Volume 6, Issue 8, August 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Profit Based Swarm Intelligence Task Scheduling for Cloud Computing

Gunjan Goyal*
Research Scholar, M.tech Computer Science
KITM, Kurukshetra University, Haryana, India

Dr. Deepak Sharma
Head of Department, Department of Computer Science
KITM, Kurukshetra University, Haryana, India

Abstract: Now a days, cloud computing has been well-known as a new mode of service in the area of computer science. This paper initially starts from the definition, also the basic technologies and corresponding characteristic that are reviewed from all the aspects of cloud computing. In both literature and practice, the cloud computing face the large quantity of the cloud users or client groups and the quantity of number of tasks and the massive data, so the concern for scheduling task processing is very significant. How to schedule the tasks efficiently has now become an important issue rather a problem to be solved in the field of cloud computing. Hence for the programming framework of cloud computing, a Deadline Aware Particle Swarm optimisation(DAPSO) Algorithm can get an efficient optimised task scheduling completion time and better results, and the results of the scheduling task average make-span completion time is far more better than existing PSO(particle swarm optimisation)algorithm due to the fast convergence property of Deadline Aware Particle Swarm optimisation Algorithm. The simulation result of this algorithm is compared with other algorithms, the results analysis show that, this algorithm DAPSO is much better than the existing PSO algorithm and also it can be used as an efficient task scheduling algorithm in cloud computing environment.

Keywords: Cloud Computing, Task Scheduling Algorithm, Deadline Aware Particle Swarm optimisation, DAPSO, Optimisation, Scheduling Types, Particle swarm optimisation (PSO).

I. INTRODUCTION

Cloud computing is mainly meant for the delivery of computing services over the Internet. Cloud services allow individuals and businesses to access or use the software and hardware that are in turn managed by third parties at remote locations. Examples of cloud services includes the online file storage, the social networking sites, a web-mail, and an online business applications.[10] The cloud computing model has the open access to information and computer resources from anywhere or any part of that a network the connection is available. [1]Cloud computing provides a shared pool of the resources, including the data storage space, networks, the computer processing power, and a specialised corporate and a user applications. Cloud computing architecture has two most important components that are generally known as the front end and other is the back end. Where the Front end is that part of cloud which is visible to the user of cloud. This include applications and the computer that user look to access the cloud. Back end of cloud computing is nothing but it is the cloud itself, which mainly comprises of computers, storage devices.[2]

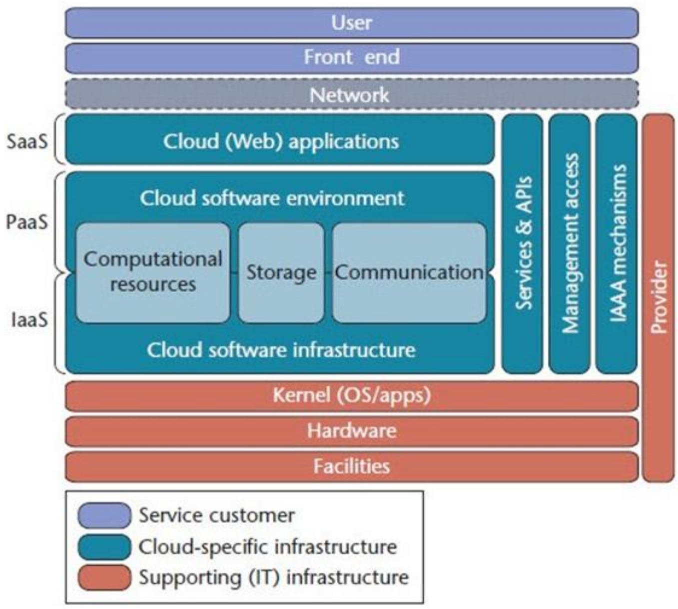

Fig.1 Layered View of Cloud Architecture

Cloud computing is an internet based computing that delivers the users an infrastructure as a service (IaaS), platform as a service (PaaS), and software as services (SaaS). The services are mostly made available on a subscription based using pay per demand user model to customers, regardless of their location. IaaS is the lower most layers of the cloud computing systems and it basically provides a virtualised resource such as the computation, communication and storage that are available on demand. PaaS makes the cloud so easily as programmable. SaaS can be called as the software delivery model. [8]

Cloud Scheduling: Traditional cloud scheduling algorithms typically aim to minimise and decrease the time and cost for processing all tasks scheduled. However, in cloud computing environment, computing capability varies from different resources and the cost of the resource usage. Therefore, it is important to take into consideration the cost. A scheduling algorithm is implemented by programmers to schedule the task with maximum estimated gain or profit and execute the task in the queue.

Need for Cloud Scheduling: In cloud computing, users may utilise hundreds of thousands of virtualised resources and it is impossible for everyone to allocate each task manually. Due to commercialisation and Virtualisation, cloud computing left the task scheduling complexity to virtual machine layer by utilising resources virtually. Hence to assign the resources to each task efficiently and effectively, Scheduling plays an important role in cloud computing. [7]

Types of Cloud Scheduling: The different types of cloud scheduling are:

- **User level scheduling:** User level scheduling comprises of market based and auction based scheduling. FIFO scheduling, priority based, non pre-emptive scheduling etc. are used in user level scheduling.
- **Cloud Service Scheduling:** Cloud service scheduling is classified at user level and system level. At user level, it mainly considers the service regarding problems between the provider and the user. At system level, scheduling and resource management is done. In addition to real time satisfaction, fault tolerance, reliability, resource sharing and QoS parameters are also taken under consideration.[12]
- **Static and Dynamic Scheduling:** Static scheduling permits pre-fetching of required data and pipelining of distant stages of task execution. Static scheduling imposes minimum runtime overhead. In case of dynamic scheduling, information of the job components or task is not known before. Thus the execution time of the task may not be known and the allocation of tasks is done only as the application executes.
- **Heuristics Scheduling:** In cloud environment, heuristic based scheduling can be done to optimize results. More accurate results can be built by heuristic methods.[4]
- **Workflow Scheduling:** For the management of workflow execution, workflow scheduling is done.
- **Real Time Scheduling:** Real Time Scheduling in cloud environment is done to increase the throughput and to decrease the average response time instead of meeting deadline.

Performance Parameters of Cloud: Various cloud scheduling parameters are discussed below:

- **Makespan:** (completion time of a schedule): The time variation between the start and finish of a sequence of schedule.
- **Resource Cost:** Resource cost in a cloud can be determined by resource capacity and occupied time. Powerful resources led to higher cost.
- **Scalability:** Capability to cope and perform with increased workload and has ability to add resources effectively.
- **Reliability:** The ability of a system to continue work even in case of failure.
- **Resource Utilisation:** This parameter defines how effectively the system utilises the resources.

II. RELATED WORK

BU Yanping et al. [8] provided an improved particle swarm optimisation (PSO) algorithm kept all the advantages of the standard PSO, such as implementation simplicity, low computational burden, and few control parameters, etc. He also tested the improved PSO algorithm against the MaxMin heuristic and found that improved PSO outperforms MaxMin by the total make-span and other performance. The function of DPSO was to find the best tasks scheduling strategy and to obtain the optimal make-span. Nuttapong Netjinda e t al. [1 2] proposed a new framework where number of purchased instance, instance type, purchasing options, and task scheduling were considered within an optimisation process. In order to identify a solution in a reasonable amount of time, they studied the use of Particle Swarm Optimisation (PSO) technique. Another benefit was that the number of alternatives for task assignment could be changed as the number of purchased instances had changed. Xingquan Zuo et al. [1 4] e stablished an integer programming model for the resources allocation problem of an IaaS cloud in a hybrid cloud environment. A self adaptive learning PSO (SLPSO)based scheduling approach for this problem was proposed. In SLPSO, each dimension of a particle represents a task and a particle as a whole represents all tasks' priorities. This approach was able to obtain a high quality scheduling solution by adaptively selecting velocity updating strategies to update each particle. Himani et al. [1 5] represented a soft real real time scheduling approach with deadline and cost constraints. It had been planned to take the concrete problems related to cloud computing. They determined that earlier approaches are not able to meet the deadline efficiently. They illustrate that a deadline-meeting methodology to schedule tasks over a cloud allows to reduce the number of missed deadline. They compare the CDB TaskScheduling Algorithm with space shared policy and shows that the given approaches was more effective in defined parameters as Task Profit, Task Penalty, ThroughPut, Provider profit and in User Loss. Azadi Khalili et al. [1 5] proposed a hybrid Particle Swarm Optimisation (PSO) for scheduling in cloud. The hybrid PSO performs better compared to Max Min Scheduling and Minimum Execution time in case of schedule length and execution ratio in this study. Average schedule length of the hybrid PSO improved in a range of 1.1 to 5.5% than

Max Min Scheduling and in a range of 1.1 to 3.3% than Minimum Execution Time. Dr. M.Sridhar et al. [1 5] proposed a hybrid Particle Swarm Optimisation (PSO) for scheduling in cloud. The hybrid PSO that an evaluation and comparative study of these approaches had been performed. Firstly the best values of parameters for each algorithm, experimentally determined. Then the algorithms in applications with the number of tasks varying from 100 to 1000 evaluated. Simulation results demonstrate that ABC, PSO and ACO algorithms achieves better resource utilisation and significantly outperforms FPLTF, random and FCFS algorithms on the basis of make-span and degree of imbalance. The experimental results proved that ABC algorithm is the superior and outperforms other algorithms. The PSO and ACO could be putted in second level and third level respectively. ChienHung Chen et al. [1 5] Investigated the DeadlineConstrained MapReduce Scheduling (DCMRS) problem in heterogeneous cloud computing systems. Considering the slots in different nodes had different amount of computing resources, we first divide a job deadline into two sub deadlines: map and reduce deadlines. The sub-deadlines were used for finding appropriate slots to run the tasks of the job. Then, they transformed the DCMRS problem to a well-known Minimum Weighted Bipartite Matching (MWBM) problem. To solved this problem, they formulated an ILP model for obtaining the optimal solution. They also presented a heuristic algorithm involving the node group technique to decrease the computational time. Lizheng Guo et al. [1 2] p resented the task scheduling optimising method in cloud computing, and they formulated a model for task scheduling to minimise the cost of the problem. By comparing and analysing particle swarm algorithm with crossover, mutation and local search algorithm based on particle swarm, they proposed the particle swarm algorithm embed in SPV, which represents better performance.

III. PROPOSED APPROACH

Task scheduling in cloud computing is implemented by using the Deadline aware particle swarm optimisation Algorithm which defines itself by an important factor of fast convergence followed by many more which made this algorithm more faster than the rest of the earlier used algorithms by decreasing the makespan factor after working upon the number of tasks and a virtual machine.

Deadline-Aware Algorithm: The algorithm is based on two parameters which are profit and deadline, then define both the parameter. The profit is calculated according to Tasks and virtual machines, same as deadline. Then sorting mechanism is used for deciding the order of tasks and tasks mapping with virtual machine is given in PSO algorithm. Therefore, proposed approach is to maximise profit, achieve deadline, minimise makespan , minimise loss. The name given to this approach is **"Deadline Aware Particle Swarm Optimisation Algorithm(DAPSO)"**.

```
Begin
        Step 1. Define Deadline Di and million instructions mi for each Tasks Ti.
        Step 2. Calculate MaxUserPay MUPi for each instruction Ti by using prediction model.
                        MUPi = k * mi/Di
        Step 3. Put all accepted Tasks Ti in queue Q.
        Step 4. Sort the queue according to the earliest deadline first.
        Step 5. If two tasks Ti having same deadline , Then Sort according to highest MaxUserPay MUPi.
        Step 6. Mapped the tasks with virtual machine using Particle swarm intelligence approach(PSO).
        Step 7. Schedule the task according to Space-shared scheduling policy.
        Step 8.  Evaluate the Net Profit Pi of tasks Ti using estimation model
                which depend upon Lateness L given below :-
                        if L< 0     Pi = MUPi – CostPerSec * ActualCPUTime
                        else        Pi = MUPi *(1-0.01*L) -  CostPerSec * ActualCPUTime
                        return Pi;
End
```

Fig.2 Pseudo-code of Deadline-Aware Algorithm

Particle Swarm Optimisation: A modern heuristic optimization techniques such as simulated annealing, evolutionary algorithms, neural networks, and ant colony have been given much attended by many researchers due to their ability to find an almost global optimal solution in EDPs(Balakrishnan et al.,2003; Coelho & Mariani, 2006; Song & Chou, 1999; Yalcinoz & Altun,2001). One of these modern heuristic optimization paradigms is the particle swarm optimization(PSO) (Eberhart & Kennedy,1995; Kothari & Dhillon, 2011; Singh et al.,2013).

PSO is a kind of evolutionary algorithm based on a population of individuals and motivated by the simulation of social behaviour instead of survival of the fittest individual. Similar to the other population-based evolutionary algorithms, PSO is initialised with a population of random solutions. Unlike the most of the evolutionary algorithm solution (individual) in PSO is associated with a randomized velocity, and the potential solutions, called particles, are then "flown" through the problem space.

As stated before, PSO simulates the behaviors of bird flocking. Suppose the following scenario: a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. So what's the best strategy to find the food? The effective one is to follow the bird which is nearest to the food.

PSO learned from the scenario and used it to solve the optimization problems. In PSO, each single solution is a "bird" in the search space. We call it "particle". All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles.

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest.

After finding the two best values, the particle updates its velocity and positions with following equation (a) and (b).

v[] = v[] + c1 * rand() * (pbest[] - present[]) + c2 * rand() * (gbest[] - present[]) (a)

present[] = present[] + v[] (b)

v[] is the particle velocity, present[] is the current particle (solution). pbest[] and gbest[] are defined as stated before. rand () is a random number between (0,1). c1, c2 are learning factors. usually c1 = c2 = 2

The pseudo code of the procedure is as follows

```
For each particle
        Initialize particle
END
Do
For each particle
    Calculate fitness value
        If the fitness value is better than the best fitness value (pBest) in history
            set current value as the new pBest
    End
    Choose the particle with the best fitness value of all the particles as the gBest
For each particle
     Calculate particle velocity according equation (a)
    Update particle position according equation (b)
End
While maximum iterations or minimum error criteria is not attained
```

Particles' velocities on each dimension are clamped to a maximum velocity Vmax. If the sum of accelerations would cause the velocity on that dimension to exceed Vmax, which is a parameter specified by the user. Then the velocity on that dimension is limited to Vmax.

IV. EXPERIMENTAL SETUP

The performance measurement has been gathered by using the CloudSim[R.N. Calheiros,2011] framework. The goal of these experiment is to show the ability of DAPSO task scheduling approach to meet deadline in conditions where the other algorithms could not able to meet deadlines and get better profit and MakeSpan. To reach the better results , we are considering the following situation. We are taking virtual machines having processing speed in range of {1000,2000,3000} MIPS. An available bandwidth of 100 Gbits/s. A Random Access Memory (RAM) of 128GB.

We process 1000 to 10000 million instructions and each cloudlet is given a randomly generated deadline. We have 100 tasks and 10 VMs to calculate and visualise all parameters i.e Existing MakeSpan, Profit, Succeed task, Failure task and loss. The following Parameters are described below with the result:

Table 1: Showing Experiment result relevant to the parameters

Algorithm	**MakeSpan**	**TaskProfit**	**TaskPenalty**	**ProviderProfit**	**UserLoss**
PSO-SCH	82.21	43	57	11538.23	408374.39
DAPSO-SCH	73.21	40	60	10922.66	190107.22

4.1 MakeSpan: It is defined as the amount of time, from start to finish for completing a set of jobs, i.e the maximum completion time of all jobs. In this Experiment we want to minimise the MakeSpan , So that the tasks execution of virtual machines will take less time. DAPSO task scheduling approach is more efficient to finish tasks before the deadline arrives. We compare number of tasks completed by each approach for the given set of instructions In Fig 3. We can see the results of experiment and give the comparison among the PSO scheduling and DAPSO scheduling.

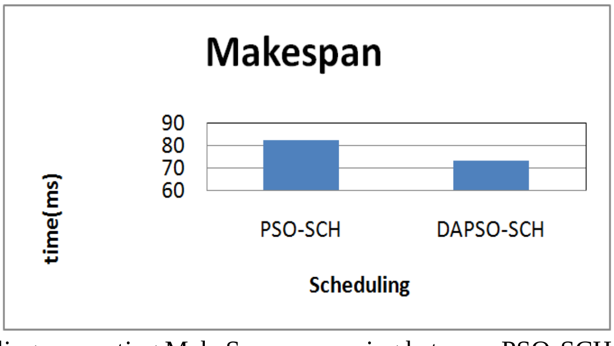

Fig 3. Task scheduling computing MakeSpan comparing between PSO-SCH and DAPSO-SCH

4.2 Task Profit: It is defined as the number of tasks which have completed successfully before they meet the deadline i.e number or tasks which meet the deadline before it decided. DAPSO task scheduling approach is more efficient to finish tasks before the deadline arrives. We compare number of tasks completed by each approach for the given set of instructions In Fig 4. We can see the results of experiment and give the comparison among the PSO scheduling and DAPSO scheduling

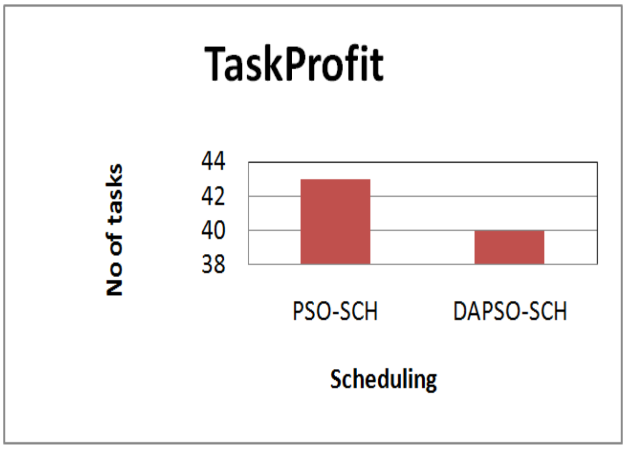

Fig 4. Task scheduling computing Task Profit comparing betweenPSO-SCH and DAPSO-SCH

4.3 Task Penalty: It is defined as number of tasks misses their deadline i.e. Number of tasks which are not completed successfully within the deadline. We analyze result of another parameter in Figure 5. In this parameter, we annoyed to find task penalty of tasks. Once again, It is notice DAPSO outperforms than existing approach. DAPSO approach missed negligible number of deadline as compared to PSO policy.

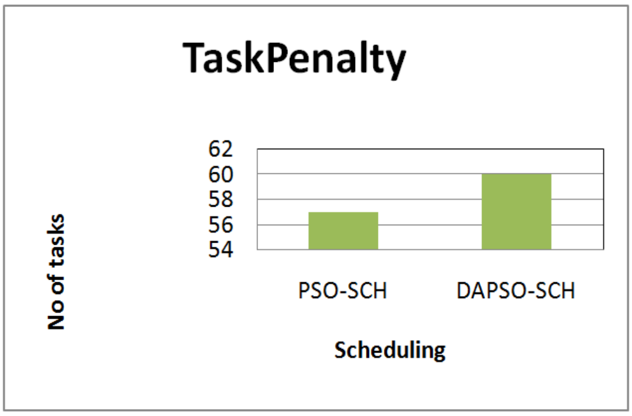

Fig 5. Task scheduling computing Task Penalty comparing between PSO-SCH and DAPSO-SCH

4.4 Provider Profit: It is depend on lateness, if lateness of tasks is less than zero then profit is calculated. DAPSO is capable to process soft deadline tasks. Such tasks has penalties when they misses their deadline. Figure 6. Results indicates that profit in our approach is maximum as compared to other policy.

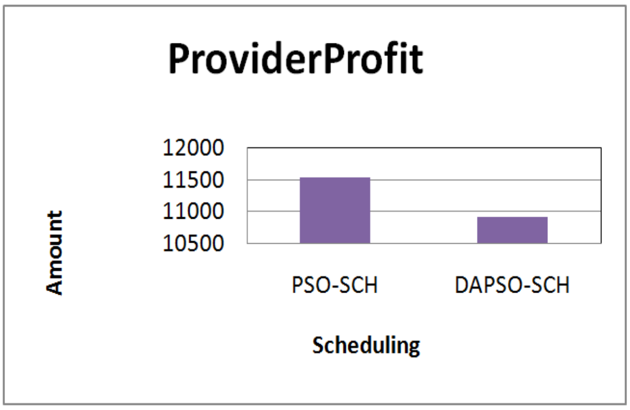

Fig 6. Task scheduling computing Provider Profit comparing betweenPSO-SCH and DAPSO-SCH

4.5 User Loss: It is also depend upon lateness of the task in which lateness greater than zero provide user loss. Figure 7. Results indicate that user loss in our approach is minimum as compared to other policy.

© 2016, IJARCSSE All Rights Reserved

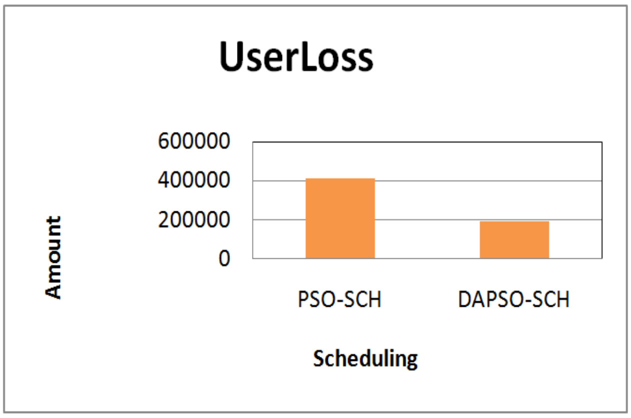

Fig 6. Task scheduling computing User Loss comparing betweenPSO-SCH and DAPSO-SCH

V. CONCLUSION

This work represents a soft real-time scheduling approach with deadline and profit constraints. It has been planned to take the concrete problems related to cloud computing. We determine that earlier approaches are not able to meet the deadline efficiently. However, we illustrate that a deadline-meeting methodology to schedule tasks over a cloud allows to reduce the number of missed deadline. We simulate our results by using cloud-sim framework. We compare the DAPSO task scheduling algorithm with PSO algorithm and shows that the given approach is more effective in defined parameters as task profit, task penalty, existing makespan, user loss and provider profit. The research includes cost optimization by a prediction model to estimate the initial value of MaxUserPay(initial cost of task) based on the length of the task, number of processing elements required and deadline. The prediction model is valuable for enhancing the profit and saving the money for users as well.

VI. FUTURE SCOPE

The future scope of proposed work includes that hybrid algorithms will be used for enhancing the optimisation of profit and deadline. The quality of service factors can be more for future enhancement and new approaches will be proposed for the same. The performance of given DAPSO algorithm can also also evaluated comparing with another algorithm.

RFERENCES

[1] BU Yanping1,2 ZHOU Wei3 YU Jinshou1 1.Research Institute of Automation,East China University of Science and Technology,Shanghai 200237 China; 2. School of Technology, Shanghai Jiaotong University, Shanghai 201101 China; 3.School of Business, East China University of Science and Technology, Shanghai 2002 37 China , 2008 International Symposium on Computer Science and Computational Technology .

[2] Nuttapong Netjinda, Booncharoen Sirinaovakul, Tiranee Achalakul Department of Computer Engineering King Mongkut's University of Technology Thonburi Bangkok, Thailand,Thailand Research Fund through the Royal Golden Jubilee Ph.D. Program (grant no. PHD/0031/2553).

[3] Xingquan Zuo, Member, IEEE, Guoxiang Zhang, and Wei Tan, Member, IEEE ,IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, VOL. 11, NO. 2, APRIL 2014.

[4] A zadi Khalili and S eyed Morteza, School of Electrical and Computer Engineering Kashan University , Babamir , 2015 23r d I ranian Conference on Electrical Engineering (ICEE).

[5] Dr. M.Sridhar and Dr. G. Rama Mohan Babu , R.V.R & J.C College of Engineering, Guntur, INDIA, 2015 IEEE International Advance Computing Conference (IACC) .

[6] ChienHung Chen, JennWei Lin, and SyYen Kuo, Fellow, I EEE

[7] Himani , Global Institute of management & emerging technology, Amritsar, India and Harmanbir Singh Sidhu , Chandigarh Group Of Colleges, India, 2015 Second International Conference on Advances in Computing and Communication Engineering.

[8] Lizheng Guo , Departm,ent of Computer Science and Engineering, Henan University of Urban Construction, Pingdingshan 467633, China, JOURNAL OF NETWORKS, VOL. 7, NO. 3, MARCH 2012

© 2016, IJARCSSE All Rights Reserved