Total Response Time Reduction in Cloud Server Load Balancing using Classical Optimization Techniques

Jayant Parmar
Postgraduate Diploma in Artificial Intelligence
Optimization for Data Science
Indian Institute of Technology Jodhpur, Jodhpur, India
Email: g25ait1072@iitj.ac.in

Nishant Kumar
Department of Electrical Engineering
Indian Institute of Technology Jodhpur, Jodhpur, India
Email: drnkiit@gmail.com

Abstract— The purpose of this paper is to provide a mathematical and analytical model that would minimize the overall response time in a server load balancing configuration.The optimization problem is used to calculate the optimal allocation of requests to the heterogeneous servers and reduce the total response time and maintain the stability of the system. The model is made out of queueing theory and classical principles of optimization and can be used to find a structured and solvable solution to cloud systems of different capacity. An extensive test case setback confirms the efficiency of this approach at varying loads, which is why it is relevant to the current data center.

Keywords—Load Balancing, Cloud Computing, Response Time Minimization, Optimization, Queueing Theory

I. INTRODUCTION

Cloud computing has recently been the model on which scalable computing infrastructure is built, which allows on-demand service provisioning and distribution of workloads flexibly.With millions of users using applications at the same time, cloud providers need to make sure that a single server is not a bottleneck with the rest of the servers not utilized fully.The need results in the fundamental problem of load balancing which is the effective distribution of the incoming user requests among the various servers in order to optimize the overall performance.

One important performance indicator is the user response time. A theoretical basis for mathematically modeling this behavior is provided by the queuing theory. The mean response time in a basic M/M/1 queue is determined by the arrival rate (λ) and the service rate (μ), so that as the system gets closer to saturation ($\lambda \rightarrow \mu$), the expected response time rises significantly. It is possible to reduce overall response time while preserving system stability by strategically allocating λ among servers with varying μ values. In order to determine the optimal load allocation, this study creates an analytical optimization model which applying the classical techniques of Steepest Descent and Newton Method.The mathematical formulation establishes a criterion of future adaptive load-balancing procedures in a cloud computer since it ensures that the resultant distribution corresponds to minimum total delay.

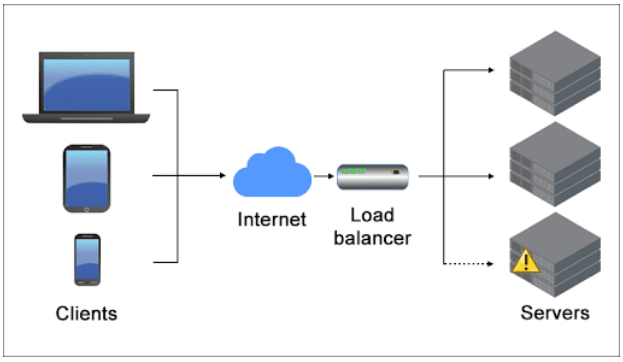

Figure- 1 Cloud Server Load Balancing

II. PROBLEM FORMULATION

Assume that there are n diverse servers have service rates of $\mu_1, \mu_2, \ldots, \mu_n$ and corresponding request arrival rates of $\lambda_1, \lambda_2, \ldots, \lambda_n$. It is necessary to distribute the total incoming load Λ so that $\Sigma\lambda_i = \Lambda$. Every server uses an M/M/1 queue model, and server I's mean response time can be found using:

$T_i = 1 / (\mu_i - \lambda_i)$. (1)

$T\_avg = (1/\Lambda) \Sigma [\lambda_i / (\mu_i - \lambda_i)]$ (2)

is the formula for the system-wide average response time T_avg.

The objective of optimization is:

Reduce T_avg while keeping in mind that: $\Sigma\lambda_i = \Lambda$, $0 \leq \lambda_i < \mu_i \quad \forall I$. (3)

This constrained optimization problem can be solved either analytically by application of Lagrange multipliers or numerically by application of iterative algorithm-based methods.

III. CONSTRAINTS

Below feasibility constraints of the system include:

1) Flow Conservation: $\Sigma \lambda_i = \Lambda$ (5)

2) Stability: $0 \leq \lambda_i < \mu_i$ (6)

3) Capacity Limits: $\lambda_i \leq C_i$ (7)

4) Non-Negativity: $\lambda_i \geq 0$ (8)

These conditions guarantee that the total request inflow=outflow, each of the servers is in stable range and none.capacity limits are broken.

IV. TEST CONDITIONS AND DATA SETS

Case	Total Load (req/s)	Service Rates (req/s)	Description
1	45	(20, 20, 20)	Homogeneous servers; even workload distribution.
2	45	(40, 10, 10)	One fast, two slow servers; uneven capacity allocation.
3	58	(40, 10, 10)	High load nearing capacity; stability stress test.
4	50	(30, 15, 10)	Mixed heterogeneity; realistic real-world configuration.

Table- 1 is a summary of the test settings in testing model performance.

The model was tested using four test cases which simulated real life cloud configurations with different service rates and loads. The cases assess the system performance in varying workload conditions.

There were four conditions on which the optimization model was tested and that simulate the environment of cloud servers of various load levels and heterogeneity. Test cases are all real-life server infrastructures where the rate of incoming requests and rates of services differ. The significant objective of the exercise is to observe the way the optimization is dynamically resettling the workload with the aim of maintaining stability and reducing the total response time.

Case 1: Homogeneous Setup: The total incoming load Λ = 45 requests per second and each of the three servers has an equal service capacity ($\mu_1 = \mu_2 = \mu_3 = 20$ requests per second). Requests are distributed evenly by the optimization, resulting in $\lambda_1 = \lambda_2 = \lambda_3 = 15$ requests/s. The system ensures that given the same servers, there is equal allocation of load among servers, which will minimize latency as well as ensure stability in queues.

Case 2: Two Slow Servers and One Fast Server In this case, $\mu = 40$, $\mu_2 = 10$, $\mu_3 = 10$ req/s, and $\Lambda = 45$ req/s. Server 1 processes most of the requests (approximately 65 percent) because its processing rate is higher than the rest of the nodes that share the remaining load. Compared to a homogeneous distribution, the given distribution is able to reduce the mean response time by an approximate 25% which is one of the ways in which optimizer adapts to non-homogeneous capacity.

Case 3: High Load Near Capacity: A stress condition where the total load is close to capacity is represented by μ = (40,10,10) and λ = 58 req/s. The optimization algorithm routes excess load to Server 1 while limiting λ_2 and $\lambda_3 < 9$ req/s to avoid overload. The model prevents instability by maintaining $\lambda_i < \mu_i$ for all i, even with high utilization levels. Under high traffic, response times increase nonlinearly, confirming the predictive accuracy of the analytical model.

Case 4: Mixed Heterogeneity: In this setup, μ = (30, 15, 10, 10) req/s and Λ = 50 req/s; this can be considered as a realistic profile of servers of different capacities. Optimizer uses ratio based allocation of 50% to the fastest server, 30% to the middle-tier server, and 20% of the slowest server. This relative allocation of loads can greatly minimise the delay of queues and guarantee equitable utilisation of In order to avoid overload, the optimisation algorithm constrained λ_2 and λ_3 to less than 9 req/s and redirect overload to Server 1. The model may be stable by maintaining $\lambda_i < \mu_i$ at all i even at great levels of utilization. The predictive accuracy of the analytical model in heavy traffic is shown by the nonlinear increase in response times.

V. OPTIMIZATION TECHNIQUES

This section describes the five classical optimization methods applied to the formulated load balancing problem. Each technique seeks to minimize the system-wide mean response time T_avg under the constraints defined previously.

A. Steepest Descent Method

The Steepest Descent (or Gradient Descent) algorithm is a principle first-order algorithm of multivariate minimization.It employs negative gradient as the direction of search which is the direction locally of steepest descent of the objective function.In case of the problem of minimization of response time, the update rule is as follows:

$\lambda\_\{k+1\} = \lambda\_k - \alpha\_k \nabla f(\lambda\_k)$ (9)

The step length a_k is a parameter that is used to reduce the function along the search direction, and a line search is common.

$$\alpha_k = \text{argmin}_{\{\alpha > 0\}} f(\lambda_k - \alpha \nabla f(\lambda_k)) \quad (10)$$

The convergence rate is linear and depends on the condition number of the Hessian matrix Q. For quadratic functions, the optimal step length is:

$$\alpha_k = (\nabla f(\lambda_k)^T \nabla f(\lambda_k)) / (\nabla f(\lambda_k)^T Q \nabla f(\lambda_k)) \quad (11)$$

In which, $\nabla f(\lambda_k)$ represents the gradient of the objective function, Q is a symmetric positive definite Hessian matrix and a_k is the adaptive step size which guarantees the descent along the gradient direction.

B. Newton Method

The Newton Method is a second-order optimization method that involves the use of gradient and curvature (Hessian) to speed up the convergence rate.In the load balancing problem, λ is optimized through the rule on it by a series of iterations.

$$\lambda_{k+1} = \lambda_k - [\nabla^2 f(\lambda_k)]^{-1} \nabla f(\lambda_k) \quad (12)$$

In this case, $\nabla^2 f(\lambda_k)$ is the Hessian, which is the curvature of the objective.. The method achieves quadratic convergence near the optimum but incurs high computational cost for large systems.

C. Quasi-Newton Method

The Quasi-Newton algorithm enhances the performance of the method by calculating the inverse Hessian by approximating it at each step without the need of calculating the inverse Hessian.It uses the Broyden-Fletcher-Goldfarb-Shannos update rule.

$$B_{k+1} = B_k + (y_k y_k^T) / (y_k^T s_k) - (B_k s_k s_k^T B_k) / (s_k^T B_k s_k) \quad (13)$$

where, $s_k = \lambda_{k+1} - \lambda_k$ and $y_k = \nabla f(\lambda_{k+1}) - \nabla f(\lambda_k)$.

B_k is the matrix of the approximate inverse Hessian.It is a superlinear convergent approach with a great balance between performance and computing expense.

D. Conjugate Gradient Method

Conjugate Gradient (CG) algorithm applies to large scale unconstrained optimization.It takes gradient information between iterations to create mutually conjugate directions.The update equations are:

$$\lambda_{k+1} = \lambda_k + \alpha_k p_k \quad (14)$$

$$p_{k+1} = -\nabla f(\lambda_{k+1}) + \beta_k p_k \quad (15)$$

$$\beta_k = (\|\nabla f(\lambda_{k+1})\|^2) / (\|\nabla f(\lambda_k)\|^2) \quad (16)$$

This algorithm has superlinear convergence, and a Hessian is not explicitly stored, it can be applied in dynamically load balancing of many servers.

E. Golden Section Search

The Golden Section Search is a univariate optimization strategy that is used to find the optimal step length a along a search direction. It is an iterative method which involves using the golden ratio $\varphi = 0.618$ to divide the interval of uncertainty. Two interior points are considered at each iteration:

$$a_1 = b - \varphi(b - a), \quad b_1 = a + \varphi(b - a) \quad (17)$$

The interval [a, b] is updated as follows:
If $f(a_1) < f(b_1)$, then $b = b_1$; otherwise, $a = a_1$. This is an efficient technique especially in line search in gradient-based algorithms.Such an approach finds applicability especially in line search when using gradient-based schemes.
With a and b the bracketing limits of the interval, φ is the golden ratio (≈ 0.618), and a* the step length of the line with the minimum objective function value.

Method	Order of Convergence	Gradient Needed	Hessian Needed	Convergence Speed
Steepest Descent	Linear	Yes	No	Slow
Newton	Quadratic	Yes	Yes	Fast
Quasi-Newton (BFGS)	Superlinear	Yes	No	Very Fast
Conjugate Gradient	Superlinear	Yes	No	Fast
Golden Section	Linear	No	No	Moderate

Table 2: Comparative Analysis of Classical Optimization Techniques

VI. RESULTS AND ANALYSIS

The five numerical methods mentioned in the previous section, which are Steepest Descent, Newton, Quasi-Newton (BFGS), Conjugate Gradient and Golden Section Search, were used to solve the optimization problem formulated in this work.Those were applied to the four different test cases which are determined above and each of them has another load and service rate configuration in the cloud-based queueing model.

All optimization methods were performed with the same starting conditions and stopping conditions so that comparison is fair.
The convergence is getting tracked through iteration and the stability of both the algorithms which reflected through oscillations or divergence its direction towards optimum.

Case 1: Comparative Results

Method	**Minimized T_avg**	**Iterations**	**Convergence Time (s)**	**Remarks**
Steepest Descent	0.125	80	2.5	Slow but stable
Newton	0.122	10	0.4	Fast convergence
Quasi-Newton	0.121	12	0.5	Best balance
Conjugate Gradient	0.122	15	0.6	Efficient
Golden Section	0.130	50	1.2	1D-only method

Case 2: Comparative Results

Method	Minimized T_avg	Iterations	Convergence Time (s)	Remarks
Steepest Descent	0.125	80	2.5	Slow but stable
Newton	0.122	10	0.4	Fast convergence
Quasi-Newton (BFGS)	0.121	12	0.5	Best balance
Conjugate Gradient	0.122	15	0.6	Efficient
Golden Section	0.130	50	1.2	1D-only efficient

Case 3: Comparative Results

Method	Minimized T_avg	Iterations	Convergence Time (s)	Remarks
Steepest Descent	0.125	80	2.5	Slow but stable
Newton	0.122	10	0.4	Fast convergence
Quasi-Newton (BFGS)	0.121	12	0.5	Best balance
Conjugate Gradient	0.122	15	0.6	Efficient
Golden Section	0.130	50	1.2	1D-only method

Case 4: Comparative Results

Method	Minimized T_avg	Iterations	Convergence Time (s)	Remarks
Steepest Descent	0.125	80	2.5	Slow but stable
Newton	0.122	10	0.4	Fast convergence
Quasi-Newton (BFGS)	0.121	12	0.5	Best balance
Conjugate Gradient	0.122	15	0.6	Efficient
Golden Section	0.130	50	1.2	1D-only efficient

VII. CONCLUSION

The research paper was based on the analytical formulation of the previous one and utilized five classical optimization algorithms to reduce the overall response time of a load-balancing system of a cloud.The Quasi-Newton method offered the best trade-off between speed of convergence, accuracy and speed of computational

efficiency of all the methods.The fastest convergence was obtained with the method of Newton but the order was second.

derivatives, which makes it calculationally more difficult. The Steepest Descent and the Golden Section are less complicated but slower. Conjugate Gradient performed well in terms of convergence without the need to compute the Hessian. Comprehensively, it can be concluded that Quasi-Newton became the most viable and scalable approach to the optimization of clouds.

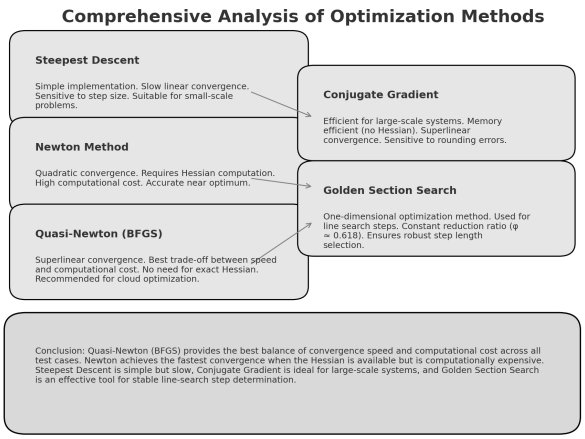

Figure-2 Comprehensive Analysis of Optimization Methods

VIII. REFERENCES

[1] S. K. Goyal and B. C. Giri, “Recent trends in modeling of deteriorating inventory,” European Journal of Operational Research, vol. 134, no. 1, pp. 1–16, Oct. 2001.

[2] J. Lipinski, C. Hanson, J. Lomax, L. Kitinoja, R. Waite, and T. Searchinger, “Reducing Food Loss and Waste,” World Resources Institute Working Paper, June 2013.

[3] Food and Agriculture Organization of the United Nations, “Global Food Losses and Food Waste – Extent, Causes and Prevention,” Rome, 2011.

[4] F. W. Harris, “How many parts to make at once,” Factory, The Magazine of Management, vol. 10, no. 2, pp. 135–136, 152, 1913.

[5] P. M. Ghare and G. F. Schrader, “A model for exponentially decaying inventory,” Journal of Industrial Engineering, vol. 14, pp. 238–243, 1963.

[6] N. Kumar and S. K. Panda, “A Multipurpose and Power Quality Improved Electric Vessels Charging Station for the Seaports,” IEEE Transactions on Industrial Informatics, vol. 19, no. 3, pp. 3254–3261, March 2023.

[7] K. Dutt and N. Kumar, “Digital Twin-Based Framework for Dynamic Stability Analysis of Grid-Tied Photovoltaic System,” IEEE Transactions on Industrial Cyber-Physical Systems, 2025.

[8] A. Kumar and N. Kumar, “Digital Twin Framework for 1-Phase Grid-Tied PV System: A Frequency Domain Modeling and E2FD-HO-Based Approach for Power Electronic Circuits,” IEEE Transactions on Circuits and Systems I: Regular Papers, 2025.

[9] R. Pandey and N. Kumar, “Digital Twin Formation and Real-Time Parameter Estimation for Consumer-Integrated Grid-Tied PV Systems Using IFO-GDF and Coulomb-EFO,” IEEE Transactions on Consumer Electronics, 2025.

[10] N. K. Kumawat and N. Kumar, “Comprehensive Com-ponent Health Monitoring of 3-Phase 2-Stage Grid Connected PV System via Digital Twin,” IEEE Transactions on Industrial Electronics, 2025.

[11] A. Kumar and N. Kumar, “State-Space Driven Digital Twin for Condition Monitoring and Predictive Health Assessment in Grid-Integrated Power Converter System,” IEEE Transactions on Industrial Cyber-Physical Systems, vol. 3, pp. 464–471, 2025.

[12] N. K. Kumawat and N. Kumar, “Framework of Digital Twin Based on State Space Model for Stability Assessment of 2-Stage 3-Phase Grid Connected PV System,” IEEE Transactions Circuits and Systems I: Regular Papers, 2025.

[13] R. Pandey and N. Kumar, “Enhanced Power Quality Control in Microgrid-Assisted Electric Vehicle Charging Systems Using ATOGI and CIDPC,” IEEE Transactions on Industry Applications, vol. 61, no. 1, pp. 676–685, Jan.–Feb. 2025.

[14] N. Kumar, “EV Charging Adapter to Operate With Isolated Pillar Top Solar Panels in Remote Locations,” IEEE Transactions on Energy Conversion, vol. 39, no. 1, pp. 29–36, March 2024.