LECTURE 4: STATES AND EVENTS (PT 2) RANDOM NUMBERS
States and Events

- A *state* is a condition of a system at some point in time
- An *event* is something that cause a system to shift from one state to another
Last Time

- FSA: Finite State Automata
  - event driven transitions
- Markov Models
  - probabilistic transitions
- Production Systems
  - rule based transitions
Conditions and Events

Petri Nets
Petri Nets

tokens
places
transitions

- Machine M1
- p1
- t1
- p3
- Assembly A1
- t3
- p5
- Machine M2
- p2
- t2
- p4

\(\text{\bullet} = \text{material}\)
Petri Nets

- Places are conditions
- Transitions are events
- Transitions can fire if all incoming places have a token
- Firing a transition results in a token in all outgoing places
Petri Nets

t1 and t2 are ready to fire
Petri Nets

t3 is ready to fire
Petri Nets

![Petri Net Diagram](image)
Inputs to Petri Nets

- Some places can be designated as inputs
  - May be a time-based function for adding tokens
  - May be an infinite source of tokens
A Manufacturing Line

Cell 1

Cell 2

A = accumulator
R = robot
L = lathe
D = drill

Storage

AGV
Manufacturing Line: Petri Net
Mutex - Semaphore

Modeling Uncertainty

Random Number Generators
Probabilistic Processes

- Many of the parameters in real-world systems are **probabilistic** rather than **deterministic**
- To simulate these parameters, we need random number generators
Example: Single Queue Checkout

- Consider the following functional model for a single-queue of customers (i.e., a 7-11)

Block K is the source of customer arrivals.
Block Q is the queue of customers.
Customers enter the queue immediately upon arrival and exit after checkout.

Both of these blocks should be replaced by probabilistic functions.
Example: Single Queue Checkout

- **Sources of Randomness:**
  - Customer Arrival: time between customer arrivals
  - Checkout: time required to checkout

- Although random, both times can be described statistically by probability distributions (PDs)

- To model the system, we need to:
  - determine the PD from the real systems
  - replicate the PD with appropriate random numbers
Probability Distributions

- **A probability distribution** describes
  - the range of possible values that a random variable can attain, and
  - the probability that the value of the random variable is within any subset of that range

- **Probability distributions may be:**
  - continuous: allowing for all values in a range (real numbers)
  - discrete: allowing for particular values in a range (integers)
A uniform distribution is one in which the probability of any of the possible values is the same.

Example: The time between arrivals of customers is uniformly distributed over the range [1, 10] minutes.
Uniform Distribution: Discrete

If we only allow for integer values in \([1,10]\), then the probability of any one of those values is 10%.

Sum of probabilities must be 1.0 (100%)
If we allow for all real values in [1,10], we have a continuous probability distribution.
Non-uniform Distributions

- Non-uniform distributions are described by functions, such as the familiar bell-shaped Gaussian distribution:

![Graph of a bell-shaped distribution showing likely and unlikely values.]

likely values

unlikely values
Discrete Random Variables

- If the number of possible values for $X$ is finite or countably infinite, then $X$ is a *discrete random variable*.
- Examples:
  - $X \in \{ 1, 2, 3, 4, 5 \}$ (finite)
  - $X \in \{ 0, 2, 4, 6, \ldots \}$ (countably infinite)
Continuous Random Variables

- If the range of $X$ is an interval (or collection of intervals) of real numbers, then $X$ is a *continuous random variable*.

- Examples:
  \[ X \in [1.0, 10.0] \]
  \[ X \in (0.0, 1.0]\) or \[ X \in [-1.0, 0.0) \]

( is exclusive, [ is inclusive)
Interval Probabilities: Continuous

- for continuous random variables, probabilities must be computed over intervals:

\[ P(a < X < b) = \int_{a}^{b} f(x) \, dx \]

where \( f(x) \) is the *probability density function* (pdf).
Interval Probabilities

The probability that \( r \) is in \( [a,b] \) is the area of the shaded region.

\[
P(a < X < b) = \int_{a}^{b} f(x) \, dx
\]
Interval Probabilities

\[ P(5 < X < 7) = \int_{5}^{7} 0.1111 \, dx = 2 \times 0.1111 = 22.22\% \]
Interval Probabilities: Discrete

- for discrete random variables, probabilities summed over intervals:

\[ P(a \leq X \leq b) = \sum_{x=a}^{b} f(x) \]

where \( f(x) \) is the pdf
Uniform Distribution: Discrete

\[ P(5 \leq X \leq 7) = \sum_{x=5}^{7} f(x) = 0.1 + 0.1 + 0.1 = 30\% \]
Common Distribution Functions

- Uniform
- Triangular
- Exponential
- Normal
Uniform Distributions

- Probability of any value is the same

\[ f(x) = \begin{cases} 
  \frac{1}{b-a}, & a \leq x < b \\ 
  0, & \text{otherwise} 
\end{cases} \]
Triangular Distributions

- Probability peaks at some *mode* value then decreases to 0 at ends of range
  - $c$ is mode, $a$ and $b$ are range

\[
f(x) = \begin{cases} 
\frac{2}{(b-a)} \left( \frac{x-a}{c-a} \right), & a \leq x \leq c \\
\frac{2}{(b-a)} \left( \frac{b-x}{b-c} \right), & c \leq x \leq b \\
0, & \text{otherwise}
\end{cases}
\]
Triangular Distribution
Exponential Distributions

- Lambda is a rate (arrival rate, failure rate)

\[ f(x) = \begin{cases} \varphi e^{-\varphi x}, & x > 0 \\ 0, & \text{otherwise} \end{cases} \]
Exponential Distributions

- $\phi = 2$
- $\phi = 1.5$
- $\phi = 1$
- $\phi = 0.5$
Normal Distributions

- aka: Gaussian Distribution, Bell curve
- \( \mu = \) mean (average)
- \( \sigma^2 = \) variance

\[
f(x) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[ -\frac{1}{2} \left( \frac{x - \mu}{\sigma} \right)^2 \right], \quad -\infty < x < \infty
\]
Normal Distributions

Figure 5.11  pdf of the normal distribution.
Cumulative Distribution Functions

- A CDF defines the probability that a random variable is less than some given value.
- The CDF is the summation of a discrete PDF or integral of a continuous PDF.

\[ F(x) = \sum_{\forall x_i \leq x} p(x_i) \]

or integral of a continuous PDF.

\[ F(x) = \int_{-\infty}^{x} f(t) \, dt \]
CDF for Uniform Distribution

Figure 5.8 pdf and cdf for uniform distribution.
Pseudorandomness

- A *pseudorandom* process appears random, but isn’t
- Pseudorandom sequences exhibit *statistical randomness*
  - but generated by a deterministic process
- Pseudorandom sequences are easier to produce than a genuine random sequences
- Pseudorandom sequences can reproduce exactly the same numbers
  - useful for testing and fixing software.
Pseudorandom Generator Seeds

- Random number generators typically compute the next number in the sequence from the previous number.
- The first number in a sequence is called the seed.
  - To get a new sequence, supply a new seed (current machine time is useful).
  - To repeat a sequence, repeat the seed.
Uniform Distribution Generation

- Most programming environments have some function for generating random numbers with a continuous uniform PDF
- C/C++: int rand() (cstdlib)
  - Java: java.util.Random
  - Python: random() (module random)
Generating Random Numbers

- If we have a uniform PDF generator, we can generate RNs for non-uniform PDFs
- Given a pdf, f(x):
  - rejection method:
    generate a random 2D point (x,y), if y <= f(X) use y, otherwise try again
  - inverse transformation method:
    compute inverse of cumulative distribution function
    generate random number N in [0.0,1.0]
    use N as probability in CDF to generate number
Rejection Method

ger get random point \((x, y)\), where \(y > 0\).
if \(y < f(x)\), accept \(y\) with correct probability.
otherwise, reject \(y\) and try again
Inverse Method

- Suppose $P(X=1) = 25\%, \ P(X=2) = 50\%, \ P(X=3) = 25\%$ then $CDF(1) = 25\%, \ CDF(2) = 75\%, \ CDF(3) = 100\%$
- generate $N$ in $[0.0,1.0]$
- if $(n \leq CDF(1))$ $X = 1$
  else if $(n \leq CDF(2))$ $X = 2$
  else $X = 3$
Resources for RN Generation

- **Python:**
  - [http://docs.python.org/lib/module-random.html](http://docs.python.org/lib/module-random.html)

- **C++ and Java:**
  - `RandomGenerator` class

- **Arena:**
  - Textbook, appendix D
SOURCES

- Petri Nets, Random Numbers:
  - *Simulation Model Design and Execution*
    by Paul A. Fishwick, Prentice Hall, 1995

- Random Number Generators
  - *Discrete-Event System Simulation, 4th Edition*
    Banks, Carson, Nelson and Nicol, Prentice-Hall, 2005

- Random Float Generator Function
  - http://www.csee.usf.edu/~christen/tools/unif.c

- Random Distribution Functions
  - http://www.cise.ufl.edu/~fishwick/simpack/howtoget.html