

**ECE 254 / CPS 225**  
**Fault Tolerant and Testable Computing Systems**

**Modeling and Evaluation**

Copyright 2011 Daniel J. Sorin  
Duke University

**Outline**

- **Experimental Methodology and Modeling**
- **Random Variables**
- **Probabilistic Models**
- **Queuing Network Models**
- **Markov Chain and Petri Net Models**
- **Evaluation Using Simulation**

**Experimental Methodology**

- **Follow the scientific rule (“strong inference”)**
  - Devise hypothesis
  - Conduct experiment that will prove or disprove hypothesis
- **Keys to good evaluations**
  - Using appropriate experimental tools
  - Using appropriate metrics
  - Designing experiments which will produce unambiguous results
  - NOT running experiments and THEN retro-fitting your hypothesis to match the results you obtained
- **Goals**
  - Conclusive and persuasive results
    - » Find situations for which your system is “better/worse” than existing systems
    - » Determine which system to use for which scenario

**Modeling**

- **Model: a representation of a system and its behavior**
  - Simulation is a form of modeling (e.g., SimpleScalar)
- **Why use models?**
  - Can study systems quickly
  - Can study designs that haven’t been built yet
- **Caveat: GIGO (garbage in, garbage out)**
  - If your model isn’t good, its outputs won’t be useful/correct
- **Analytical model = mathematical model**
  - Numerous types of analytical modeling techniques exist

## Modeling Techniques

- **Simulation**
- **Analytical**
  - Combinatorial
  - Probabilistic
  - Queuing network
  - Markov chain
  - Petri Net
  - Etc.
- **Each type of model has pros and cons**
  - More complexity → can model more detail, but tougher to solve

## Inputs and Outputs

- **Inputs**
  - System model (# of components, how they're connected, etc.)
  - Component characteristics
    - » Size, speed, failure rate, etc.
  - Mean Time To Repair (MTTR) is often an input
    - » But not if system self-repair is something we're studying
- **Outputs**
  - Performance (latency & throughput)
  - Reliability  $R(t)$
  - Availability  $A(t)$
  - Mean Time To Failure (MTTF)
  - Mean Time Between Failures (MTBF)

## A Paper with a Simple Model

- **“Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic”**
  - by Shivakumar, Kistler, Keckler, Burger, and Alvisi
  - U. of Texas and IBM/Austin

## Outline

- **Experimental Methodology and Modeling**
- **Random Variables**
- **Probabilistic Models**
- **Queuing Network Models**
- **Markov Chain and Petri Net Models**
- **Evaluation Using Simulation**

## Random Variables

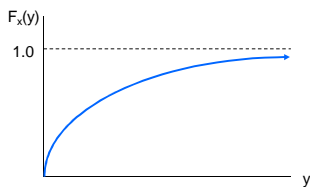
- Informally, a **random variable**  $X$  is a variable that takes on a value as a result of an experiment
  - Time between failures
  - Time to repair a failure
  - Time between requests made to a server
- Many types of random variables
  - Deterministic
  - **Exponential** (aka **Markovian**)
  - Gaussian (aka Normal)
  - Weibull
  - Hyper-exponential
  - Coxian
  - Etc.

## Characterizing Random Variables

- Can characterize random variable  $X$  in several ways
  - Mean (aka "expected value"), often denoted as  $\mu$  or  $E[X]$
  - Variance, often denoted as  $\sigma^2$ 
    - »  $\sigma^2 = E[(X - \mu)^2] = E[X^2] - \mu^2$
    - » Note that  $E[X^2] \neq (E[X])^2$
  - Standard deviation, often denoted as  $\sigma$ 
    - »  $\sigma = \text{sqrt}(\sigma^2)$
  - Coefficient of variation, often denoted as CV
    - »  $CV = \sigma/\mu$
  - **Probability density function (pdf)**
  - **Cumulative distribution function (CDF)**
- The pdf or CDF completely describes a RV
  - Looking at pdf or CDF, we can determine mean, variance, etc.

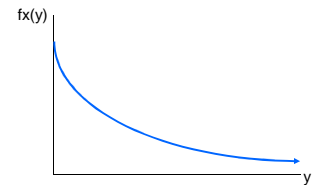
## CDF: Cumulative Distribution Function

- $X$  is a random variable
- CDF:  $F_x(y) = P[X \leq y]$ 
  - Monotonically increasing function
- For an exponential RV:  
 $F_x(y) = 1 - e^{-\lambda y}$  ( $X$  is characterized by single parameter  $\lambda$ )



## pdf: Probability Density Function

- $X$  is a random variable
- pdf:  $f_x(y)$  corresponds to probability of  $X=y$ 
  - Rough intuitive definition, not mathematically precise
- $P[a \leq X < b] = \int_a^b f_x(y) dy$  (more precisely)
- $f_x(y) \geq 0$  for all  $y$
- $\lim_{y \rightarrow \text{infinity}} f_x(y) = 0$
- For exponential RV:  
 $f_x(y) = \lambda e^{-\lambda y}$



### Expected Value (mean)

- $E[X] = \text{mean} = \int y f_x(y) dy$
- For exponential RV (do the math at home):
  - $E[X] = 1/\lambda$
- Important notes
  - In an experiment,  $E[X]$  is the expected value, but not the one that will always occur! There is a distribution of values around the expected value.
  - Exception to above rule: deterministic distributions always have the same value for every experiment

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

13

### Poisson Processes

- Let  $X(t)$  = a random variable that is the number of events that occur between time 0 and time  $t$
- $X(t)$  is a **Poisson process** if events occur at a constant rate,  $\lambda$ .
  - I.e.,  $X(t)$  does not depend on  $t$ .
- Many system behaviors can be modeled as Poisson processes (even if they aren't exactly Poisson in reality)
- $X(t)$  = Poisson process  $\leftrightarrow$  the time between events is an Exponential random variable

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

14

### Exponential Distributions

- Denoted by:  $X(t) \sim \text{Exponential}(\lambda)$
- Can show that  $P[X \leq t+x \mid X > t] = 1 - e^{-\lambda x}$ 
  - Not dependent on  $t \rightarrow$  "memory-less" or "Markovian" property
  - Markov chains (later!) depend on this property
- "Bus stop" analogy for memory-less property
  - How long you will wait for a bus is independent of how long you've already been waiting for a bus

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

15

### ECE 254 Just Skims Surface of RVs

- Random variables are a big and important topic that we can't cover in great depth in this class
  - Very useful in many aspects of science and engineering  $\rightarrow$  not just math for the sake of math
- If interested in learning more, take ECE 255 and ECE 257 (Prof. Trivedi) and/or courses on Probability and Stochastic Processes (most likely in Math Dept)

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

16

## Outline

- Experimental Methodology and Modeling
- Random Variables
- Probabilistic Models
- Queuing Network Models
- Markov Chain and Petri Net Models
- Evaluation Using Simulation

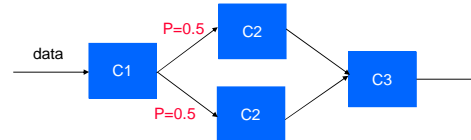
(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

17

## Probabilistic Models

- Model system behaviors and failure rates (inputs with probabilities (probability distributions))
- Simple example:



- Assume C2 components are redundant paths
- Assume you're given failure rates for each component
- What is the probability of getting correct output?

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

18

## Hazard Rate

- Hazard rate  $z(t)$  = failure rate at time  $t$
- Simplest model is non-time-varying:  $z(t) = \lambda$ 
  - Models failures as a Poisson process
  - Leads to reliability function that has Exponential distribution
- More sophisticated model:  $z(t) = \alpha\lambda(\lambda t)^{\alpha-1}$ 
  - Choose  $\alpha$  and  $\lambda$  to model the component's failure behavior
  - Leads to reliability function that has Weibull distribution
  - Denoted Weibull( $\alpha, \lambda$ )
- For  $z(t) = \lambda$ 
  - $R(t) = \text{Prob}[0 \text{ failures from time } 0 \text{ to time } t] = e^{-\lambda t}$
  - Intuitively,  $R(t) \rightarrow 0$  as  $t \rightarrow \text{infinity}$
  - $R(t)$  decreases at an exponential rate determined by  $\lambda$

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

19

## Weibull Reliability

- Weibull: for  $z(t) = \alpha\lambda(\lambda t)^{\alpha-1}$ 
  - $R(t) = \text{Prob}[0 \text{ failures from time } 0 \text{ to time } t] = e^{-(\lambda t)^\alpha}$
  - Intuitively,  $R(t) \rightarrow 0$  as  $t \rightarrow \text{infinity}$
  - $R(t)$  decreases at a rate determined by  $\alpha$  and  $\lambda$

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

20

## Other Probability Distributions

- We have only looked at Exponential and Weibull
  - Other distributions exist, but we won't cover them in this course
- In general, many behaviors can be represented reasonably well by Exponential distributions
  - This is convenient since they're very easy to work with
  - We'll see more about this later with Markov chains
- But if Exponential is not a good match for behavior you're studying, you can use other distributions

## Mean Time To Failure (MTTF)

- $MTTF = \int R(t)dt$
- For exponential hazard function
  - $MTTF = \int e^{-\lambda t} dt = 1/\lambda$
  - Intuitively, MTTF shrinks as failure rate ( $\lambda$ ) increases

## Availability

- Availability  $A = MTTF / (MTTF + MTTR)$ 
  - Denominator is often called MTBF (mean time between failures)
- Rewriting this equation a bit ...
  - Let  $\lambda$  = failure rate
  - Let  $\mu$  = repair rate
  - Then  $A = (1/\lambda)/(1/\lambda + 1/\mu) = \mu / (\mu + \lambda)$

## Example Application of Modeling

- In backward error recovery (BER), a system that detects an error recovers to a previous checkpoint and resumes execution from there
- What is the performance penalty of BER (as opposed to a FER system that doesn't recover backwards)?
- Assume:
  - Error rate is  $\lambda$ , and recovery rate is  $\mu$
  - Note that  $RecoveryTime = 1/(RecoveryRate)$
  - These are Exponential distributions (and not functions of  $t$ )
- What's the answer?

### Another Example Application of Modeling

- In a BER system, do we need to consider the possibility of an error occurring during recovery?
- Assume:
  - Error rate is  $\lambda$ , and recovery rate is  $\mu$
- What's the answer?

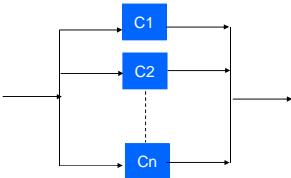
### Series System Models

- Series connection of components



- Every component must work for system to work
- $R(\text{system}) = \prod_{i=1 \text{ to } n} R(i) = e^{-\sum \lambda_i t}$
- Intuitively, R increases as:
  - Number of components decreases
  - Failure rates of components decrease

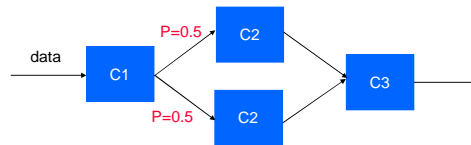
### Parallel System Models



- Only one component must work for system to work
- $R(\text{system}) = 1 - \prod_{i=1 \text{ to } n} [1 - R(i)]$
- Intuitively, R increases as:
  - Number of components increases
  - Failure rates of components decrease

### Mixed System Models

- It's rare that real systems are either purely serial or purely parallel
- We must construct models that are hybrids and analyze them accordingly



## Outline

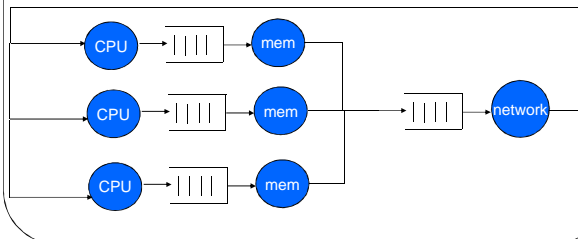
- Experimental Methodology and Modeling
- Random Variables
- Probabilistic Models
- **Queuing Network Models**
- Markov Chain and Petri Net Models
- Evaluation Using Simulation

## Queuing Networks

- **Modeling technique used in many situations**
  - More often used for performance than reliability
  - Thus we'll skim this topic in this class
- **Good reference textbook: "Quantitative System Performance: Computer System Analysis Using Queueing Network Models", by Lazowska, Zahorjan, Graham, and Sevcik (1984)**
  - Entire book available in PDF online at:  
<http://www.cs.washington.edu/homes/lazowska/qsp/>

## Queuing Networks

- Can model a computer system as a bunch of **service centers** with **queues**, in which **customers** flow from service center to service center
- Example: multi-chip multiprocessor



## Queuing Network Issues

- **Two types of queuing networks**
  - Open: customers arrive from outside world at some rate
  - Closed: fixed number of customers in system
    - » Example on previous slide was closed
- **Service times and arrival rates (for open systems) generally modeled probabilistically**
- **Common distributions used:**
  - Deterministic
  - Exponential (Markovian)
- **When might each of these be appropriate?**
- **When might neither be appropriate?**

## Solving Queuing Networks

- Can sometimes derive closed-form solution, if we make certain assumptions about distributions (and other more subtle issues)
- Usually pretty easy to solve numerically, if model isn't too enormous and we make a few simplifying assumptions
- Outputs of solution:
  - Customer throughput, response times, waiting times
  - Service center utilization (e.g., which one is the bottleneck!)
- Several approaches for solving models:
  - Mean value analysis (MVA): simple solution technique, but only reveals mean values

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

33

## Outline

- Experimental Methodology and Modeling
- Random Variables
- Probabilistic Models
- Queuing Network Models
- **Markov Chain and Petri Net Models**
- Evaluation Using Simulation

(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

34

## Markov Chains and Petri Nets

- Modeling techniques used in many situations
  - Not just for modeling reliability
  - ECE 255 & 257 cover these topics in much more detail
- Good reference textbook: "Probability and Statistics with Reliability, Queuing, and Computer Science Applications", by Kishor Trivedi, 2002 (2<sup>nd</sup> edition)

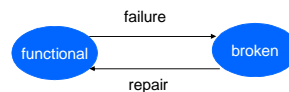
(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

35

## Markov Chains

- Models states and transitions between them
- Not necessarily "states" in the sense of a finite state machine, but often more global states that represent features of the entire system
- Example: 2 states, with 2 transitions



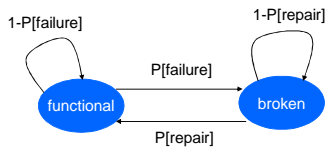
(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

36

### Discrete Time Markov Chains

- Edges in graph (i.e., arcs) represent **probabilities**
- At each **discrete time** step, there is a certain probability of transitioning to each other connected state



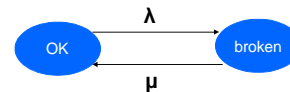
(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

37

### Continuous Time Markov Chains

- Edges in graph (i.e., arcs) represent **rates**
- **Continuously**, there is a certain rate of transitioning to each other connected state (no need for self-edges)
- Rates are assumed to be **Exponentially distributed**
  - Exponential is synonym for Markovian (hence "Markov chains")



(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

38

### Markov Chain Example

- **Memory scrubbing** walks through memory (reading each line) to uncover latent errors
- Let's create Markov chain for a given line of memory
  - Assume errors occur one at a time with rate  $\lambda$  per second
  - Assume we scrub lines with rate  $\mu$  lines per second
  - Assume we can repair all single-bit errors



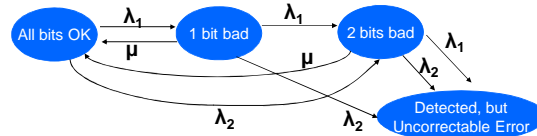
(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

39

### Twist on Markov Chain Example

- Still looking at memory scrubbing
- **New assumptions**
  - Assume single-bit errors occur with rate  $\lambda_1$  per second
  - Assume double-bit errors occur with rate  $\lambda_2$  per second
  - Assume we scrub lines at rate  $\mu$  per second
  - Assume we can repair all 1-bit and 2-bit errors, can detect >2-bit



(C) 2011 Daniel J. Sorin

ECE 254 / CPS 225

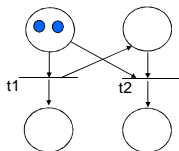
40

## Solving Markov Chains

- After creating a Markov chain model, we'd like to be able to solve it and determine:
  - Probability distribution of being in each state
  - Reachability (can we actually reach all states)
- Except for very simple models, it is impossible to derive a closed form solution
- In general, Markov chains are solved with software tools that can crunch through them numerically (i.e., without finding a closed form solution)
- This course will not address how to solve Markov chains → take ECE 255 and/or ECE 257

## Petri Nets

- Markov chains are not always the most intuitive way to represent a system
- Petri Nets (1962) developed to provide more intuitive interface, while still mathematically equivalent to Markov chains



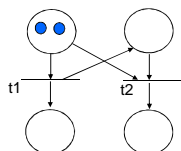
Petri net components:  
 places (states) -- large black circles  
 transitions -- arcs  
 multiplicities (widths of transitions) -- not shown  
 tokens -- small blue circles

A transition is **enabled** if each of its input places has at least as many tokens as the multiplicity of the corresponding input arc

Thus, only t1 is currently enabled in example

## Petri Nets

- Markov chains are not always the most intuitive way to represent a system
- Petri Nets (1962) developed to provide more intuitive interface, while still mathematically equivalent to Markov chains



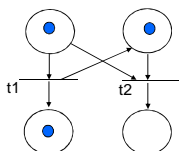
Once a transition is enabled, it can **fire**

At firing, a number of tokens equal to the multiplicity of the input arc are removed from each input place, and a number of tokens equal to the multiplicity of the output arc are placed in each output place

So, where will that leave us?

## Petri Nets

- Markov chains are not always the most intuitive way to represent a system
- Petri Nets (1962) developed to provide more intuitive interface, while still mathematically equivalent to Markov chains



Notice that we started with 2 tokens and now we have 3 → no conservation law for tokens

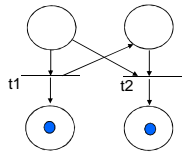
Now both t1 and t2 are enabled and can fire

Petri Net Rule: either is allowed to fire first, but they are NOT allowed to fire at the same time

What will happen if t2 fires first?

## Petri Nets

- Markov chains are not always the most intuitive way to represent a system
- Petri Nets (1962) developed to provide more intuitive interface, while still mathematically equivalent to Markov chains



This is the result if t2 fired first

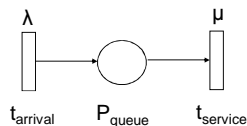
No more transitions are possible when tokens are in this global state (called a **marking** in Petri Net lingo)

## Stochastic Petri Nets

- The Petri net we just looked at had no concept of time
  - Transitions could fire immediately (no delay)
- **Stochastic Petri Nets (SPNs)** add timing information by adding a **firing time** to each transition
- Firing times often restricted to exponential distributions
- A SPN is mathematically equivalent to a **Continuous Time Markov Chain**
  - Can be automatically converted by software
  - Can be solved by software

## Very Simple SPN Example

- **Rectangles represent non-instantaneous transitions**
  - Labeled with mean of their exponential distribution
- This example is an **M/M/1 queue**
  - M stands for Markov (= exponential distribution)
  - First M means arrival process is Markovian
  - Second M means service process is Markovian
  - The 1 means that there is one service center
- **Note:**  $t_{\text{arrival}}$  has no input arc  $\rightarrow$  always enabled



## SPN Example From Software Aging

Models system with  $n$  nodes, initially all up. Tokens represent nodes.

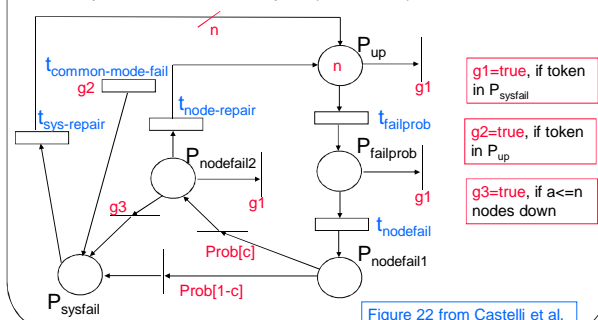


Figure 22 from Castelli et al.

## More About Petri Nets

- There are many other flavors of Petri Nets and many features that you can add to them
- There are many software tools for solving them (by first converting them to Markov chains)
- But this is all beyond the scope of this course (i.e., you will not be tested on it)
- Comment to the curious: Petri nets should remind you of dataflow computers

## Outline

- Experimental Methodology and Modeling
- Random Variables
- Probabilistic Models
- Queuing Network Models
- Markov Chain and Petri Net Models
- Evaluation Using Simulation

## Simulating

- Write a program that mimics system behavior
- Advantages
  - + Very flexible
  - + Relatively quick to develop
- Disadvantages
  - Runs slowly (e.g., 30,000 times slower than hardware)
- Some examples from computer engineering:
  - SimpleScalar: simulates microprocessors
  - nsim: simulates networks
  - Simics: simulates entire computer system

## Describing Simulated System

- How detailed must our simulator be?
- Model every transistor in the system?
  - Would take too long
- Abstract away details of system?
  - Could miss important effects
  - Could achieve wrong conclusion
- Need balance
  - Model in detail only where necessary
  - E.g., model memory system in detail, but abstract disks

## Simulators for Evaluating Fault Tolerance

- Simulators for evaluating fault tolerance may be quite a bit different than simulators for evaluating performance (which is what most computer engineers are used to using)
- Issue #1: need to be able to model behavior of faults
- Issue #2: often need to be able to model probabilistic behaviors

## Fault/Error Injection

- To simulate how a system tolerates faults/errors, we must “inject” them into the system
  - E.g., periodically have the simulator flip bits in the system
- Issues in fault/error injection
  - What kinds of faults/errors to inject
  - Where to inject them
  - When to inject them
    - » Once
    - » Periodically
    - » According to probability distribution

## Monte Carlo Simulation

- To simulate systems with probabilistic behaviors, we often use **Monte Carlo Simulation**
- Idea: each time we need the value of a probabilistic behavior (e.g., memory access time), we randomly choose it from the distribution
- Example
  - Assume memory access times are either 1 cycle (L1 hit), 5 cycles (L2 hit), or 200 cycles (memory hit)
  - Assume 90% hit rate in L1, 80% hit rate in L2 (if miss in L1), and 100% hit rate in memory (if miss in L1 and L2)
  - For each memory access, simulator will choose among 100 options:
    - » 90 options are 1 cycle
    - » 8 options are 5 cycles
    - » 2 options are 200 cycles

## Outline

- Experimental Methodology and Modeling
- Random Variables
- Probabilistic Models
- Queuing Network Models
- Markov Chain and Petri Net Models
- Evaluation Using Simulation