Lesson 16. The Birth-Death Process – Performance Measures

1 Overview

- A **birth-death process** is a Markov process with state space $\mathcal{M} = \{0, 1, 2, \dots\}$ with generator matrix

$$\mathbf{G} = \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & 0 & \cdots \\ \mu_1 & -(\lambda_1 + \mu_1) & \lambda_1 & 0 & \cdots \\ 0 & \mu_2 & -(\lambda_2 + \mu_2) & \lambda_2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

- Last lesson: modeling queueing systems as birth-death processes
- This lesson:
  - How to compute steady-state probabilities
  - How to use the steady-state probabilities to compute long-run performance measures

2 Steady-state probabilities of a birth-death process

- Recall the steady-state equations:

$$\pi^\top \mathbf{G} = \mathbf{0}$$
$$\pi^\top \mathbf{1} = 1$$

- Rewrite the steady-state equations as scalar equations:

- Solve recursively in terms of π_0:

- Define

$$d_0 = 1 \qquad d_j = \frac{\lambda_0 \lambda_1 \cdots \lambda_{j-1}}{\mu_1 \mu_2 \cdots \mu_j} \quad \text{for } j = 1, 2, \ldots \qquad D = \sum_{i=0}^{\infty} d_i$$

- Simplify and solve for π_0:

- Finally, solve for all the steady-state probabilities $\pi_0, \pi_1, \pi_2, \ldots$:

- Note: the steady-state probabilities exist provided that $0 < \sum_{j=0}^{\infty} d_j < \infty$

Example 1. [slightly modified from Lesson 16] A small ice-cream shop competes with several other ice-cream shops in a busy mall. If there are too many customers already in line at the shop, then potential customers will go elsewhere. Potential customers arrive at a rate of 20 per hour. The probability that a customer will go elsewhere is $n/3$ when there are $n \leq 3$ customers already in the system, and 1 when there are $n > 3$ customers already in the system. The server at the shop can serve customers at a rate of 10 per hour. Approximate the process of potential arrivals as Poisson, and the service times as exponentially distributed.

We can model this queueing system as a birth-death process:

$$\text{Arrival rates:} \quad \lambda_i = \begin{cases} 20\left(1 - \frac{i}{3}\right) & \text{if } i = 0, 1, \ldots, 3 \\ 0 & \text{if } i = 4, 5, \ldots \end{cases} \qquad \text{Service rates:} \quad \mu_i = 10 \quad \text{if } i = 0, 1, 2, \ldots$$

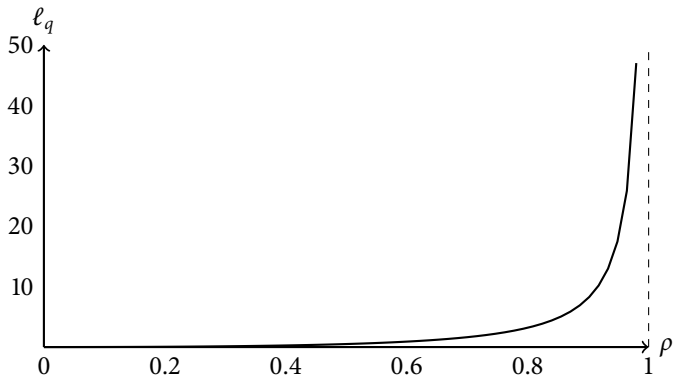
a. Over the long run, what is the probability there are n customers in the shop? ($n = 0, 1, 2, 3$)
b. Over the long run, what fraction of the time is the server busy?

3 System-level performance measures

- How do we measure the workload or congestion in the entire system?
- **Expected number of customers in the system** ℓ:
- **Expected number of customers in the queue** ℓ_q:

  where s is the number of customers that can be served simultaneously
  - Note that when there are $j = 0, 1, \ldots, s$ customers in the system, there are 0 customers in the queue
- **Traffic intensity** ρ:

  where λ is the arrival rate in all states, and μ is the service rate of s identical servers
  - $\rho < 1 \Rightarrow$ the system is **stable**: customers can be served faster than they are arriving
    - ρ = fraction of time each server is busy, a.k.a. the long-run **utilization** of each server
  - $\rho \geq 1 \Rightarrow$ customers are arriving faster than they can be served
- **Offered load** o, or the expected number of busy servers:
- Relationship between ρ and ℓ_q: ℓ_q explodes as ρ approaches 1



  - This example: Poisson arrivals with rate λ, one server with constant service rate μ: $\ell_q = \dfrac{\rho^2}{1-\rho}$

Example 2. In Example 1, how many customers are waiting to be served (not including the customer in service)?

4 Customer-level performance measures

- **Effective arrival rate** to the system λ_{eff}, or the "average arrival rate" =

- **Little's law** (system-wide):

  where w is the **expected waiting time**: the expected time a customer spends in the system from arrival to departure

  - Deep result, difficult to prove rigorously
  - Intuitively, why does this hold?
    - ◇ Average departure rate for an individual customer ≈
    - ⇒ Average departure rate for the whole system ≈
    - ◇ System is stable ⇒ arrival rate = departure rate ("conservation of customers"):

- **Little's law** (queue only):

  where w_q is the **expected delay**: the expected time a customer spends in the queue

- If the service rate is a constant μ, then waiting time and delay are related like so:

Example 3. Consider the setting described in Example 1.

a. What is the effective arrival rate?
b. What is the expected customer delay?
c. What is the expected customer waiting time?

5 Exercises

Problem 1. Consider the setting described in Example 1.

a. Over the long run, at what rate are customers lost?
b. Suppose that the shop makes a revenue of \$2 per customer served and pays the server \$4 per hour. What is the shop's long-run expected profit per hour?

Problem 2. Turingtown has a small urgent care center. Due to the recent increase in patients visiting the center, Turingtown is considering expanding the center. You have been asked to evaluate the current configuration.

The center has two doctors. You have estimated that the time a doctor spends with a patient is exponentially distributed with a mean of 30 minutes. Based on your observations and interviews, you have also estimated that patients arrive outside the center according to a Poisson process with a rate of 2 per hour. However, a patient will only enter the center if there is at most 1 patient already waiting (in other words, at most three patients total in the center); otherwise, the patient will opt to go to the hospital emergency room in nearby Gaussville.

a. Model this setting as a birth-death process by defining
    - the state space and what each state means,
    - the arrival rate in each state, and
    - the service rate in each state.
b. Find the steady-state distribution of the number of patients in the urgent care center.
c. What is the long-run expected number of patients waiting for a doctor?
d. What is the long-run expected time a patient waits for a doctor?
e. What fraction of the arriving patients opt to go to Gaussville instead?