Module 3: State-Based Methods
Availability: A Motivation for State-Based Methods

- Recall that *availability* quantifies the alternation between proper and improper service.
  - $A(t)$ is 1 if service is proper, 0 otherwise.
  - $E[A(t)]$ is the probability that service is proper at time $t$.
  - $A(0,t)$ is the fraction of time the system delivers proper service during $[0,t]$.

- For many systems, availability is a more “user-oriented” measure than reliability.

- However, it is often more difficult to compute, since it must account for repair and/or replacement.
Availability Example

• A radio transmitter has an expected time to failure of 500 days. Replacement takes an average 48 hours.

• A cheaper and less reliable transmitter has an expected time to failure of 150 days, but due to the cost, a replacement is readily available and replacement can be done in 8 hours on average.

• For $t \to \infty$, $A(t) = .996$ for the more reliable transmitter, and $A(t) = .998$ for the less reliable transmitter.

• Higher reliability does not necessarily mean higher availability!

• But how do we compute these numbers??
Availability Modeling using Combinatorial Methods

- Availability modeling can be done with combinatorial methods, but only with the independent repair assumption, and exponential life-time distributions.
- This uses the theory of ON/OFF processes.

\[ \text{\text{\textit{On}}} \text{ time distribution is reliability distribution, obtained using combinatorial methods, say mean is } E[\text{\textit{On}}] \]

\[ \text{\text{\textit{Off}}} \text{ distribution is repair time distribution, say mean is } E[\text{\textit{Off}}] \]

Availability is the fraction of time in the On state.

Asymptotically, if instances of On periods are independent and identically distributed (i.i.d.) and instances of Off periods are i.i.d., then

\[ \Pr\{\text{in On state}\} = \frac{E[\text{\textit{On}}]}{E[\text{\textit{On}}]+E[\text{\textit{Off}}]} \]
State-Based Methods

• More accurate modeling with state-based methods relaxes the independence assumptions needed for combinatorial modeling
  
  – Failures need not be independent. Failure of one component may make another component more or less likely to fail.
  
  – Repairs need not be independent. Repair and replacement strategies are an important component that must be modeled in high-availability systems.
  
  – High-availability systems may operate in a degraded mode. In a degraded mode, the system may deliver only a fraction of its services, and the repair process may start only after the system is sufficiently degraded.

• We use random processes to model these systems.
• We use “state” to “remember” the conditions leading to dependencies
Random Processes

Random processes are useful for characterizing the behavior of real systems.

A random process is a collection of random variables indexed by time.

Example: $X(t)$ is a random process. Let $X(1)$ be the result of tossing a die. Let $X(2)$ be the result of tossing a die plus $X(1)$, and so on. Notice that time ($T$) = \{1,2,3, ...\}.

One can ask:

$P[X(2) = 12] = \frac{1}{36}$

$P[X(3) = 14 | X(1) = 2] = \frac{1}{36}$

$E[X(n)] = 3.5n$
Random Processes, cont.

If $X$ is a random process, $X(t)$ is a random variable.

Remember that a random variable $Y$ is a function that maps elements in “sample space” $\Omega$ to numbers in $\mathbb{R}$.

Therefore, a random process $X$ maps elements in the two-dimensional space $\Omega \times T$ to elements in $\mathbb{R}$.

A sample path of $X$ is the history of sample space values $X$ adopts as a function of time.

- When we fix $t$, then $X$ becomes a function of $\Omega$ to $\mathbb{R}$.
- However, if we fix $\omega$, then $X$ becomes a function of $T$ to $\mathbb{R}$.
- By fixing $\omega$ (e.g., the system is available) and observing $X$ as a function of $T$, we see a trajectory of the process sampling $\omega$ or not.
Describing a Random Process

Recall that for a random variable \( X \), we can use the cumulative distribution \( F_X \) to describe the random variable.

In general, no such simple description exists for a random process.

However, a random process can often be described succinctly in various different ways. For example, if \( Y \) is a random variable representing the roll of a die, and \( X(t) \) is the sum after \( t \) rolls, then we can describe \( X(t) \) by

\[
X(t) - X(t - 1) = Y,
\]

\[
P[X(t) = i | X(t - 1) = j] = P[Y = i - j],
\]

or \( X(t) = Y_1 + Y_2 + \ldots + Y_t \), where the \( Y_i \)’s are independent.
Classifying Random Processes: Characteristics of $T$

If the number of time points defined for a random process, i.e., $|T|$, is finite or countable (e.g., integers), then the random process is said to be a *discrete-time random process*.

If $|T|$ is uncountable (e.g., real numbers) then the random process is said to be a *continuous-time random process*.

Example: Let $X(t)$ be the number of fault arrivals in a system up to time $t$. Since $t \in T$ is a real number, $X(t)$ is a continuous-time random process.
**Classifying Random Processes: State Space Type**

Let $X$ be a random process. The *state space* of a random process is the set of all possible values that the process can take on, i.e.,

$$S = \{y: X(t) = y, \text{ for some } t \in T\}.$$

If $X$ is a random process that models a system, then the state space of $X$ can represent the set of all possible configurations that the system could be in.
Random Process State Spaces

If the state space $S$ of a random process $X$ is finite or countable (e.g., $S = \{1,2,3, \ldots\}$), then $X$ is said to be a *discrete-state random process*.

Example: Let $X$ be a random process that represents the number of bad packets received over a network. $X$ is a discrete-state random process.

If the state space $S$ of a random process $X$ is infinite and uncountable (e.g., $S = \mathbb{R}$), then $X$ is said to be a *continuous-state random process*.

Example: Let $X$ be a random process that represents the voltage on a telephone line. $X$ is a continuous-state random process.

We examine only discrete-state processes in this lecture.
## Stochastic-Process Classification Examples

<table>
<thead>
<tr>
<th>Time</th>
<th>State</th>
<th>Continuous</th>
<th>Discrete</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Continuous</td>
<td>Analog signal</td>
<td>A to D converter</td>
</tr>
<tr>
<td></td>
<td>Discrete</td>
<td>Computer availability model</td>
<td>round-based network protocol model</td>
</tr>
</tbody>
</table>
A special type of random process that we will examine in detail is called the **Markov process**. A Markov process can be informally defined as follows.

Given the state (value) of a Markov process $X$ at time $t$ ($X(t)$), the future behavior of $X$ can be described completely in terms of $X(t)$.

Markov processes have the very useful property that their future behavior is independent of past values.
Markov Chains

A *Markov chain* is a Markov process with a discrete state space.

We will always make the assumption that a Markov chain has a state space in \( \{1, 2, \ldots\} \) and that it is time-homogeneous.

A Markov chain is *time-homogeneous* if its future behavior does not depend on what time it is, only on the current state (i.e., the current value).

We make this concrete by looking at a *discrete-time Markov chain* (hereafter *DTMC*). A DTMC \( X \) has the following property:

\[
P[X(t + k) = j | X(t) = i, X(t - 1) = n_{t-1}, X(t - 2) = n_{t-2}, \ldots, X(O) = n_O] = P^{(k)}_{ij}
\]  

(1)

(2)
Notice that given $i$, $j$, and $k$, $P_{ij}^{(k)}$ is a number!

$P_{ij}^{(k)}$ can be interpreted as the probability that if $X$ has value $i$, then after $k$ time-steps, $X$ will have value $j$.

Frequently, we write $P_{ij}$ to mean $P_{ij}^{(1)}$. 
State Occupancy Probability Vector

Let $\pi$ be a row vector. We denote $\pi_i$ to be the $i$-th element of the vector. If $\pi$ is a state occupancy probability vector, then $\pi_i(k)$ is the probability that a DTMC has value $i$ (or is in state $i$) at time-step $k$.

Assume that a DTMC $X$ has a state-space size of $n$, i.e., $S = \{1, 2, \ldots, n\}$. We say formally

$$\pi_i(k) = P[X(k) = i]$$

Note that $\sum_{i=1}^{n} \pi_i(k) = 1$ for all times $k$. 
Computing State Occupancy Vectors: A Single Step Forward in Time

If we are given $\pi(0)$ (the initial probability vector), and $P_{ij}$ for $i, j = 1, \ldots, n$, how do we compute $\pi(1)$?

Recall the definition of $P_{ij}$.

$$P_{ij} = P[X(k+1) = j \mid X(k) = i] = P[X(1) = j \mid X(0) = i]$$

Since $\sum_{i=1}^{n} \pi_i(0) = 1$,

$$\pi_j(1) = P[X(1) = j] = P[X(1) = j \mid X(0) = 1]P[X(0) = 1] + \ldots + P[X(1) = j \mid X(0) = n]P[X(0) = n]$$

$$= \sum_{i=1}^{n} P[X(1) = j \mid X(0) = i]P[X(0) = i]$$

$$= \sum_{i=1}^{n} P_{ij} \pi_i(0)$$

$$= \sum_{i=1}^{n} \pi_i(0) P_{ij}$$
Transition Probability Matrix

We have \( \pi_j(1) = \sum_{i=1}^{n} \pi_i(0)P_{ij} \), which holds for all \( j \).

Notice that this resembles vector-matrix multiplication.

In fact, if we arrange the matrix \( P = \{P_{ij}\} \), that is, if

\[
P = \begin{bmatrix}
p_{11} & \cdots & p_{1n} \\
\vdots & \ddots & \vdots \\
p_{n1} & \cdots & p_{nn}
\end{bmatrix},
\]

then \( p_{ij} = P_{ij} \), and \( \pi(1) = \pi(0)P \), where \( \pi(0) \) and \( \pi(1) \) are row vectors, and \( \pi(0)P \) is a vector-matrix multiplication.

The important consequence of this is that we can easily specify a DTMC in terms of an occupancy probability vector \( \pi \) and a transition probability matrix \( P \).
Transient Behavior of Discrete-Time Markov Chains

Given $\pi(0)$ and $P$, how can we compute $\pi(k)$?

We can generalize from earlier that

$$\pi(k) = \pi(k - 1)P.$$  

Also, we can write $\pi(k - 1) = \pi(k - 2)P$, and so

$$\pi(k) = [\pi(k - 2)P]P$$

$$= \pi(k - 2)P^2$$

Similarly, $\pi(k - 2) = \pi(k - 3)P$, and so

$$\pi(k) = [\pi(k - 3)P]P^2$$

$$= \pi(k - 3)P^3$$

By repeating this, it should be easy to see that

$$\pi(k) = \pi(0)P^k$$
A Simple Example

Suppose the weather at Urbana-Champaign, Illinois can be modeled the following way:

- If it’s sunny today, there’s a 60% chance of being sunny tomorrow, a 30% chance of being cloudy, and a 10% chance of being rainy.
- If it’s cloudy today, there’s a 40% chance of being sunny tomorrow, a 45% chance of being cloudy, and a 15% chance of being rainy.
- If it’s rainy today, there’s a 15% chance of being sunny tomorrow, a 60% chance of being cloudy, and a 25% chance of being rainy.

If it’s rainy on Friday, what is the forecast for Monday?
Simple Example, cont.

Clearly, the weather model is a DTMC.

1) Future behavior depends on the current state only
2) Discrete time, discrete state
3) Time homogeneous

The DTMC has 3 states. Let us assign 1 to sunny, 2 to cloudy, and 3 to rainy. Let time 0 be Friday.

\[ \pi(0) = (0,0,1) \]

\[ P = \begin{pmatrix}
.6 & .3 & .1 \\
.4 & .45 & .15 \\
.15 & .6 & .25 
\end{pmatrix} \]
Simple Example Solution

The weather on Saturday $\pi(1)$ is

$$\pi(1) = \pi(0)P = (0,0,1) \begin{pmatrix} .6 & .3 & .1 \\ .4 & .45 & .15 \\ .15 & .6 & .25 \end{pmatrix} = (.15,.6,.25),$$

that is, 15% chance sunny, 60% chance cloudy, 25% chance rainy.

The weather on Sunday $\pi(2)$ is

$$\pi(2) = \pi(1)P = (.15,.6,.25) \begin{pmatrix} .6 & .3 & .1 \\ .4 & .45 & .15 \\ .15 & .6 & .25 \end{pmatrix} = (.3675,.465,.1675).$$

The weather on Monday $\pi(3)$ is

$$\pi(3) = \pi(2)P = (.4316, .42, .1484),$$

that is, 43% chance sunny, 42% chance cloudy, and 15% chance rainy.
Solution, cont.

Alternatively, we could compute $P^3$ since we found

$$\pi(3) = \pi(0)P^3.$$  

Working out solutions by hand can be tedious and error-prone, especially for “larger” models (i.e., models with many states). Software packages are used extensively for this sort of analysis.

Software packages compute $\pi(k)$ by $\ldots ((\pi(0)P)P)\ldots P$ rather than computing $P^k$, since computing the latter results in a large “fill-in.”
Graphical Representation

It is frequently useful to represent the DTMC as a directed graph. Nodes represent states, and edges are labeled with probabilities. For example, our weather prediction model would look like this:

1 = Sunny Day
2 = Cloudy Day
3 = Rainy Day
"Simple Computer" Example

\[
P = \begin{bmatrix}
P_{idle} & P_{arr} & P_{fi} \\
P_{com} & P_{busy} & P_{fb} \\
P_{r} & 0 & P_{ff}
\end{bmatrix}
\]

- \( X = 1 \) computer idle
- \( X = 2 \) computer working
- \( X = 3 \) computer failed
Limiting Behavior of DTMCs

It is sometimes useful to know the time-limiting behavior of a DTMC. This translates into the “long term,” where the system has settled into some steady-state behavior.

Formally, we are looking for \( \lim_{n \to \infty} \pi(n) \).

To compute this, what we want is \( \lim_{n \to \infty} \pi(0)P^n \).

There are various ways to compute this. The simplest is to calculate \( \pi(n) \) for increasingly large \( n \), and when \( \pi(n + 1) \equiv \pi(n) \), we can believe that \( \pi(n) \) is a good approximation to steady-state. This can be rather inefficient if \( n \) needs to be large.
Classifications

It is much easier to solve for the steady-state behavior of some DTMC’s than others. To determine if a DTMC is “easy” to solve, we need to introduce some definitions.

Definition: A state $j$ is said to be accessible from state $i$ if there exists an $n \geq 0$ such that $P_{ij}^{(n)} > 0$. We write $i \rightarrow j$.

Note: recall that $P_{ij}^{(n)} = P[X(n) = j | X(0) = i]$

If one thinks of accessibility in terms of the graphical representation, a state $j$ is accessible from state $i$ if there exists a path of non-zero edges (arcs) from node $i$ to node $j$. 
**State Classification in DTMCs**

Definition: A DTMC is said to be *irreducible* if every state is accessible from every other state.

Formally, a DTMC is irreducible if

\[ i \rightarrow j \text{ for all } i,j \in S. \]

A DTMC is said to be *reducible* if it is not irreducible.

It turns out that irreducible DTMC’s are simpler to solve. One need only solve one linear equation:

\[ \pi = \pi P. \]

We will see why this is so, but first there is one more issue we must confront.
Periodicity

Consider the following DTMC:

\[
\begin{align*}
&\begin{array}{c}
1 \\
\end{array} & \xrightarrow{1} & \begin{array}{c}
2 \\
\end{array} \\
\end{align*}
\]

\[\pi(0) = (1, 0)\]

Does \(\lim_{n \to \infty} \pi(n)\) exist? No!

However, \(\lim_{n \to \infty} \frac{\sum \pi(n)}{n}\) does exist; it is called the time-averaged steady-state distribution, and is denoted by \(\pi^*\).

Definition: A state \(i\) is said to be periodic with period \(d\) if it can return to itself after \(n\) transitions only when \(n\) is some multiple of \(d > 1\), \(d\) the smallest integer for which this is true. If \(d = 1\), then \(i\) is said to be aperiodic.

Formally, \(\exists d > 1, j \neq kd \Rightarrow P^{(j)}_{ii} = 0\)

All states in the same strongly connected component have the same periodicity.

An irreducible DTMC is said to be periodic if all its states are periodic.

A steady-state solution for an irreducible DTMC exists if all the states are aperiodic.

A time-averaged steady-state solution for an irreducible DTMC always exists.
Transient and Recurrent States

- The probability of *first return* to state \( s \) after exactly \( n \) epochs is:

\[
f_s(n) = \Pr\{ X(n) = s, X(n-1) \neq s, \ldots, X(1) \neq s \mid X(0) = s \}
\]

- not to be confused with \( p_{s,s}(n) = P^n(s, s) \)
- relationship:

\[
p_{s,s}(n) = \sum_{k=1}^{n} f_s(k) \cdot p_{s,s}(n-k) \quad \text{for } n \geq 1
\]

- Probability to *eventually return* to \( s \): \( f_s = \sum_{n=1}^{\infty} f_s(n) \)

- state \( s \) is called *transient* if \( f_s < 1 \)

\[\Rightarrow\] there is a non-zero probability that the DTMC will not return to \( s \)

- state \( s \) is called *recurrent* if \( f_s = 1 \)

\[\Rightarrow\] it is impossible to never come back to a recurrent state
Mean Recurrence Time

• For recurrent state $s$, the mean number of epochs between two successive visits to $s$:

$$m_s = \sum_{n=1}^{\infty} n \cdot f_s(n)$$

  – state $s$ is called positive recurrent if $m_s < \infty$
  – state $s$ is called null recurrent if $m_s = \infty$

• State $s$ is **ergodic** if $s$ is aperiodic and positive recurrent
  – a DTMC is called ergodic when all its states are ergodic

• Fact: in a **finite**, aperiodic and irreducible DTMC all states are ergodic
  – there are also infinite DTMCs that are ergodic
Connected States and Type

Let \( s \) and \( s' \) be mutually reachable from each other. Then:

\[
\begin{align*}
\text{\( s \) is transient} & \quad \text{iff} \quad \text{\( s' \) is transient} \\
\text{\( s \) is null-recurrent} & \quad \text{iff} \quad \text{\( s' \) is null-recurrent} \\
\text{\( s \) is positive recurrent} & \quad \text{iff} \quad \text{\( s' \) is positive recurrent} \\
\text{\( s \) has period} d & \quad \text{iff} \quad \text{\( s' \) has period} d
\end{align*}
\]
Examples on Board

Which of these are (a) irreducible? periodic?
Steady-State Solution of DTMCs

The steady-state behavior can be computed by solving the linear equation \( \pi = \pi P \), with the constraint that \( \sum_{i=1}^{n} \pi_i = 1 \). For irreducible DTMC’s, it can be shown that this solution is unique. If the DTMC is periodic, then this solution yields \( \pi^* \).

One can understand the equation \( \pi = \pi P \) in two different ways.

- In steady-state, the probability distribution \( \pi(n + 1) = \pi(n)P \), and by definition \( \pi(n + 1) = \pi(n) \) in steady-state.
- “Flow” equations.

Flow equations require some visualization. Imagine a DTMC graph, where the nodes are assigned the occupancy probability, or the probability that the DTMC has the value of the node.
Flow Equations

Let $\pi_i P_{ij}$ be the "probability mass" that moves from state $j$ to state $i$ in one time-step. Since probability must be conserved, the probability mass entering a state must equal the probability mass leaving a state.

Probability must be conserved, i.e.,

$$\sum \pi_i = 1.$$

Written in matrix form, $\pi = \pi P$. 

$$\sum_{j=1}^{n} \pi_j P_{ji} = \sum_{j=1}^{n} \pi_i P_{ij} = \pi_i \sum_{j=1}^{n} P_{ij} = \pi_i$$
Continuous Time Markov Chains (CTMCs)

For most systems of interest, events may occur at any point in time. This leads us to consider continuous time Markov chains. A *continuous time Markov chain (CTMC)* has the following property:

\[
P[X(t + \tau) = j | X(t) = i, X(t - t_1) = k_1, X(t - t_2) = k_2, \ldots, X(t - t_n) = k_n] = P[X(t + \tau) = j | X(t) = i] = P_{ij}(\tau)
\]

for all \( \tau > 0, 0 < t_1 < t_2 < \ldots < t_n \)

A CTMC is completely described by the initial probability distribution \( \pi(0) \) and the transition probability matrix \( P(t) = [p_{ij}(t)] \). Then we can compute \( \pi(t) = \pi(0)P(t) \).

The problem is that \( p_{ij}(t) \) is generally very difficult to compute.
CTMC Properties

This definition of a CTMC is not very useful until we understand some of the properties.

First, notice that $p_{ij}(\tau)$ is independent of how long the CTMC has previously been in state $i$, that is,

$$P[X(t + \tau) = j | X(u) = i \text{ for } u \in [0, t]]$$

$$= P[X(t + \tau) = j | X(t) = i]$$

$$= p_{ij}(\tau)$$

There is only one random variable that has this property: the exponential random variable. This indicates that CTMCs have something to do with exponential random variables. First, we examine the exponential r.v. in some detail.
Exponential Random Variables

Recall the property of the exponential random variable. An exponential random variable $X$ with parameter $\lambda$ has the CDF

$$P[X \leq t] = F_X(t) = \begin{cases} 0 & t \leq 0 \\ 1 - e^{-\lambda t} & t > 0 \end{cases}.$$

The distribution function is given by $f_X(t) = \frac{d}{dt} F_X(t)$;

$$f_X(t) = \begin{cases} 0 & t \leq 0 \\ \lambda e^{-\lambda t} & t > 0 \end{cases}.$$  

The exponential random variable is the only random variable that is “memoryless.”

To see this, let $X$ be an exponential random variable representing the time that an event occurs (e.g., a fault arrival).

We will show that $P[X > t + s | X > s] = P[X > t]$. 
Memoryless Property

Proof of the memoryless property:

\[
P[X > t + s | X > s] = \frac{P[X > t + s, X > s]}{P[X > s]}
\]

\[
= \frac{P[X > t + s]}{P[X > s]}
\]

\[
= \frac{1 - F_X(t + s)}{1 - F_X(s)}
\]

\[
= \frac{e^{-\lambda(t+s)}}{e^{-\lambda s}}
\]

\[
= e^{-\lambda t} e^{-\lambda s}
\]

\[
= e^{-\lambda t}
\]

\[
= P[X > t]
\]
**Event Rate**

The fact that the exponential random variable has the memoryless property indicates that the “rate” at which events occur is constant, i.e., it does not change over time.

Often, the event associated with a random variable $X$ is a failure, so the “event rate” is often called the *failure rate* or the *hazard rate*.

The *event rate* of random variable $X$ is defined as the “time-averaged probability” that the event associated with $X$ occurs within the small interval $[t, t + \Delta t]$, given that the event has not occurred by time $t$, per the interval size $\Delta t$:

$$
\frac{P[t < X \leq t + \Delta t|X > t]}{\Delta t}.
$$

This can be thought of as looking at $X$ at time $t$, observing that the event has not occurred, and measuring the expected number of events (probability of the event) that occur per unit of time at time $t$. 
Observe that:

\[
P[t < X \leq t + \Delta t \mid X > t] = \frac{P[t < X \leq t + \Delta t, X > t]}{P[X > t] \cdot \Delta t}
\]

\[
= \frac{P[t < X \leq t + \Delta t]}{P[X > t] \cdot \Delta t}
\]

\[
= \frac{F_X(t + \Delta t) - F_X(t)}{(1 - F_X(t)) \Delta t}
\]

\[
= \frac{F_X(t + \Delta t) - F_X(t)}{\Delta t} \cdot \frac{1}{1 - F_X(t)}
\]

\[
= \frac{f_X(t)}{1 - F_X(t)} \quad \text{in general.}
\]

In the exponential case,

\[
\frac{f_X(t)}{1 - F_X(t)} = \frac{\lambda e^{-\lambda t}}{1 - (1 - e^{-\lambda t})} = \frac{\lambda e^{-\lambda t}}{e^{-\lambda t}} = \lambda.
\]

This is why we often say a random variable $X$ is “exponential with rate $\lambda$.”
Minimum of Two Independent Exponentials

Another interesting property of exponential random variables is that the minimum of two independent exponential random variables is also an exponential random variable.

Let $A$ and $B$ be independent exponential random variables with rates $\alpha$ and $\beta$ respectively. Let us define $X = \min\{A,B\}$. What is $F_X(t)$?

\[
F_X(t) = P[X \leq t] = P[\min\{A,B\} \leq t] = P[A \leq t \text{ OR } B \leq t] = 1 - P[A > t \text{ AND } B > t] - \text{ see lecture 2}
\]

\[
= 1 - P[A > t] P[B > t] = 1 - (1 - P[A \leq t])(1 - P[B \leq t]) = 1 - (1 - F_A(t))(1 - F_B(t))
\]

\[
= 1 - (1 - [1 - e^{-\alpha t]})(1 - [1 - e^{-\beta t}]) = 1 - e^{-\alpha t} e^{-\beta t} = 1 - e^{-(\alpha + \beta)t}
\]

Thus, $X$ is exponential with rate $\alpha + \beta$. 
Competition of Two Independent Exponentials

If $A$ and $B$ are independent and exponential with rate $\alpha$ and $\beta$ respectively, and $A$ and $B$ are competing, then we know that the time until one of them wins is exponentially distributed time (with rate $\alpha + \beta$). But what is the probability that $A$ wins?

$$P[A < B] = \int_0^\infty P[A < B | A = x]P[A = x]dx$$

$$= \int_0^\infty P[A < B | A = x]f_A(x)dx$$

$$= \int_0^\infty P[A < B | A = x]\alpha e^{-\alpha x} dx$$

$$= \int_0^\infty P[x < B]\alpha e^{-\alpha x} dx$$

$$= \int_0^\infty (1 - P[B \leq x])\alpha e^{-\alpha x} dx$$

$$= \int_0^\infty (1 - [1 - e^{-\beta x}])\alpha e^{-\alpha x} dx$$

$$= \int_0^\infty e^{-\beta x}\alpha e^{-\alpha x} dx$$

$$= \alpha \int_0^\infty e^{-\beta x} dx = \frac{\alpha}{\alpha + \beta}$$
Competing Exponentials in CTMCs

Imagine a random process $X$ with state space $S = \{1, 2, 3\}$. $X(0) = 1$. $X$ goes to state 2 (takes on a value of 2) with an exponentially distributed time with parameter $\alpha$. Independently, $X$ goes to state 3 with an exponentially distributed time with parameter $\beta$. These state transitions are like competing random variables.

We say that from state 1, $X$ goes to state 2 with rate $\alpha$ and to state 3 with rate $\beta$.

$X$ remains in state 1 for an exponentially distributed time with rate $\alpha + \beta$. This is called the holding time in state 1. Thus, the expected holding time in state 1 is $\frac{1}{\alpha + \beta}$.

The probability that $X$ goes to state 2 is $\frac{\alpha}{\alpha + \beta}$. The probability $X$ goes to state 3 is $\frac{\beta}{\alpha + \beta}$.

This is a simple continuous-time Markov chain.
Competing Exponentials vs. a Single Exponential With Choice

Consider the following two scenarios:

1. Event $A$ will occur after an exponentially distributed time with rate $\alpha$. Event $B$ will occur after an independent exponential time with rate $\beta$. One of these events occurs first.

2. After waiting an exponential time with rate $\alpha + \beta$, an event occurs. $A$ occurs with probability $\frac{\alpha}{\alpha + \beta}$, and event $B$ occurs with probability $\frac{\beta}{\alpha + \beta}$.

These two scenarios are indistinguishable. In fact, we frequently interchange the two scenarios rather freely when analyzing a system modeled as a CTMC.
State-Transition-Rate Matrix

A CTMC can be completely described by an initial distribution $\pi(0)$ and a state-transition-rate matrix. A 
\textit{state-transition-rate} matrix $Q = [q_{ij}]$ is defined as follows:

$$q_{ij} = \begin{cases} 
\text{rate of going from state } i \text{ to state } j & i \neq j, \\
-\sum_{k \neq i} q_{ik} & i = j.
\end{cases}$$

Example: A computer is idle, working, or failed. When the computer is idle, jobs arrive with rate $\alpha$, and they are completed with rate $\beta$. When the computer is working, it fails with rate $\lambda_w$, and with rate $\lambda_i$ when it is idle.
“Simple Computer” CTMC

Let $X = 1$ represent “the system is idle,” $X = 2$ “the system is working,” and $X = 3$ a failure.

\[
Q = \begin{bmatrix}
-(\alpha + \lambda_i) & \alpha & \lambda_i \\
\beta & -(\beta + \lambda_w) & \lambda_w \\
0 & 0 & 0 \\
\end{bmatrix}
\]

If the computer is repaired with rate $\mu$, the new CTMC looks like

\[
Q = \begin{bmatrix}
-(\alpha + \lambda_i) & \alpha & \lambda_i \\
\beta & -(\beta + \lambda_w) & \lambda_w \\
\mu & 0 & -\mu \\
\end{bmatrix}
\]
Analysis of “Simple Computer” Model

Some questions that this model can be used to answer:

– What is the availability at time $t$?
– What is the steady-state availability?
– What is the expected time to failure?
– What is the expected number of jobs lost due to failure in $[0,t]$?
– What is the expected number of jobs served before failure?
– What is the throughput of the system (jobs per unit time), taking into account failures and repairs?
Poisson Process

• $N(t)$ is said to be a “counting process” if $N(t)$ represents the total number of “events” that have occurred up to time $t$

• A counting process $N(t)$ is said to be a Poisson Process if
  – $N(0) = 0$
  – $\{N(t)\}$ has independent increments, e.g. for all $s < t < v$, $N(v)-N(t)$ is independent of $N(t)-N(s)$
  – The number of events in any interval of length $t$ has a Poisson distribution with mean $\lambda t$. $\lambda$ is the rate of the process

Equivalent statements

• $\{N(t)\}$ is a Poisson process if $N(0) = 0$, $\{N(t)\}$ has stationary and independent increments, and
  – $Pr\{N(t) \geq 2\} = o(t)$,
  – $Pr\{N(t) = 1\} = \lambda t + o(t)$
  (Function $f$ is $o(t)$ if limit as $t$ goes to 0 of $f(t)/t$ also goes to 0)

• $N(t)$ counts events when the distribution of time between events is exponential with rate $\lambda$
CTMC Transient Solution

We have seen that it is easy to specify a CTMC in terms of the initial probability distribution $\pi(0)$ and the state-transition-rate matrix.

Earlier, we saw that the transient solution of a CTMC is given by $\pi(t) = \pi(0)P(t)$, and we noted that $P(t)$ was difficult to define.

Due to the complexity of the math, we omit the derivation and show the relationship

$$\frac{d}{dt} P(t) = QP(t) = P(t)Q,$$

where $Q$ is the state transition rate matrix of the Markov chain.

Solving this differential equation in some form is difficult but necessary to compute a transient solution.
Transient Solution Techniques

Solutions to $\frac{d}{dt} P(t) = QP(t)$ can be done in many (dubious) ways*:

- Direct: If the CTMC has $N$ states, one can write $N^2$ PDEs with $N^2$ initial conditions and solve $N^2$ linear equations.
- Laplace transforms: Unstable with multiple “poles”
- Nth order differential equations: Uses determinants and hence is numerically unstable
- Matrix exponentiation: $P(t) = e^{Qt}$, where $e^{Qt} = I + \sum_{n=1}^{\infty} \frac{(Qt)^n}{n!}$.

Matrix exponentiation has some potential. Directly computing $e^{Qt}$ by performing $I + \sum_{n=1}^{\infty} \frac{(Qt)^n}{n!}$ can be expensive and prone to instability.

If the CTMC is irreducible, it is possible to take advantage of the fact that $Q = ADA^{-1}$, where $D$ is a diagonal matrix. Computing $e^{Qt}$ becomes $Ae^{Dt}A^{-1}$, where $e^{Dt} = \text{diag}(e^{d_{11}t}, e^{d_{22}t}, ..., e^{d_{nn}t})$.

Standard Uniformization

Starting with CTMC state transition rate matrix \((Q)\) construct

1. Poisson process: rate \(\lambda\), \(\lambda \geq |q(i,i)|\)

2. DTMC: \(P = I + \frac{Q}{\lambda}\)

Then:

\[
\pi(t) = \pi(0)P(t) = \pi(0) \sum_{k=0}^{\infty} \frac{(\lambda t)^k}{k!} e^{-\lambda t} P^k.
\]

In actual computation:

\[
\pi(t) = \sum_{k=0}^{N_s} \frac{(\lambda t)^k}{k!} e^{-\lambda t} \pi(k),
\]

with \(\pi(k + 1) = \pi(k)P\).

Probability of \(k\) transitions in time \(t\) (Poisson distribution)

\(k\)-step state transition probability

Choose truncation point to obtain desired accuracy

Compute \(\pi(k)\) iteratively, to avoid fill-in
Error Bound in Uniformization

• Answer computed is a lower bound, since each term in summation is positive, and summation is truncated.
• Number of iterations to achieve a desired accuracy bound can be computed easily.

\[
Error \text{ for each state } \leq 1 - \sum_{k=0}^{N_s} \frac{(\lambda t)^k}{k!} e^{-\lambda t}
\]

⇒ Choose error bound, then compute \(N_s\) on-the-fly, as uniformization is done.
A Simple Example

- Consider the simple CTMC below

1
\[Q = \begin{bmatrix} -1.4 & 1.4 \\ 1 & -1 \end{bmatrix}\]

- Note that with \( \lambda = 2 \), \( \lambda \) exceeds largest magnitude on diagonal

- Define

\[P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -1.4/2 & 1.4/2 \\ 1/2 & -1/2 \end{bmatrix} = \begin{bmatrix} .3 & .7 \\ .5 & .5 \end{bmatrix}\]

- Discrete time Markov chain
A Simple Example (continued)

CTMC

\[ Q = \begin{bmatrix} -1.4 & 1.4 \\ 1 & -1 \end{bmatrix} \]

- Discrete time Markov chain

\[ P = \begin{bmatrix} 0.3 & 0.7 \\ 0.5 & 0.5 \end{bmatrix} \]

- Make sense?
  - Look at sum of a geometric number of exponentials (geometric with parameter \( r \))
  - Result: exponential with rate \( r\lambda \).
  - DTMC steps in state 1 is geometric, \( p=0.7 \). Note \( \lambda=2 \) : CTMC rate from state 1 is \( 0.7*2 = 1.4 \). Ditto state 2.
  - Matches that for CTMC.
Uniformization cont.

Doing the infinite summation is of course impossible to do numerically, so one would stop after a sufficiently large number.

Uniformization has the following properties:

- Numerically stable
- Easy to calculate the error and stop summing when the error is sufficiently small
- Efficient
- Easy to program
- Error will decrease slowly if there are large differences (orders of magnitude) between rates.
Steady-State Behavior of CTMCs

Suppose we want to look at the long-term behavior of a CTMC.

A solution can be derived from the differential equation \( \frac{d}{dt} P(t) = P(t)Q \).

In steady state, \( \frac{d}{dt} P(t) = 0 \). To see this, recall

\[ \pi(t) = \pi(0)P(t). \]

Taking the limit as \( t \to \infty \),

\[ \lim_{t \to \infty} \pi(t) = \lim_{t \to \infty} \pi(0)P(t), \]

and differentiating

\[ \lim_{t \to \infty} \frac{d}{dt} \pi(t) = \lim_{t \to \infty} \frac{d}{dt} \pi(0)P(t) \]

so

\[ 0 = \lim_{t \to \infty} \pi(0) \frac{d}{dt} P(t) \]

\[ 0 = \frac{d}{dt} \pi(0) \lim P(t) = \pi(0) \lim \frac{d}{dt} P(t) = \lim \pi(0) P(t)Q = \lim \pi(t)Q = \pi^*Q \]
Steady-State Behavior of CTMCs via Flow Equations

Another way to arrive at the equation $\pi^* Q = 0$, where $\pi^* = \lim_{i \to \infty} \pi(t)$, is to use the flow equations. The global flow rate of transitions into a state $i$ must equal the global flow rate of transitions out of state $i$.

The global flow rate from state $i$ to state $j$ is simply $\pi_i q_{ij}$, which is the probability of being in state $i$ times the rate at which transitions from $i$ to $j$ take place.

Flow into state $i$:
$$\sum_{j=1}^{n} \pi_j q_{ji}$$

Flow out of state $i$:

Re-arrange and combine sums

(1) $\sum_{j=1, j\neq i}^{n} \pi_i q_{ij} = \pi_i \sum_{j=1, j\neq i}^{n} q_{ij} = \pi_i (-q_{ii})$

(2) $\sum_{j=1, j\neq i}^{n} \pi_j q_{ji} = \pi_i (-q_{ii})$

(3) $\sum_{j=1, j\neq i}^{n} \pi_j q_{ji} + \pi_i q_{ii} = 0$

(4) $\sum_{j=1, j\neq i}^{n} \pi_j q_{ji} = 0$

Flow in equals flow out

In matrix form, for all $i$, we get $\pi Q = 0$. 
Steady-State Behavior of CTMCs, cont.

This yields the elegant equation $\pi^* Q = 0$, where $\pi^* = \lim_{t \to \infty} \pi(t)$, the steady-state probability distribution. If the CTMC is irreducible, then $\pi^*$ can be computed with the constraint that $\sum_{i=1}^{n} \pi_i^* = 1$.

If the CTMC is not irreducible, then more complex solution methods are required.

Notice that for irreducible CTMCs, the steady-state distribution is independent of the initial-state distribution.
Steady-State Solution Methods

It is convenient to write the linear equations for the steady-state solution as $\pi^* Q = 0$.

Solving $\pi^* Q = 0$ will not give a unique solution; we must add the constraint that $\sum_{i=1}^{n} \pi_i^* = 1$ to guarantee uniqueness. This leads to two approaches:

- Replace $i^{th}$ column of $Q$ with $(1, 1, \ldots, 1)^T$ to form $\tilde{Q}$ and solve $\pi^* \tilde{Q} = e_i^T$ to guarantee a unique solution. This typically leads to worse numerical properties.
- Find any solution to $\pi^* Q = 0$ and then normalize the results. Not all solution methods can find a non-unique solution.
Direct Methods for Computing $\pi^* Q = 0$

A simple and useful method of solving $\pi^* Q = 0$ is to use some form of Gaussian elimination. This has the following advantages:

- Numerically stable methods exist
- Predictable performance
- Good packages exist
- Must have a unique solution

The disadvantage is that many times $Q$ is very large (thousands or millions of states) and very sparse (ones or tens of nonzero entries per row). This leads to very poor performance and extremely large memory demands.
Stationary Iterative Methods

Stationary iterative methods are solution methods that can be written as

$$\pi^{(k+1)} = \pi^{(k)} M,$$

where $M$ is a constant (stationary) matrix. Computing $\pi^{(k+1)}$ from $\pi^{(k)}$ requires one vector-matrix multiplication, which is one iteration.

Recall that for DTMCs, $\pi^* = \pi^* P$. For CTMCs, we can let $\hat{P} = Q + I$ and then $\pi^* = \pi^* \hat{P}$. Converting this to an iterative method, we write

$$\pi^{(k+1)} = \pi^{(k)} P.$$

This is called the power method.

- Simple, natural for DTMCs
- Gets to an answer slowly
- Can find a non-unique solution
Convergence of Iterative Methods

We say that an iterative solution method converges if \( \lim_{k \to \infty} \| \pi^{(k)} - \pi^* \| = 0. \)

Convergence is of course an important property of any iterative solution method.

The rate at which \( \pi^{(k)} \) converges to \( \pi^* \) is an important problem, but a very difficult one.

Loosely, we say method A converges faster than method B if the smallest \( k \) such that \( \| \pi^{(k)} - \pi^* \| < \varepsilon \) is less for A than for B.

Which iterative solution method is fastest depends on the Markov chain!
Stopping Criteria for Iterative Methods

An important consideration in any iterative method is knowing when to stop. Computing the solution exactly is often wasteful and unnecessary.

There are two popular methods of determining when to stop.

1. \( \| \pi^{(k+1)} - \pi^{(k)} \| < \varepsilon \) called the difference norm
2. \( \| \pi^{(k+1)} Q \| < \varepsilon \) called the residual norm

The residual norm is usually better, but is sometimes a little more difficult to compute. Both norms do have a relationship with \( \| \pi^{(k+1)} - \pi^* \| \), although that relationship is complex. The unfortunate fact is that the more iterations necessary, the smaller \( \varepsilon \) must be to guarantee the same accuracy.
Gauss-Seidel

From $\pi^{(k+1)} = \pi^{(k)}U(D + L)^{-1}$, we can write an element-wise algorithm:

for $k = 1$ to convergence
  for $i = 1$ to $n$
    $\pi^{(k+1)}_i = -\frac{1}{q_{ii}} \left( \sum_{j=1}^{i-1} \pi^{(k+1)}_j q_{ji} + \sum_{j=i+1}^{n} \pi^{(k)}_j q_{ji} \right)$
  end for
end for

Gauss-Seidel has the following properties:

- Memory-efficient; needs only $Q$ and $\pi$
- Typically converges faster than Power or Jacobi (not covered)
- May not converge at all in some extreme situations
- A way of guaranteeing convergence exists
- Convergence may be very slow
- Can find a non-unique solution
The above CTMC serves as an example of some of the issues involved in numerical solution, where $\lambda << 1$.

We illustrate an experiment of solving the CTMC using Gauss-Seidel, ranging $\lambda$ over many values, and stopping when $\|\pi^{(k+1)} - \pi^{(k)}\| < 10^{-7}$. 
Analysis

We range $\lambda$ from $10^{-7}$ to 1.

The measure we examine for this model is the steady-state probability of being in state 1 or 2, i.e., $\pi_1^* + \pi_2^*$.

For $\lambda << 1$, it is easy to approximate by observing that it is twice as likely to be in states 1 and 2 than in 3 and 4, so the measure is approximately $2/3$.

For our experiments, we used the $\|\pi^{(k+1)} - \pi^{(k)}\| < 10^{-7}$ stopping criterion.
## Results

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>number of iterations</th>
<th>computed solution</th>
<th>exact solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-7}$</td>
<td>1</td>
<td>0.5000</td>
<td>0.6667</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>305432</td>
<td>0.6000</td>
<td>0.6667</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>107298</td>
<td>0.6600</td>
<td>0.6667</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>18408</td>
<td>0.6660</td>
<td>0.6667</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>2612</td>
<td>0.66648</td>
<td>0.6666</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>342</td>
<td>0.66556</td>
<td>0.66556</td>
</tr>
<tr>
<td>$10^{-1}$</td>
<td>46</td>
<td>0.6562495</td>
<td>0.65625</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>0.600000</td>
<td>0.6</td>
</tr>
</tbody>
</table>
**Discussion**

- For $\lambda = 10^{-7}$, the first Gauss-Seidel iteration resulted in $\|\pi^{(2)} - \pi^{(1)}\| < 10^{-7}$, so it stopped immediately. 0.5 is the initial guess.

- For smaller $\lambda$, more iterations are required, and less accuracy is achieved.

- For larger $\lambda$, much fewer iterations are required, and very high precision is achieved.

- The property of the CTMC has more effect on the rate of convergence than the size of the CTMC does.
A Second Example: Multiprocessor Failure/Repair Model

System Description:

1) \( n \) processors, 1 needed for system to be up.

2) Each processor fails with rate \( \lambda \).

3) Processors repaired (one at a time) with rate \( \mu \).

4) After failure, reboot with prob(1 - \( c \)), reconfigure with probability \( c \).

5) Reconfiguration time exponential (rate \( \delta \)) results in system with one fewer processor.

6) Reboot (exponential, rate \( \gamma \)) results in fully functioning system.

7) Second failure during reconfiguration, or all processors failed, results in crash.

8) Crash repair (exponential, rate \( \beta \)) results in fully functioning system.

9) System unavailable if reboot or crash.
### Nominal System Parameter Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda )</td>
<td>( \frac{1}{6000 \text{ hours}} )</td>
<td>processor failure rate</td>
</tr>
<tr>
<td>( \mu )</td>
<td>( \frac{1}{1 \text{ hour}} )</td>
<td>processor repair rate</td>
</tr>
<tr>
<td>( \delta )</td>
<td>( \frac{1}{0.01 \text{ hours}} )</td>
<td>processor reconfiguration rate</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>( \frac{1}{0.5 \text{ hours}} )</td>
<td>reboot rate</td>
</tr>
<tr>
<td>( \beta )</td>
<td>( \frac{1}{20 \text{ hours}} )</td>
<td>crash repair rate</td>
</tr>
<tr>
<td>( c )</td>
<td>0.99</td>
<td>coverage probability</td>
</tr>
</tbody>
</table>
State Transition Rate Diagram

Reconfig1

Reconfig2

n

n - 1

n - 2

Crash

Reboot

\( \mu \)

\( \delta \)

\( \beta \)

\( \lambda \)

\( (n-1)\lambda(1-c) \)

\( n\lambda(1-c) \)

\( n\beta_c \)

\( (n-2)\lambda \)

\( (n-2)\lambda c \)

\( \gamma \)

\( \lambda \)

\( (n-1)\lambda \)
Unavailability vs. Number of Processors (varying failure rate)
Unavailability vs. Number of processors (varying coverage)
Review of State-Based Modeling Methods

• Random process
  – Classifications: continuous/discrete state/time

• Continuous-time Markov chain
  – Definition
  – Properties
  – Exponential random variable
    • Memoryless property
    • Minimum of two exponentials
    • Competing exponentials
    • Event/failure/hazard rate
  – State-transition-rate matrix
  – Transient solution
  – Steady-state behavior: $\pi^* Q = 0$

• Discrete-time Markov chain
  – Definition
  – Transient solution
  – Classification: reducible/irreducible
  – Steady-state solution

• Solution methods
  – Direct
  – Iterative: power, Gauss-Seidel
  – Stopping criterion
  – Example