B.E COMPUTER ENGINEERING
CLOUD COMPUTING

Project Title

Load Balancing Algorithm in Cloud

Group Member:

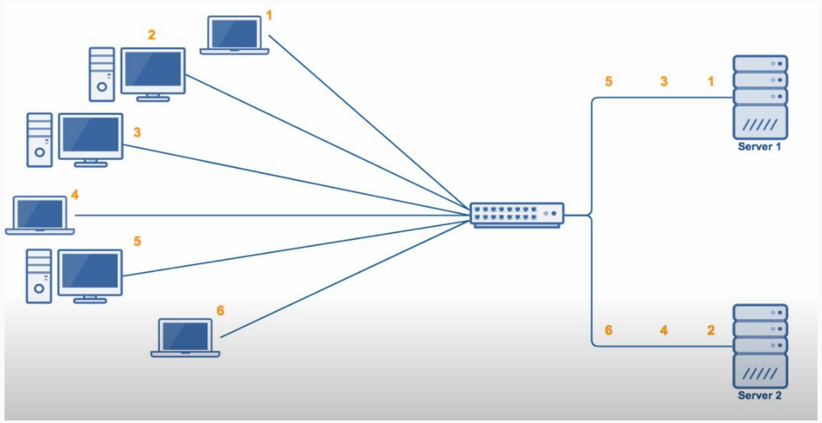
Paras Singh Maravi : 0801cs181056
Nayanika Vyas : 0801cs181048
Ashwin Kaurav : 0801cs181018

Load Balancing

- Load Balancing is a technique to distribute the workload across various nodes, or other resources like central processing units, network links, etc.

- The main aim is to achieve efficient resource utilization, increase in the throughput, decrease the response time and minimize the overhead. It also helps to achieve the fairly distribution of the workload.

- The important things which are to be considered while developing such algorithms are selection of nodes, nature of work to be transferred, performance of the system, comparison of load etc.



Why load balancing is vital to cloud environments

Load balancers are highly beneficial to cloud environments, where massive workloads can easily overwhelm a single server and high levels of service availability and response times are critical to certain business processes or mandated by SLAs.

Load balancing also plays a key role in a cloud's scalability. By nature, cloud infrastructures are supposed to easily scale up to accommodate any uptick or surge in traffic. When a cloud "scales up", it typically spins up multiple virtual servers and runs multiple application instances. The main network component responsible for distributing traffic across these new instances is/are the load balancer(s).

Without load balancers, newly spun virtual servers wouldn't be able to receive the incoming traffic in a coordinated fashion or if at all. Some virtual servers might even be left handling zero traffic while others became overloaded.

Load Balancing Techniques and Algorithms

Network layer Algorithms

- **Round robin** – A batch of servers are programmed to handle load in a rotating sequential manner. The algorithm assumes that each device is able to process the same number of requests and isn't able to account for active connections.
- **Weighted round robin** – Servers are rated based on the relative amount of requests each is able to process. Those having higher capacities are sent more requests.
- **Least connections** – Requests are sent to the server having the fewest number of active connections, assuming all connections generate an equal amount of server load.
- **Weighted least connections** – Servers are rated based on their processing capabilities. Load is distributed according to both the relative capacity of the servers and the number of active connections on each one.
- **Source IP hash** – Combines the source and destination IP address in a request to generate a hash key, which is then designated to a specific server. This lets a dropped connection be returned to the same server originally handling it.

Load Balancing Techniques and Algorithms

Application Layer algorithms

- **Accurate load distribution** – Unlike network layer algorithms distributing requests according to preset rules, LPR intelligently selects the server best suited to process an incoming connection in real-time.
- **Request specific distribution** – LPR can acknowledge that connection requests take different processing times and distributes load accordingly. As a result, traffic isn’t routed to busy servers.

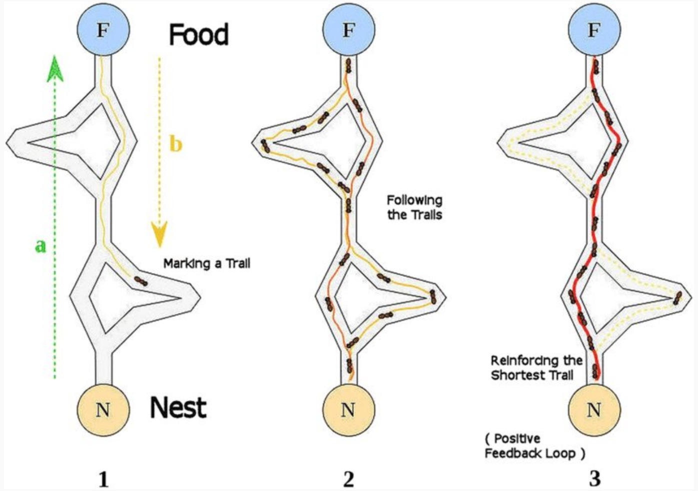
Aim and Objective

Aim:
The aim of this project is to reduce average response time and average datacentre processing time with the help of proposed ACO VM load balancing algorithm.

Objective:
To implement the proposed Ant Colony Optimization(ACO) algorithm and compare it with the existing load balancing algorithm.

Ant Colony Optimization(ACO)

- Originally proposed by Marco Dorrigo in 1992.
- The inspiring source os ACO is the foraging behaviour of real ants. When searching for food , ants initially explore the area surrounding their nest in a random manner.
- Indirect communication between the ants via pheromone trails enables them to find shortest paths between their nest and food sources.



Ant Colony Load Balancing Algorithm

The inputs which are considered in the algorithm are: n = VM count, pheromone[n][n], probability[n]

Parameters considered in the algorithm are: alpha=1; beta=1

Number of ants=10;
Evaporation factor =0.5;

Steps for Ant Colony Load Balancing Algorithm

Step 1: Initialize pheromones[n][n].

Step 2: Move ants through VMs to lay pheromone.

Step 3: Calculate probability [n] according to VM visited.

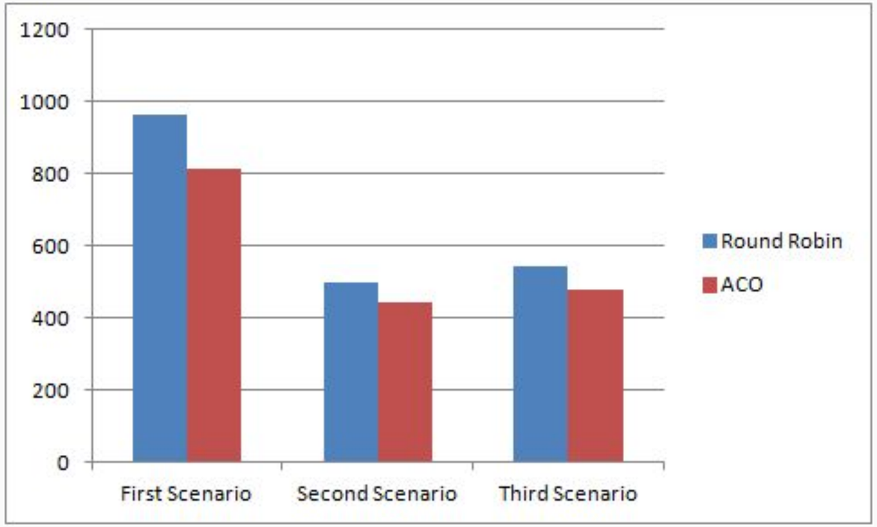
Step 4: Update pheromone[n] table.

Step 5: Select the underload VM with highest probability from pheromone[n].

Step 6: Allocate cloudlet to the selected underload VM.

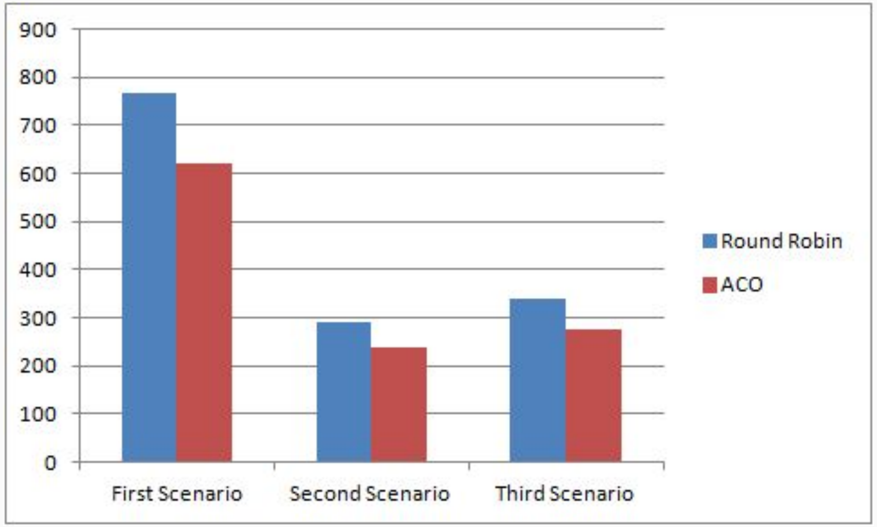
Comparative analysis of Average Response Time

Scenarios	DataCenter	Round Robin(ms)	ACO(ms)
First Scenario	1 DC with 50VMs	963.87	814
Second Scenario	2 DC with 25VMs each	494.94	444.02
Third Scenario	2 DC with 50VMs each	542.43	479.62



Comparative analysis of Average Datacentre Response Time

Scenarios	DataCenter	Round Robin(ms)	ACO(ms)
First Scenario	1 DC with 50VMs	767.82	621.35
Second Scenario	2 DC with 25VMs each	291.01	239.54
Third Scenario	2 DC with 50VMs each	338.76	275.06



THANK YOU!