Process/Thread Migration and Load Balancing

Operating Systems Seminar Report

Saumya Shah 121046

Process/Thread Migration and Load Balancing

Process Migration

Process Migration is a specialized form of process management whereby processes are moved from one computing environment to another. The most common application of process migration is in computer clusters where processes are moved from machine to machine.

Process migration in computing comes in two flavors

Non-preemptive process migration

Process migration that takes place before execution of the process starts (i.e. migration whereby a process need not be preempted). This type of process migration is relatively cheap, since relatively little administrative overhead is involved.

Preemptive process migration

Process migration whereby a process is preempted, migrated and continues processing in a different execution environment. This type of process migration is relatively expensive, since it involves recording, migration and recreation of the process state as well as the reconstructing of any inter-process communication channels to which the migrating process is connected.

Load Balancing

To understand Load balancing, it is necessary to understand load. Load may be defined as number of tasks running in queue, CPU utilization, load average, I/O utilization, amount of free CPU time/memory, etc., or any combination of the above indicators. Load balancing can be done among interconnected workstations in a network or among individual processors in a parallel machine. Load balancing is nothing but the allocation of tasks or jobs to processors to increase overall processor utilization and throughput. Actually load balancing is done by process migration. But to balance the load it is

necessary to measure the load of individual node in network or in a distributed environment. After calculating the load of individual node, mark the underloaded/free and overloaded/busy node. The process form the overloaded node is migrated to the underloaded node for load balancing.

Load Balancing Algorithms

Static

- Deterministic
- Probablistic

Dynamic

- Centralized
- Distributed
    - Co-operative
    - Non co-operative

Static versus Dynamic

- Static algorithms use only information about the average behavior of the system
- Static algorithms ignore the current state or load of the nodes in the system
- Dynamic algorithms collect state information and react to system state if it changed
- Static algorithms are much more simpler
- Dynamic algorithms are able to give significantly better performance

Deterministic versus Probabilistic

- Deterministic algorithms use the information about the properties of the nodes and the characteristic of processes to be scheduled
- Probabilistic algorithms use information of static attributes of the system (e.g. number of nodes, processing capability, topology) to formulate simple process placement rules

- Deterministic approach is difficult to optimize
- Probabilistic approach has poor performance

Issues and policies in designing Load-balancing algorithms

Load estimation policy determines how to estimate the workload of a node

Process transfer policy determines whether to execute a process locally or remote

State information exchange policy determines how to exchange load information among nodes

Location policy determines to which node the transferable process should be sent

Priority assignment policy determines the priority of execution of local and remote processes

Migration limiting policy determines the total number of times a process can migrate

Load estimation policy

Most of the algorithms use the *threshold policy* to decide on whether the node is lightly-loaded or heavily-loaded. Threshold value is a limiting value of the workload of node which can be determined by

Static policy: predefined threshold value for each node depending on processing capability

Dynamic policy: threshold value is calculated from average workload and a predefined constant

Below threshold value node accepts processes to execute, above threshold value node tries to transfer processes to a lightly-loaded node.

Bidding method and Pairing are also used for load estimation policy.

Priority assignment policy

Selfish- Local processes are given higher priority than remote processes. Worst response time performance of the three policies.

Altruistic-Remote processes are given higher priority than local processes. Best response time performance of the three policies.

Intermediate-When the number of local processes is greater or equal to the number of remote processes, local processes are given higher priority than remote processes. Otherwise, remote processes are given higher priority than local processes.

Migration Limiting Policy

This policy determines the total number of times a process can migrate

Uncontrolled-A remote process arriving at a node is treated just as a process originating at a node, so a process may be migrated any number of times

Controlled-Avoids the instability of the uncontrolled policy

Use a migration count parameter to fix a limit on the number of time a process can migrate

Irrevocable migration policy: migration count is fixed to 1

For long execution processes migration count must be greater than 1 to adapt for dynamically changing states.

Disadvantages of Load Balancing

The attempt to equalize the workload on all the nodes is not an appropriate object since big overhead is generated by gathering exact state information.

Load balancing is not achievable since number of processes in a node is always fluctuating and temporal unbalance among the nodes exists every moment.

References

http://www.ijsce.org/attachments/File/v2i1/A0480032212.pdf

http://coltech.vnu.edu.vn/httt/media/sach/nlhdh/TanenbaumModernOperatingSystems.pdf