The M/M/k Queue: A First-Principles Derivation

Abstract

This document presents a rigorous derivation of the M/M/k queueing system from first principles. We extend the M/M/1 analysis to multiple servers, carefully handling the state-dependent service rates that arise when the number of busy servers varies. We derive the steady-state distribution, the Erlang-C formula for waiting probability, and key performance metrics including expected queue length and waiting times.

Contents

1 Introduction: The Multi-Server Model

The M/M/k queue generalizes the single-server M/M/1 queue to a system with k identical parallel servers. This model captures scenarios such as call centers with multiple agents, bank branches with multiple tellers, or computing clusters with multiple processors.

1.1 Model Assumptions

We make the following assumptions:

1. **Arrivals:** Customers arrive according to a Poisson process with rate $\lambda > 0$.
2. **Service times:** Service times are exponentially distributed with rate $\mu > 0$ per server.
3. **Number of servers:** There are k identical servers operating in parallel.
4. **Queue discipline:** Customers who find all servers busy wait in a single infinite queue. Service is first-come, first-served (FCFS).
5. **Independence:** Servers work independently. Each server serves one customer at a time with exponential service time of rate μ.

Definition 1.1 (State Variable)**.** Let n denote the number of customers in the system (both in service and waiting). The state space is $n \in \{0, 1, 2, 3, \ldots\}$.

1.2 Key Insight: State-Dependent Service Rate

The crucial difference between M/M/1 and M/M/k lies in how the total service rate depends on the system state. In M/M/1, the service rate is simply μ whenever $n \geq 1$. In M/M/k, the total service completion rate depends on *how many servers are currently busy.*

- Each busy server completes service at rate μ
- Total departure rate = (number of busy servers) $\times \mu$

This leads to two distinct regimes based on whether customers are waiting.

2 The Two Regimes: Service Rate Analysis

2.1 Case 1: $n \leq k$ (No Waiting)

When there are n customers in the system and $n \leq k$:

- All n customers are being served (no one is waiting)
- Number of busy servers = n
- Number of idle servers = $k - n$
- Total service rate = $n\mu$

Key observation: Departures happen *faster* as n increases because more servers are active. More customers means more parallel service.

2.2 Case 2: $n > k$ (Queue Forms)

When there are n customers and $n > k$:

- All k servers are busy
- Number waiting in queue $= n - k$
- Total service rate $= k\mu$ (capped at maximum capacity)

The service rate is now *saturated*—adding more customers to the queue doesn't speed up departures.

2.3 Unified Service Rate

We can express the total service (death) rate as:

$$\mu_n = \begin{cases} n\mu & \text{if } n \leq k \\ k\mu & \text{if } n > k \end{cases} = \min(n, k) \cdot \mu$$

Remark 2.1 (Contrast with M/M/1). In M/M/1, there is exactly one server, so the service rate is always μ when $n \geq 1$. In M/M/k, the service rate *changes with* n until saturation at $k\mu$. This state-dependence is what makes the M/M/k analysis more intricate.

3 Stability Condition

3.1 Traffic Intensity

Definition 3.1 (Traffic Intensity for M/M/k)**.** The traffic intensity is defined as:

$$\rho = \frac{\lambda}{k\mu} = \frac{\text{arrival rate}}{\text{maximum service capacity}}$$

This represents the average fraction of time each server is busy.

Note: We can also write $\lambda/\mu = k\rho$, which will be useful in the derivations.

3.2 Stability Analysis

The parameter ρ compares how fast work arrives to how fast the system can process it at full capacity.

Case $\rho \geq 1$ (Unstable): When $\lambda \geq k\mu$, customers arrive as fast as or faster than they can be served. The consequences are:

- Accumulation never stops
- Queue length $\to \infty$
- Waiting time $\to \infty$
- No steady-state probabilities exist
- The queue explodes

Case $\rho < 1$ (Stable): When $\lambda < k\mu$, on average, the system can serve customers faster than they arrive:

- Queue does not grow without bound

- Steady-state probabilities π_n exist
- The system reaches equilibrium

$$\boxed{\textbf{Stability requirement: } \rho = \frac{\lambda}{k\mu} < 1}$$

We assume $\rho < 1$ throughout the remainder of this document.

4 Birth-Death Process Framework

4.1 Birth and Death Events

The M/M/k queue is a **birth-death process** where the state n (number of customers) can only change by ± 1:

- **Birth (arrival):** State goes from n to $n+1$. This occurs at rate λ for all states n.
- **Death (service completion):** State goes from n to $n-1$. This occurs at rate $\mu_n = \min(n,k)\mu$.

In an infinitesimally small time interval:

- At most one arrival can happen (Poisson process property)
- At most one service completion can happen (exponential service property)

4.2 Birth and Death Rates Summary

State	Birth Rate	Death Rate
$n = 0$	λ	0
$1 \le n \le k$	λ	$n\mu$
$n > k$	λ	$k\mu$

5 Deriving the Balance Equations

5.1 Steady-State and Flow Balance

Let $\pi_n = P(N = n)$ denote the steady-state probability of having n customers in the system. In steady state, for each state, the rate of probability flow in equals the rate of probability flow out.

Rate of transition from state i to state j = (arrival or service rate) $\times$ (probability of being in state i) = (rate) $\times \pi_i$.

5.2 State $n = 0$

$$\begin{aligned} \text{Flow in:} &\quad \text{Service completion from state } 1 = \mu\pi_1 \\ \text{Flow out:} &\quad \text{Arrival to state } 1 = \lambda\pi_0 \end{aligned}$$

Balance equation:

$$\boxed{\mu\pi_1 = \lambda\pi_0}$$

5.3 State $1 \le n \le k-1$

Service rate in this regime is $n\mu$.

$$\begin{aligned} \text{Flow in:} \quad & \text{Arrival from } n-1 : \lambda\pi_{n-1} \\ & \text{Service from } n+1 : (n+1)\mu\pi_{n+1} \\ \text{Flow out:} \quad & \text{Arrival to } n+1 : \lambda\pi_n \\ & \text{Service to } n-1 : n\mu\pi_n \end{aligned}$$

Balance equation:

$$\boxed{\lambda\pi_{n-1} + (n+1)\mu\pi_{n+1} = (\lambda + n\mu)\pi_n, \quad 1 \le n \le k-1}$$

5.4 State $n \ge k$

Service rate is capped at $k\mu$.

$$\begin{aligned} \text{Flow in:} \quad & \text{Arrival from } n-1 : \lambda\pi_{n-1} \\ & \text{Service from } n+1 : k\mu\pi_{n+1} \\ \text{Flow out:} \quad & \text{Arrival to } n+1 : \lambda\pi_n \\ & \text{Service to } n-1 : k\mu\pi_n \end{aligned}$$

Balance equation:

$$\boxed{\lambda\pi_{n-1} + k\mu\pi_{n+1} = (\lambda + k\mu)\pi_n, \quad n \ge k}$$

6 Solving the Balance Equations

For birth-death processes, there is a powerful shortcut: **local balance**. Instead of solving the global balance equations directly, we use the fact that in steady state, the flow between any two adjacent states must balance.

6.1 Local Balance Principle

Consider transitions between states $n-1$ and n:

$$\begin{aligned} \text{Rate } n-1 \to n &= \lambda\pi_{n-1} \\ \text{Rate } n \to n-1 &= \mu_n\pi_n \end{aligned}$$

Local balance gives us:

$$\lambda\pi_{n-1} = \mu_n\pi_n \implies \pi_n = \frac{\lambda}{\mu_n}\pi_{n-1}$$

6.2 Case 1: $n \le k$

For $n \le k$, we have $\mu_n = n\mu$, so:

$$\lambda\pi_{n-1} = n\mu\pi_n \implies \pi_n = \frac{\lambda}{n\mu}\pi_{n-1}$$

Iterating from π_0:

$$\pi_1 = \frac{\lambda}{\mu}\pi_0$$

$$\pi_2 = \frac{\lambda}{2\mu}\pi_1 = \frac{\lambda}{2\mu}\cdot\frac{\lambda}{\mu}\pi_0 = \frac{\lambda^2}{2!\mu^2}\pi_0$$

$$\pi_3 = \frac{\lambda}{3\mu}\pi_2 = \frac{\lambda^3}{3!\mu^3}\pi_0$$

The pattern is clear:

$$\boxed{\pi_n = \frac{\lambda^n}{n!\mu^n}\pi_0 = \frac{(\lambda/\mu)^n}{n!}\pi_0 = \frac{(k\rho)^n}{n!}\pi_0, \quad n \le k}$$

Remark 6.1. This has the form of a **Poisson distribution truncated at** $n = k$. The factorial in the denominator arises because the service rate increases with n.

6.3 Case 2: $n > k$

For $n > k$, we have $\mu_n = k\mu$, so:

$$\lambda\pi_{n-1} = k\mu\pi_n \implies \pi_n = \frac{\lambda}{k\mu}\pi_{n-1} = \rho\pi_{n-1}$$

This is now a simple geometric recursion with ratio ρ!
Starting from π_k:

$$\pi_{k+1} = \rho\pi_k$$
$$\pi_{k+2} = \rho\pi_{k+1} = \rho^2\pi_k$$

In general, for $n > k$, let $m = n - k$ (number of customers waiting):

$$\pi_{k+m} = \rho^m\pi_k$$

Or equivalently:

$$\pi_n = \rho^{n-k}\pi_k, \quad n > k$$

Substituting $\pi_k = \frac{(k\rho)^k}{k!}\pi_0$:

$$\boxed{\pi_n = \frac{k^k\rho^n}{k!}\pi_0 = \frac{(k\rho)^k}{k!}\rho^{n-k}\pi_0, \quad n > k}$$

7 Finding π_0: Normalization

The probabilities must sum to 1:

$$\sum_{n=0}^{\infty}\pi_n = 1$$

We split this into two parts:

7.1 Sum for $n = 0$ to $k - 1$ (Not All Servers Busy)

$$\sum_{n=0}^{k-1}\pi_n = \sum_{n=0}^{k-1}\frac{(k\rho)^n}{n!}\pi_0 = \pi_0\sum_{n=0}^{k-1}\frac{(k\rho)^n}{n!}$$

This is a finite sum (partial sum of the exponential series).

7.2 Sum for $n \geq k$ (All Servers Busy, Queue Forms)

$$\sum_{n=k}^{\infty} \pi_n = \sum_{n=k}^{\infty} \frac{k^k \rho^n}{k!} \pi_0 = \frac{k^k}{k!} \pi_0 \sum_{n=k}^{\infty} \rho^n$$

For the infinite sum, let $m = n - k$, so $n = k + m$ and as n goes from k to ∞, m goes from 0 to ∞:

$$\sum_{n=k}^{\infty} \rho^n = \sum_{m=0}^{\infty} \rho^{k+m} = \rho^k \sum_{m=0}^{\infty} \rho^m = \rho^k \cdot \frac{1}{1-\rho}$$

(This geometric series converges because $|\rho| < 1$.)
Therefore:

$$\sum_{n=k}^{\infty} \pi_n = \frac{k^k}{k!} \pi_0 \cdot \frac{\rho^k}{1-\rho} = \frac{(k\rho)^k}{k!(1-\rho)} \pi_0$$

7.3 Final Expression for π_0

Combining both parts:

$$\pi_0 \left[\sum_{n=0}^{k-1} \frac{(k\rho)^n}{n!} + \frac{(k\rho)^k}{k!(1-\rho)} \right] = 1$$

Theorem 7.1 (Probability System is Empty)**.**

$$\boxed{\pi_0 = \left[\sum_{n=0}^{k-1} \frac{(k\rho)^n}{n!} + \frac{(k\rho)^k}{k!(1-\rho)} \right]^{-1}}$$

Remark 7.1. The expression inside the brackets has two interpretable parts:

- **Finite sum:** Contribution from states where not all servers are busy (no waiting)
- **Geometric tail:** Contribution from states where all servers are busy and a queue forms

8 The Erlang-C Formula

A key quantity in M/M/k analysis is the probability that an arriving customer must wait (i.e., all servers are busy).

8.1 Probability of Waiting

An arriving customer must wait if and only if $n \geq k$ (all servers are busy). This probability is:

$$P(\text{wait}) = P(N \geq k) = \sum_{n=k}^{\infty} \pi_n$$

From our earlier calculation:

$$\sum_{n=k}^{\infty} \pi_n = \frac{(k\rho)^k}{k!(1-\rho)} \pi_0$$

Definition 8.1 (Erlang-C Formula)**.** The **Erlang-C formula** gives the probability that an arriving customer finds all servers busy and must wait:

$$C(k, \lambda/\mu) = P(\text{wait}) = \frac{\dfrac{(k\rho)^k}{k!} \cdot \dfrac{1}{1-\rho}}{\displaystyle\sum_{n=0}^{k-1} \frac{(k\rho)^n}{n!} + \frac{(k\rho)^k}{k!} \cdot \frac{1}{1-\rho}}$$

This can be written more compactly as:

$$\boxed{P(\text{wait}) = \frac{(k\rho)^k}{k!(1-\rho)} \cdot \pi_0}$$

Remark 8.1. The Erlang-C formula is fundamental in telecommunications and service operations. It answers the question: "What fraction of customers will experience a delay?"

9 Performance Metrics

9.1 Expected Queue Length $E[L_q]$

The queue length L_q is the number of customers *waiting* (not including those in service):

$$L_q = \begin{cases} 0 & \text{if } n \leq k \\ n-k & \text{if } n > k \end{cases}$$

The expected queue length is:

$$E[L_q] = \sum_{n=k}^{\infty} (n-k)\pi_n$$

Substituting $\pi_n = \frac{k^k \rho^n}{k!}\pi_0$ for $n \geq k$:

$$E[L_q] = \frac{k^k \pi_0}{k!} \sum_{n=k}^{\infty} (n-k)\rho^n$$

Let $m = n - k$, so $n = m + k$:

$$\sum_{n=k}^{\infty} (n-k)\rho^n = \sum_{m=0}^{\infty} m\rho^{m+k} = \rho^k \sum_{m=0}^{\infty} m\rho^m$$

Using the standard result $\sum_{m=0}^{\infty} mx^m = \frac{x}{(1-x)^2}$ for $|x| < 1$:

$$\sum_{m=0}^{\infty} m\rho^m = \frac{\rho}{(1-\rho)^2}$$

Therefore:

$$E[L_q] = \frac{k^k \pi_0}{k!} \cdot \rho^k \cdot \frac{\rho}{(1-\rho)^2} = \frac{(k\rho)^k}{k!} \cdot \frac{\rho}{(1-\rho)^2} \cdot \pi_0$$

Recognizing that $\frac{(k\rho)^k}{k!(1-\rho)}\pi_0 = P(\text{wait})$:

Theorem 9.1 (Expected Queue Length)**.**

$$\boxed{E[L_q] = \frac{\rho}{1-\rho} \cdot P(wait)}$$

Remark 9.1. This elegant formula shows that the expected queue length is the product of:

- $P(\text{wait})$: The probability that a queue exists (all servers busy)
- $\frac{\rho}{1-\rho}$: The expected queue length *given* that all servers are busy

9.2 Expected Waiting Time in Queue $E[W_q]$

By **Little's Law** applied to the queue:

$$E[L_q] = \lambda \cdot E[W_q]$$

Solving for expected waiting time:

$$E[W_q] = \frac{E[L_q]}{\lambda} = \frac{P(\text{wait})}{\lambda} \cdot \frac{\rho}{1-\rho}$$

Since $\rho = \frac{\lambda}{k\mu}$, we have $\lambda = \rho k\mu$:

$$E[W_q] = \frac{P(\text{wait})}{\rho k\mu} \cdot \frac{\rho}{1-\rho} = \frac{P(\text{wait})}{k\mu(1-\rho)}$$

Noting that $k\mu - \lambda = k\mu(1-\rho)$:

Theorem 9.2 (Expected Waiting Time in Queue)**.**

$$\boxed{E[W_q] = \frac{P(wait)}{k\mu - \lambda}}$$

9.3 Expected Total Time in System $E[W]$

The total time in the system is the sum of waiting time and service time:

$$W = W_q + S$$

where S is the service time with $E[S] = 1/\mu$.
Taking expectations (using linearity):

$$E[W] = E[W_q] + E[S] = E[W_q] + \frac{1}{\mu}$$

Theorem 9.3 (Expected Time in System)**.**

$$\boxed{E[W] = \frac{P(wait)}{k\mu - \lambda} + \frac{1}{\mu}}$$

10 Sanity Checks and Limiting Cases

10.1 Case 1: $k = 1$ (Reduces to M/M/1)

When $k = 1$, we have a single server and $\rho = \lambda/\mu$.
Checking P(wait):
For $k = 1$:

$$P(\text{wait}) = \frac{(1 \cdot \rho)^1}{1!(1-\rho)} \cdot \pi_0 = \frac{\rho}{1-\rho} \cdot \pi_0$$

The normalization gives:

$$\pi_0 = \left[\frac{(1\cdot\rho)^0}{0!} + \frac{\rho}{1-\rho}\right]^{-1} = \left[1 + \frac{\rho}{1-\rho}\right]^{-1} = \left[\frac{1}{1-\rho}\right]^{-1} = 1-\rho$$

(Note: the sum $\sum_{n=0}^{k-1}$ for $k = 1$ is just the $n = 0$ term.)

Therefore:

$$P(\text{wait}) = \frac{\rho}{1-\rho} \cdot (1-\rho) = \rho$$

This matches M/M/1: the probability of waiting equals the server utilization ρ.

Checking $E[W_q]$:

$$E[W_q] = \frac{P(\text{wait})}{k\mu - \lambda} = \frac{\rho}{\mu - \lambda}$$

For M/M/1, we know $E[W_q] = \frac{\lambda}{\mu(\mu-\lambda)} = \frac{\rho}{\mu-\lambda}$. ✓

Checking $E[W]$:

$$E[W] = \frac{\rho}{\mu - \lambda} + \frac{1}{\mu} = \frac{\rho + (\mu-\lambda)/\mu}{\mu - \lambda} = \frac{\rho\mu + \mu - \lambda}{\mu(\mu - \lambda)}$$

Since $\rho = \lambda/\mu$:

$$E[W] = \frac{\lambda + \mu - \lambda}{\mu(\mu - \lambda)} = \frac{1}{\mu - \lambda}$$

This matches the M/M/1 result. ✓

10.2 Case 2: $\rho \to 0$ (Light Traffic)

When $\rho \to 0$, arrivals are rare relative to service capacity ($\lambda \to 0$).

- **P(wait) $\to 0$:** Servers are almost always idle, so arriving customers rarely wait.
- **$E[W_q] = \frac{P(\textbf{wait})}{\lambda} \cdot \frac{\rho}{1-\rho} \to 0$:** No waiting time.
- **$E[W] \to \frac{1}{\mu}$:** Customers experience only service time, no waiting.

This makes intuitive sense: in a lightly loaded system, customers go directly into service.

10.3 Case 3: $\rho \to 1^-$ (Heavy Traffic)

When $\rho \to 1^-$, the arrival rate approaches the maximum service capacity ($\lambda \to k\mu$).

- **P(wait) $\to 1$:** Almost every arriving customer finds all servers busy.
- **$E[W_q] = \frac{P(\textbf{wait})}{k\mu-\lambda} \to \frac{1}{0^+} = \infty$:** Waiting times explode.
- **$E[W] \to \infty$:** Total time in system diverges.

Remark 10.1. Even though the system is technically stable when $\rho < 1$, operating close to capacity leads to:

- The queue cannot be cleared because there is no spare capacity
- Queue grows without bound as $\rho \to 1$
- Heavy traffic means operating very close to capacity
- As utilization approaches 100%, waiting time blows up to ∞

11 Summary of Key Results

For an M/M/k queue with arrival rate λ, service rate μ per server, and k servers, where $\rho = \lambda/(k\mu) < 1$:

Quantity	**Formula**
Steady-state probability ($n \leq k$)	$\pi_n = \frac{(k\rho)^n}{n!}\pi_0$
Steady-state probability ($n > k$)	$\pi_n = \frac{k^k \rho^n}{k!}\pi_0$
Probability system is empty	$\pi_0 = \left[\sum_{n=0}^{k-1} \frac{(k\rho)^n}{n!} + \frac{(k\rho)^k}{k!(1-\rho)}\right]^{-1}$
Probability of waiting (Erlang-C)	$P(\text{wait}) = \frac{(k\rho)^k}{k!(1-\rho)}\pi_0$
Expected queue length	$E[L_q] = \frac{\rho}{1-\rho} \cdot P(\text{wait})$
Expected waiting time in queue	$E[W_q] = \frac{P(\text{wait})}{k\mu - \lambda}$
Expected time in system	$E[W] = \frac{P(\text{wait})}{k\mu - \lambda} + \frac{1}{\mu}$

These formulas form the foundation for analyzing multi-server queueing systems and are widely used in operations research, telecommunications, and service system design.