www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 5 may, 2014 Page No. 5897-5900

A Survey on: Pre-Emptive Migration of a video process using Genetic Algorithm on Virtual machine

Ranjitha K N[1], Sandhya S[2], N K Cauvery[3]

[1] Asst. Professor, Dept. of Information Science and Engg,
Global Academy of Technology, Bangalore,India
ranju.kn@gmail.com
[2] Asst. Professor, Dept. of Computer Science and Engg,
RVCE, Bangalore, India
sandhya.sampangi@rvce.edu.in
[3]Prof & Head, Dept. of Information Science and Engg
RVCE, Bangalore, India
cauverynk@rvce.edu.in

Abstract: In a distributed computer system the performance of the system is estimated how efficiently the work is divided across the participating nodes. Process migration is a specialized form of process management where by processes are moved from one computing host to another computing host. The most common application of dynamic process migration is load balancing. Load balancing is a computer networking method for distributing workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units or disk drives. Using this both resource utilization and job response time is improved and avoiding a situation in which some nodes are heavily loaded and others are idle. To develop an effective load balancing algorithm so many important issues are considered like load levels comparison, load estimation, system stability, amount of information exchanged among nodes, job resource requirements estimation, job's selection for transfer, remote nodes selection etc. In this paper survey of different memory migration techniques and uses of genetic algorithm in the field of dynamic load balancing are discussed

Keywords: Load balancing schemes, Virtualization, Memory migration techniques, Use of GA in load balancing

1. Introduction

Process migration and load balancing are used in a network of workstations to share the load of heavily loaded processors with more lightly loaded processors, instead of overloading a few machines with a lot of tasks, it makes sense to allocate some of these tasks to under-used machines to increase performance. Process migration in computing comes in two flavors Preemptive and non-preemptive. In pre-emptive process migration, a process is suspended, migrated and then resumes processing on the migrated node. In non-preemptive process migration, migration is done before the execution of a process begins. Any running process can be considered as a victim for migration, but better to choose a process which is consuming too many resources like multimedia application.

The load balancing is divided into two main approaches static and dynamic. In the static approach [1], priori knowledge about the global status of the distributed system, job resource requirement, and communication time are assumed. This has a major impact on the overall system performance due to the unpredictability of load fluctuation of the distributed system.

In the dynamic approach [2], load balancing decisions are based on the current state of the system; tasks are allowed to move dynamically from an overloaded node to an under-loaded node to receive faster service. Dynamic load balancing can produce a better performance because it makes load balancing decisions based on the current load of the system.

Genetic algorithms are powerful search techniques that are used to successfully to solve problems in different discipline. Genetic Algorithms (GAs) are based on principles of natural selection and genetics. Genetic algorithm in dynamic load balancing is used to find the lightly loaded node among the available nodes using the fitness factor.

2. Virtualization

Virtualization [3] is a technology that provides a layer of abstraction between computing, storage and networking hardware, and the applications running on it. Its main advantages include isolation, consolidation and multiplexing of resources. Other benefits of virtualization include saving on power by consolidation of different virtual machines on a single physical machine, migration of virtual machine for load balancing etc. Virtualization provides full control of resource allocation to administrator, resulting in optimum use of resources. More recently, another advantage of virtualization is live migration of virtual machine, which is increasingly used to handle workload balancing across physical machines in data center, and where in which the available resources in physical machine are not sufficient for VMs.

3. Load Balancing Schemes

A Local v/s Global

The local scheduling method is applied to the processes which are present on a single CPU and executed. A global scheduling policy uses information about system to allocate processes to multiple processors to optimize a system wide performance.

B Static v/s Dynamic

Static load balancing uses only information about the average behavior of the system, and strictly based on a fixed and pre-configured set of rules relating to characteristics of the input traffic. Dynamic Load Balancing reacts to the system state that changes dynamically.

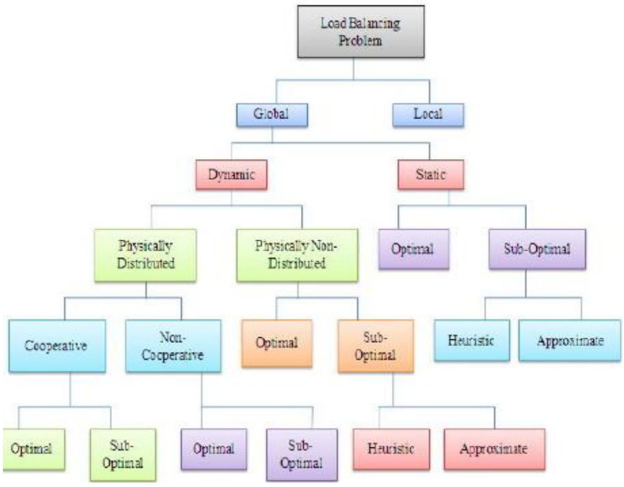

Figure I: Load balancing schemes

C Optimal v/s suboptimal

Static load balancing can be divided into two categories optimal and sub-optimal. When all the information regarding the state of the system as well as the resource needs is known then an optimal assignment can be made based on some criterion function. When for some of computations, optimal solution does not exist then sub-optimal methods can be applied. These methods rely on the rules-of-thumb and heuristics to guide a scheduling process.

D Approximate v/s Heuristic

If optimal solutions are hard to achieve then Approximate and heuristic scheduling is necessary. Suboptimal solutions are reached either by approximating the search space with its subset or by using heuristics.

E Distributed v/s Centralized

Distributed scheduling uses the information provided by the load information management module to make migration decisions , There are a few well-known classes of distributed scheduling policies:

- **A sender-initiated policy[10]** in this heavily loaded nodes search for lightly loaded nodes to which work may be transferred. That is in this method , when a node's load becomes more than the threshold value , it broadcast a message to find a lightly loaded node. It is preferable for low and medium loaded systems, which have a few overloaded nodes.
- **A receiver-initiated policy** in this lightly loaded nodes search for heavily loaded nodes from which work may be transferred .It is preferable for high load systems, with many overloaded nodes and few under loaded ones.
- **A symmetric policy** is the combination of the sender-initiated and receiver initiated policies, to take advantage of the good characteristics of both of them.
- **A random policy** chooses the destination node randomly from all nodes in a distributed system.

In a Centralized dynamic scheduling, the responsibility of scheduling physically reside on a single node. A problem associated with the centralized mechanism is that of reliability. If the centralized server fails, all scheduling system in the system would cease.

4. Memory Migration Techniques

Memory migration techniques can be categorized as stop and copy, precopy, postcopy and freeze free algorithm.

In [9] stop and copy method is explained. In this method, a process execution is stopped while its address space is being transferred. In this method the VM completely stop running on source machine where it is heavily loaded . After coping all memory pages, VM is started on lightly loaded destination machine. Here Migration time and downtime is same because VM is not started on destination host until its all pages are sent to destination machine. Drawback of this method is increased downtime due to the VMs services are completely unavailable until it is started on destination

In pre-copy[5] memory migration, the address space is transferred while the process is still running on the source node. Therefore, once the decision has been made to migrate a process, it continues to run on its source node until its address space has been transferred to the destination node. During this process if some memory pages become dirty then they will be copied once again until the rate of re-copied pages is not less than dirtying page rate. Due to the dirty pages the algorithm works iteratively i.e modified memory pages in the source host since last page transferred must be sent again to the destination. Migration time will increase if the rate of updating of pages is very high. The advantage of this approach is that all updating are available at the destination host and it can be activated any time. Therefore Pre-copy algorithm shortens VM downtime and also avoids the unpredictable overhead and errors caused by too long VM downtime.

Although pre-copy reduces the downtime of the process, but it may increase the total migration time due to the possibility of redundant page transfers, leading to large performance degradation of virtual machine services. So that Hai Jin et al[4] presents the design and implementation of a novel memory-compression based VM migration approach (MECOM) , In this implementation first memory is compressed to provide fast, stable virtual machine migration and guaranteeing the virtual machine services to be slightly affected. Based on memory page characteristics, they design an adaptive zero-aware compression algorithm for balancing the performance and the cost of virtual machine migration. In this design pages are quickly compressed in batches on the source and exactly recovered on the target.

Erik Elmorth et al[6] introduces migration through the combined dynamic page transfer reordering and compression algorithm to overcome the extended migration downtime that may cause service interruption or even failure, and prolonged total migration time that is harmful for the overall system performance. They proposed a novel algorithm that dynamically adapts the transfers order of VM memory pages during migration. Dynamic page transfer reordering is one in which the page update frequency is sampled and this information is used to calculate a page weight, which is then used to prioritize the transfer of less frequently updated pages before the busy ones. By leaving busy pages until last, the number of page re-transfers is reduced, thereby reducing the amount of data to be transferred and thus the total migration time. By combining this technique with a compression scheme that increases the migration throughput the migration downtime is also reduced.

In another method which is a modification of precopy approach[3] which uses LRU and Splay tree, it consists of two feasible implementation stacks and counters, Stack implementation keeps a stack of page numbers of most recently used memory pages, so the top of the stack contains memory pages used in the last recently. So working set list of VM is predicted dynamically by monitoring memory accesses and constructing the LRU list. A splay tree[13] is a self-adjusting binary search tree and which has additional property that recently accessed elements are quick to access again. They proposed framework for live VM migration and it consists of pre-processing phase, push phase , stop and copy phase. In pre-processing phase the system applies the working set algorithm and it is based on LRU replacement algorithm and splay tree algorithm to define the working set list. In push phase , at first iteration the system transfer memory pages except working set list and in second iteration memory pages modified during the previous iteration are send to the destination . In stop and copy phase , the source system suspend the VM for a final transfer, then the source system discards the source VM and transfer the CPU state and last modified pages. Lastly the VM is activated on the target machine. Advantages of this approach is reduced migration time i.e 11.45% on average compared with Xen;s default migration algorithm.

Post-copy[7] VM migration, the VM execution is stopped while its address space is being transferred and is simple and easy to implement. Its main disadvantage is that if a process is suspended for a long time during migration, timeouts may occur. Initially, a small subset of the execution state of VM i.e CPU registers and non-page able memory is transferred to the target machine. Then the VM is resumed at the target machine. At the target, if the VM tries to access pages that have not yet been transferred then it generates page-fault. At target machine these faults are recognized and redirected to the source through the network. **.** Then the source system responds to the network-fault by sending the faulted page. This technique can degrade performance since each page fault of the running VM is redirected towards the source node. This can be improved by accompanied pure demand paging with techniques such as pre-paging.

In [7] the author combined the use of post-copy with adaptive pre-paging in order to eliminate all duplicate page transmissions, and able to reduce the number of network-bound page faults. Pre-paging refers to a more proactive form of pre-fetching from disk in which the memory subsystem can try to hide the latency of high-locality page faults or cache misses by intelligently sequencing the perfected pages. Additionally, to avoid transmitting the free pages, they employ a "dynamic self ballooning". Ballooning is an existing technique that allows a guest Operating System to reduce its memory footprint by releasing its free memory pages back to the hypervisor. DSB automates the ballooning mechanism so it can trigger frequently (every 5 seconds) without degrading application performance (DSB) mechanism.

In[8] freeze free algorithm is explained, in this all hosts are homogeneous and run the same operating system. In freeze free algorithm , the old host can assume the new host will accept the process migration and send the program counter and execution state to the new host. This saves time waiting for a reply which is a costly round trip message. Of course, the algorithm still must restart the migration process if the new host rejects the migration. Other improvements made by the Freeze Free algorithm include concurrent file cache management, initial structure allocation, and increased process modularity. The Freeze Free algorithm discards clean pages and flushes modified file cache pages in parallel with process execution at the new site.

5. Use of Genetic Algorithm in Load Balancing

Genetic algorithm [9] is a kind of random search algorithm which is not only a one-point search but also combines a multipoint search feature. It is difficult to find a suitable receiver which is ready to receive additional task, when a system becomes heavily loaded. So Genetic algorithm is used for determining a destination processor that can receive a task. There are various factors which are needed in load balancing with genetic algorithm such as load measure, fitness function, coding method and algorithm.

a) Load Measure: basically 3-level measure scheme is used to represent the load state of processor. Lightly loaded processor, normally loaded processor, and heavily loaded processor.

b) Coding Methods: There are many kinds of coding methods, binary encoding, character and real value encoding, tree encoding are some examples which are used on population, onto which genetic operations are applied. Binary coded vector <v0, v1,…, vn-1>, can be used to define a string in population, which represent a set of processors to which request messages are sent off.

c) Fitness Function: following formula will evaluate fitness [10]:

$$F= 1/ \alpha \, X \, TMP+ \beta \, X \, TMT+ \lambda X \, TTP \qquad (1)$$

Where α, β and γ are the weights of parameters TMP, TMT, and TTP respectively. Where TMP is the Total Message Processing Time,TMT is Total Message Transfer Time, and TTP is Total Task Processing Time .

Sender initiated load balancing scheme using GA[10], is a part of dynamic load balancing scheme, with the help of genetic algorithm (GA),since GA is easily adaptable and robust search technique. In sender-initiated algorithms, sender having high load tries to transfer task to another processor (receiver) which have low load i.e. the node which is congested tries to move some workload to the node which is not congested. In this genetic algorithm is used to find the destination processor. Each processor contains its own population onto which selection, crossover and mutation, genetic operations are applied. Binary coded vector is used to define a string which represents a set of processors.

The Enhanced Genetic Algorithm [12] is designed based on the standard GAs. The method requires an encoding scheme which can represent all legal solutions to the optimization problem. Any particular solution is uniquely represented by a particular chromosome (or schedule). Chromosomes are manipulated in various ways by applying two genetic operators until the termination condition is met. In order for this manipulation to proceed in the right direction, a quality function called fitness function, is required. The population is generated consisting of S schedules in which 1st and 2nd schedules are generated using optimal strategies and remaining schedules on random basis. Fitness function is based on To achieve maximum load balance, we first introduce the concept of average resource utilization. The average resource utilization is defined as the sum of all resources utilization divided by total number of resources.

Fitness function

The fitness function in dynamic load balancing will have a fitter node in which the task should be transferred such that it has less execution time and communication cost, higher processor utilization and system throughput.

Selection

It is one of the stage of a genetic algorithm and is based on the survival-of-the-fittest mechanism. Chromosome are selected based on the fitness value. There are many methods to select the best chromosomes, for example roulette wheel selection, Boltzman selection, tournament selection, rank selection, steady state selection etc . Most commonly used is Roulette wheel selection, in which each chromosome in the population has a slot sized in proportion to its fitness.

Crossover

Crossover selects genes from parent chromosomes and creates a new offspring. A *Single-Point Crossover* operator randomly selects a point, called Crossover point, on the selected chromosomes, then swaps the bottom halves after crossover point, including the gene at the crossover point and generates two new chromosomes called children

Mutation

Mutation is used to change the genes randomly in a chromosome. Mutation of a bit involves flipping it changing 0 to 1 and vice versa with a small probability.

6. Conclusion

This paper explains a survey of different memory migration techniques and use of GA in load balancing. Process migration involves transferring a running process between machines There are many algorithms to implement process migration which attempt to minimize the process migration and message freeze times. Finally, although process migration is a well-developed area with limited areas of future research, research into compiler technology and programming languages which better support distribution, migration, and heterogeneity, when combined with a suitable process migration mechanism, may allow for better utilization of networks of workstations. With the help of genetic algorithm suitable candidate receiver is decided to which request message should be sent off. In terms of achieving the goals of maximum processor utilization and minimum total completion time, genetic based load balancing algorithm performs really well.

References

[1] X. Tang and S.T. Chanson, "Optimizing Static Job Scheduling in aNetwork of Heterogeneous Computers," Proc. of the Intl. Conf. On Parallel Processing, pp. 373-382, August 2000.

[2] Ali M. Alakeel, “A Guide to Dynamic Load Balancing in Distributed Computer Systems”, IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.6, June 2010.

[3] Ei PhyuZaw “improved Live Virtual migration using LRU and spaly tree algorithm” International . journal of computer science and telecommunications.S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, “A novel ultrathin elevated channel low-temperature poly-Si TFT,” *IEEE Electron Device Lett.*, vol. 20, pp. 569–571, Nov. 1999.

[4] H. Jin, L. Deng, S. Wu, X. Shi and X. Pan X, Live Virtual Machine Migration with Adaptive Memory Compression, IEEE International Conference on Cluster Computing (2009).

[5] Clarck, C., Fraser, k., Hand, S., Hansen, J., Jule, E., Limpach, C., Pratti, I., And warfield, A.” Live migration of virtual machines”. In Network System Design and Implementation 2005

[6] Erik Elmroth, Benoit Hudzia, Petter Sv¨ard and Johan Tordsson “High performance live migration through dynamic page transfer reordering and compression “2011 Third IEEE International Conference on Coud Computing Technology and Science.

[7] Michael R. Hines and Kartik Gopalan “Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning”, *In VEE ’09, pages 51-60. ACM,2009.*

[8] Ellard Thomas Roush, “The freeze free algorithm for process migration” ,Technical report 1924, University of Illinois at Urbana-Champaign, Urbana, Illinois, 1995.

[9] Kulvinder Singh Handa, Chetna Kukreja.*”* Load Balancing With Genetic Algorithm”, International Journal of Emerging Technologies in Computational and Applied Sciences, Aug-Nov. 2012, pp. 1-4

[10] Kulvinder Singh1,Chetna Kukreja University Institute of Engineering and Technology Kurukshetra University Kurukshetra, Haryana .*“*A Sender-Initiated Load Balancing Approach with Genetic Algorithm*”.* International Journal of Software and Web Sciences 12-201, 2012.

[11] S. Lee and C. Hwang, “A dynamic load balancing approach using genetic algorithm in distributed systems”, IEEE, 1998, pp. 639-644.

[12] Sandip Kumar Goyal and Manpreet Singh “Enhanced Genetic Algorithm Based Load Balancing in Grid”, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 3, No 2, May 2012.

[13] D. Dominic Sleator, R. Endre Tarjan. Self-Adjusting Binary Search Trees. In Journal of the Association for Computing Machinery, Vol. 32, No.3, July 1985, pp.652-686.