Intra-Datacenter Load Balancing in a Federated Cloud with Throttled Algorithm

Instructor- Dr. Sumit Kalra

Presented By-
Himani (M23CSA516)
Ankit Kumar Chauhan (M23CSA509)

Problem Statement

Problem Addressed: The project addresses uneven load distribution within datacenters in a federated cloud environment, which leads to reduced Quality of Service (QoS) and Service Level Agreement (SLA) violations.

Context: Federated clouds aggregate resources from multiple cloud providers but struggle with intra-datacenter load imbalance, causing inefficient resource utilization and delayed response times.

Importance: This issue is critical in cloud computing as effective load balancing ensures optimal performance, scalability, and cost-efficiency while meeting user demands.

Literature Review

- **Hybrid Load Balancing Algorithms:**
    a. **MMRR Algorithm:** Combines maximum-minimum and round-robin methods to improve response times but fails to consider dynamic VM states during high-load periods.
    b. **HAMM Algorithm:** Hybrid of min-min and max-min algorithms, achieving 89.6% makespan and 89.5% resource utilization but lacks AI integration.
- **Nature-Inspired Approaches:**
    a. Genetic algorithms and honeybee-inspired strategies are more adaptive but often require complex parameter tuning and computational overhead.
- **Throttled Load Balancing Algorithm:**
    a. The throttled algorithm serves as the basis for this work, showing promise in dynamically allocating tasks based on real-time resource availability but requiring adaptation for federated environments.
- **Comparative Reviews of Cloud Simulators:**
    a. Evaluations of tools like CloudAnalyst and CloudSim highlight the importance of simulation-based validation.

Existing Results

Test Case 1:

- **Scenario 1 to Scenario 5:** Varying the number of VMs from 10 to 100 resulted in a decrease in both response time and processing time.
- **Results:**
    - With 100 VMs, the intra-balancer achieved an average response time of 98.93 ms and processing time of 2.26 ms.

Scenario ID	Scenario	Overall Average Response Time (milliseconds)	Overall Average Processing Time (milliseconds)
Scenario 1	2 VMs in DC 1, 4 VMs in each DC 2 and DC 3	140.98	44.53
Scenario 2	5 VMs in DC 1, 10 VMs in each DC 2 and DC 3	111.91	15.20
Scenario 3	10 VMs in DC 1, 15 VMs in each DC 2 and DC 3	105.54	8.94
Scenario 4	10 VMs in DC 1, 20 VMs in each DC 2 and DC 3	102.69	6.17
Scenario 5	20 VMs in DC 1, 40 VMs in each DC 2 and DC 3	98.93	2.26

Table 1. Intra-balancer results (Case 1)

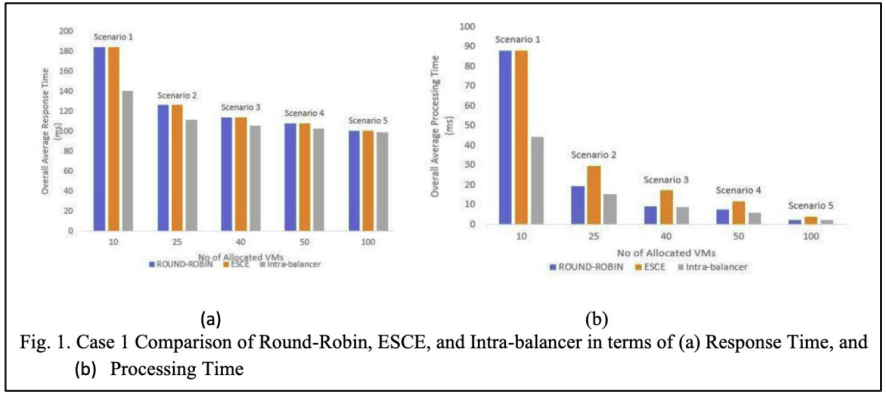


Fig. 1. Case 1 Comparison of Round-Robin, ESCE, and Intra-balancer in terms of (a) Response Time, and (b) Processing Time

Test Case 2:

Scenario 1 to Scaneario 5: By varying the number of requests per user base while keeping the total number of VMs constant (100 VMs).

Results

The intra-balancer consistently maintained response times around **98–103 ms** and processing times near **1.7–2.3 ms**.

Scenario ID	Requests Per Hour	Overall Average Response Time (milliseconds)	Overall Average Processing Time (milliseconds)
Scenario 1	5	98.94	2.27
Scenario 2	10	103.17	1.80
Scenario 3	15	103.40	1.73
Scenario 4	20	103.40	1.67
Scenario 5	25	102.54	1.69

Table 2. Intra-balancer results (Case 2)

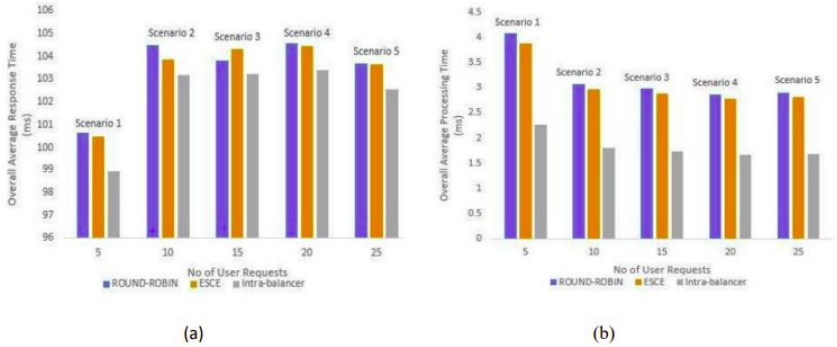


Fig. 2. Case 2 Comparison of Round-Robin, ESCE, and Intra-balancer in terms of (a) Response Time, and (b) Processing Time

Our Implementation

- **Content:**
  - **Step 1: Environment Setup:**
    - Installation of Java SDKs and Java IDE.
    - Downloading and importing the CloudAnalyst project into the IDE.
  - **Step 2: Algorithm Adaptation:**
    - The throttled algorithm was adapted for intra-datacenter balancing. It continuously monitors VM states (Available vs. Busy).
    - A table of VMs is maintained, allowing the load balancer to assign requests to the first available VM. If none is available, the request is queued.
  - **Step 3: Simulation Execution:**
    - Simulated on CloudAnalyst by altering the number of VMs and user request rates. Performance metrics like response time and processing time were recorded.
  - **Step 4: Validation:**
    - Results were compared with Round-Robin and ESCE algorithms to evaluate improvements.

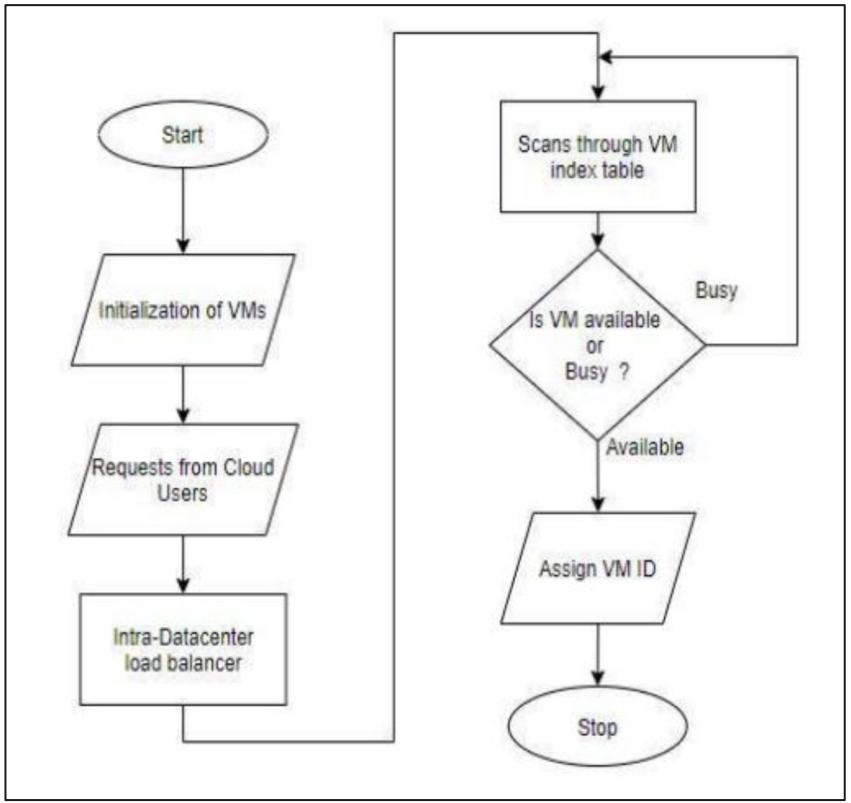


Fig: The Flowchart of the Proposed Intra-Datacenter Load Balancing Algorithm

Our Results

Test Case 1:

- **Improved Latency:** Achieved an average response time as low as 98.03 ms in high resource scenarios.
- **Enhanced Processing Efficiency:** Processing times as low as 2.12 ms.
- **Scalability:** Response and processing times improved as the number of VMs increased.

Scenario ID	Scenario	Overall Average Response Time (milliseconds)	Overall Average Processing Time (milliseconds)
Scenario 1	2 VMs in DC 1, 4 VMs in each DC 2 and DC 3	140.27	43.36
Scenario 2	5 VMs in DC 1, 10 VMs in each DC 2 and DC 3	111.46	14.54
Scenario 3	10 VMs in DC 1, 15 VMs in each DC 2 and DC 3	104.87	7.98
Scenario 4	10 VMs in DC 1, 20 VMs in each DC 2 and DC 3	102.97	6.05
Scenario 5	20 VMs in DC 1, 40 VMs in each DC 2 and DC 3	98.94	2.12

Table 3. Intra-balancer results (Case 1)

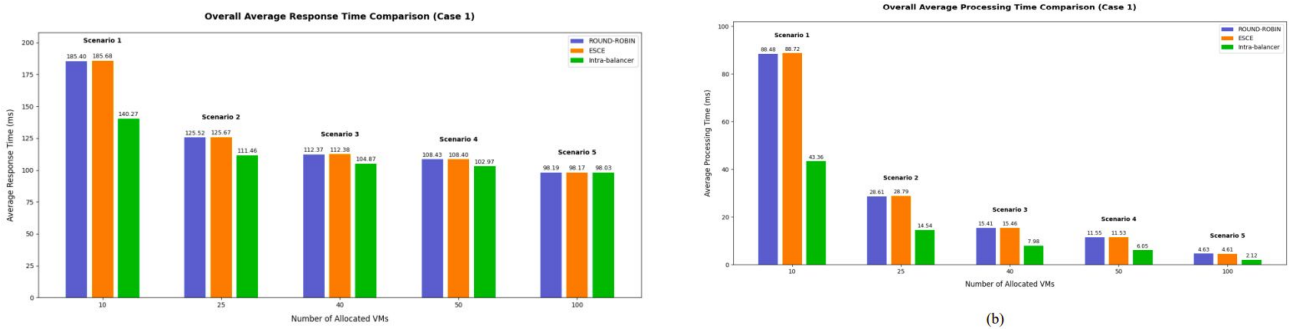
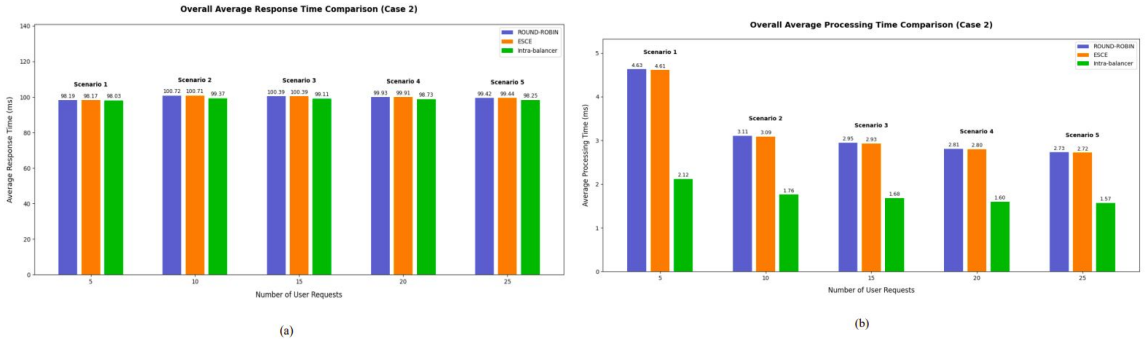


Fig. 3. Case 1 Comparison of Round-Robin, ESCE, and Intra-balancer in terms of (a) Response Time, (b) Processing Time, and (c) Simulation Result

Test Case 2: By varying the number of requests per user base while keeping the total number of VMs constant (100 VMs), the intra-balancer consistently maintained response times around 98–99 ms and processing times near 1.6–2.0 ms. Table 4 and Fig. 4 denotes this.

Scenario ID	Requests Per Hour	Overall Average Response Time (milliseconds)	Overall Average Processing Time (milliseconds)
Scenario 1	5	98.94	2.12
Scenario 2	10	99.37	1.76
Scenario 3	15	99.11	1.68
Scenario 4	20	98.73	1.60
Scenario 5	25	98.25	1.57

Table 4. Intra-balancer results (Case 2)



(a) (b)

Fig. 4. Case 2 Comparison of Round-Robin, ESCE, and Intra-balancer in terms of (a) Response Time, and (b) Processing Time

Comparison and Analysis

- **Key Findings (Test Case 1):**

- **Response Time:** The intra-balancer consistently achieved lower response times (98.03 ms with 100 VMs) compared to Round-Robin and ESCE.

- **Processing Time:** The intra-balancer reduced processing time to 2.12 ms, outperforming traditional methods.

- **Efficiency in Resource Utilization:** The algorithm minimizes idle time and optimizes the use of cloud resources.

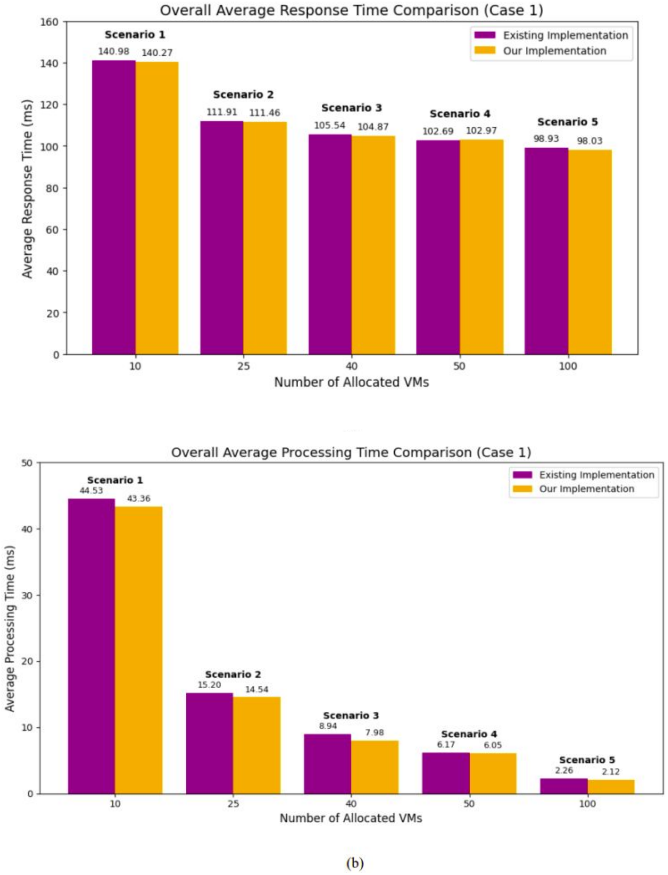


Fig. 5. Case 1 Comparison of existing and our implementation of Intra-balancer in terms of (a) Response Time, and (b) Processing Time

Key Findings (Test Case 2):

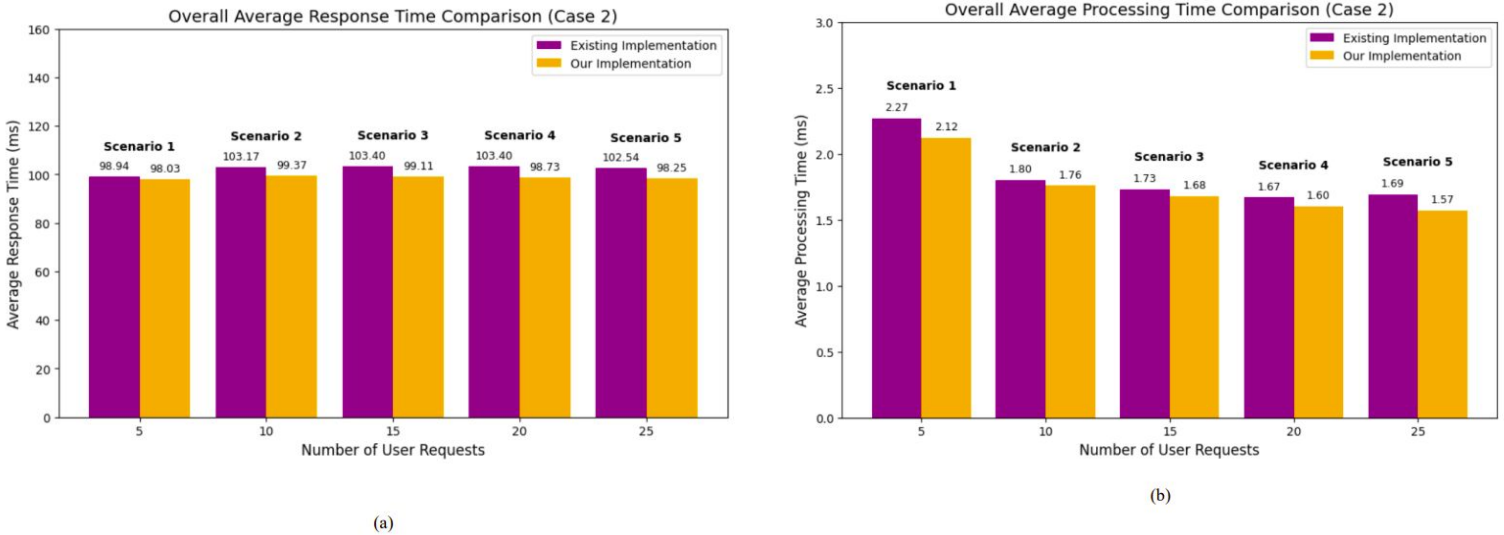


Fig. 6. Case 2 Comparison of existing and our implementation of Intra-balancer in terms of (a) Response Time and (b) Processing Time

Future Scope

- **AI/ML Integration:** Implement machine learning algorithms to dynamically predict VM workload and optimize request distribution.
- **Enhanced Security:** Implement security protocols and fault tolerance mechanisms.
- **Advanced Simulation Models:** Use real-time monitoring systems to refine the load balancing strategy.

Conclusion

- The proposed intra-datacenter load balancing algorithm successfully adapts the throttled algorithm to federated clouds, improving response and processing times.
- Future work includes AI/ML enhancements and advanced optimization techniques.

Thank You