Analysis of Steady-State Behavior in Server Queues using Markov Chains and Eigenvalues in the M/M/1 Model

Nicholas Wise Saragih Sumbayak - 13524037
Department of Informatics Engineering
School of Electrical Engineering and Informatics
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
nicholasaragih@gmail.com — 13524037@std.stei.itb.ac.id

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert.**

I. Introduction

Queueing behavior arises naturally in nearly every computing environment where resources are shared among multiple tasks. Whenever incoming work arrives faster than it can be immediately processed, the excess work must wait, forming a queue. This phenomenon appears in a broad range of systems, including CPU scheduling, packet forwarding in routers, job dispatching in cloud infrastructures, disk I/O scheduling, and networked application servers. The performance of these systems is heavily influenced by their queueing characteristics, making analytical models essential for understanding and improving real-world performance [1].

Modern computing workloads are highly variable and unpredictable. Task arrivals do not occur at fixed intervals, and service times fluctuate due to user behavior, network delays, resource contention, and software-level scheduling. These uncertainties make deterministic approaches nonoptimal. However, probabilistic theory enables engineers to determine whether a server will remain stable under a particular load, estimate average waiting times, and understand how performance degrades as traffic increases.

To formally analyze such behavior, queueing systems are commonly modeled as stochastic processes, with one of the simplest and most fundamental being the M/M/1 queue. In this model, arrivals follow a Poisson process with service times that follow an exponential distribution, all handled by a single server. Despite its simplicity, the model can capture trends and predict behaviors such as stability, queue buildup, and performance spikes. [2].

The M/M/1 queue can be naturally represented as a Markov Chain, where each state corresponds to the current number of jobs in the system. Transitions between states reflect arrivals and completions of jobs, producing a stochastic matrix representing the dynamic behavior of the queue. The steady-state distribution corresponds to the eigenvector associated with eigenvalue 1, consistent with standard results in Markov Chain theory [3]. This connection between queueing systems and linear algebra provides a powerful framework for analyzing server load and predicting performance under varying traffic conditions.

In this paper, we examine the steady-state behavior of the M/M/1 queue using Markov Chains and eigenvalue analysis. We first derive the transition matrix of the system and compute its stationary distribution using the eigenvector method. We then compare this theoretical distribution with the known closed-form solution of the M/M/1 queue. To validate the results, a queue simulation is implemented in C++, allowing us to observe empirical queue behavior under different arrival and service rates. By comparing theoretical predictions with simulation outcomes, we illustrate how mathematical modeling can determine whether a server can withstand a particular load and how queue lengths evolve over time.

II. Theoretical Foundation

A. Matrices

A matrix is a rectangular array of numbers, symbols, or expressions, arranged in rows and columns. The individual items in a matrix are called its elements or entries. The size of a matrix is described in terms of the number of rows and columns it contains. Generally, a general $m \times n$ matrix may be denoted as:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

with A denoting the matrix, m the number of rows, n the number of columns, and a_{ij} the element in row i and column j. A matrix with size $n \times n$ is called a square matrix of order n and the elements a_{ii} (where the row and column indices are equal) form the main diagonal of matrix A.

1) Row and Column Vectors: A matrix with only one row or one column is called a row matrix (or a row vector) or column matrix (or a column vector), respectively. A general $m \times 1$ column matrix and a $1 \times n$ row matrix b may be denoted as:

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix}$$

2) Matrix Addition and Subtraction: If A and B are matrices of the same size, their sum (A + B) and difference (A - B) are obtained by adding or subtracting their corresponding entries. Matrices of different sizes cannot be added or subtracted. In matrix notation, if $A = [a_{ij}]$ and $B = [b_{ij}]$ have the same size, then $(A \pm B)_{ij} = a_{ij} \pm b_{ij}$.

3) Scalar Multiplication: If A is any matrix and k is any scalar, then the scalar multiple kA is the matrix obtained by multiplying every entry of A by k. In matrix notation, if $A = [a_{ij}]$, then $(kA)_{ij} = ka_{ij}$.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \rightarrow 2A = \begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \end{bmatrix}$$

4) Matrix Multiplication: If A is an $m \times n$ matrix and B is an $n \times p$ matrix, then the product AB is defined to be the $m \times p$ matrix C whose entries are given by:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

In simpler terms, to find entry c_{ij} of the product matrix C, multiply the corresponding entries of the i^{th} row of matrix A with the j^{th} column of matrix B and add the results. Given the example below:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 0 \end{bmatrix}, B = \begin{bmatrix} 2 & 1 \\ 4 & 3 \\ 5 & 0 \end{bmatrix}$$

$$C = AB = \begin{bmatrix} 25 & 7 \\ 22 & 15 \end{bmatrix}$$

Since A is a 2×3 matrix and B is a 3×2 matrix, the resulting product AB is a 2×2 matrix. For example, to determine entry c_{11} of the product matrix AB, we multiply the corresponding entries of the first row of matrix A with the first column of matrix B and add the results.

5) Transpose of a Matrix: Given a matrix A of size $m \times n$, the transpose of A, denoted by A^T, is the $n \times m$ matrix obtained by interchanging the rows and columns of A. In matrix notation, if $B = A^T$, then the entries of B are defined as $b_{ij} = a_{ji}$, $1 \leq i \leq n$, $1 \leq j \leq m$.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \rightarrow A^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

B. Eigenvalues and Eigenvectors

If A is an $n \times n$ matrix, then a nonzero vector $\mathbf{v}$ in R^n is called an eigenvector of A if there exists a scalar λ such that:

$$A\mathbf{v} = \lambda\mathbf{v}$$

The scalar λ is called the eigenvalue of A corresponding to the eigenvector $\mathbf{v}$. In other words, multiplying the matrix A by the vector $\mathbf{v}$ results in a new vector that is a scalar multiple of the original vector $\mathbf{v}$.

Given a matrix A with size $n \times n$, the eigenvalues and eigenvectors are found by solving the following characteristic equation:

$$\begin{gathered} Ax = \lambda x \\ IAx = \lambda Ix \\ Ax = \lambda Ix \\ (A - \lambda I)x = 0 \end{gathered}$$

Since $x = 0$ is the only trivial solution, for $(A - \lambda I)x = 0$ to have non-trivial solutions, the matrix $(A - \lambda I)$ must be singular, and therefore $det(A - \lambda I)$ must be zero. The polynomial given by $det(A - \lambda I) = 0$ is called the characteristic equation of A, and the solutions to such equation are the eigenvalues of A, otherwise denoted as the characteristic roots.

Geometrically, an eigenvector of a matrix A represents a direction in R^n that is preserved under the linear transformation defined by A. While most vectors are both rotated and scaled by a linear transformation, eigenvectors are only scaled by the corresponding eigenvalue. If $|\lambda| > 1$, the transformation stretches vectors in the direction of the eigenvector, if $|\lambda| < 1$, it contracts them, and if $\lambda < 0$, it reverses their direction.

C. Markov Chains

A Markov Chain is a mathematical system used to model systems that transition between different states over time. The defining characteristic of a Markov Chain is that the probability of transitioning to the next state depends only on the current state, and not on the sequence of states that preceded it. This property is known as the Markov property.

A Markov Chain is commonly represented using a matrix formed by transition probabilities between states. This matrix is called the transition matrix, and the state vectors at successive time intervals are defined by

$$x(n+1) = Px(n),$$

where $x(n)$ is a probability vector describing the state distribution at time n, and P_{ij} is the probability that the system will be in state i at time $n+1$ given that it was in state j at time n [4].

The transition matrix P is a stochastic matrix, meaning that all of its entries are nonnegative and that the sum of each column is equal to one. These properties ensure that multiplying a probability vector by P produces another valid probability vector.

As an example, consider a system with three states. A possible transition matrix for this system is

$$P = \begin{bmatrix} 0.6 & 0.2 & 0.1 \\ 0.3 & 0.5 & 0.2 \\ 0.1 & 0.3 & 0.7 \end{bmatrix}.$$

Each column of this matrix sums to one, and each entry represents the probability of transitioning from one state to another in a single time step.

From a linear algebra perspective, the long-term behavior of a Markov Chain is determined by the eigenvalues and eigenvectors of its transition matrix. In particular, a steady-state distribution is a probability vector x that remains unchanged under the transition matrix, satisfying

$$Px = x.$$

This corresponds to an eigenvector associated with the eigenvalue $\lambda = 1$.

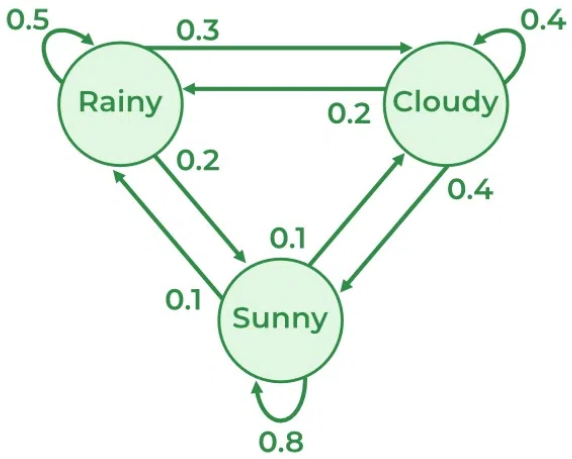

Fig. 1. Example of a Markov Chain representing weather states [6].

III. Methodology

A. The M/M/1 Queue Model

The $M/M/1$ queue is one of the simplest stochastic models used to represent server-based systems. The notation $M/M/1$ indicates that arrivals follow a (Markovian) Poisson process (M), service times are exponentially distributed (M), and the system consists of a single server (1).

Let λ denote the average arrival rate and μ the average service rate. At any time, the state of the system is defined as the number of jobs currently present in the system, including the job being served. Thus, the system is considered stable if $\lambda < \mu$ and if otherwise, the queue will grow indefintely long over time.

Expanding over such, the utilization of the queue's buffer can be written as $\rho = \frac{\lambda}{\mu}$ and for it to be stable, it must hold that $\rho < 1$. The steady-state probability of having n jobs in the system is given by:

$$P_n = (1-\rho)\rho^n, \quad n = 0, 1, 2, \ldots$$

where P_n represents the probability of having n jobs in the system at steady state [2].

Fig. 2. An M/M/1 queueing node [7].

B. Markov Chain Representation

Building on the Markov Chain framework introduced in the theoretical foundation, the M/M/1 queue can be modeled as a discrete-state system in which each state represents the number of jobs currently present in the system. Let state n denote the condition where $n \geq 0$ jobs are either waiting or being served.

Transitions between states occur due to two possible events: the arrival of a new job or the completion of a service. An arrival causes the system to move from state n to $n+1$, while a service completion causes a transition from state n to $n-1$ when $n > 0$. These transitions form a birth–death process, a special class of Markov Chains commonly used to model queueing systems.

To enable matrix-based analysis, the infinite state space is truncated to a finite number of states $\{0, 1, \ldots, N\}$. This results in a finite stochastic transition matrix P whose structure directly reflects the arrival rate λ and service rate μ. The evolution of the queue state distribution can therefore be analyzed using the matrix formulation established earlier [4].

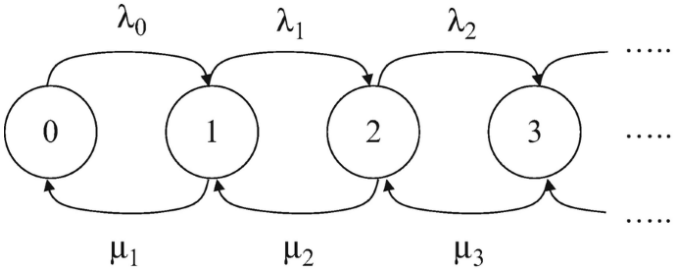

Fig. 3. A Markov Chain representation of a queue [9].

C. Steady-State Analysis via Eigenvalues and Eigenvectors

The primary objective of the Markov Chain representation is to determine the long-term behavior of the queue. In steady-state operation, the distribution of queue lengths no longer changes over time, indicating that the system has reached equilibrium.

From a linear algebra perspective, a steady-state distribution is a probability vector x that satisfies

$$Px = x,$$

meaning that x is an eigenvector of the transition matrix P corresponding to the eigenvalue $\lambda = 1$ [4]. Since P is

a stochastic matrix, the existence of such an eigenvalue is guaranteed.

For the M/M/1 queue, the existence of a valid steady-state eigenvector depends on the relationship between the arrival rate λ and the service rate μ. When $\lambda < \mu$, the system is stable and the eigenvector associated with $\lambda = 1$ represents the long-term probabilities of observing each queue length. This provides a direct linear algebraic criterion for evaluating whether a server can sustain a given workload.

D. C++ Queue Simulation

IV. Results and Discussion

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity "Magnetization", or "Magnetization, M", not just "M". If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write "Magnetization (A/m)" or "Magnetization {A[m(1)]}", not just "A/m". Do not label axes with a ratio of quantities and units. For example, write "Temperature (K)", not "Temperature/K".

Acknowledgment

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

References

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first ..."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

References

[1] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[2] L. Kleinrock, *Queueing Systems, Volume 1: Theory*. New York, NY, USA: Wiley, 1975.

[3] J. R. Norris, *Markov Chains*. Cambridge, U.K.: Cambridge Univ. Press, 1997.

[4] H. Anton and C. Rorres, *Elementary Linear Algebra: Applications Version*, 11th ed. Hoboken, NJ, USA: Wiley, 2013.

[5] Munir, Rinaldi. "Nilai Eigen dan Vektor Eigen (Bagian 1)", School of Electrical Engineering and Informatics (STEI) ITB, 2025. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2025-2026/Algeo-19-Nilai-Eigen-dan-Vektor-Eigen-Bagian1-2025.pdf

[6] "Markov Chains in NLP," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/nlp/markov-chains-in-nlp/. [Accessed: Dec. 15, 2025].

[7] "M/M/1 queue," Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/M/M/1_queue. [Accessed: Dec. 16, 2025].

[8] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[9] G. Giambene, "Markov chains and queuing theory," in *Queuing Theory and Telecommunications*, Textbooks in Telecommunication Engineering, Cham, Switzerland: Springer, 2021, doi: 10.1007/978-3-030-75973-5_4.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove the template text from your paper may result in your paper not being published.