scheduling task with current node and NeighborNode $S = (s_1, s_2, \ldots \ldots s_i, \ldots \ldots . s_v)$ to represent the completed task of scheduling procedure, and the i^{th} constituent of vector S indicate that task i is assigned on server s_i where $1 \le s_i \le N$ and $i = 1,2,\ldots,v$. Traversing throughout the scheduling process S and achieve the set of map and reduce results for each and every job A_q which are assigned on server k, it is denoted as M_q^k and R_q^k, correspondingly, where $k = 1,2,\ldots,N$ and $q = 1,2,\ldots,J$ Let $NM_q^k = |M_q^k|$ and $NR_q^k = |R_k^q|$. Here give the PSO optimization model for the multi-job scheduling difficulty with energy efficiency based on MapReduce structure for cloud computing

$$\min \sum_{k=1}^{N} \left(CO_K - \left(CS_K + \sum_{q=1}^{F} (NM_K^q \times CM_q) + \sum_{q=1}^{F} (NR_K^q \times CR_q) \right) \right)^2 \qquad (1)$$

satifies that for scheduling schema S,

$$\begin{cases} s_i \in [p_{i1}, p_{i2} p_{i3}] & \text{for } i = 1,2, \ldots \ldots . \bar{m} \\ s_i \in [1, N] & \text{for } i = \bar{m}+1, \bar{m}+2, \ldots \ldots v \end{cases} \qquad (2)$$

$$NM_K^q = \left| \left\{ s_i | s_i = k, i = \sum_{j=0}^{q-1} m_j + 1, \sum_{j=0}^{q-1} m_j + 2, \ldots, \sum_{j=0}^{q-1} m_j + m_q \right\} \right| \qquad (3)$$

where $m_o = 0, k = 1,2,,, N, q = 1,2, \ldots F$

$$NR_K^q = \left| \left\{ s_i | s_i = k, i = \bar{m} + \sum_{j=0}^{q-1} r_j + 1, \sum_{j=0}^{q-1} r_j + 2, \ldots, \sum_{j=0}^{q-1} r_j + r_q \right\} \right| \qquad (4)$$

where $r_o = 0, k = 1,2,,, N, q = 1,2, \ldots F$

$$CS_k + \sum_{q=1}^{F} (NM_K^q \times CM_q) + \sum_{q=1}^{F} (NR_K^q \times CR_q) \le 1 \qquad (5)$$

where $k = 1,2,,, N, CM_q \in [0,1], q = 1,2, . F$

The objective funtion of the PSO algorithm needs to calculate the CPU utilization after scheduling and their optimal points. Equation (2) express to facilitate if there is map tasks$_i$ is assigned to server s_i and then it stores the corresponding data of this map tasks$_i$.These types of scheduling schema avoid a large-scale data progress by not only reducing the bandwidth, but also map and process the data in efficient manner. Equation (3) and (4) compute the number of map tasks NM_K^q and reduce tasks NR_K^q of job A_q which are assigned to server k. Equation (5) denotes to utilization of CPU for any server must not go beyond earlier than and following the task scheduling.

A particle swarm optimization (PSO) is the general comparable to a behaivour of fish or bird moving in dimensional search or investigation search space. Bases on the individuality of this energy efficient problem, agree to the integer coding, convert the map and reduce task data into vector form of representation that is $v = \sum_{q=1}^{F} m_q + \sum_{q=1}^{F} r_q$ including $\bar{m} = \sum_{q=1}^{F} m_q$ map tasks and $\bar{r} = \sum_{q=1}^{F} r_q$ reduce tasks. The intialization of every one particle is the to consider the tasks to be scheduled with vector $S = (s_1, \ldots . . s_v)$ as an individual to represent a scheduling scheme S indicate that task$_i$ is assigned on servers_i. All particles have fitness values demonstrating their performances of each and every server K and velocities which straight the journey of particles. Each particle location of each server energy efficiency best values are inclined by together both local and global best energy efficiency optimial results, it is referred to as pbest , gbest. Therefore particles have a tendency to fly towards an improved best energy efficiency exploration area throughout the search procedure. A particle position on the search freedom is specified by two basics, namely its velocity and position, which are updated in each and every generation as follows.

3.1 Energy efficiency particle swarm optimization based energy-efficient multi-job scheduling algorithm PSOBEMJS ( )

Step 1: Initializing the particles with scheduling vector $S = (s_1, \ldots . . s_v)$ and mapreduce functions NR_K^q and NM_K^q let $NM_K^q = 0$ and $NR_K^q = 0$ where $k = 1,2..N$ and $q = 1,2,..q$ empty set M_k and R_k

Step 2: For $i = 1,2, \ldots \ldots \ldots . N$ where N is the size of the paritcles that corresponds to the scheduling vector. For each element s_i of individual S set $k = s_i$ and $m_0 = 0$

Step 3: For a job A_q where $q = 1,2, \ldots \ldots J$

3.1. If $i = \sum_{j=0}^{q-1} m_j + 1, \sum_{j=0}^{q-1} m_j + 2, \ldots \ldots . . \sum_{j=0}^{q-1} m_j + m_q$, NM_K^q plus one and put i into set M_k

3.2. Else

3.3. If $i = \bar{m} + \sum_{j=0}^{q-1} r_j + 1, \sum_{j=0}^{q-1} r_j + 2, \ldots \ldots . . \sum_{j=0}^{q-1} r_j + r_q$, NR_K^q plus one and put i into set R_k

Step 4: For $j = 1,2,,, K$ where k is the number of servers and randomly initializes frequency to the servers

Step 5: Initialize a population of particles with random positions and velocities V[i] in the search space.

Step 6: Initialize the local best energy efficiency as p$Best$ of each scheduling vector

pbest [i] $= S_i$

Step 7: Evaluate objectives of each particle using the equation (2,3,4,5)

Step 8: Initialize the Global Best particle (gBest) with the best one among N the particles: g$Best$ = Best particle found in S_i

Step 9: End For

Step 10: Add the no dominated solutions found in S into objective function in step (7)

Step 11: Initialize the iteration number $t = 0$

Step 12: *Repeat until* $t > G$ (G is the maximum number of iterations)

12.1. for $i = 1,2, \ldots \ldots \ldots . N$

12.2. Randomly select the global best particle for S_i from the map reduce vector v store its position in *gBest*.

12.3. Calculate the new velocity $V[i]$ according to following equation

$$V_i^{k+1} = \omega V_i^k + c_1 r_1 (pbest_i - S_i^K) + c_2 r_2 (gbest - S_i^K)$$

$$S_i^{k+1} = S_i^k + V_i^{k+1}$$

12.4. Compute the new position of S_i according to following equation

$$V_i^{k+1} = V_i^k + ((R_1 \otimes (pbest_i \ominus S_i^K) \oplus (R_2 \otimes (gbest \ominus S_i^K) S_i^{k+1} = S_i^k \oplus V_i^{k+1}$$

Step 13: $if(t < G * PUE)$ *then* (PUE *is the* Power usage effectiveness*)* Perform mutation on S_i

Step 14: Find best S_i

Step 15: End For

Step 16: Update the personal best solution of each particle S_i

Step 17: *if* $S_i \leq pbest_i \vee S_i \sim pbests_i$ *then Then* $pbests_i = S_i$ *update current solution*

Step 18: *Add non relevant scheduling task to reduce* R_k

Step 19: if $S_i \neq x \; \forall \; x \in S$ *Then* $S = S \cup S_i$

Step 20: Remove all particles dominated by S_i in S

$S = S - \{y \in S | Y < S_i\}$

Step 21: If the archive is full then randomly select the article to be replaced from *map reduce with scheduling task* S_i with respect to NR_K^q and NM_K^q

Step 22: If the stopping criterion is met step 12, then output $gBest$ and its objective value; otherwise, go to Step (6)

Step 23: Return the best result

Power usage effectiveness (PUE) is used to evaluate the energy efficeincy of each data and it is the ration of sum of all power used by data center delivered to cloud computing equipment. It is to determine of how professionally a processor data center use its power:

$$PUE = \frac{Total\ faciltiy\ number}{IT\ equipment\ power}$$

Then now proceed how the energy efficiency concept is performed for each and every phase of the group discovery algorithm both energy efficiency and network latency .The procedure of these three phase as follows:

3.1.1 Initialization

Procedure definition:

```
CheckForGroupableToVisitNodeRoot (VisitNodem, NeighborNode),

CheckForEnergyEfficiency (VisitNodem, NeighborNode) from PSOBEEMJS( )

if sumOfLatency(neighborOfVisitNode,rootOfVisitNode) > groupThershold and sumOfLatency(neighborOfVisitNode,

rootOfVisitNode) > groupThreshold and if sumOfEnergyEfficiency(neighborOfVisitNode, rootOfVisitNode) < PUE then

increase moreThanGroupThresholdCounter

end

if anylatency (visitnode, neighbornode) < latency (visitNode, previous OfVisitNode) and anyenergy efficiency (visitnode, neighbornode) > latency (visitNode, previousOfVisitNode) then isVisitNodeRemovable = true

add visitnodeto tempgroup

End
```

3.1.2 Group Discovery Algorithm - Configuration Phase

Data: nodes, latency, threshold, groupThreshold,enegry efficiency ,PMU

Result: Groups of Nodes with minimal inter-node latency and less energy efficiency

Initialization:

```
for each node in registry do

select visitNode

If numberOfneighborOfVisitNode =0 OR ( previous ofVisitNode = null &visitnode ≠ grouped ) then

create newGroup

add visitNode to newGroup

end

for each neighbourNode of visitNode do

input latency(visitNode, neighborNode),

energy(visitNode,neighborNode)

if latency(visitnode,neighbornode)threshold & energyefficiency (visitnode, neighbornode) ≤ PMU then

if neighbornode ≠ grouped then

add neighbornode into tempGroup

call_proc: checkForGroupableToVisitNodeRoot
           (visitNode, neighborNode)

call_proc:CheckForEnergyEfficiency(VisitNodem,
           NeighborNode) from PSOBEEMJS( )

end

add neighborNode to tempGroup

end else
```

```
iflatency(visitnode,neighbornode) ≤ latency (visitnode, neighbornode) ≤ threshold & energyefficiency (visitnode, neighbornode) ≤ PMU then
if neignbornode ≠ grouped then
create newgroup
add neighbornode to newgroup
end
end
end
checkthat the tempgroup
is group table to groupOfRootNode or not
if morethanGroupThersholdCounter ≤ regroupingIndex then
copy tempgroup to groupOfVisitNode
end
else if ThersholdCounter > regroupingIndex then
create newgroup
if is VisitNodeRemovable = true then
add visitNode to newGroup
end
copy tempGroup to newGroup
end
rank visitNode to zero
end
```

3.1.3 Group Discovery Algorithm—Reconfiguration Phase:

Data: nodes, groups, latency, threshold, pairLatency, prevMinLatency,PMU,Energy efficiency

Result: Groups of Nodes with minimal inter-node latency and less energy efficiency

Initialization:

```
for each group in registry do
PrevMinLatency ← 0, PMU ← 0
if groupSize = 1 then
for each potentialGroup in registry do
if potentialGroupSize > 1 then
pairLatency←findPairLatency(currentNode, potentialGroupLeader)
CheckForEnergyEfficiency(VisitNodem,
            NeighborNode) from PSOBEEMJS( )
if pairLatency ≤ threshold and (pairLatency <
prevMinLatency or prevMinLatency = 0)
if energyefficiency≤ $PMU$ then
prevMinLatency←pairLatency prevMinPMU←taskenergy //from step 22 in PSOBEEMJS( )
move candidateNode to newGroup
newGroupNumber←potentialGroup
end
end
end
end
end
update latency and energy efficiency in registry
```

4. EXPERIMENTAL RESULTS

Experimental results are conducted for group discovery algorithm (GDA) with network latency and energy efficiency based on the three clusters. These clusters are physically co-located in the similar experiment excluding in the dissimilar building. Thus the network latency among their nodes and energy efficiency is not extremely high. On standard network latency among the nodes belong to the similar cluster is not a large amount dissimilar than the latency among the nodes belonging to the dissimilar clusters. This network model is moderately diverse from the Grid5000; where nodes belong to the similar site contain much smaller inter-node latency and less energy efficiency than the nodes belong to the dissimilar sites. The node allocation on the cluster was like this.

Cluster-1: Node 00–19

Cluster-2: Node 20–33

Cluster-3: Node 34–51.

In these experiments, inter-node latencies among the nodes belong to the similar substantial cluster is moderately low as compared to the inter-node latencies among the nodes belonging to the dissimilar physical group. In the experiment, Preprocessing App three times by construction of three variations in the inter-node communication instances. The average communication instances were 15, 30, and 49 for every test run that is 5 and 10 nodes. Our group discovery algorithm (GDA) formed the groups during these variations. Results of these experiments are shown in Table 1, the figure 1, 2 shows the performance comparison of experiment one and two with test node 5,10 with corresponding Execution Time GDA with latency and Execution Time GDA with latency and energy , figure 3 and figure 4 shows the performance evaluation result of Network Latency and energy efficiency. It shows that proposed energy efficieny with network latency group discovery algorithm shows less execution time, more gain ratio and less energy consumption

Table 1: Performance Evaluation on Local Clusters

Experiment	Preproc.app community Instance	Test application node request	Execution time with GDA(ms) (Latency)	Execution time with GDA(ms) (latency and energy)	%Gain ratio (latency)	Energy (KWh)
Test LC-1.1	15	5	3932	3781	4.23	1.57
Test LC-1.2	30	5	3932	3565	7.27	1.42
Test LC-1.3	39	5	3932	3545	7.89	1.24
Test LC-2.1	15	10	5611	5394	3.67	1.45
Test LC-2.2	30	10	5611	5251	5.94	1.32
Test LC-2.3	39	10	5611	5214	6.20	1.04

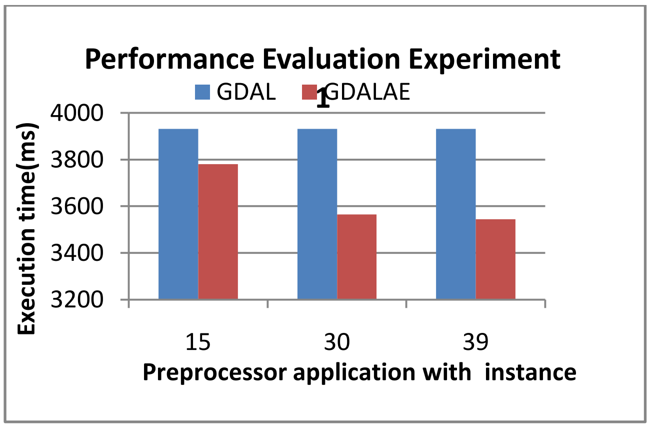

Figure 1: Performance Evaluation of Experiment 1 vs Execution Time

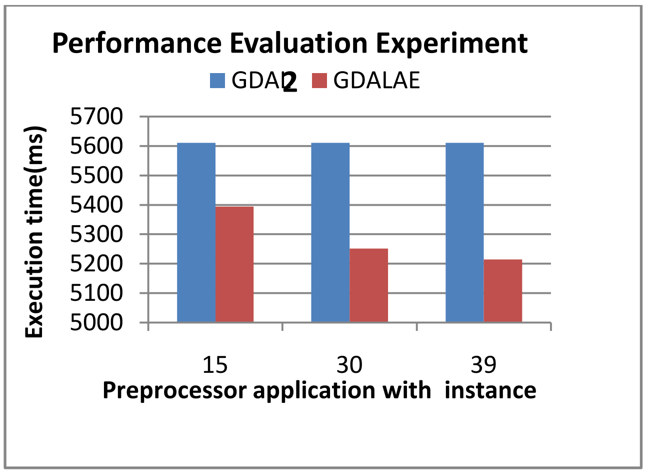

Figure 2: Performance Evaluation of Experiment 2 vs Execution Time

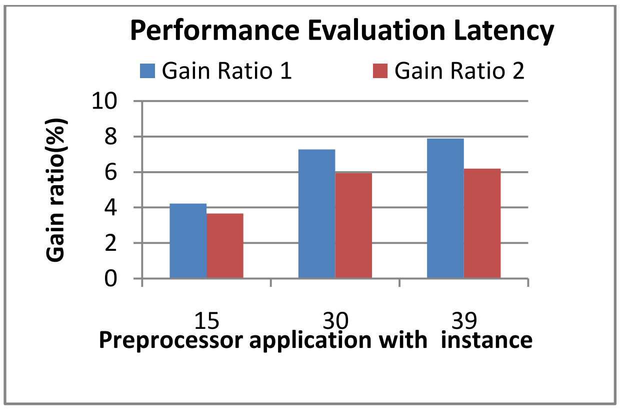

Figure 3: Performance Evaluation of Experiment vs Network Latency

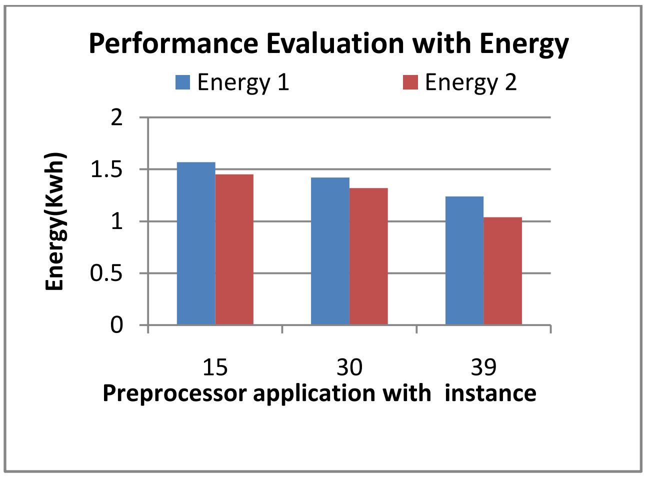

Figure 4: Performance Evaluation of Experiment vs Energy efficiency

5. CONCLUSION AND FUTURE WORK

The energy efficiency of the each and every cloud plays a major important role to improve the overall energy utilization. This paper majorly focuses on how to reduce the energy utilization of cloud servers throughout suitable scheduling schema. Taking full contemplation of the association among the result of cloud server with less network latency and less energy utilization of servers, propose a novel network latency and energy-efficient particle swarm optimizaation based multi-job scheduling representation based on the data processing structure, MapReduce, and give the equivalent algorithm. In the meantime, propose a Particle Swarm Optimization (PSO) method for those scheduling tasks and make a general energy efficiency purpose of the servers as the fitness value of the particles. It specially target the state when do not contain entire information concerning all the inter-node latencies and energy efficient information from PSOBEEMJS algorithm. The proposed system provides a possible solution for grouping of nodes with respect towards their inter-node latency and less energy efficiency.

Proposed optimization based workflow scheduling in cloud computing is not a full-grown field. Most of the obtainable mechanism efforts reduce either the make span or cost only. Though, in future work will map to believe additional objectives such as dependability, protection in adding together to the energy utilization mean to apply our algorithm in the real cloud.

6. REFERENCES

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, vol. 53, no. 6, 2009.

[2] J. Hamilton, "Cooperative expendable micro-slice servers CEMS: low cost, low power servers for internet-scale services," Citeseer.

[3] B. Karrer, E. Levina, M.E.J. Newman, Robustness of community structure in networks, Physical Review E 77 (2008) http://dx.doi.org/10.1103/PhysRevE. 77.046119.

[4] G.W. Flake, S. Lawrence, C.L. Giles, F.M. Coetzee, Self-organization and identification of web communities, IEEE Computer 35 (2007) 66–71. http://dx.doi.org/10.1109/ 2.989932.

[5] L.H. Hartwell, J.J. Hopfield, S. Leibler, A.W. Murray, From molecular to modular cell biology, Nature 402 (1999) C47–C52. http://dx.doi.org/10.1038/35011540.

[6] S. Redner, Citation statistics from more than a century, Physical Review (2004).

[7] C.T. Zahn, Graph-theoretical methods for detecting and describing gestalt cluster, IEEE Transactions on Computers C-20 (1971) 68–86. http://dx.doi.org/ 10.1109/T-C.1971.223083.

[8] S. Malik, F. Huet, D. Caromel, RACS: a framework for resource aware cloud computing, in: Proceedings of the 7th IEEE International Conference for Internet Technology and Secured Transactions, ICITST 2012, 2012.

[9] R. Andersen, F. Chung, K. Lang, Local graph partitioning using pagerank vectors, in: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, FOCS'06, 2006, pp. 475–486. http://dx.doi.org/10.1109/ FOCS.2006.44.

[10] K. Lang, S. Rao, A flow-based method for improving the expansion or conductance of graph cuts, in: D. Bienstock, G. Nemhauser (Eds.), Proceedings of the 10th International IPCO Conference on Integer Programming and Combinatorial Optimization, in: Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2004, pp. 383–400.

[11] R.J. Gil-Garcia, J.M. Badia-Contelles, A. Pons-Porrata, A general framework for agglomerative hierarchical clustering algorithms, in: Proceedings of the 18th International Conference on Pattern Recognition, ICPR 2006, 2006, pp. 569–572. http://dx.doi.org/10.1109/ICPR.2006.69.

[12] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid '10), pp. 577–578, Melbourne, Australia, May 2010.

[13] A. Berl, E. Gelenbe, M. Di Girolamo et al., "Energy-efficient cloud computing," Computer Journal, vol. 53, no. 7, pp. 1045–1051, 2010.

[14] R. Buyya, A. Beloglazov, and J. Abawajy, "Energy-Efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges," Distributed, Parallel, and Cluster Computing, http://arxiv.org/abs/1006.0308/, pp. 6–17, 2010.

[15] J. Baliga, R. W. A. Ayre, K. Hinton, and R. S. Tucker, "Green cloud computing: balancing energy in processing, storage and transport," Proceedings of the IEEE, vol. 99, no. 1, pp. 149–167, 2011.

[16] L. A. Barroso and U. H¨ olzle, "The datacenter as a computer: an introduction to the design of warehouse-scale machines," Synthesis Lectures on Computer Architecture, vol. 4, no. 1, pp. 1–108, 2009.

[17] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," in Proceedings of the Conference on Power aware computing and systems, p. 10, USENIX Association, San Diego, Calif, USA, 2008.