DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4337

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

# MULTI CORE BASED REAL TIME RESOUCE ALLOCATION AND SCHEDULING IN CLOUD

Shantashri Mallikarjun Pujari
Dept of studies in computer application (MCA)
VTU centre for PG studies, Kalaburagi
Kalaburagi, Karnataka, India

Bharati S Pochal
Dept of studies in computer application (MCA)
VTU centre for PG studies, Kalaburagi
Kalaburagi, Karnataka, India

***Abstract:*** As the smart cities emerged for more comfortable urban spaces, services, such health, Transportation, and so on, need to be promoted. In addition, the cloud computing provides exible allocation, migration of services, and better security isolation; therefore, it is the infrastructure for the smart cities. Single instruction set architecture ISA heterogeneous multi-core processors have higher performance and are popular in current processors, To increase the effectiveness of these systems is to accelerate sequential CPU bound threads using fast cores, and to improve the throughput of parallel memory-bound threads using slow cores.

***Keywords:*** Dynamic resources partitioning; heterogeneous multi-core processors; memory partitioning, memory scheduling; Real Time; resource allocation; scheduling in cloud.

## I. INTRODUCTION

Cloud computing is a technology which has grown and has been used in many fields which also initiates the developments of the smart cities ,a single ISA heterogeneous multicourse consists of the same architecture but explore different features . The HMP are generally known for their high speed, high power, small and low cores, and scalability. The system which consists of the threads which are running concurrently consists of both SMP and HMP .which considering the HMP systems it is quite hard to increase the speed of resource sharing among threads. So it is important to manage the shared resources properly. The effectiveness of the HMP system can be evaluated by three objectives: power efficiency, system performance and fairness. The most important one is the power efficiency, which is the most advantage of the HMP. The second is the overall system throughput, which should remain high. And the last is fair scheduling, which is no single thread, especially the thread running on fast core, should be incorrectly slowed down.

As the smart cities grow day by day it has also become very important to manage the requirements of the resources of the system to fulfill the user's requirements. Along with maintaining the resources providing the securing and the balancing the resource request efficiently is also an important task in the smart cities. Balancing the request and providing the efficient throughput of the system can be done by using the scheduling algorithms .In the current cloud computing environment where the cloud shows the features such as dynamic, non transparent, isolated. The clouds are maintained using few out of order cores, small cores and all these cores consist of the same ISA. We can crease then efficiency of the system by increasing the CPU bound threads with the help of the fast cores. We have proposed a dynamic resources allocation for the heterogeneous cores where the multicores have different architecture.

In the current system where a large number of users are increase in the cloud this has become a challenging task to fulfill the users request along with providing security in the system .While it is also important to provide quality of service and efficient scheduling technique to provide fairness while processing the users' request.

## II. REALATED WORK

[1] The problem while dealing with the thread scheduling is solved by the asymmetric multicourse processors (AMPs). Whenever we perform the operation in cloud, the system creates the thread for the process. The need of one thread is different from another thread during the execution; therefore reassigning the thread to the core after analyzing the needs of the thread improves the efficiency of the AMP's power. Identifying the best thread in the AMP is done in the traditional online learning schemes .The calculation of performance is done by comparing the thread in the other core type where as thread scheduling can be done by avoiding the difficulties related to the sampling. [2] The main motivation this paper is recent methods which are used in different ways. Firstly, chip multicourse processor (CMPs).Secondly, multithreaded software and lastly by increasing the level of on-chip integration. The first methods are very common which has a increasing number of CPU cores. Second method will take the help of CMPs which will soon become common as the multithread processes have a sequential execution. The last method has a problem of reducing power consumption and thus making it as a challenge. The main of this paper is to minimize the execution time of the threads in the core which consists of sequential and parallel executing threads without reducing the CMP's power consumption. In [3] author proposed a method where mechanism to decrease the processor power dissipation by avoiding the single ISA heterogeneous multicourse architecture. The proposed system contains the heterogeneous core components which show the different needs and performance in the results. During the execution of the system it automatically chooses the appropriate core to meet the specification required for the execution of the operation and the power consumption. The experimental results of this architecture show the efficient power consumption. [4] the memory scheduling is also an

important task while developing or allocating the resources. Because of the memory scheduling method can avoid the variance in the memory allocation occurring during the execution of the threads which are executing concurrently and also it is necessary to allocate the memory faster so that the throughput of the system should be high. Previously proposed algorithm provides only one of the features mentioned above. So, here we propose a new memory scheduling algorithm which provides both efficient scheduling as well as faster memory allocation resulting into high throughput. Here in the proposed system we divide the threads into clusters and apply the different memory scheduling schemes to each cluster. [5] the challenge for the computer designers is the difference between the microprocessor speed and the DRAM speed. The dram speed decides the efficiency of the system. So it is necessary to increase the DRAM speed as well as the throughput of the system. The memory of the dram is shared among cores of the system which may cause problem such as memory contention and interference problem which will gradually decrease the efficiency of the system and unfair resources allocation to the threads which are executing inside the system. This proposed system also contains the memory optimization framework along with the thread scheduling and the memory scheduling. these all techniques helps the proposed system to increase the efficiency of the system by providing the fairness among the resource allocation among the threads, decreases the memory allocation time and reduce the interference among the multi-cores. [6] Interference among a application which are running concurrently and the DRAM memory will reduce the system performance. The memory scheduling schemes keeps the details of the request sent for the memory allocation. This will improve the throughput of the system but it will not avoid the problem of interference between the multi-cores. Memory partitioning method is used to reduce the interference between the multi-cores. The memory partitioning method divides the memory among the threads through which the interaction among the threads for the memory resource will be reduced. Memory channel will analyze the threads which requires the interaction with each other to different channels , by doing this will increase the unfairness of the scheduling and it will also physically exacerbates memory contention. Memory bank partitioning will divide the memory among the cores and avoids the interference between the threads but however the memory bank are limited in size.. in this paper we have proposed the dynamic memory bank partitioning ,this technique will divide the memory dynamically based on the requirement of the thread .This technique emphases the bank level parallelism. The main of the goal of the proposed system is to estimate the resource requirement of the thread during runtime and dynamically allocate the required resources to the thread. This will gradually decrease the interference among the threads and increases the system performance. [7] the usage of the DRAM and its power efficiency has becoming important now a days. The Memory bank partitioning method will partition the memory among the multi-cores by which the interference among the threads will decreased resulting into the increase in system performance. But the memory bank partitioning method does not consider the power consumption or the efficiency of the DRAM. The main objective of the proposed system is that to dynamically assign the page policies according to their requirement and memory characteristics .which deals with both interference among the threads and the power consumption of the DRAM. This experimental result shows the proposed system will show the better performance. [8] coordinate Channel-Aware Page Mapping Policy and Memory Scheduling for Reducing Memory Multi-core system generally share the memory among different programs which are running concurrently so, there will lot of variance and interference among the different programs which are running in the system. These kinds of problems will gradually decrease the system performance by low thread performance in this paper we will propose a method of coordinating channel-aware page mapping policy and memory scheduling (CCPS) to reduce inter multimedia application interference in a memory system. The idea is to map the data of different threads to different channels, together with memory scheduling.

## III. SYSTEM DESIGN

### A. *Proposed System*

- We propose a dynamic resource partitioning to maximize effectiveness of the single-ISA heterogeneous multi-core system. DRP dynamically partitions shared resources, containing shared cache, memory banks and memory bandwidth, according to core/thread pair's resource requirement.
- We combine DRP with memory scheduling policies to improve power efficiency, throughput and fairness simultaneously, because resource partitioning is orthogonal to memory scheduling. This system focus on increasing the system performance by using threads to access the memory. Since threads are the lightweight process. It will easy for the system to switch among multiple memory threads
- We estimate the core/thread's resources requirement according to the characteristic of both the core and the thread running on it, not just the thread's characteristic. It is important to character the core's performance in the HMP for maximizing its effectiveness.

***Advantages of Proposed System***

- Makes the resource allocation faster.
- Makes the resource allocation easier.
- Provides quality of service to the user.

### B. *Objectives of the Study*

- The main objective of the proposed system is to create the dynamic resource allocation system for the heterogeneous system
- The main focus of the system is to dynamically analysis the resource and the requirement of the thread and to perform the task by allocating the required resources
- The system also uses the scheduling algorithm to increase the throughput of the system.
- Then, we use the estimation to direct our resource partitioning.

### C. *Methodology Used*

In the earlier System all the resources will be allocated from one system. In the proposed system each cloud will be allocated with separate system. The allocated system will be having a fixed number of resources.

let the cloud be named as c1,c2,c3 with allocated system s1,s2,s3 respectively .let the u1, u2 and u3 be the users of c1,u4,u5,u6 and u7 be the users of c2,and u8,u9,u10 and u11 be the users of c3.

Step 1: if a u6 requests of some resources the system will

allocate the required resources to the u6 from the s1 if it is available.

Step2: if at the same time u4 request for some resources. The s2 will allocate the required resources to the u4 from s2.

Step 3: if the u8 from c3 requests for the file from the c2. Then he requests the file to the s2 then the s2 checks For the available resources if the resources are sufficiently not available. First it will allocate the available resources after the completion of the u4's work it will allocate the resources to the u8.

Step 4: finally the u8 will perform its operation and gets the requested file.

### *D. System Architecture*

To maximize the effectiveness of the HMP system, we propose Dynamic Resource Partitioning (DRP).It contains four parts of the DRP. First, to estimate threads' needs for shared resources, shared caches, memory banks and memory bandwidth, thread's characteristic paroling pro let threads' three components, memory intensity, row-buffer locality and bank level parallelism. Second, to estimate cores' needs for shared resources in HMP, core's characteristic paroling is to cores' three components, CPU frequency, cache size, and out-of-order window size. Through the six components, we can estimate the. Shared resources requirements for every thread running on different core .Third, to take advantages of heterogeneous performance cores, based on the estimation of both thread and core characteristic, dynamic partition shared resource to prevent interfering to realize accelerating sequential CPU both reads using fast cores, and improving throughput of parallel memory-bound threads using slow cores. Final, to improve fairness, we combine memory scheduling police into our DRP, which can improve through put and fairness simultaneously to maximize effectiveness of HMP. Figure 1 demonstrates our DRP framework.

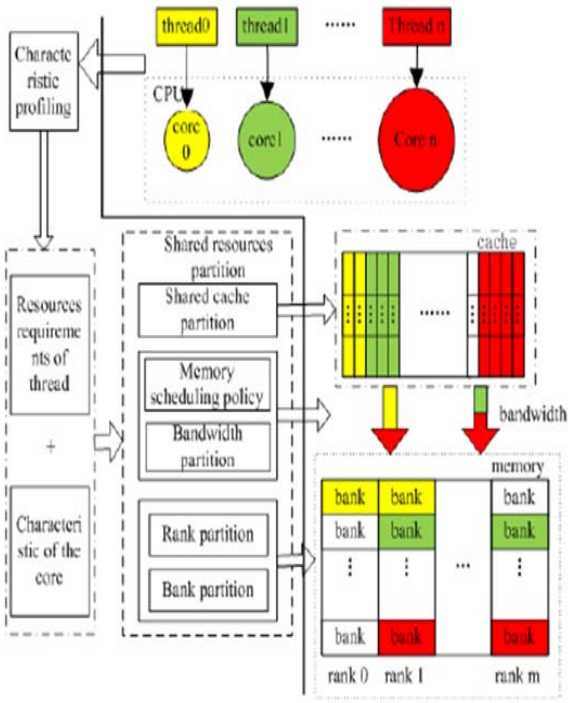


Figure.1: The framework of DRP.

### *E. Sequence Diagram*

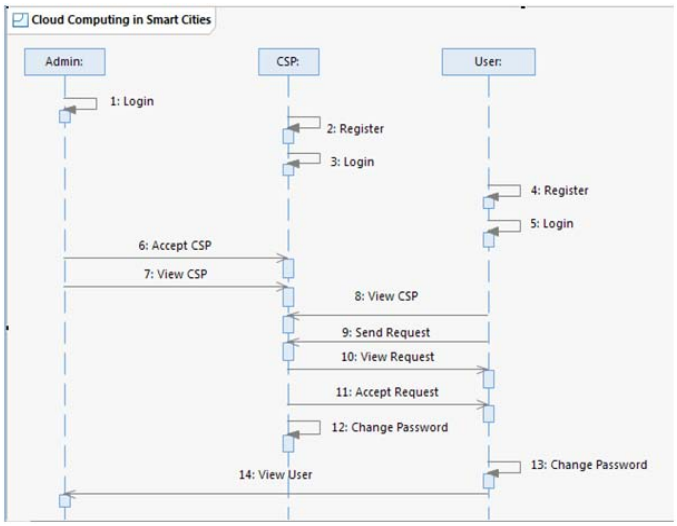


Figure. 2: sequence diagram

Figure 2 shows how processes operate with each other and in which organize they operate the sequence diagram. The structure or the blueprint of Chart, The object interactions arranged in time sequence is shown.

The life span of the actor or the entities are shown using the straight line and the horizontal line shows the interaction between the entities and also the direction the data flows. This diagram generally shows the flow of data during runtime.

## IV. IMPLEMENTATION

### *A. Module Description*

Cloud Partitioning: the cloud partitioning decides the place where the users request has to be sent in order to fulfill the request for the resources This will send the users request to the appropriate cloud where the resource is available.

*Scheduling*

This module provides the way for the system to schedule the request of the users in the cloud. The scheduling technique increase the systems quality of service and the user's request an be processed fairly.

*User Module*

The user is the one who will be using the facilities of the system. The users is allowed to upload their data to the system, he allowed viewing the details of the file uploaded in the system, allowed to place the request to the required file.

*Dynamic Server*

Migrating Server is responsible to move the request from one place to another. He allowed to view the request for file and allowed to view approve the request of the file.

### *B. Algorithm*

Our Memory Bank Partition Algorithm

**De_nition:**

N: the total cores in the processor

M: the total memory banks in the server

NL: the number of low memory intensive threads

NHL: the number of high memory intensive with low row buffer locality threads

NHH: the number of high memory intensive with high row

buffer locality threads
N: the total cores in the cloud computing
**Our memory bank partition:**
If NL==N
Allocate M/N banks for every thread;
Else if NL==0
Allocate M/(NHLCNHH) banks for every threads in high memory intensive with high row buffer locality type;
If M/(NHLCNHH) <16
Each two threads in high memory intensive with low row buffer locality type sharing M/(NHLCNHH)*2 banks;
Else
Allocate M/(NHLCNHH) banks for every thread in high memory intensive with low row buffer locality type;
Else
Allocate M/(NHLCNHH) banks for every thread in high memory intensive with high row buffer locality type;
Threads in low memory intensive type can access all banks allocated for high memory intensive with high row buffer locality threads;
If M/(NHLCNHH)<16
Each two threads in high memory intensive with low row buffer locality type sharing M/(NHLCNHH)*2 banks;
Else
Allocate M/(NHLCNHH) banks for every thread in high memory intensive with low row buffer locality type;

## V. PERFORMANCE ANALYSIS

### A. *Tooles and Technology*

Ns2 is a diver network simulator which allows us creates a dynamic structure of a system. The network simulator helps the users to understand the system more deeply. The simulation can be implemented in different filed such wired and wireless networks even while demonstrating the network topologies and the protocols and algorithm. Ns2 has been considered as one of the strongest and versatile simulator across the period of time. This is generally used to get the knowledge of the system's behavior. In this project we using in our project, the operating system is window xp and technology we are using ns2 in our work.

***Simulation workflow***

- ***Topology*:** To make the development easy while creating the basic properties and the relationships among the network ,ns-2 contains containers and helpers
- ***Model development*:** For the development of the system models have been added to ns architecture.
- ***Node and link configuration:*** During the configuration of the system models are set to the default values
- **'Execution:** The simulator takes events as an input and they perform the operation one by one.
- **Performance:** The results are shown in the form of time-stamped event trace.
- ***Graphical Visualization:*** *It* can show the graph based on the raw or the processed data.

B. ***Evaluation Metrics****:* In ours project we measure system throughput using weighted speedup and fairness using maximum slowdown. *weighted speedup* and fairness using *maximum slowdown*.

$$weighted\_speedup = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}}$$

$$maximum\_slowdown = max_i \frac{IPC_i^{alone}}{IPC_i^{shared}}.$$

### B. *Results and Discusstion*

In our project work we discussing result and description .in ours works how the cloud is partitioning the multi resources to multiple users. in our project work, we are showing the how all work is going on.

In the description in first diagram all nodes are grouping by the help of cloud next it will group around the server .like that many servers are grouped around many user or nodes. After that the rely on a node will help transfer any type of data from source to destination.

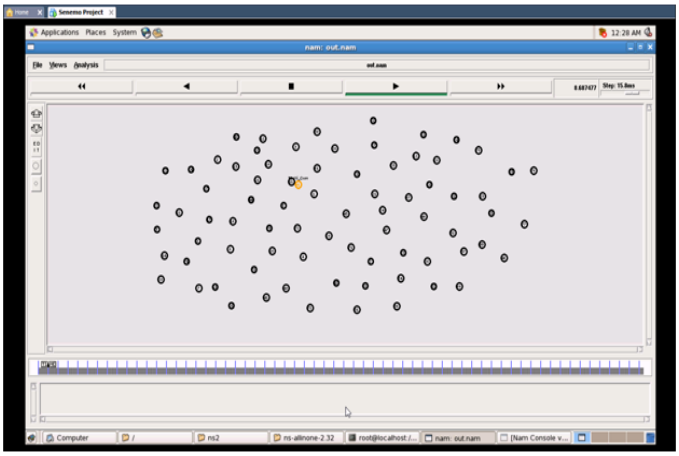

Figure. 3: All nodes are grouping by the help of cloud Partitioning

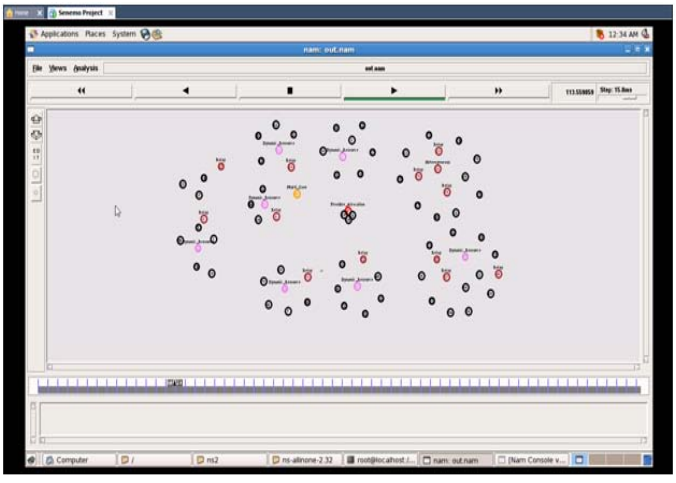

Figure. 4: After grouping the multiple server are in pink node in one server many users are available, the relays are help to transmits the data from one server to another, red color nod it is flexibly allocation in this all relay will came together and in here it will decide in where and which resources will allocated and it will send a any format of data to user from source to destination

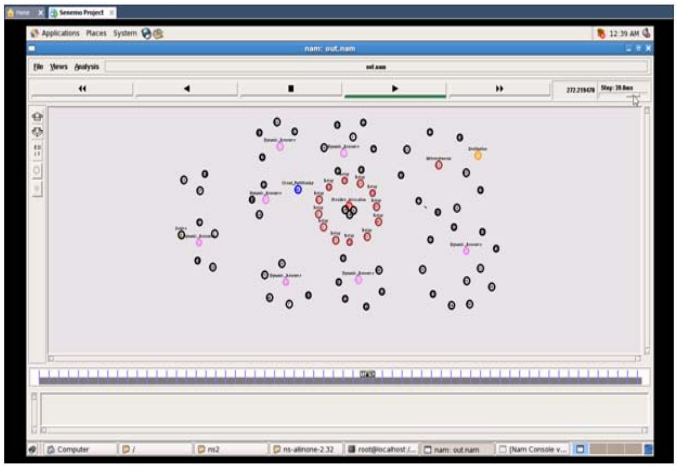

Figure . 5 : All data collected now it deciding resource allocation

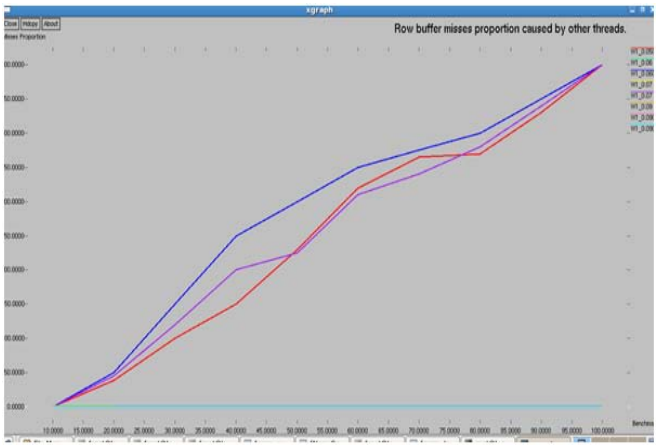


Figure. 6: Bench Mark VS Misses proportion caused by other thread


Figure. 7: Work load VS normalized power efficiency

## VI. CONCLUSION

In future, the cloud computing will be the infrastructure of the smart cities to cater more and better services while cutting down the budget. So the cloud computing adopts the better single-ISA heterogeneous multi-core. However, it needs properly management to leverage the effectiveness of the HMP. In this paper, we propose a dynamic resource partitioning (DRP) for single-ISA heterogeneous.Multicoure, which partitions shared resources according to both threads' requirements for shared resources and the performance of their running cores In the next version of the system we will show the graph of the resources used by the clients and we can even increase the capacity of the resources. To develop a system this allows the user to select the required resources out of many and provides the trust among the cloud and the users. We can also implement the SDN which controls the networking very efficiently. We can also provide a service allows the users to use the resources based on the cost they have paid.

.

## VII. REFERENCES

[1] R. Rodrigues, A. Annamalai, I. Koren, and S. Kudu, ``Scalable thread scheduling in asymmetric multicourse for power efficiency," in Proc. IEEE24th Int. Sump. Compute. Archit. High Perform Compute, Oct. 2012, pp. 59_66

[2] M. Annavaram, E. Grochowski, and J. Shen, ``Mitigating Amdahl's law through EPI throttling," in *Proc. ISCA*, 2005, pp. 298_309.

[3] R. Kumar, K. I. Freaks, N. P. Juppe, P. Ranganathan, and D. M. Tulsan, ``Single-ISA heterogeneous multi-core architectures: The potential for processor power reduction," in *Proc. MICRO*, vol. 36. 2003,p.81.K. Elissa, "Title of paper if known," unpublished.

[4] Y. Kim, M. Papamichael, O. Mutlu, and M. HarcholBalter, ``Thread cluster memory scheduling: Exploriting differences in memory access behavior,"in Proc.MICRO,2010,pp-65_76.

[5] G Jai, G. Han, L. Shi, J. Wan, and D. Dai, ``Combine thread with memory scheduling for maximizing performance in multi-core systems," in *Proc. ICPADS*, 2014, pp. 298_305.

[6] M. Xie, D. Tong, K. Huang, and X. Cheng, ``Improving system throughput and fairness simultaneously in shared memory CMP systems via dynamic bank partitioning," in *Proc. HPCA*, 2014, pp. 344_355.

[7] M. Xie, D. Tong, Y. Feng, K. Huang, and X. Cheng, ``Page policy control with memory partitioning for DRAM performance and power ef_ciency," in *Proc. ISLPED*, 2013, pp. 298_303.

[8] G. Jia, G. Han, A. Li, and J. Lloret, ``Coordinate channel-aware page mapping policy and memory scheduling for reducing memory interference among multimedia applications," *IEEE Syst. J.*, 2015, doi: 10.1109/JSYST.2015.2430522.