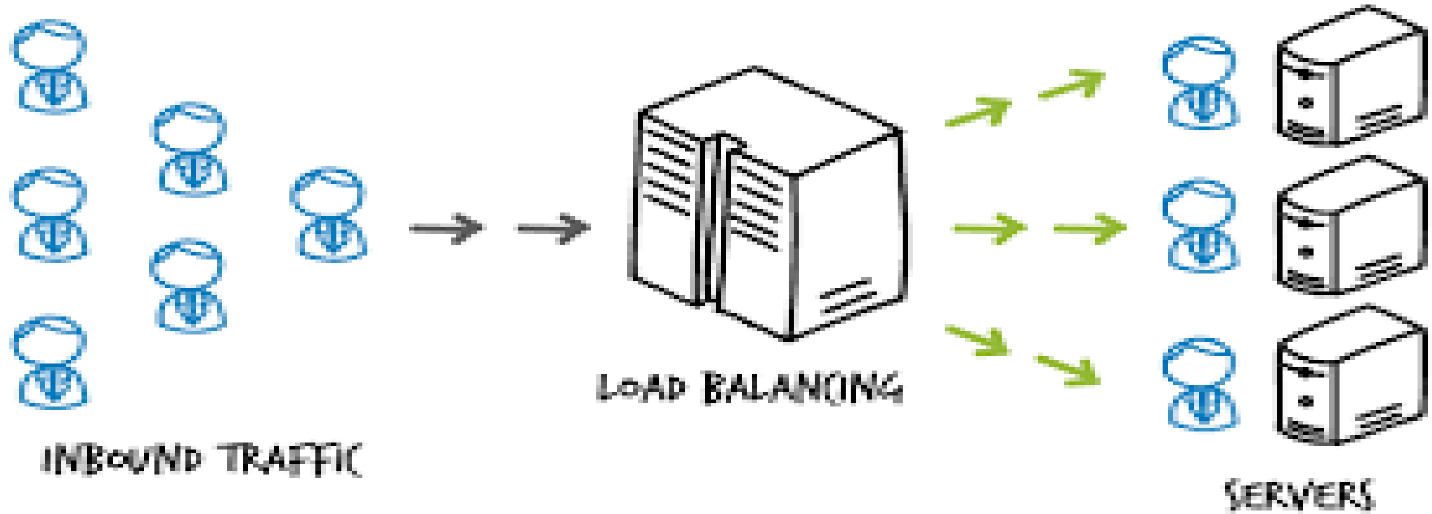


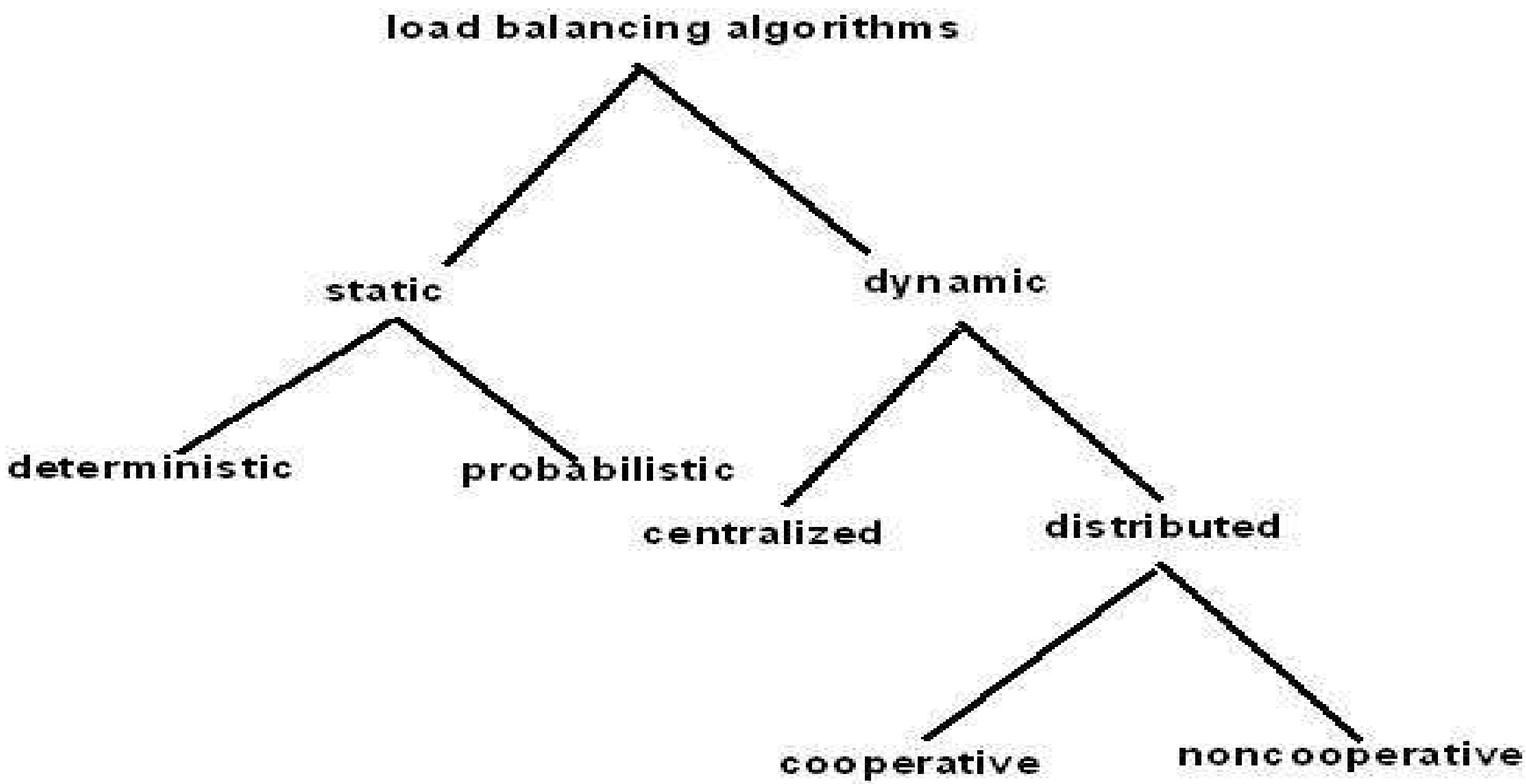
Load Balancing

Reference: Pradeep K Sinha,
"Distributed Operating Systems: Concepts and Design",
Prentice Hall of India, 2007

Load balancing approach

- Load balancing Algorithms are also known as load-leveling algorithms.
  - Based on the intuition of better resource utilization.
- Algorithm tries to balance the total system load by transparently transferring the workload from heavily loaded nodes to lightly loaded nodes.
- Goal: maximize the total system throughput

Taxonomy of load balancing



Static vs. Dynamic

- Static algorithms:
  - Use only information about the average behavior of the system, ignoring the current state of the system.
  - Simpler because no need to maintain and process system state information.
  - Do not react to the current system state.

Static vs. Dynamic

- Dynamic algorithms:
  - React to the system state that changes dynamically.
  - Able to avoid states with unnecessarily poor performance.
  - More complex than static algorithms.

Deterministic vs. Probabilistic

- Both are Static load balancing algorithms.
- Deterministic algorithms:
  - Use the information about the properties of the node and characteristics of the processes.
  - Difficult to optimize and cost more to implement.

Deterministic vs. Probabilistic

- Probabilistic algorithms:
  - Use information regarding static attributes of the system.
  - Easier to implement.
  - Suffer from having poor performance.

Centralized vs. Distributed

- Centralized algorithm:
  - The responsibility of scheduling physically resides on a single node.
  - System state information is collected at a single node at which all the scheduling decisions are made.
    - Known as Centralized server node.

Centralized vs. Distributed

Centralized algorithm:

- Problem : reliability
  - If the centralized server fails, all scheduling in the system would cease.

- Solution : replicate the server on K+1 nodes if it is to survive k faults.

Centralized vs. Distributed

- Distributed algorithms:
  - The work involved in making process assignment decisions is physically distributed among the various nodes of the system.
  - Avoids the bottleneck of collecting state information at a single node.
  - Allows the scheduler to react quickly to dynamic changes in the state.

Centralized vs. Distributed

- Distributed algorithms:
  - Algorithm is composed of entities known as local controllers.
  - Each entity is responsible for making scheduling decisions for the processes of its own node.

Co-operative vs. Non-Cooperative

- **Non-cooperative algorithms :**
  - Individual entities act as autonomous entities and make scheduling decisions independently of the actions of other entities.

Co-operative vs. Non-Cooperative

- Cooperative algorithms:
  - Distributed entities cooperate with each other to make scheduling decisions.
  - More complex and involve larger overhead than non-cooperative.

Issues in designing load balancing algorithms

- Load estimation policy
- Process transfer policy
- State information exchange policy
- Location policy
- Priority assignment policy
- Migration limiting policy

Issues in designing load balancing algorithms

- Local Process
  - A process which is processed at its originated node.

- Remote Process
  - A process which is processed at a node different than the one on which it originated.

Load Estimation Policies

- Estimation based on parameters like:
  1. Total no. of processes on the node at the time of load estimation.
  2. Resource demands of these processes.
  3. Instruction mixes of these processes.
  4. Architecture and speed of the node's processor.

Load Estimation Policies

- Sum of the remaining service times of all the processes on a node can be a measure for estimating a node's workload.

- Issue: how to estimate the remaining service time of the processes?

Load Estimation Policies

- Solutions:
  1. Memoryless method
     - This method assumes that all processes have the same expected remaining service time, independent of the time used so far.
     - It reduces the load estimation method to that of total number of processes.

Load Estimation Policies

2.Pastrepeats

- This method assumes that the remaining service time of a process is equal to the time used so far by it.

3.Distribution method

- If the distribution of service times is known, the associated process's remaining service time is the expected remaining time conditioned by the time already used.

Process Transfer Policies

- Load balancing algorithms use the threshold policy to decide whether a node is lightly or heavily loaded.

- The threshold value of a node:
  - the limiting value of its workload,
  - used to decide whether a node is lightly or heavily loaded.

Process Transfer Policies

- Methods to determine the threshold value of a node:
  1. Static policy
     - Each node has a predefined threshold value depending on its processing capability.
     - This value does not vary with the dynamic changes in workload at local or remote nodes.
     - No exchange of state information among the nodes to decide this value.

Process Transfer Policies

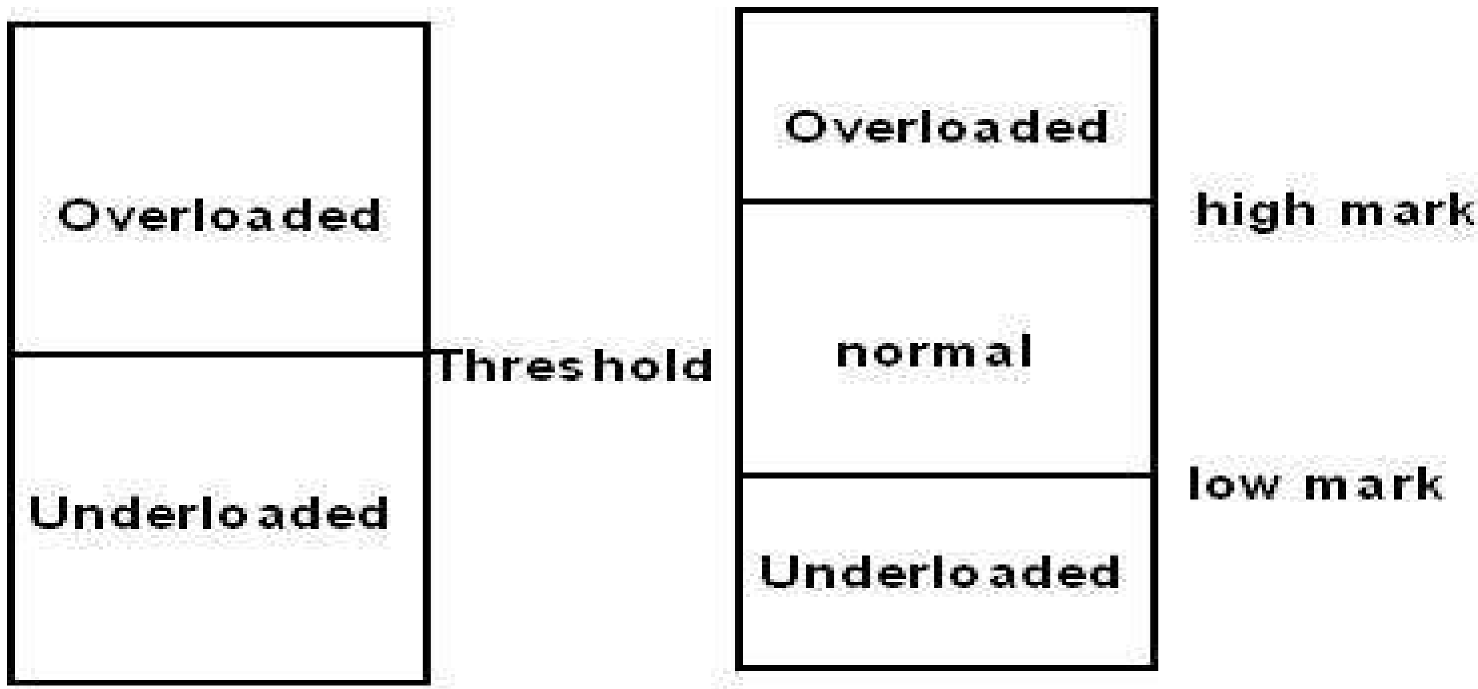
2. Dynamic policy

- The threshold value of a node is calculated as a product of the average workload of all the nodes and a predefined constant (ci).
- Nodes exchange state information by using one of the state information exchange policies.
- It gives a more realistic value of threshold for each node.
- Involves overhead in exchange of state information.

Process Transfer Policies

- Most load balancing algorithms uses a single threshold and thus only have overloaded and under loaded regions.

Load regions



Single threshold policy
and
double threshold policy

Single-threshold policy

- A node accepts new processes (either local or remote) based on its load.
  - Accepts if load is below the threshold value.
  - Rejects if load is above the threshold value.
- It makes scheduling algorithms unstable.

Single-threshold policy

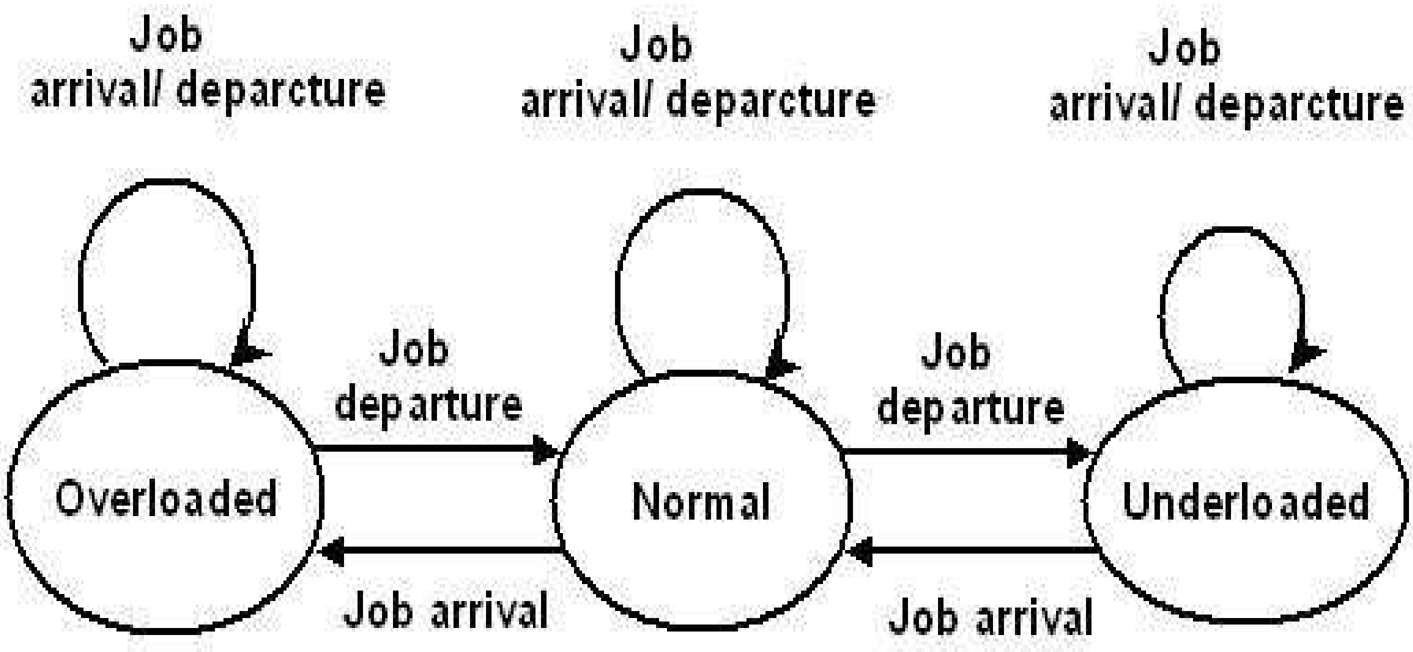
- A node should only transfer one or more of its processes to another node if such transfers greatly improves the performance of the rest of its local processes.

- A node should accept remote processes if its load is such that the added workload of processing these incoming processes does not significantly affect the service to the local ones.

Double-threshold policy

- Also known as high-low policy.
- Use of two threshold values: high mark and low mark.
- Three regions :
  - Overloaded
  - Normal
  - Under loaded

Double-threshold policy



State transition diagram of the load of a node in case of double threshold policy

Double-threshold policy

- For overloaded region:
  - New local processes are sent to be run remotely and requests to accept remote processes are rejected.
- For normal region:
  - New local processes run locally and requests to accept remote processes are rejected.
- For underloaded region:
  - New local processes run locally and requests to accept remote processes are accepted.

Location policies

1. Threshold
2. Shortest
3. Bidding
4. Pairing

Threshold

- A destination node is selected at random.
- A check is made to determine
  - whether the transfer of the process to that node would place it in a state that prohibits the node to accept remote processes.
  - If not, the process is transferred to the selected node, which must execute the process regardless of its state when the process actually arrives.

Threshold

- If the check indicates that the selected node is in a state that prohibits it to accept remote processes, another node is selected at random and probed in the same manner.

- A static probe limit L_P is used here.

Shortest

- L_P distinct nodes are chosen at random and each is polled in turn to determine its load.

- The process is transferred to the node having the minimum load value, unless that node's load is such that it prohibits the node to accept remote processes.

Shortest

- If none of the polled node can accept the process, it is executed at its originating node.

- Discontinue probing whenever a node with zero load is encountered.

Bidding

- Each node in the network is responsible for two roles : manager and contractor.
- The Manager represents a node having a process in need of a location to execute.
- The Contractor represents a node that is able to accept remote processes.

Bidding

- To select a node for its processes, a manager broadcasts a request-for-bids message to all other nodes in the system.
- The contractors return bids to the manager node.
- Manager transfers the process to the node with best bid.

Bidding

- Problem:
  - A contractor may win many bids from many other manager nodes and thus becomes overloaded.

- Solution:
  - On choosing best bid, manager node may send a message to the owner of that bid and send process on acknowledgement.

Bidding

- Both manager and contractor are free to take decisions.
- Drawback of bidding policy:
  - Communication overhead
  - Difficult to decide a good pricing policy.

Pairing

- This policy reduces the variance of loads only between pairs of nodes of the system.
- Two nodes that differ greatly in load are temporarily paired with each other.
- The load-balancing operation is carried out between the nodes belonging to the same pair.

Pairing

- A node only tries to find a partner if it has at least two processes; otherwise migration from this node is never reasonable.
- Use of random selection of pair.
- The pair is broken as soon as the process migration is over.

State information exchange policies

1. Periodic broadcast
2. Broadcast when state changes
3. On- demand exchange
4. Exchange by polling

Periodic broadcast

- Each node broadcasts its state information after the elapse of every t units of time.
- Generates heavy traffic.
- Possibility of fruitless messages being broadcast.
- Poor scalability problem.

Broadcast when state changes

- A node broadcasts its state information only when its state changes.
  - When a process arrives at that node or when a process departs from that node.
  - When its state switches from the normal load region to either the underloaded region or the overloaded region.
    - Works with two-threshold transfer policy.

On- demand exchange

- A node broadcasts a StateInformationRequest message when its state switches from the normal load region to either the underloaded region or the overloaded region.
- Receiving nodes send their current state to the requesting node.
- Policy works with two-threshold transfer policy.

On- demand exchange

- The status of the requesting node is included in the StateInformationRequest message.

- If this status is
  - Underloaded, only overloaded nodes will respond to it.
  - Overloaded, only underloaded nodes will respond to it.

Exchange by polling

- No need for a node to exchange its state information with all other nodes in the system.
- When a node needs the cooperation of some other node for load balancing, it can search for a suitable partner by randomly polling the other nodes one by one.

Priority Assignment Policies

1. Selfish
2. Altruistic
3. Intermediate

Priority Assignment Policies

- Selfish:
  - Local processes are given higher priority than remote processes.
  - Yields the worst response time performance among other policies.
    - Poor performance of remote processes.
    - Best response time for local processes.

Priority Assignment Policies

- Altruistic:
  - Remote processes are given higher priority than local processes.
  - Policy has best response time of the three policies.

Priority Assignment Policies

- Intermediate:
  - The priority of processes depends on the number of local processes and the number of remote processes at the concerned node.
  - If no. of local nodes is greater than or equal to the no. of remote processes, priority will be given to local processes otherwise to remote processes.
  - Overall response time performance is much closer to that of the altruistic policy.

Migration- limiting policies

- A decision about the total no. of times a process should be allowed to migrate.

- Two migration-limiting policies:
  - Uncontrolled
  - Controlled

Migration- limiting policies

- Uncontrolled
  - A remote process arriving at a node is treated just as a process originating at the node.
  - A process may be migrated any no of times.

Migration- limiting policies

- Controlled
  - To overcome the instability problem of the uncontrolled policy, most system treat remote processes different from local processes and use a migration count parameter.

Thank You