Module 3: State-Based Methods

Availability: A Motivation for State-Based Methods

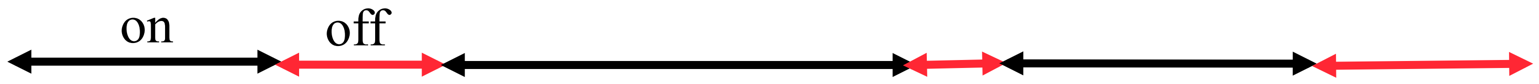
- Recall that *availability* quantifies the alternation between proper and improper service.
  - $A(t)$ is 1 if service is proper, 0 otherwise.
  - $E[A(t)]$ is the probability that service is proper at time t.
  - $A(0,t)$ is the fraction of time the system delivers proper service during $[0,t]$.
- 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 \$1000 radio transmitter has an expected time to failure of 500 days. Replacement takes an average 48 hours.
- A \$200 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 $t \rightarrow \infty$, $A(t) = .996$ for the more reliable transmitter, and $A(t) = .998$ for the less reliable transmitter.
- Higher reliability does not necessarily mean higher availability!
- For either choice, what is the cost we should budget for, including replacement?
- 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



"On" time distribution is reliability distribution, obtained using combinatorial methods, say mean is E[On]

"Off" distribution is repair time distribution, say mean is E[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{in On state} = E[On]/(E[On]+E[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

What is “State”

Informally, what you need to remember in order to be able to checkpoint, go away, come back, and resume where you started.

Examples:

- Game of chess (perhaps timer value, when timers are used)
- Election votes….number of votes for each candidate
- Football game
  - Team that has possession of the ball
  - Number of downs
  - Position of the ball on the field
  - First down line
- Grocery store
  - For each line
    - Remaining amount of time customer being served will still be present
    - Number of customers in queue

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) = $\{1,2,3, \ldots\}$.

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

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

$$E[X(n)]=3.5n$$

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" Ω to numbers in $\Re$.

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

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 Ω to $\Re$.

• However, if we fix ω, then X becomes a function of T to $\Re$.

• By fixing ω (e.g., the system is available) and observing X as a function of T, we see a trajectory of the process sampling ω or not

Examples

Fix $t = t_0$, $X(t_0)$ is the random number of customers in queue at t_0

Note that when $t_1 \mathrel{!}= t_0$, then $X(t_1)$ may have a different distribution than $X(t_0)$

Examples

Fix ω (e.g. ω = 0), then $X_\omega(t) = 1$ *if* $X(t) = \omega$ and $X_\omega(t) = 0$ otherwise is an interesting stochastic process

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

$$X(t) - X(t - 1) = Y,$$

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

$$\text{or } X(t) = Y_1 + Y_2 + \ldots + Y_t, \text{ where the } Y_i\text{'s are independent.}$$

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., $S = \Re$), 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

State \ *Time*	*Continuous*	*Discrete*
Continuous	Analog signal	A to D converter
Discrete	Computer availability model	round-based network protocol model

Markov Process

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, . . .} 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:

$$\begin{aligned} &P\left[X(t+k)=j \mid X(t)=i, X(t-1)=n_{t-1}, X(t-2)=n_{t-2}, \ldots, X(O)=n_O\right] \\ &\quad = P\left[X(t+k)=j \mid X(t)=i\right] \qquad (1) \\ &\quad = P_{ij}^{(k)} \qquad (2) \end{aligned}$$

DTMCs

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 π be a row vector. We denote π_i to be the *i*-th element of the vector. If π 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 $i, j = 1, \dots, n$, how do we compute $\pi(1)$?

Recall the definition of P_{ij}.

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

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

$$\begin{aligned} \pi_j(1) &= P[X(1) = j] \\ &= P[X(1) = j \mid X(0) = 1]P[X(0) = 1] + \dots + P[X(1) = j \mid X(0) = n]P[X(0) = n] \\ &= \sum_{i=1}^{n} P[X(1) = j \mid X(0) = i]P[X(0) = i] \\ &= \sum_{i=1}^{n} P_{ij}\pi_i(0) \\ &= \sum_{i=1}^{n} \pi_i(0)P_{ij} \end{aligned}$$

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

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

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 π 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

$$\begin{aligned}\pi(k) &= [\pi(k-2)P]P \\ &= \pi(k-2)P^2\end{aligned}$$

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

$$\begin{aligned}\pi(k) &= [\pi(k-3)P]P^2 \\ &= \pi(k-3)P^3\end{aligned}$$

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

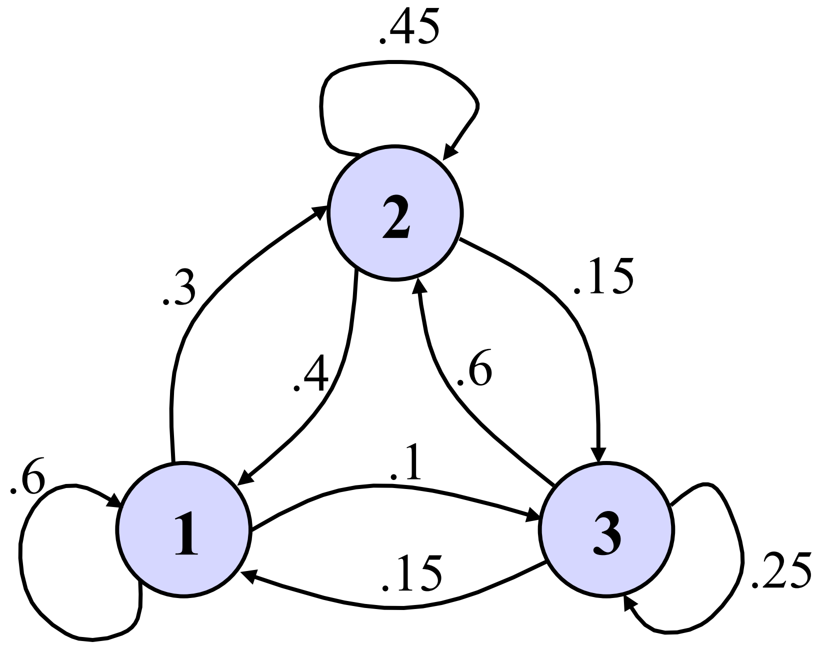
$$\pi(3) = \pi(0)P^3.$$

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 $(\ldots((\pi(0)P)P)P\ldots)P$ 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

$$\begin{tikzcd}
1 \arrow[loop left, "P_{idle}"] \arrow[rr, bend left, "P_{arr}"] \arrow[dr, bend left, "P_{fi}"] & & 2 \arrow[ll, bend left, "P_{com}"] \arrow[loop right, "P_{busy}"] \arrow[dl, "P_{fb}"] \\
& 3 \arrow[ul, bend left, "P_{r}"] \arrow[loop below, "P_{ff}"] &
\end{tikzcd}$$

$X = 1$	computer idle
$X = 2$	computer working
$X = 3$	computer failed

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