

ENME 601  
Manufacturing Systems Design and Control  
Spring 2003 Course Notes

Jeffrey W. Herrmann  
Department of Mechanical Engineering  
University of Maryland  
College Park, Maryland 20742  
(301) 405-5433  
jwh2@eng.umd.edu

January 24, 2003

Copyright 2003 by Jeffrey W. Herrmann. All rights reserved. These notes may not be reproduced without the written permission of Jeffrey W. Herrmann.

# Contents

<b>1</b>	<b>Systems Modeling and Analysis</b>	<b>4</b>
1.1	System Life Cycle . . . . .	5
1.2	Types of Systems . . . . .	5
1.3	Problem Solving and System Design . . . . .	5
1.4	Modeling . . . . .	7
1.5	Types of Models . . . . .	8
1.6	Input-Output Modeling . . . . .	10
1.7	Deterministic and Non-deterministic . . . . .	11
1.8	Applications of Modeling and Analysis . . . . .	12
<b>2</b>	<b>Simulation</b>	<b>13</b>
2.1	Simulation Basics . . . . .	13
2.2	Steps in a simulation study . . . . .	15
2.3	Building a valid simulation model . . . . .	15
2.4	Selecting input probability distributions . . . . .	16
2.5	Generating random numbers . . . . .	20
2.6	Simple simulation . . . . .	21
2.7	Experimental design. . . . .	22
2.8	Analyzing output data . . . . .	22
2.9	Comparing two systems . . . . .	24
<b>3</b>	<b>Discrete Random Variables</b>	<b>25</b>
3.1	Bernoulli . . . . .	25
3.2	Binomial . . . . .	25
3.3	Geometric . . . . .	25
3.4	Poisson . . . . .	26
<b>4</b>	<b>Continuous Random Variables</b>	<b>26</b>
4.1	Uniform . . . . .	26
4.2	Exponential . . . . .	27
4.3	Gamma . . . . .	27
4.4	Normal . . . . .	27
4.5	Triangular . . . . .	28
<b>5</b>	<b>Discrete Event Systems: Deterministic Models</b>	<b>28</b>
5.1	Definitions . . . . .	28
5.2	Finite State Automata . . . . .	28
5.3	State Automata . . . . .	29
5.4	Timed State Automata . . . . .	29
<b>6</b>	<b>Discrete-Event Systems: Stochastic Models</b>	<b>31</b>
6.1	Stochastic Timed State Automata . . . . .	31
6.2	Stochastic Processes . . . . .	32
6.3	Generalized Semi-Markov Process . . . . .	33

6.4	Poisson Process . . . . .	33
6.4.1	Superposition . . . . .	35
6.4.2	Decomposition . . . . .	35
6.5	Markov Chain . . . . .	35
6.6	Discrete-Time Markov Chains . . . . .	36
6.6.1	Basic Properties . . . . .	36
6.6.2	State Classification . . . . .	38
6.6.3	Steady-State Analysis . . . . .	39
6.7	Continuous-Time Markov Chains . . . . .	39
6.8	Queueing Systems: Introduction . . . . .	42
6.9	M/M/1 Queue . . . . .	43
6.10	M/M/m Queueing Systems . . . . .	45
6.11	Other Single-Stage Queueing Systems . . . . .	47
6.12	Markovian Queueing Networks . . . . .	48
6.13	Complex Queueing Networks . . . . .	49

# 1 Systems Modeling and Analysis

What is a system?

A system is a combination of elements intended to act together to accomplish an objective. The systems approach to problem solving views a system as a set of interconnected elements and tries to understand the system performance (Palm, page 4).

Systems are composed of components, attributes, and relationships. Components are the operating parts of the system, consisting of input, process, and output. Attributes are the properties or discernible manifestations of the system components. Relationships are the links between components and attributes. A system is interconnected: the components affect each other and the overall system behavior (Blanchard, page 1).

What is systems engineering?

Our institute defines systems engineering as the discipline that develops and exploits structured, efficient approaches to analysis and design to solve complex engineering problems. Systems engineering focuses on methods to solve problems. For more information, see the Institute for Systems Research web site at [www.isr.umd.edu](http://www.isr.umd.edu). The International Council on Systems Engineering (INCOSE) is a professional organization of systems engineers.

According to Blanchard (page 23), there are many definitions of systems engineering, but they have some common threads:

1. A top-down approach that views the system as a whole and understands how the components effectively fit together.
2. A life-cycle orientation that addresses all phases to include system design and development, production and construction, distribution, operations, maintenance and support, retirement, phaseout, and disposal.
3. Definition of system requirements.
4. An interdisciplinary or team approach so that all design objectives are addressed effectively.

## 1.1 System Life Cycle

The system life cycle contains many steps. The acquisition phase includes conceptual design, detailed design, and production and construction. The utilization phase includes distribution, operations, maintenance and support, retirement, phaseout, and disposal (Blanchard, page 19).

For example, the following stages are a typical manufacturing system life cycle:

- Birth of the system: How does the system support the goals of the firm?
- Product design and process selection: What is the form of the manufactured product? How is it produced?
- Design of the system: What capacity is needed? Where should the facility be located? What physical arrangement is the best layout?
- System control: Managing day-to-day activities: production planning, scheduling, inventory.
- System improvements: Responding to change. Organizing the system to perform well in an environment of change.

## 1.2 Types of Systems

Different classification schemes exist for systems.

Clearly, some systems are natural, and others are human-made, though the latter exist in the natural world. A physical system consumes physical space, while a conceptual system organizes ideas. A static system does not move, but a dynamic system changes over time. A bridge is a static system, but traffic is a dynamic system. A closed system does not interact with its environment; it is self-contained. An open system allows information, energy, and matter to enter and leave (Blanchard, page 4).

This course focuses on discrete-event systems such as queueing systems, computer systems, communication systems, manufacturing systems, and traffic systems.

## 1.3 Problem Solving and System Design

Systems modeling and analysis is used to solve problems associated with the design and control of complex engineering systems. The following steps are a general problem-solving approach:

1. Formulating the problem: Identify the time horizon, the decision variables (which represent the controllable factors: how much to make?), the parameters (the uncontrollable factors like cost, although sometimes, these can be changed to model different environments), the constraints (the relationships between the decision variables and the parameters), and the objective function. (Note that some objectives might be expressed as constraints.)
2. Gathering data (for the parameters), which can be the most costly and time-consuming step.
3. Solution: Solve the optimization problem exactly or employ a heuristic to get an approximate solution
4. Sensitivity analysis: Determine what happens to the solution if certain parameters change. (A dynamic environment may cause change, or parameter values may be uncertain.)
5. Testing & implementation: Analyze the solution and its implications. Enact a policy (locate a plant or change schedule) or build computer software to provide decision support repeatedly in the future.

Problem-solving is related to the process of design. According to Blanchard (page 29), after one or more possible designs are created, one needs to evaluate them and select the most appropriate design (the one that best meets the design criteria and system requirements). Optimization can be viewed as a systematic method that generates and evaluates design alternatives and identifies an optimal one. In some cases, the problem can be formulated concisely and one can apply a mathematical programming technique. In general, generation and evaluation is an iterative procedure that requires significant effort.

Buzacott (page 8) presents a modeling process that is quite similar:

1. Identify the issues to be addressed.
2. Learn about the system.
3. Choose a modeling approach (mathematical, simulation, scale model).
4. Develop and test the model. Collect data.
5. Verify and validate the model. Verification checks that the model is working correctly. Validation checks the model assumptions against the real world.
6. Develop a model interface for the user.
7. Experiment with the model.
8. Present the results.

## 1.4 Modeling

Consider the following scenario: A manufacturing firm is planning to construct a manufacturing system. The analyst's job is to investigate this problem and to suggest a system design and control policies that, when implemented correctly, will yield optimal performance (measured on some scale). The analyst will usually construct a model as an analysis tool.

Why build a model?

According to Blanchard (page 147), a model represents a system and the essential relationships involved. Models are useful to obtain information about a system being designed, when it is hard or impossible to experiment directly with a prototype. Experiments with the model are easier and less expensive than directly manipulating the system itself. Examples include building a scale model and testing it in a wind tunnel, creating a factory layout (or living room layout) by manipulating templates on graph paper.

According to Bradley, Hax, Magnanti (Chapter 1), analysts use models to guide management decisions. Models are simplified representations of the real world. Models can be used to evaluate a strategy or to generate the best alternative within a class of strategies. For example, simulation is an evaluation model. Linear programming is a generation model.

In general, modeling has many uses. The following are the goals of modeling and system theory (Cassandras, page 55):

- Modeling and Analysis. Develop the model and study its properties.
- Design and Synthesis. Create a system that meets performance requirements.
- Control. Select the input functions that ensure that the system will perform correctly under a variety of conditions.
- Performance Evaluation. Measuring performance over a time interval.
- Optimization. Designing a system and controlling it achieve the best possible performance.

Buzacott (pages 12 and 17) give the following reasons to build a model:

- Understanding: The model explains why and how, develops insight, shows sensitivity.

- Learning: The model teaches managers or workers about the factors that influence system performance.
- Improvement: The model may help an analyst find better system designs or operating policies.
- Optimization: A mathematical programming model used to find the optimal values of decision variables.
- Decision making: The model predicts the impact of different alternatives and helps the decision-maker select a course of action.

## 1.5 Types of Models

There are many types of models and various ways to classify them.

Blanchard (page 145) presents the following classification of models:

- Physical models look like what they represent (a globe or a scale model or a model of a molecule).
- Analogue models behave like the original (an electrical circuit that represents a mechanical system or a clay mockup of an automobile).
- Schematic models graphically describe the system or process (an organization chart or a diagram of a football play).
- Mathematical models symbolically represent the attributes and are used to predict performance. Because systems involve phenomena that are unpredictable, the mathematical model may incorporate random variables. Also, the model includes controllable variables (design parameters) and uncontrollable variables.

According to Bradley, Hax, Magnanti (Chapter 1), there is a range of model abstraction:

- Real world
- Operational exercise: experimentation in the real world (try something, evaluate it).
- Gaming: role playing in an artificial environment, which can be a good learning experience.

- Simulation: a computer program used to evaluate a system.
- Analytical model: a completely mathematical formulation of the system.

These have increasing degree of abstraction and speed, but a decreasing degree of realism and cost. The first are realistic, slow, and expensive, while the last are abstract, fast, and cheap.

One can classify mathematical models (both analytical and simulation) by their intent and assumptions of certainty. The following table presents four primary types of mathematical models (Bradley, Hax, Magnanti, page 6). Strategy generation models are optimization models that find the best solution (or strategy or design). Of course, one can use a strategy evaluation model to find the best alternative by iteratively trying different alternatives.

	Strategy evaluation	Strategy generation
Certainty	Deterministic simulation Econometric models Systems of simultaneous equations Input-output models	Linear programming Network models Integer programming Nonlinear programming Control theory
Uncertainty	Monte Carlo simulation Discrete-event simulation Econometric models Stochastic processes Queueing theory Reliability theory	Decision theory Dynamic programming Inventory theory Stochastic programming Stochastic control theory

For discrete parts manufacturing there are three common types of models (Buzacott, page 11): A physical model is a physical system that has a smaller scale, like a toy model. Simulation models are computer programs that represent the events that occur as a system operates. Analytical models describe the system using mathematical or symbolic relationships. These are then used to derive a formula or define an algorithm or computational procedure to estimate the the system performance.

Which model is best? There is no available theory to select the best model for a given situation. The choice of an appropriate model is determined by the system and the background of the system analyst (Blanchard, page 149).

## 1.6 Input-Output Modeling

For more information on Input-Output Modeling, see Cassandras (Chapter 1). When modeling (describing) a system, it is useful to identify variables that we can measure. Some of these variables we can control: these we call the input variables, and we denote them as  $u(t)$ . A comprehensive set of variables that describe the system is denoted the state  $x(t)$ . Some variables represent the system's performance, these output variables are denoted  $y(t)$ .

The state of the system at time  $t_0$  is the information  $x(t_0)$  required at  $t_0$  such that the output  $y(t)$  for all  $t \geq t_0$  is uniquely determined from  $x(t_0)$  and the input  $u(t)$ ,  $t \geq t_0$ .

The state equations specify the state  $x(t)$ ,  $t \geq t_0$ , given  $x(t_0)$  and the input  $u(t)$ ,  $t \geq t_0$ .

The state space  $X$  is the set of all possible states. That is,  $x(t) \in X$  for all  $t$ .

It is common to write the state equations in the following way (see also Figure 1.6 from Cassandras, page 12):

$$x'(t) = f(x(t), u(t), t)$$

$$y(t) = g(x(t), u(t), t)$$

One way to classify systems is by considering how we can represent its state and the state equations.

In a static model, the state never changes.

In a linear model, both  $g$  and  $f$  are linear functions, and we can define matrices  $A(t)$ ,  $B(t)$ ,  $C(t)$ , and  $D(t)$  to achieve the following state equations:

$$x'(t) = A(t)x(t) + B(t)u(t)$$

$$y(t) = C(t)x(t) + D(t)u(t)$$

Linear systems are a small but interesting subset of systems. Much of system and control theory is based on analyzing linear models.

Some models have continuous states, where the variables have values that are real numbers. In other models, the variables take on discrete values. And there are many hybrids with some discrete and some continuous variables.

In a deterministic model, none of the output variables is a random variable. In a stochastic model, uncertainty about the input causes uncertainty about the state and output, so one or more of the output variables is a random variable.

Often, the system is controlled by something selecting the correct input so that the system performs correctly. That is, there is a reference signal  $r(t)$  and a controller that determines the input as follows:

$$u(t) = \gamma(r(t), t)$$

Feedback uses information about the system state to set the control, and this yields a closed-loop system:

$$u(t) = \gamma(r(t), x(t), t)$$

Often it is useful to discretize time so that time is not a continuous quantity but a sequence of points  $t_0 < t_1 < t_2 < \dots < t_k < t_{k+1} < \dots$ . Discretizing time does not imply discretizing the state space. In a discrete-time model, the state equations change slightly:

$$x(k+1) = f(x(k), u(k), k)$$

$$y(k) = g(x(k), u(k), k)$$

In a time-driven model, the state changes as time changes (whether continuously or at discrete clock ticks). In an event-driven model, the state changes when an event occurs. Different types of events can occur, and the underlying processes are asynchronous and concurrent, though not necessarily independent.

A model of a continuous-variable dynamic system (CVDS) has two key properties: it has continuous states, and the state transition mechanism is time-driven. Analysis proceeds by examining the differential equations (or difference equations). A model of a discrete event system (DES), also called a discrete-event dynamic system, is different in two ways: it has a discrete state space, and the state transition mechanism is event-driven. Analysis is more complex.

## 1.7 Deterministic and Non-deterministic

According to Hazelrigg (page 36), uncertainty exists any time that we conduct an experiment for which we cannot predict the outcome. That is, when the sample space contains more than one element with nonzero probability, we say that there is uncertainty. The following things can cause uncertainty:

1. Insufficient understanding of nature.
2. Insufficient computational power (e.g., fluid dynamics).
3. Insufficient data about initial conditions (e.g., coin toss).
4. The system is inherently random (e.g., quantum mechanics).

Variability describes how a set of objects are different. Variability is associated with randomness. Controllable variation occurs because of decisions. Random variation occurs from events beyond our immediate control, and certain results are unpredictable or uncertain. It is important to develop intuition about random variation (Hopp and Spearman, page 249).

## 1.8 Applications of Modeling and Analysis

There exist many applications for systems engineering (Blanchard, page 39): aerospace systems, hydroelectric systems, information processing systems, electronic systems, transportation systems, manufacturing systems, infrastructure systems, communications systems, and health care systems.

There have been many applications of system modeling and analysis. Miser (page 79) describes how modeling and analysis was used to help the Wilmington, Delaware fire department locate engine and ladder companies. The city had seven old firehouses and one new one, and the fire department was considering building more. However, administrators were unsure where to locate the new ones and which companies to assign to them. Selecting the best deployment policy involved many considerations. The most important performance measures were the number of fire fatalities and injuries and the amount of property loss. Since there were no reliable ways to estimate the impact of a deployment policy on these measures, the analysts used three substitute measures: maximum travel time to individual locations, average travel time in a region, and company workload. The department wanted companies to be both available and close to fires. The analysts estimated travel times by correlating travel time with travel distance and used a linear regression based on actual company responses. The analysts used a well-known theoretical model that estimates average travel distance in a region from the region area and the average number of available companies. Then the analysts were able to create a model that evaluated a deployment policy on the relevant measures. Policymakers compared over 100 different deployments and selected the one that was the best compromise on all objectives.

Buzacott (page 19) presents multiple models for a machine interference system. A machine shop contains several automatic machines. A machine requires attention only when a tool must be changed, it fails, or it has no more material. Management needs to find an acceptable staffing level. Assigning too many machines to one operator will cause interference because multiple machines will be stopped waiting for the operator, which reduces throughput. Assigning too few machines to one operator will require too many operators, which costs money. Analysis can determine how the number of machines assigned to an operator affects operator utilization, how long a down machine must wait for an operator, and the resulting throughput. More complex models can predict performance under a variety of conditions about the time

to fail, time to repair, dissimilar machines, sharing machines between operators, operator travel time, and spare machines.

## 2 Simulation

### 2.1 Simulation Basics

Simulation uses a model of the system to reproduce the dynamic behavior of the system, such as flight simulators or virtual reality. For studying discrete-event systems we use discrete-event simulation. However, there are other types of simulation.

According to Law and Kelton (page 6), simulation models have three important characteristics:

1. Static or Dynamic. A static simulation model represents a static system (or a dynamic system at a particular point in time). A Monte Carlo model is an example. A dynamic simulation model includes the system behavior over time.
2. Deterministic or Stochastic. A deterministic simulation model contains no random variables. A stochastic simulation model contains random variables. Thus, the output of a stochastic simulation model is a random variable.
3. Continuous or Discrete. A continuous simulation model uses continuous variables to represent the state of the system. A discrete simulation model uses discrete variables to represent the state of the system. One can use either type of model to represent a discrete-event system, since aggregation can make a discrete-event system appear continuous.

A discrete-event simulation is a dynamic simulation (stochastic or deterministic) in which the state variables change instantaneously at separate points in time (Law and Kelton, page 7).

Monte Carlo simulation (Law and Kelton, page 113) uses random numbers to solve certain problems about static systems. Certain problems in multi-variable integration and statistics have no analytical solution but can be solved approximately using Monte Carlo simulation.

Simulation has certain advantages over other modeling approaches (Law and Kelton, page 114):

- Simulation can represent complex, real-world systems.
- Simulation can estimate system performance in an actual or hypothetical scenario and can be used to compare alternative system designs or operating policies.
- Simulation allows better control over experimental conditions.
- Simulation can scale time by evaluating performance over a long time frame or by allowing one to study each event that occurs, no matter how quickly.

Simulation also has disadvantages:

- Each run of a stochastic simulation estimates the system performance. Better estimates require multiple independent simulation experiments.
- Developing simulation models can be expensive and time-consuming.
- Impressive quantities of simulation results and realistic animations do not guarantee the simulation model's validity.

The following factors should be considered when choosing an analytical model versus a simulation model (Buzacott, page 15):

- Complexity. Simulation models can describe very complex systems, where analytical models are limited.
- Flexibility. Unless the simulation model is well-designed, changing the system parameters can be difficult. It is usually very easy to change the parameters of an analytical model, but a change to the structure requires a new model.
- Data requirements. Analytical models usually require less data than a simulation model.
- Transparency. Neither are very transparent. Non-experts cannot read the simulation code or understand the mathematics.
- Efficiency. An analytical model is usually quick to execute, but simulation models can require considerable time and computational effort. As computing power becomes less expensive and more available, bigger models can be run in reasonable time.
- User interface. Both simulation models and analytical models require user interfaces to allow changes to parameters and to view and compare results. Simulation sometimes includes animation, which aids understanding.

Effective modeling of discrete part manufacturing systems requires both types of models. Simulation is cheaper than experimenting with the real system and more realistic than an analytical model.

Simulation is also used to create schedules that the manufacturing system can then try to follow. (Autosched is a simulation-based scheduling.) This environment requires the ability to get up-to-date information about manufacturing resources and the jobs to do and the ability to reschedule.

## **2.2 Steps in a simulation study**

According to Law and Kelton (page 106), the following steps compose a typical, sound simulation study:

1. Formulate problem and plan the study
2. Collect data and define a model
3. Check validity by consulting decision-maker/user/other expert and by testing goodness-of-fit for probability distributions
4. Construct computer program and verify/debug: top-down programming, have someone check code, run and see if results make sense, trace program, observe animation
5. Make pilot runs
6. Check validity by testing model's sensitivity to input parameters
7. Design experiments (which alternatives, initial conditions and warmup, simulation length, replications)
8. Run experiments
9. Analyze output data using statistical techniques
10. Document, present, and implement results

## **2.3 Building a valid simulation model**

Model validation is the process of establishing model credibility. Credibility is the model's ability to provide good credible answers to the right questions.

There are various ways to improve the validity of a simulation model.

Usually it is appropriate to start by developing a reasonable model based on discussions with experts, observations of the system, accepted theories about system behavior, and experience with modeling similar systems.

Then, one can deliberately plan to present analysis results in the middle of the project so that the customer can see initial results and identify problems early.

It is also recommended to document all model assumptions and regularly review the assumptions. Determine if those assumptions are still reasonable. Undocumented assumptions are invisible sources of errors.

Also, one can use sensitivity analysis to identify one or two sets of input variables that significantly influence the results. The values of these variables need to be checked carefully. Data gathered by automated systems is useful but may contain missing values or outliers that skew the results.

Finally, determine if the output of the model is representative by comparing it to actual results, if available.

## 2.4 Selecting input probability distributions

Simulation is often used to study stochastic systems, where there is some uncertainty about the input parameters. Arrival times, processing times, failure times, repair times, order sizes, and the number of arrivals are typically random variables. One can model these random variables by their probability distributions. Sometimes one must determine which probability distribution to use given a sample of actual values (from a time study, for instance). Common distributions include the normal, exponential, poisson, weibull, gamma, triangular, binomial, geometric, and the uniform. See Chapters 3 and 4 for more information about these probability distributions.

- **Step 1: Hypothesize** the family by considering what the random variable is modeling: interarrival times are often exponential, the number of defects in a batch is often binomial, heights are normal. Consider a histogram of the data to identify a distribution that has a similar probability density function, but be careful to select the correct intervals. Let  $X_1, \dots, X_n$ , represent  $n$  independent observations of the variable.
- **Step 2: Estimate the parameters.** Any distribution has one or more parameters  $\theta$  (e.g., mean, standard deviation), and we want to choose parameter values so that the distribution will closely match the underlying population. We

estimate the distribution's parameters with the maximum-likelihood estimators  $\hat{\theta}$  (MLE), which maximize the probability that we have guessed the parameter  $\theta$ . Note that  $\hat{\theta}$  is a random variable, since it is a function of  $X_1, \dots, X_n$ , which are random variables. This forms the estimated distribution function  $\hat{F}$ .

- **Step 3: Check the goodness-of-fit.** We form the hypothesis that the  $X_i$ 's are independent, identically distributed (IID) random variables with the distribution function  $\hat{F}$ . We use the chi-square test to determine if we should reject the hypothesis. Divide the theoretical range of the variable into  $k$  intervals and compare the actual and expected number of samples in each interval. For  $j = 1, \dots, k$ , let  $N_j$  be the actual number of  $X_i$  that occur in the  $j$ -th interval  $[a_{j-1}, a_j)$  (where  $a_0 = -\infty$  or  $a_k = +\infty$  if necessary). Note that  $\sum_{j=1}^k N_j = n$ . Define the interval's probability  $p_j$  as follows:

$$p_j = \int_{a_{j-1}}^{a_j} \hat{f}(x) dx \quad \text{if the random variable is continuous}$$

$$p_j = \sum_{a_{j-1} \leq x < a_j} \hat{f}(x) \quad \text{if the random variable is discrete}$$

Compare  $N_j$  to  $np_j$  (the expected number in that interval) as follows:

$$\chi^2 = \sum_{j=1}^k \frac{(N_j - np_j)^2}{np_j}$$

Ideally, this should be close to zero. We reject the hypothesis if this number is too large. Specifically, with approximate **level**  $\alpha$ , we reject the hypothesis if  $\chi^2 > \chi_{k-1, 1-\alpha}^2$ . The level  $\alpha$  is the probability of a Type I error, the error that we reject the hypothesis when it is true. A smaller  $\alpha$  requires that we accept fairly high test statistics, so the threshold goes up as the level goes down, implying that we reject less often. Levels of 10% are very common. Also note that we should choose the correct number of intervals. Previous work has shown that the best intervals are equally likely, so  $p_j = \frac{1}{k}$  for all  $j$ , and that  $np_j = \frac{n}{k} \geq 5$ , so choose  $k \leq \frac{n}{5}$ . (Choosing unusually sized or too many intervals is bad.)

The following example is from Law and Kelton (page 386).

Value	Frequency	Value	Frequency	Value	Frequency	Value	Frequency
0.01	8	0.02	2	0.03	3	0.04	6
0.05	10	0.06	4	0.07	10	0.08	4
0.09	2	0.10	9	0.11	5	0.12	4
0.13	2	0.14	4	0.15	6	0.17	1
0.18	1	0.19	3	0.20	1	0.21	5
0.22	3	0.23	5	0.24	1	0.25	5
0.26	5	0.27	1	0.28	2	0.29	2
0.30	1	0.31	2	0.32	1	0.35	3
0.36	3	0.37	2	0.38	5	0.39	1
0.40	2	0.41	2	0.43	3	0.44	1
0.45	2	0.46	1	0.47	3	0.48	1
0.49	4	0.50	3	0.51	3	0.52	2
0.53	3	0.54	2	0.55	2	0.56	1
0.57	2	0.60	1	0.61	2	0.63	2
0.64	1	0.65	3	0.69	2	0.70	1
0.72	3	0.74	1	0.75	1	0.76	1
0.77	1	0.79	1	0.84	1	0.86	1
0.87	1	0.88	2	0.90	1	0.93	2
0.95	1	0.97	1	1.03	1	1.05	2
1.06	1	1.09	1	1.10	1	1.11	1
1.12	1	1.17	1	1.18	1	1.24	2
1.28	1	1.33	1	1.38	1	1.44	1
1.51	1	1.72	1	1.83	1	1.96	1

We are given  $n = 219$  interarrival times with sample mean  $\bar{X}(n) = \frac{1}{n} \sum_{i=1}^n X_i = 0.399$ . From the histogram (Figure 1), we guess that the interarrival times are exponentially distributed with mean 0.399. The estimated density function is  $\hat{f}(x) = 0.399e^{-\frac{x}{0.399}}$ ,  $x \geq 0$ . And the estimated distribution function is  $\hat{F}(x) = 1 - e^{-\frac{x}{0.399}}$ ,  $x \geq 0$ .

To determine if this distribution is reasonable, we can choose  $k = 20$  equally likely intervals (with  $p_j = \frac{1}{20}$ ) as follows:

$$\begin{aligned}
 a_0 &= 0 \\
 \hat{F}(a_j) &= \frac{j}{20}, j = 1, \dots, 20 \\
 1 - e^{-\frac{a_j}{0.399}} &= \frac{j}{20}, j = 1, \dots, 20 \\
 a_j &= -0.399 \ln \left( 1 - \frac{j}{20} \right), j = 1, \dots, 20 \\
 \text{Note } a_{20} &= \infty
 \end{aligned}$$

### Interarrival Time Frequency

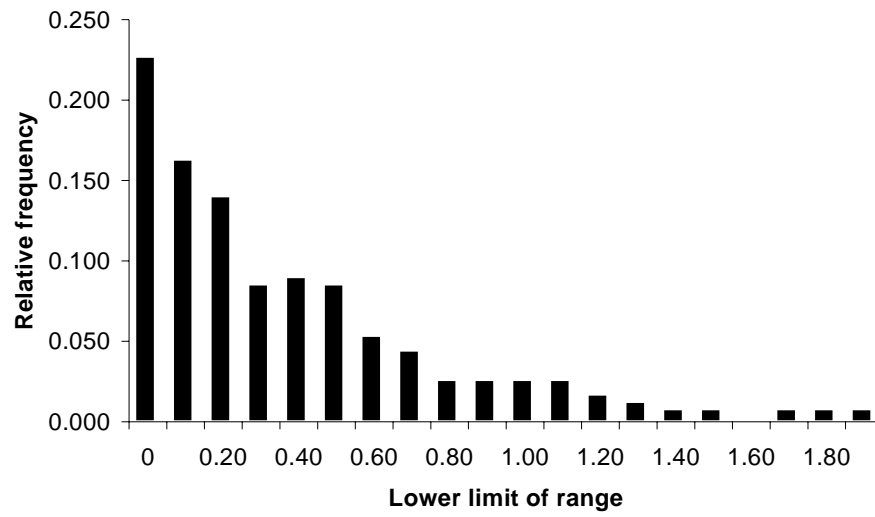


Figure 1: Histogram of Interarrival Times

$j$	$a_{j-1}$	$a_j$	$N_j$
1	0	0.020	8
2	0.020	0.042	11
3	0.042	0.065	14
4	0.065	0.089	14
5	0.089	0.115	16
6	0.115	0.142	10
7	0.142	0.172	7
8	0.172	0.204	5
9	0.204	0.239	13
10	0.239	0.277	12
11	0.277	0.319	7
12	0.319	0.366	7
13	0.366	0.419	12
14	0.419	0.480	10
15	0.480	0.553	20
16	0.553	0.642	9
17	0.642	0.757	11
18	0.757	0.919	9
19	0.919	1.195	14
20	1.195	$\infty$	10

$np_j = \frac{219}{20} = 10.95$  for each interval. We can calculate  $\chi^2$  as follows:

$$\chi^2 = \sum_{j=1}^k \frac{(N_j - np_j)^2}{np_j} = 22.188$$

$$\chi_{k-1,1-\alpha}^2 = \chi_{19,0.90}^2 = 27.204$$

Because  $\chi^2 < \chi_{k-1,1-\alpha}^2$ , we do not reject the hypothesis that the underlying distribution is exponential. The simulation model should generate interarrival times by sampling an exponentially distributed random variable with mean 0.399.

## 2.5 Generating random numbers

How does a computer program generate random numbers?

Computers create pseudo-random numbers by using a modulus function. Given an initial integral seed  $Y_0$ , which the user (or software) specifies, the program generates a sequence of integers  $Y_1, Y_2, Y_3, \dots$  using a function that looks like this:

$$Y_i = (aY_{i-1} + b) \bmod m$$

Thus, each element  $Y_i$  is in the set  $\{0, \dots, m-1\}$ . Dividing the number by  $m$  gives a rational number between 0 and 1, and it resembles a random variable uniformly distributed between 0 and 1. Note, however, that this pseudo-random number is not continuous (it has  $m$  different values).

Consider the following example with these parameters:  $m = 10,000$ .  $a = 101$ .  $b = 7$ .  $Y_0 = 4567$ .

$i$	$Y_{i-1}$	$aY_{i-1} + b$	$Y_i$	$U_i$	$X_i = \frac{\ln(1-U_i)}{-1/3}$
1	4567	461274	1274	0.1274	0.4088
2	1274	128681	8681	0.8681	6.0771
3	8681	876788	6788	0.6788	3.4071
4	6788	685595	5595	0.5595	2.4595

From this (pseudo-random) uniform distribution, we can generate samples from other distributions. Any random variable  $X$  has a distribution  $F(x) = P\{X \leq x\}$ .

$U = F(X)$  is a random variable that has a uniform distribution between 0 and 1. Thus, if  $U$  is a random variable with a uniform[0,1] distribution,  $F^{-1}(U)$  has the same distribution as  $X$ .

For example, if  $X$  has an exponential distribution and the mean of  $X$  is  $1/\lambda$ ,  $F(x) = 1 - e^{-\lambda x}$ , if  $x \geq 0$ . Let  $u = 1 - e^{-\lambda x}$ , which leads to  $x = \ln(1 - u)/(-\lambda)$ . If the mean of  $X$  is 3, then  $\lambda = 1/3$ . See the above table for values of  $X_i$ .

## 2.6 Simple simulation

Consider a simple queueing system. There is a single server. Customers arrive over time. If the server is idle when the customer arrives, then the server begins processing. When the server finishes, the customer leaves. If the server is busy when the customer arrives, then the customer waits in a queue. When the server finishes the current customer, the next customer in the queue is served.

The following steps describe an algorithm to simulate the behavior of the system:

1. Initialization: Let  $t = t_0$ . The event list is empty. The server is available. Generate the first interarrival time  $a_1$ . Add (customer 1 arrives at time  $t + a_1$ ) to the event list.
2. Step: Take next event from the event list. Let  $t$  be the time at which this event occurs. If this event is (customer  $j$  arrives at time  $t$ ), go to Arrival. If this event is (customer  $j$  completes service at time  $t$ ), go to Departure.
3. Arrival: Customer  $j$  enters the system and enters the queue.  
If a server is available, customer  $j$  leaves the queue and begins service. Generate service time  $p_j$  and add (customer  $j$  completes service at time  $t + p_j$ ) to the event list. The server is now busy.  
In any case, generate the next interarrival time  $a_{j+1}$ . Add (customer  $j + 1$  arrives at time  $t + a_{j+1}$ ) to the event list. Go to Step.
4. Departure: Customer  $j$  completes service and leaves the system. The server is available.  
If there are any customers in the queue, select next customer  $k$ . Customer  $k$  leaves the queue and begins service. Generate service time  $p_k$  and add (customer  $k$  completes service at time  $t + p_k$ ) to the event list. The server is now busy. Go to Step.

The following example illustrates the algorithm:

- $t = 9:00$  (initialization).  $a_1 = 5$  minutes. Add (customer 1 arrives at 9:05).
- $t = 9:05$  (customer 1 arrives).  $p_1 = 4$  minutes. Add (customer 1 completes service at 9:09).  $a_2 = 3$  minutes. Add (customer 2 arrives at 9:08).
- $t = 9:08$  (customer 2 arrives).  $a_3 = 4$  minutes. Add (customer 3 arrives at 9:12).
- $t = 9:09$  (customer 1 completes service).  $p_2 = 7$  minutes. Add (customer 2 completes service at 9:16).

- $t = 9:12$  (customer 3 arrives).  $a_4 = 8$  minutes. Add (customer 4 arrives at 9:20).
- $t = 9:16$  (customer 2 completes service).  $p_3 = 2$  minutes. Add (customer 3 completes service at 9:18).
- $t = 9:18$  (customer 3 completes service). Server available.
- $t = 9:20$  (customer 4 arrives).  $p_4 = 6$  minutes. Add (customer 4 completes service at 9:26).  $a_5 = 5$  minutes. Add (customer 5 arrives at 9:25).
- $t = 9:25$  (customer 5 arrives). no more arrivals
- $t = 9:26$  (customer 4 completes service).  $p_5 = 7$  minutes. Add (customer 5 completes service at 9:33).
- $t = 9:33$  (customer 5 completes service). Server available.

The following table summarizes the life of each customer:

Customer	Arrival	Begin Service	Departure
1	9:05	9:05	9:09
2	9:08	9:09	9:16
3	9:12	9:16	9:18
4	9:20	9:20	9:26
5	9:25	9:26	9:33

## 2.7 Experimental design.

Simulation can be used incorrectly. One simulation run of a model does not prove anything. A proper simulation study requires careful experimental design and statistical inference. Recall that probabilities are information about a population, and they describe what a sample might look like. Statistics are information about a sample, and they describe what a population might look like.

## 2.8 Analyzing output data

To predict the actual system performance, we run a set of simulations (replications) and measure the system performance during each simulation. (The system performance could be an “average” number like the average queue time per job, measured during one simulation run.)

Suppose we have run  $n$  replications and collected the system performance in each run:  $X_1, \dots, X_n$ . We can calculate the sample mean and sample variance as follows:

$$\begin{aligned}\bar{X}(n) &= \frac{1}{n} \sum_{i=1}^n X_i \\ S^2(n) &= \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X}(n))^2\end{aligned}$$

If  $\mu = E(X)$ , then the  $100(1 - \alpha)\%$  confidence interval for  $\mu$  is this:

$$\bar{X}(n) \pm t_{n-1, 1-\frac{\alpha}{2}} \sqrt{\frac{S^2(n)}{n}}$$

$\widehat{Var}(\bar{X}(n)) = \frac{S^2(n)}{n}$  and  $t_{n-1, 1-\frac{\alpha}{2}}$  is the  $1 - \frac{\alpha}{2}$  critical point for the t distribution with  $n - 1$  degrees of freedom. (As  $n \rightarrow \infty$ , the t distribution approaches the z distribution.)

Consider the following example, which presents the results for  $n = 10$  replications of a simulation, and the system performance  $X_i$  is the average time in queue for all customers.

$$\begin{array}{ll} X_1 = 1.53 & X_6 = 1.69 \\ X_2 = 1.66 & X_7 = 2.69 \\ X_3 = 1.24 & X_8 = 2.86 \\ X_4 = 2.34 & X_9 = 1.70 \\ X_5 = 2.00 & X_{10} = 2.60 \end{array}$$

We can estimate the actual average time in queue and construct a 90 percent confidence interval as follows:

$$\begin{aligned}\bar{X}(n) &= 2.03 \\ S^2(n) &= \frac{1}{9} \sum_{i=1}^n (X_i - 2.03)^2 = 0.31 \\ \alpha &= 0.10 \\ t_{9, 0.95} &= 1.833 \\ \bar{X}(n) \pm t_{n-1, 1-\frac{\alpha}{2}} \sqrt{\frac{S^2(n)}{n}} &= 2.03 \pm 0.32 = [1.71, 2.35]\end{aligned}$$

## 2.9 Comparing two systems

Suppose we collect the same measure of performance for two systems and we want to determine if either system is significantly better than the other.

Let  $X_{ij}$  be the performance of system  $i$  in replication  $j$ ,  $i = 1, 2$ ;  $j = 1, \dots, n$ . Suppose  $\mu_i = E(X_i)$  and  $\delta = \mu_1 - \mu_2$ . We want to know whether  $\delta < 0$  ( $\mu_1 < \mu_2$ ). Calculate  $Z_j = X_{1j} - X_{2j}$ , for  $j = 1, \dots, n$ . (Then  $\delta = E(Z)$ .)

$$\begin{aligned}\bar{Z}(n) &= \frac{1}{n} \sum_{j=1}^n Z_j \\ S^2(n) &= \frac{1}{n-1} \sum_{j=1}^n (Z_j - \bar{Z}(n))^2\end{aligned}$$

Because  $\widehat{Var}(\bar{Z}(n)) = \frac{S^2(n)}{n}$ , the  $100(1 - \alpha)\%$  confidence interval for  $\delta$  is this:

$$\bar{Z}(n) \pm t_{n-1, 1-\frac{\alpha}{2}} \sqrt{\frac{S^2(n)}{n}} = [Z_{\min}, Z_{\max}]$$

If  $Z_{\min} > 0$ , then we can say that  $\delta > 0$  and thus  $\mu_1 > \mu_2$ . If  $Z_{\max} < 0$ , then we can say that  $\delta < 0$  and thus  $\mu_1 < \mu_2$ . Otherwise, our results are inconclusive.

Consider the following example, using data from Law and Kelton (page 588). The system performance is the average total monthly cost, and we are comparing two policies. Which policy has a lower monthly cost?

$j$	$X_{1j}$	$X_{2j}$	$Z_j$
1	126.97	118.21	8.76
2	124.31	120.22	4.09
3	126.68	122.45	4.23
4	122.66	122.68	-0.02
5	127.23	119.40	7.83

$$\begin{aligned}\bar{Z}(n) &= \frac{1}{5} \sum_{j=1}^5 Z_j = 4.98 \\ S^2(n) &= \frac{1}{4} \sum_{j=1}^5 (Z_j - 4.98)^2 = 12.19 \\ \alpha &= 0.10 \\ t_{4, 0.95} &= 2.132 \\ [Z_{\min}, Z_{\max}] &= 4.98 \pm 3.33 = [1.65, 8.31]\end{aligned}$$

Because  $Z_{\min} > 0$ , we can say that policy 1 has a higher average total monthly cost and policy 2 has a lower average total monthly cost.

## 3 Discrete Random Variables

### 3.1 Bernoulli

A Bernoulli is a useful random variable with only two outcomes: fail or succeed. The only parameter is  $p$ , the probability of success.

$$P\{X = 0\} = 1 - p$$

$$P\{X = 1\} = p$$

$$E[X] = p$$

The most likely estimator is  $\hat{p}$ .

$$\hat{p} = \bar{X}(n)$$

The sum of  $t$  Bernoulli variables is a binomial random variable with parameters  $t$  and  $p$ .

### 3.2 Binomial

A binomial random variable describes the number of successes in multiple trials (e.g., the yield of a batch). The parameters are  $t$ , the number of trials, and  $p$ , the probability of success. For  $x = 0, 1, \dots, t$ :

$$P\{X = x\} = \frac{t!}{x!(t-x)!} p^x (1-p)^{(t-x)}$$

$$E[X] = pt$$

If  $t$  is known, the most likely estimator is  $\hat{p}$ .

$$\hat{p} = \bar{X}(n)/t$$

### 3.3 Geometric

A geometric random variable describes the number of failures before the first success (e.g., the number of items inspected before the first defect) or the demand in a period of time. The only parameter is  $p$ , the probability of success. For  $x = 0, 1, \dots$ :

$$P\{X = x\} = p(1-p)^x$$

$$E[X] = \frac{1-p}{p}$$

The most likely estimator is  $\hat{p}$ .

$$\hat{p} = \frac{1}{\bar{X}(n) + 1}$$

### 3.4 Poisson

A Poisson random variable describes the number of events (arrivals, orders, events) that occur in a unit period of time. The only parameter is  $\lambda$ , the rate. For  $x = 0, 1, \dots$ :

$$P\{X = x\} = \frac{e^{-\lambda} \lambda^x}{x!}$$

$$E[X] = \lambda$$

The most likely estimator is  $\hat{\lambda}$ .

$$\hat{\lambda} = \bar{X}$$

The sum of multiple Poisson random variables is a Poisson random variable whose rate is the sum of the component rates.

## 4 Continuous Random Variables

### 4.1 Uniform

The uniform random variable  $U(a, b)$  can be used to approximate a random variable that varies between the values  $a$  and  $b$ . The  $U(0, 1)$  distribution is used to generate other distributions. For  $a \leq x \leq b$ ,

$$f(x) = \frac{1}{b-a}$$

$$E[X] = \frac{a+b}{2}$$

The most likely estimators are  $\hat{a}$  and  $\hat{b}$ .

$$\hat{a} = \min\{X_i\}$$

$$\hat{b} = \max\{X_i\}$$

## 4.2 Exponential

Exponential random variables can be used to describe interarrival times of customers, processing times, and time to failure and time to repair. The only parameter is  $\beta$ , the mean (e.g., hours), which is the reciprocal of  $\lambda$ , the rate (e.g., customers per hour).

For  $x \geq 0$ ,

$$f(x) = \frac{1}{\beta} e^{-x/\beta}$$

$$E[X] = \beta$$

The most likely estimator is  $\hat{\beta}$ :

$$\hat{\beta} = \bar{X}(n)$$

The exponential distribution is a special case of the gamma and Weibull distributions.

## 4.3 Gamma

Gamma random variables can be used to describe service and repair times. There are two parameters:  $\alpha$  and  $\beta$ . If  $\alpha$  is an integer  $m$ , then the gamma distribution is called the  $m$ -Erlang distribution, which describes the sum of  $m$  exponential random variables with mean  $\beta$ . If  $\alpha = 1$ , then the gamma distribution is the exponential distribution. For  $x > 0$ ,

$$f(x) = \frac{\beta^{-\alpha} x^{\alpha-1} e^{-x/\beta}}{\Gamma(\alpha)}$$

$$E[X] = \alpha\beta$$

If  $\alpha$  is a positive integer,  $\Gamma(\alpha) = (\alpha - 1)!$ . For the most likely estimators, see Law and Kelton (page 332).

## 4.4 Normal

Normal random variables can be used to describe random errors and variables that are the sum of many other variables. There are two parameters: the mean  $\mu$  and the variance  $\sigma^2$ . The  $N(0, 1)$  is the standard normal random variable  $Z$ . If  $X = \sigma Z + \mu$ , then  $X$  has mean  $\mu$  and variance  $\sigma^2$ . The most likely estimators are  $\hat{\mu}$  and  $\hat{\sigma}^2$ :

$$\hat{\mu} = \bar{X}(n)$$

$$\hat{\sigma}^2 = \frac{1}{n} \sum (X_i - \bar{X}(n))^2$$

## 4.5 Triangular

A triangular random variable can be used to approximate a random variable that varies between the values  $a$  and  $b$  and has a definite mode  $c$ . It can also be used when there are no data present: Let  $a$  be the minimum possible (or pessimistic) value, let  $b$  be the maximum possible (or optimistic) value, and let  $c$  be the most likely value.

For  $a \leq x \leq c$ ,

$$f(x) = \frac{2(x - a)}{(b - a)(c - a)}$$

For  $c \leq x \leq b$ ,

$$f(x) = \frac{2(b - x)}{(b - a)(b - c)}$$

$$E[X] = \frac{a + b + c}{3}$$

# 5 Discrete Event Systems: Deterministic Models

## 5.1 Definitions

An event is something that occurs at some time and changes the state of the system. An event can be a deliberate action, a spontaneous occurrence, or a condition being satisfied.

A discrete-event system has a discrete state space, and the transitions between states occur at discrete times, and these transitions are events. Since the states form a discrete set, there are no time-based derivatives that control the evolution of the system. Instead, we have events that cause system to jump from one state to another.

Examples include queueing systems, computer systems, communication systems, manufacturing systems, traffic systems, and database management systems. A machine in a factory may be busy, idle, or down. A warehouse has a certain number of items. A restaurant has a certain number of customers. Many board games have a discrete state space (chess, Trivial Pursuit, Scrabble).

## 5.2 Finite State Automata

Finite state automata (FSA) are the simplest models of discrete-event systems. These models can be used to answer the following types of questions: What happens if a

certain sequence of events occurs? How does the state change? How can we prevent the system from reaching undesirable states?

A FSA  $G$  can be described by the parameters  $(E, X, \delta, x_0, F)$ .  $E$  is the set of events.  $X$  is the finite state space.  $\delta$  is a transition function  $x_{i+1} = \delta(x_i, e_{i+1})$ , where  $e_{i+1} \in E$ .  $x_0$  is the initial state.  $F$  is a subset of  $X$ , and it includes the “final” states.

Given an initial state  $x_0$  and a sequence of events  $e_1, \dots, e_n$ , the new state  $x_n$  is a function of these events:  $x_1 = \delta(x_0, e_1)$ ,  $x_2 = \delta(x_1, e_2)$ ,  $\dots$ ,  $x_n = \delta(x_{n-1}, e_n)$ . For now we ignore when each event occurs.

Consider the following machine repair example.  $E$  contains four events: start processing ( $a$ ), complete processing without fail ( $b$ ), complete processing with an error ( $c$ ), and repair machine ( $d$ ).  $X$  has four states: idle ( $x_n = 1$ ), busy ( $x_n = 2$ ), down ( $x_n = 3$ ), and jammed ( $x_n = 4$ ). The following transitions are feasible:

- $\delta(1, a) = 2$  Start processing
- $\delta(2, b) = 1$  Complete processing without fail
- $\delta(2, c) = 3$  Complete processing with an error
- $\delta(3, d) = 1$  Repair machine
- $\delta(3, a) = 4$  Cause jam by trying to start a down machine

If  $x_0 = 1$ , then the event sequence  $abacda$  is feasible and leads to  $x_6 = 2$ . For another example of a FSA, see Sethi *et al.* [14].

### 5.3 State Automata

A state automata (SA) is similar to a FSA, but the state space  $X$  and the set of events  $E$  can be countably infinite.

### 5.4 Timed State Automata

A Timed State Automata (TSA) adds a clock structure to a SA. A clock structure defines the time from the activation of an event until its occurrence. For event  $i \in E$ ,  $V_{ij}$  is the time between the  $j$ -th activation and the  $j$ -th occurrence.

We can specify a TSA with the following parameters:  $(E, X, \Gamma, f, x_0, V)$ .  $E$  is the countable set of events  $\{1, 2, \dots\}$ .  $X$  is the countable set of states.  $\Gamma(x)$  is the set of events that are feasible when the system is in state  $x$ .  $f(x, i)$  is the state transition function that specifies the next state when event  $i \in \Gamma(x)$  occurs.  $x_0$  is the initial state.  $V = \{V_1, V_2, \dots\}$  is the clock structure. Associated with event  $i \in E$  is

$V_i = \{V_{i1}, V_{i2}, \dots\}$ . All  $V_{ij}$  are positive real numbers.

An event  $i$  is activated in two cases:

1. From state  $x$ ,  $i \in \Gamma(x)$ , event  $i$  occurs, the new state is  $x'$ , and  $i \in \Gamma(x')$ .
2. From state  $x$ ,  $i$  is not in  $\Gamma(x)$ , event  $j$  occurs, the new state is  $x'$ , and  $i \in \Gamma(x')$ .

Note that  $i$  is not activated if, from state  $x$ ,  $i \in \Gamma(x)$  and event  $j$  occurs, the new state is  $x'$ , and  $i \in \Gamma(x')$ . (It just remains alive from a previous activation.)

Let  $N_i$  be the number of times that event  $i$  has been activated.

The TSA dynamics proceed as follows.

1. The initial state is  $x = x_0$ , and for all  $i \in \Gamma(x_0)$ ,  $y_i = V_{i1}$  and  $N_i = 1$ . For all  $i$  not in  $\Gamma(x_0)$ ,  $y_i = \infty$  and  $N_i = 0$ .  
 $t = 0$ .
2. Identify the triggering event  $e'$ , which is the event  $i \in \Gamma(x)$  with the minimum  $y_i$ . Let  $y^* = y_{e'}$ . (If there is a tie, choose the lowest-numbered event.)
3. Determine the time  $t'$  of the next event and the next state  $x'$ :

$$t' = t + y_{e'}$$

$$x' = f(x, e')$$

4. Update the count and the residual clock times for all  $i \in E$ .
  - If  $i \in \Gamma(x)$  and  $i \in \Gamma(x')$ ,  $e' \neq i$ , then subtract  $y^*$  from  $y_i$ .
  - If  $e' = i$  and  $i \in \Gamma(x')$ , then add one to  $N_i$  and  $y_i = V_{i, N_i}$ .
  - If  $i$  is not in  $\Gamma(x)$  and  $i \in \Gamma(x')$ , then add one to  $N_i$  and  $y_i = V_{i, N_i}$ .
  - If  $i$  is not in  $\Gamma(x')$ , then  $y_i = \infty$ .
5. Let  $t = t'$  and  $x = x'$ , and go to Step 2.

Consider the following example, with two events  $\{1, 2\}$  and three states  $\{0, 1, 2\}$ . The initial state  $x_0 = 0$ . The following transitions are feasible:

$$\delta(0, 1) = 1$$

$$\delta(0, 2) = 2$$

$$\delta(1, 1) = 2$$

$$\delta(2, 1) = 1$$

$$\delta(2, 2) = 0$$

Let  $V_1 = \{1.0, 1.5, 1.5, \dots\}$ . Let  $V_2 = \{2.0, 0.5, 1.5, \dots\}$ .

$e'$	$y^*$	$t$	$x$	$y_1$	$y_2$	$N_1$	$N_2$
		0.0	0	1.0	2.0	1	1
1	1.0	1.0	1	1.5	$\infty$	2	1
1	1.5	2.5	2	1.5	0.5	3	2
2	0.5	3.0	0	1.0	1.5	3	3
1	1.0	4.0	1	$v_{14}$	$\infty$	4	3

A specified clock structure determines a unique sequence of events. One can view the clock structure  $V$  as the input. Whereas the state of the TSA at time  $t$  is  $x$  and  $y$ . (We don't have to know  $N_i$  as long as we know the future input, the remaining clock structures.)

## 6 Discrete-Event Systems: Stochastic Models

### 6.1 Stochastic Timed State Automata

Stochastic Timed State Automata (STSA) add a stochastic clock structure to Timed State Automata. A STSA generates a generalized semi-Markov process (GSMP). The STSA replaces the deterministic clock structure  $V_i$  with a sequence of independent, identically distributed random variables  $\{V_{i1}, V_{i2}, \dots\}$ . These random variables have a common distribution function  $G_i$ . We assume that the variables in  $V_i$  are independent from the variables in any other  $V_j$  and that  $G_i$  is a continuous function.

Also, the initial state  $x_0$  may be a random variable with discrete distribution  $p_0(x)$ ,  $x \in X$ .

Also, the transition function may be stochastic, so that  $p(x'; x, e')$  is the probability that the new state is  $x'$  if event  $e'$  occurs from state  $x$ .

Thus, a STSA has the following parameters:  $(\bar{E}, \bar{X}, \Gamma, p, p_0, G)$ .  $\bar{E}$  is the countable set of events,  $\bar{X}$  is the countable set of states,  $\Gamma(x)$  is the set of events enabled in state  $x$ ,  $p(x'; x, e')$  is the state transition probability, and  $G = \{G_i, i \in \bar{E}\}$  is the stochastic clock structure.

Note that if  $e'$  is not in  $\Gamma(x)$ , we define  $p(x'; x, e')$  as zero.

STSA dynamics are similar to TSA dynamics, but the following differences exist:

- The initial state  $x_0$  is obtained by sampling the distribution function  $p_0$  of  $X_0$ .

- When event  $e'$  occurs in state  $x$ , the next state  $x'$  is obtained by sampling the condition distribution  $p(x'; x, e')$ .
- When an event  $i$  is activated, the new clocktime  $V_{i, N_i}$  is obtained by sampling the distribution  $G_i$ .

For example, consider a two-machine, one-person machine interference system. There are three events:  $a$  occurs when Machine 1 fails.  $b$  occurs when Machine 2 fails.  $c$  occurs when the attendant repairs a machine. There are five states:  $x = 0$  when both machines are working.  $x = 1$  when Machine 1 is down.  $x = 2$  when Machine 2 is down.  $x = (1, 2)$  when both machines are down, and the attendant is trying to repair Machine 1.  $x = (2, 1)$  when both machines are down, and the attendant is trying to repair Machine 2. See the Figure 1 for the state transitions.

Suppose that  $G_a$  and  $G_b$  are exponential distributions with rate  $\lambda_a = \lambda_b = 0.5$  (and thus mean 2), and  $G_c$  is an exponential distribution with rate  $\lambda_c = 2$  (and thus mean 0.5).  $p_0(0) = 1$ , so we know the system starts with two working machines.

We sample the random distributions to get the following clock structures:  
 $V_a = \{1.1, 1.0, 10.3, \dots\}$ .  $V_b = \{1.8, 0.9, 4.2, \dots\}$ .  $V_c = \{0.6, 1.2, 0.5, \dots\}$ .

Event  $a$  occurs at  $t = 1.1$  and  $t = 2.7$ . Event  $b$  occurs at  $t = 1.8$  and  $t = 3.9$ . Event  $c$  occurs at  $t = 1.7$  and  $t = 3.0$  and  $t = 3.5$ . See Figure 2.

## 6.2 Stochastic Processes

A stochastic process is a collection of random variables that describe the evolution of a system over time:

$$\{X(t), t \in T\}$$

where  $T$  is an index set that represents time.  $T$  can be either discrete or continuous.  $X(t)$  is the state of the system at time  $t$ . The state space is the set of all possible values of  $X(t)$ . The state space can be discrete or continuous.

For example, in a queueing system,  $X(t)$  is the number of customers in the system at time  $t \geq 0$ . This process has discrete state space and continuous time. Or,  $X(t)$  could be the total amount of rain that has fallen by the  $t$ -th day of the year. This process has a continuous state space and discrete time.

A Markov process is a special case where the state of the system at some time  $t_k$  is the only information needed to predict the state of the system at time  $t > t_k$ . All past information is irrelevant, and how long the system has been in the current state is irrelevant. When the state space is discrete, this type of process is called a Markov

chain. This memoryless property is also called the Markov property. (See also Cassandras, page 194.)

### 6.3 Generalized Semi-Markov Process

STSA model systems and describe their dynamics. A Generalized Semi-Markov Process (GSMP) is the (continuous-time) stochastic process  $\{X(t), t \geq 0\}$  generated by a STSA  $(\bar{E}, \bar{X}, \Gamma, p, p_0, G)$ . GSMPs describe the behavior of many stochastic discrete-event systems.

For example, a simple queueing system can be modeled by the following STSA:

- $\bar{E}$  has two events: arrival ( $a$ ) and departure ( $d$ ).
- $\bar{X} = \{0, 1, 2, \dots\}$ .
- $\Gamma(0) = \{a\}$ .
- $\Gamma(x) = \{a, d\}$ , for  $x > 0$ .
- $p(x'; x, a) = 1$  if  $x' = x + 1$  and 0 otherwise.  $p(x'; x, d) = 1$  if  $x' = x - 1$  and  $x > 0$ , and 0 otherwise.
- $p_0(x) = 1$  if  $x = 0$  and 0 otherwise.
- $G_a$  is the distribution of interarrival times.  $G_d$  is the distribution of service times.

$X(t)$ , the number of customers in the system at time  $t$ , is the corresponding stochastic process. Note that if  $G_a$  and  $G_d$  are both exponential distributions, then we get the important M/M/1 queueing system.

### 6.4 Poisson Process

A Poisson process is a useful special case. Consider the following STSA:

- $\bar{E}$  has one event  $e$ .
- $\bar{X} = \{0, 1, 2, \dots\}$ .
- $\Gamma(x) = \{e\}$ , for  $x \geq 0$ .

- $p(x'; x, e) = 1$  if  $x' = x + 1$  and 0 otherwise.
- $p_0(x) = 1$  if  $x = 0$  and 0 otherwise.
- $G_e$  is an exponential distribution with rate  $\lambda$ .

The corresponding stochastic process  $N(t)$ , for  $t \geq 0$ , is the number of times that event  $e$  has occurred in the interval  $(0, t]$ .

It has the following properties.  $N(0) = 0$ .  $N(t)$  is a non-negative number.  $N(s) \leq N(t)$  if  $s \leq t$ .  $N(t) - N(s)$  is independent of  $N(u) - N(t)$  if  $s \leq t \leq u$ . (The time intervals are independent.)  $N(t) - N(s)$  depends upon only the difference  $t - s$  and not the absolute value of  $s$  or  $t$ . (The process is stationary.)

$$P_n(t) = P\{N(t) = n\} = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$$

for  $t \geq 0$  and  $n = 0, 1, 2, \dots$ . Then,  $E[N(t)] = \lambda t$  and  $Var[N(t)] = \lambda t$ . (Note that both go to  $\infty$  as  $t$  approaches  $\infty$ .)

$G_e$ , the distribution of the time between two events, is an exponential distribution with rate  $\lambda$ .  $E[V_{ek}] = 1/\lambda$ .  $Var[V_{ek}] = 1/\lambda^2$ . The interevent times are memoryless:

$$P\{V_{ek} > t : V_{ek} > s\} = P\{V_{ek} > t - s\} = e^{-\lambda(t-s)}$$

Proof. Let  $U$  be an exponentially distributed random variable with rate  $\lambda$ .  $F(u) = P\{U \leq u\} = 1 - e^{-\lambda u}$ ,  $u \geq 0$ .  $P\{U > u\} = 1 - F(u) = e^{-\lambda u}$ ,  $u \geq 0$ .

$$\begin{aligned} P\{U > t + u : U > t\} &= \frac{P\{U > t + u, U > t\}}{P\{U > t\}} \\ &= \frac{P\{U > t + u\}}{P\{U > t\}} \\ &= \frac{e^{-\lambda(t+u)}}{e^{-\lambda t}} \\ &= e^{-\lambda u} \\ &= P\{U > u\} \end{aligned}$$

Thus,  $F(t + u : U > t) = F(u)$ .

The time  $t$  between the  $n$ -th event and the  $n + k$ -th event is a  $k$ -Erlang random variable. This has a gamma distribution with  $\alpha = k$  and  $\beta = 1/\lambda$ . The mean is  $\alpha\beta = k/\lambda$ .

### 6.4.1 Superposition

If a system has a set  $E'$  of two (or more) events that are always enabled, and each event's occurrence is a Poisson process, then we can form a new Poisson process by superposition. Let  $\lambda_i$  be the rate of the Poisson process corresponding to event  $i$ . Let  $N(t)$  be the number of occurrences of ALL events in  $E'$ . Then  $N(t)$  is a Poisson process whose rate  $\Lambda = \sum_{i \in E'} \lambda_i$ .

This makes sense intuitively. Consider a queueing system with two types of customers. If type 1 customers are arriving at a rate of  $\lambda_1 = 4$  customers per hour, and type 2 customers are arriving at a rate of  $\lambda_2 = 3$  customers per hour, then customers are arriving at a rate  $\lambda_1 + \lambda_2 = 7$  customers per hour.

### 6.4.2 Decomposition

If the event  $e$  occurs at a rate  $\lambda$ , and when  $e$  occurs, there is a probability  $p_a$  that event  $a$  occurs and a probability  $p_b$  that event  $b$  occurs ( $p_a + p_b = 1$ ), then this forms two new Poisson processes. Event  $a$  occurs at a rate  $p_a\lambda$ . Event  $b$  occurs at a rate  $p_b\lambda$ .

To see this, consider  $P\{N_a(t) = n\}$ , the probability that event  $a$  occurs  $n$  times by time  $t$ .

$$\begin{aligned}
 P\{N_a(t) = n\} &= \sum_{k=n}^{\infty} P\{N_a(t) = n : N_e(t) = k\} P\{N_e(t) = k\} \\
 &= \sum_{k=n}^{\infty} \frac{k!}{n!(k-n)!} p_a^n p_b^{k-n} \frac{e^{-\lambda t} (\lambda t)^k}{k!} \\
 &= \frac{(p_a \lambda t)^n}{n!} e^{-\lambda t} \sum_{j=0}^{\infty} \frac{(p_b \lambda t)^j}{j!} \\
 &= \frac{(p_a \lambda t)^n}{n!} e^{-\lambda t} e^{p_b \lambda t} \\
 &= \frac{(p_a \lambda t)^n}{n!} e^{-p_a \lambda t}
 \end{aligned}$$

## 6.5 Markov Chain

In general, if all of the  $G_i$  in a STSA are exponential distributions (each with rate  $\lambda_i$ ), then we have a Poisson clock structure (Cassandras, page 228).

Let  $Y$  be the time that the system remains in state  $x$ . Let  $\Lambda(x) = \sum_{i \in \Gamma(x)} \lambda_i$ .

$$\begin{aligned}
 P\{Y \leq t\} &= 1 - P\{Y > t\} \\
 &= 1 - P\{\min_{i \in \Gamma(x)} Y_i > t\} \\
 &= 1 - P\{Y_1 > t\}P\{Y_2 > t\} \dots \\
 &= 1 - e^{-\lambda_1 t} e^{-\lambda_2 t} \dots \\
 &= 1 - e^{-\Lambda(x)t}
 \end{aligned}$$

Thus,  $Y$  is exponentially distributed with a rate  $\Lambda(x)$  that is the sum of the rates of the events enabled in state  $x$ . The time until the next event is independent of how long the system has been in the current state. The probability of event  $i$  being the next event is  $\lambda_i/\Lambda(x)$ , if  $i \in \Gamma(x)$ .

The GSMP generated by an STSA with a Poisson clock structure is a Markov chain.

Given that the system is in state  $x$ , the probability  $p(x', x)$  that the next state is  $x'$  can be derived as follows:

$$p(x', x) = \sum_{i \in \Gamma(x)} p(x'; x, i) \frac{\lambda_i}{\Lambda(x)}$$

Note that the determination of the next state is completely independent of all past history and the time spent in the current state.

The set of these transition probabilities (for all pairs of states) is a concise way to describe the behavior of a Markov chain.

## 6.6 Discrete-Time Markov Chains

### 6.6.1 Basic Properties

Let us start by considering systems where the state changes only at discrete points in time:  $\{t_0, t_1, t_2, t_3, \dots\}$ . Let  $X_k$  be the state of the system at time  $t_k$ ,  $k = 0, 1, 2, \dots$ . Let  $p_{ij}(k)$  be the probability that  $X_{k+1} = j$  given that  $X_k = i$ . Let  $p_{ij}(k, k+n)$  be the probability that  $X_{k+n} = j$  given that  $X_k = i$ . This is an  $n$ -step transition probability.

Then, the Chapman-Kolmogorov equation provides a way to calculate  $p_{ij}(k, k+n)$ . For any  $u$  such that  $k < u < k+n$ :

$$p_{ij}(k, k+n) = \sum_{r \in X} p_{ir}(k, u) p_{rj}(u, k+n)$$

Now, if the transition probabilities  $p_{ij}$  do not change over time, then we can create a matrix  $P$  of these one-step transition probabilities. Element  $i, j$  of the matrix  $P^2$  is the two-step transition probability that  $X_{k+2} = j$  given  $X_k = i$ . In general, element  $i, j$  of the matrix  $P^n$  is the  $n$ -step transition probability that  $X_{k+n} = j$  given  $X_k = i$ .

Consider the following example of a two-processor computer system that operates in distinct time slots. The two processors are identical. In any time slot, the probability that a job is submitted equals  $\alpha$ , and the probability that no job is submitted equals  $1 - \alpha$ . If both processors are free, then processor 1 takes the job. If both processors are busy, then the job leaves without service. Otherwise, the free processor takes the job. In any time slot, the probability that a busy processor completes a job is  $\beta$ . Let  $X_k$  be the number of busy processors at slot  $k$ .

- $p_{00} = 1 - \alpha$ . No job arrives.
- $p_{01} = \alpha$ . A job arrives.
- $p_{02} = 0$ .
- $p_{10} = \beta(1 - \alpha)$ . Job completes and no job arrives.
- $p_{11} = \beta\alpha + (1 - \beta)(1 - \alpha)$ . Job completes and job arrives, or no job completes and no job arrives.
- $p_{12} = (1 - \beta)\alpha$ . No job completes but job arrives.
- $p_{20} = \beta^2(1 - \alpha)$ . Both jobs complete and no job arrives.
- $p_{21} = \beta^2\alpha + 2\beta(1 - \beta)(1 - \alpha)$ . Both jobs complete and one job arrives; or one job completes, one does not, and no job arrives.
- $p_{22} = 2\beta(1 - \beta)\alpha + (1 - \beta)^2$ . One job completes, one does not, and one job arrives; or neither job completes.

If  $\alpha = 0.5$  and  $\beta = 0.7$ , then the matrix  $P$  has the following entries:

$$\begin{array}{ccc} 0.5 & 0.5 & 0 \\ 0.35 & 0.5 & 0.15 \\ 0.245 & 0.455 & 0.3 \end{array}$$

If we knew that the system started empty,  $X_0 = 0$ . If  $\Pi(k)$  is the vector of state probabilities, then  $\Pi(0) = [1, 0, 0]$ .  $\Pi(1) = \Pi(0)P = [0.5, 0.5, 0]$ , so the probability that  $X_1 = 1$  equals 0.5. For any point  $k$ ,  $\Pi(k) = \Pi(0)P^k$ :

$$\begin{aligned} \Pi(0) &= [1, 0, 0] \\ \Pi(1) &= [0.5, 0.5, 0] \end{aligned}$$

$$\begin{aligned}\Pi(10) &= [0.399, 0.495, 0.106] \\ \Pi(100) &= [0.399, 0.495, 0.106]\end{aligned}$$

Now  $\Pi(100)$  remains the same even if  $\Pi(0) = [1/3, 1/3, 1/3]$ . This “steady-state” distribution is independent of the initial state.

**State Holding Times.** Let  $V(i)$  be the number of time steps spent in state  $i$  after the system visits that state and before it leaves.  $V(i)$  is a discrete random variable that can be any positive integer.  $V(i) - 1$  (the number of failures before a success) has a geometric distribution, with  $p = 1 - p_{ii}$ .

$$P\{V(i) = n\} = (1 - p_{ii})p_{ii}^{n-1}$$

### 6.6.2 State Classification

A state  $j$  is reachable from state  $i$  if there exists some  $n \geq 1$  such that  $(P^n)_{ij} > 0$ .

A subset  $S$  of  $X$  is closed if, for all  $i \in S$  and  $j$  not in  $S$ ,  $p_{ij} = 0$ . If the subset  $S$  has one state  $i$ , this state is called an absorbing state.  $p_{ii} = 1$ .

A closed subset  $S$  is irreducible if, for all  $i \in S$  and  $j \in S$ ,  $j$  is reachable from  $i$ . If the entire state space  $X$  is irreducible, then we have an irreducible Markov chain. The two-processor computer example is an irreducible Markov chain.

Assume that  $X_0 = i$ . Consider the probability  $p_i$  that the system will return to state  $i$ . If this probability is equal to 1, the state is a recurrent state. Otherwise, the state is a transient state.

Example, consider the following transition matrix  $P$ :

$$\begin{array}{ccc} 0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0 \\ 0 & 0 & 1 \end{array}$$

State 2 is a recurrent state (it is also absorbing). States 0 and 1 are transient:  $p_0 = 0.5$  and  $p_1 = 0.75$ .

If  $X$  is a finite state space, then at least one state is a recurrent state. If  $i$  is a recurrent state, and  $j$  is reachable from  $i$ , then  $j$  is a recurrent state.

If the subset  $S$  is a finite, closed, irreducible set, then every state in  $S$  is recurrent.

However, the expected time until the next visit might be infinity. If so, the state is null-recurrent. If the expected next visit time is finite, then the state is positive recurrent.

A state can be periodic as well. Consider the set  $\{n > 0 : (P^n)_{ii} > 0\}$ . If the greatest common divisor (GCD) of the  $n$  in this set is greater than one, state  $i$  has a period equal to the GCD. Otherwise the state is aperiodic.

If a Markov chain is irreducible, then every state has the same period. Thus, if any state is aperiodic, then all of the states are aperiodic.

### 6.6.3 Steady-State Analysis

If a Markov chain is irreducible and aperiodic, then the steady-state distribution  $\Pi$  exists and it is independent of  $\Pi(0)$ .

If the Markov chain has transient or null-recurrent states, then  $\Pi_j = 0$  for all  $j$ .

If the irreducible, aperiodic Markov chain has only positive recurrent states, then  $\Pi_j > 0$  for all  $j$ ,  $\sum_j \Pi_j = 1$ , and  $\Pi = \Pi P$ .

That is,  $\Pi$  is a left eigenvector of  $P$  corresponding to the eigenvalue of  $\lambda = 1$ .

If the irreducible, aperiodic Markov chain is finite, then all of the states are positive recurrent and the steady-state distribution exists. In addition to solving the above system of equations, we can find  $\Pi$  by calculating  $\Pi(k) = \Pi(0)P^k$  for large values of  $k$ , since  $\Pi(k)$  converges to  $\Pi$ .

## 6.7 Continuous-Time Markov Chains

A STSA with a Poisson clock structure generates a continuous-time Markov chain. Denote as  $e_{ij}$  the event that causes the system to move from state  $i$  to state  $j$ . Let  $\lambda_{ij}$  be the rate at which this event occurs. Let  $\Lambda(i) = \sum_{j \neq i} \lambda_{ij}$ .

For example, consider a two-machine system with one operator. Machine failures occur at a rate of 0.5 failures per hour per machine. The operator can repair machines at a rate of 2.0 repairs per hour. The state is the number of down machines. The state space is  $\{0, 1, 2\}$ . There are four events:  $e_{01}$ ,  $e_{12}$ ,  $e_{10}$ , and  $e_{21}$ .  $\lambda_{01} = 1$ .  $\lambda_{12} = 0.5$ .  $\lambda_{10} = 2$ .  $\lambda_{21} = 2$ .

In general, let  $p_{ij}(s, s+t)$  be the probability that  $X(s+t) = j$  if  $X(s) = i$ . For

time-homogeneous systems,  $p_{ij}(s, s + t) = p_{ij}(t)$  for all  $s$ .

$$\begin{aligned}\sum_j p_{ij}(t) &= 1 \\ p_{ii}(0) &= 1 \\ p_{ij}(0) &= 0 \text{ if } j \neq i\end{aligned}$$

If  $P(t)$  is the matrix with elements  $p_{ij}(t)$ , then  $P(0) = I$ . The Chapman-Kolmogorov equation states that  $P(s + t) = P(s)P(t)$ . We can derive the following differential equations for  $P(t)$ :

$$\begin{aligned}P(t + \delta t) &= P(t)P(\delta t) \\ P(t) &= P(t)P(0) \\ \frac{P(t + \delta t) - P(t)}{\delta t} &= P(t) \frac{P(\delta t) - P(0)}{\delta t} \\ \frac{dP(t)}{dt} &= P(t)Q\end{aligned}$$

Note  $Q = \frac{dP(t)}{dt}$  at  $t = 0$ . Thus, for  $t \geq 0$ ,

$$P(t) = e^{Qt}$$

Note that  $e^{Qt} = I + Qt + Q^2t^2/2! + \dots$ . Let  $w$  be the column vector with all elements equal to 1. Now, because  $\sum_j p_{ij}(t) = 1$ ,  $P(t)w = w$ . Differentiating at  $t = 0$  yields  $Qw = 0$ , so  $\sum_j q_{ij} = 0$  for all  $i$ . Thus,

$$q_{ii} = -\sum_{j \neq i} q_{ij}$$

What is  $q_{ij}$ ?  $p_{ij}(t)$  is the probability that event  $e_{ij}$  occurs before time  $t$ . This equals the probability that some event occurs multiplied by the probability that the event is  $e_{ij}$ .

$$\begin{aligned}p_{ij}(t) &= P\{X(t) = j : X(0) = i\} \\ &= (1 - e^{-\Lambda(i)t}) \frac{\lambda_{ij}}{\Lambda(i)} \\ q_{ij} &= \frac{dp_{ij}(t)}{dt} \\ &= \Lambda(i)e^{-\Lambda(i)t} \frac{\lambda_{ij}}{\Lambda(i)} \\ &= \lambda_{ij}e^{-\Lambda(i)t}\end{aligned}$$

Since  $t = 0$ ,  $q_{ij} = \lambda_{ij}$ .

$$q_{ii} = - \sum_{j \neq i} \lambda_{ij} = -\Lambda(i)$$

If  $V(i)$  is the amount of time spent in state  $i$  when it is visited,  $V(i)$  is exponentially distributed with a rate  $\Lambda(i)$ .

Now consider the state distribution  $\Pi(t)$ , where  $\Pi_i(t)$  is the probability that  $X(t) = i$ .

$$\begin{aligned} \Pi(t) &= \Pi(0)P(t) \\ \frac{d\Pi(t)}{dt} &= \Pi(0)\frac{dP(t)}{dt} \\ &= \Pi(0)P(t)Q \\ &= \Pi(t)Q \end{aligned}$$

If the Markov chain is irreducible and has positive recurrent states, then a steady-state distribution  $\Pi$  exists. Since  $\frac{d\Pi}{dt} = 0$ ,

$$\Pi Q = 0$$

Also,  $\sum_i \Pi_i = 1$ .

This equation implies that, for all  $j$ ,

$$\begin{aligned} \sum_i \Pi_i q_{ij} &= 0 \\ \sum_{i \neq j} \Pi_i q_{ij} + \Pi_j q_{jj} &= 0 \\ \sum_{i \neq j} \Pi_i q_{ij} &= -\Pi_j q_{jj} \\ \sum_{i \neq j} \Pi_i q_{ij} &= \Pi_j \sum_{k \neq j} q_{jk} \end{aligned}$$

This last equation states that the total rate of transitions into state  $j$  equals the total rate of transitions out of state  $j$ , in the steady-state.

In the two-machine example described above,  $Q$  has the following elements:

$$\begin{array}{ccc} -1 & 1 & 0 \\ 2 & -2.5 & 0.5 \\ 0 & 2 & -2 \end{array}$$

Setting  $\Pi Q = 0$  yields the following equations:

$$\begin{aligned}
-\Pi_0 + 2\Pi_1 &= 0 \\
\Pi_0 - 2.5\Pi_1 + 2\Pi_2 &= 0 \\
0.5\Pi_1 - 2\Pi_2 &= 0 \\
\Pi_0 + \Pi_1 + \Pi_2 &= 1
\end{aligned}$$

With a little algebra, we get  $3.75\Pi_1 = 1$ .  $\Pi = [8/13, 4/13, 1/13]$ . The expected number of machines down is  $4/13 + 2(1/13) = 6/13$ .

## 6.8 Queueing Systems: Introduction

Queues are the symptom of a problem: insufficient or inefficient service. Everyday examples include customers waiting for service, cars waiting at an intersection, jobs waiting for processing or transport, airplanes waiting to takeoff, people waiting for organ transplants, and court cases waiting for judges.

The customer's arrival time is the time when the customer enters the system. Then, the customer departs from the queue, and then (after service) the customer departs from the system. The customer's time in system includes the time in queue and the time in service (or processing time). If the customer has a due date, the the customer's tardiness is the departure time minus the due date (if positive) and zero otherwise.

The arrival rate is the average number of customers that enter the system in a period of time. A server's throughput is the actual rate of service, the average number of customers served in a period of time. Under most conditions, the throughput equals the arrival rate. The queue length is the average number of customers waiting for service. Estimating queue lengths is important when designing a system, as we need to know how much space to allocate to buffers and queues. Server utilization is the fraction of time that each server is busy serving customers.

Let  $W_q(i)$  be the time that the  $i$ -th customer spends in the queue. Let  $W_s(i)$  be the time that the  $i$ -th customer spends in the system. Then  $W_q(i) \leq W_s(i)$  for all  $i$ . Let  $W_q$  be the average queue time per customer. Let  $W_s$  be the average time in system per customer.

Let  $A(t)$  be the number of customers that have arrived by time  $t$ . Let  $D_s(t)$  be the number of customers that have left the system by time  $t$ . Let  $D_q(t)$  be the number of customers that have left the queue by time  $t$ . Then, for all  $t$ ,  $A(t) \geq D_q(t) \geq D_s(t)$ .

Let  $L_s(t)$  be the number of customers that are in the system at time  $t$ . Let  $L_q(t)$  be

the number of customers that are in the queue at time  $t$ .

$$L_q(t) = A(t) - D_q(t)$$

$$L_s(t) = A(t) - D_s(t)$$

Let  $L_q$  be the time-average number of customers in the queue. Let  $L_s$  be the time-average number of customers in the system.

Suppose  $n$  customers arrive and are served in the time period  $[a, b]$ .

$$W_q = \frac{1}{n} \sum_{i=1}^n W_q(i)$$

$$W_s = \frac{1}{n} \sum_{i=1}^n W_s(i)$$

$$L_q = \frac{1}{b-a} \int_a^b L_q(t) dt$$

$$L_s = \frac{1}{b-a} \int_a^b L_s(t) dt$$

Let the arrival rate  $\lambda = n/(b-a)$ . Since  $\int_a^b L_q(t) dt = \sum_{i=1}^n W_q(i)$  and  $\int_a^b L_s(t) dt = \sum_{i=1}^n W_s(i)$ , we have Little's Law:

$$\begin{aligned} L_q &= \frac{1}{b-a} \sum_{i=1}^n W_q(i) \\ &= \frac{n}{b-a} W_q \\ &= \lambda W_q \end{aligned}$$

Similarly,

$$L_s = \lambda W_s$$

Little's Law holds for any queue discipline, including first-in-first-out (FIFO). It also holds for any configuration of interconnected queues and servers, where  $L_s$  is the average number of customer in the system, and  $W_s$  is the average time in the system (the cycle time), and  $\lambda$  is the rate at which customers enter the system.

See Example A.

## 6.9 M/M/1 Queue

The M/M/1 queueing system is the simplest queueing system, but understanding its behavior provides much insight into other queueing systems. It is called the M/M/1

because interarrival times are exponentially distributed (and thus have the Markov property) and service times are exponentially distribution (and have the Markov property). The “1” states that the system has just one server.

The interarrival times are exponentially distributed with rate  $\lambda$ , and the service times are exponentially distributed with rate  $\mu$ . Define  $\rho = \lambda/\mu$ . This fraction is a very important quantity.

The M/M/1 queueing system is a continuous-time Markov chain with  $\lambda_{i,i+1} = \lambda$ , for  $i \geq 0$ , and  $\lambda_{i,i-1} = \mu$ , for  $i \geq 1$ . If  $\mu > \lambda$ , the chain has positive recurrent states, and it is irreducible, so the steady-state distribution exists. From  $\pi Q = 0$  we can derive the following relationships:

$$\begin{aligned} -\lambda\pi_0 + \mu\pi_1 &= 0 \\ \lambda\pi_0 - (\mu + \lambda)\pi_1 + \mu\pi_2 &= 0 \\ \lambda\pi_1 - (\mu + \lambda)\pi_2 + \mu\pi_3 &= 0 \\ \lambda\pi_2 - (\mu + \lambda)\pi_3 + \mu\pi_4 &= 0 \end{aligned}$$

And so forth. This simplifies into the following set of relationships:

$$\begin{aligned} -\lambda\pi_0 + \mu\pi_1 &= 0 \\ -\lambda\pi_1 + \mu\pi_2 &= 0 \\ -\lambda\pi_2 + \mu\pi_3 &= 0 \\ -\lambda\pi_3 + \mu\pi_4 &= 0 \end{aligned}$$

Thus,  $\pi_{i+1} = \rho\pi_i$  for all  $i \geq 0$ .

$$\begin{aligned} \sum_{i \geq 0} \pi_i &= 1 \\ \sum_{i \geq 0} \rho^i \pi_0 &= 1 \\ \frac{\pi_0}{1 - \rho} &= 1 \\ \pi_0 &= 1 - \rho \end{aligned}$$

Note that the above infinite sum converges if and only if  $\rho < 1$ . Therefore,  $\pi_n$  is the steady-state probability that the system has  $n$  customers, for  $n \geq 0$ , IF  $\rho < 1$ .

$$\pi_n = (1 - \rho)\rho^n$$

Since  $\pi_0 = (1 - \rho)$ , the probability of the system being empty decreases as  $\lambda$  approaches  $\mu$ . The server utilization  $u = 1 - \pi_0 = \rho$ . The following equations describe  $L_q$ , the average number in the queue, and  $L_s$ , the average number in system.

$$L_q = \sum_{n=1}^{\infty} (n-1)\pi_n = \frac{\rho^2}{1-\rho}$$

$$L_s = \sum_{n=0}^{\infty} n\pi_n = \frac{\rho}{1-\rho}$$

From Little's Law, we can determine the average queue time  $W_q$  and the average time in system (cycle time)  $W_s$ :

$$W_q = \frac{L_q}{\lambda} = \frac{\rho}{\mu(1-\rho)}$$

$$W_s = \frac{L_s}{\lambda} = \frac{1}{\mu(1-\rho)}$$

Note that all terms approach  $\infty$  as  $\lambda$  approaches  $\mu$  and  $\rho$  approaches 1. As  $\lambda$  and  $\rho$  approach 0,  $W_q$  approaches 0, and  $W_s$  approaches  $1/\mu$ . See Figure 3 for an example.

The distribution of  $W_s(i)$  is an exponential distribution with rate  $1/W_s = \mu(1 - \rho)$ . The distribution of  $W_q(i)$  is more complicated.

Consider the departure rate  $\delta$ . If the system is empty, no customers leave the system. If the system is busy, customers leave at rate  $\mu$ . Thus, the average departure rate  $\delta = (1 - \rho)0 + \rho\mu = \lambda$ , so the departure process is equivalent to the arrival process. The throughput equals the arrival rate (IF  $\rho < 1$ ).

For example, if  $\lambda = 100$  customers per hour, and  $\mu = 140$  customers per hour,  $\rho = 5/7$ . The probability of the system being empty is  $2/7$ . The throughput is 100 customers per hour.

$$L_q = (25/49)/(2/7) = 1.786 \text{ customers}$$

$$L_s = (5/7)/(2/7) = 2.5 \text{ customers}$$

$$W_q = 1.786/100 = 0.018 \text{ hours}$$

$$W_s = 2.5/100 = 0.025 \text{ hours}$$

## 6.10 M/M/m Queuing Systems

A M/M/m queuing system has exponentially distributed interarrival times and exponentially distributed service times. There are  $m$  identical servers. Arriving customers can be served by any server. If all  $m$  servers are busy, then the customers form one queue.

The interarrival times are exponentially distributed with rate  $\lambda$ , and the service times are exponentially distributed with rate  $\mu$ . Redefine  $\rho$  as follows:

$$\rho = \frac{\lambda}{m\mu}$$

The M/M/m queueing system is a continuous-time Markov chain with  $\lambda_{i,i+1} = \lambda$ , for  $i \geq 0$ .  $\lambda_{i,i-1} = i\mu$ , for  $1 \leq i \leq m$ .  $\lambda_{i,i-1} = m\mu$ , for  $i > m$ . If  $m\mu > \lambda$ , the chain has positive recurrent states, and it is irreducible, so the steady-state distribution exists. From  $\pi Q = 0$  we can derive the following relationships, if  $\rho < 1$  ( $\lambda < m\mu$ ):

$$\begin{aligned}\pi_0 &= \left(1 + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!} + \frac{(m\rho)^m}{m!(1-\rho)}\right)^{-1} \\ \pi_n &= \pi_0 \frac{(m\rho)^n}{n!} \text{ for } 1 \leq n \leq m \\ \pi_n &= \pi_0 \frac{m^m \rho^n}{m!} \text{ for } n > m\end{aligned}$$

For  $m = 2$ ,  $\pi_0 = \frac{1-\rho}{1+\rho}$ . For  $m = 3$ ,  $\pi_0 = \frac{2(1-\rho)}{2+4\rho+3\rho^2}$ .

The average number of busy servers is  $\frac{\lambda}{\mu}$ , so the server utilization is  $\rho$ . The following equations describe the average queue time  $W_q$  and the average time in system (cycle time)  $W_s$ :

$$\begin{aligned}W_q &= \frac{1}{\mu} \frac{(m\rho)^m \pi_0}{m! m(1-\rho)^2} \\ W_s &= \frac{1}{\mu} + W_q\end{aligned}$$

From Little's Law, we can determine  $L_q$ , the average number in the queue, and  $L_s$ , the average number in system.

$$\begin{aligned}L_q &= \lambda W_q \\ L_s &= \lambda W_s\end{aligned}$$

Note that all terms approach  $\infty$  as  $\lambda$  approaches  $m\mu$  and  $\rho$  approaches 1.

For  $m = 2$ ,  $W_q = \frac{1}{\mu} \frac{\rho^2}{1-\rho^2}$ .

Which is better? Two servers or one server that is twice as fast?

$$W_s = \frac{1}{\mu} + \frac{1}{\mu} \frac{\rho^2}{1-\rho^2} = \frac{1}{\mu(1-\rho^2)}$$

For the M/M/1, let  $\mu^+ = 2\mu$ , then  $\rho^+ = \rho$ .

$$W_s^+ = \frac{1}{\mu^+(1-\rho^+)} = \frac{1}{2\mu(1-\rho)}$$

So the ratio of cycle time in M/M/2 to cycle time in M/M/1 is a function of  $\rho$ :

$$\frac{W_s}{W_s^+} = \frac{2}{1 + \rho}$$

When  $\lambda$  and  $\rho$  are small, the M/M/1 system is better, because there is little queueing and the server is twice as fast. As  $\lambda$  and  $\rho$  increase, the advantage decreases.

## 6.11 Other Single-Stage Queueing Systems

The general queueing system GI/G/m has  $m$  servers. The interarrival times are independent and have a general distribution. The service times have a general distribution. In general this type of system does not form a Markov chain, and exact analytical models are hard to find. Thus, many workers have created approximate mathematical models. There are a range of approximations. See, for instance, Buzacott and Shanthikumar [3], Hall [11], or Hopp and Spearman [7].

The following approximations are based on the models for the M/M/1 and M/M/m systems. Let  $t_a$  be the mean interarrival time and  $c_a^2$  be the squared coefficient of variation (SCV) for the interarrival times. Let  $t_e$  be the mean service time and  $c_e^2$  be the SCV of the service times. Let  $\lambda = 1/t_a$  and  $\mu = 1/t_e$ . Then,  $\rho = t_e/(mt_a)$ . (For the M/M/m systems,  $c_a^2 = c_e^2 = 1$ .)

Let  $q(m, \rho) = \mu W_q$  for an M/M/m system. For instance,  $q(1, \rho) = \rho/(1 - \rho)$  and  $q(2, \rho) = \rho^2/(1 - \rho^2)$ . For larger values of  $m$ , the formula is more complex, but it can be calculated exactly. See Section 6.10 or Panico [13].

The following equations describe the average queue time  $W_q$  and the average time in system (cycle time)  $W_s$ :

$$\begin{aligned} W_q &= \frac{c_a^2 + c_e^2}{2} q(m, \rho) t_e \\ W_s &= W_q + t_e \end{aligned}$$

From Little's Law, we can determine  $L_q$ , the average number in the queue, and  $L_s$ , the average number in system.

$$\begin{aligned} L_q &= \lambda W_q \\ L_s &= \lambda W_s \end{aligned}$$

## 6.12 Markovian Queueing Networks

A queueing network has a set of workstations through which customers flow. If all of the workstations are M/M/1 and M/M/m servers, then the queueing network is a Markovian queueing network.

For example, consider a simple two-stage network. Each stage has a single server. The processing times at the first station are exponentially distributed with a rate  $\mu_1$ . The processing times at the second station are exponentially distributed with a rate  $\mu_2$ . Arriving customers require processing at the first station and then at the second station. The interarrival times at the first station are exponentially distributed with a rate  $\lambda$ . Let  $\rho_1 = \lambda/\mu_1$  and  $\rho_2 = \lambda/\mu_2$ . (Note that  $\lambda < \mu_1$  and  $\lambda < \mu_2$ .)

It can be shown that the departure process of a M/M/1 or M/M/m queueing system is also a Poisson process (and of course it has the same rate). The arrivals to the second stage are the departures from the first stage, so it is also a Poisson process, so the second stage is also a M/M/1 queueing system.

Let  $\pi(n_1, n_2)$  be the steady-state probability that the first stage has  $n_1$  customers and that the second stage has  $n_2$  customers. One can show the following:

$$\begin{aligned}\pi(n_1, n_2) &= (1 - \rho_1)\rho_1^{n_1}(1 - \rho_2)\rho_2^{n_2} \\ &= \pi_1(n_1)\pi_2(n_2)\end{aligned}$$

where  $\pi_1$  and  $\pi_2$  are the steady-state probability distributions of each M/M/1 queueing system. This decomposition of the steady-state probabilities for the system is called the product form solution.

The product form solution exists for a wide range of open queueing networks. This makes analyzing the entire system quite easy. First determine the flows  $\lambda_i$  entering and leaving the workstations and then examine each workstation individually to determine its performance based on the number of servers  $m_i$  and the service rate  $\mu_i$ . Of course,  $\lambda_i < m_i\mu_i$  for all workstations.

The average number of customers in the whole system equals the sum of the average number at each workstation. Little's Law can then be used to determine the average time in system.

Consider the following three-node example: Arriving customers go to station 1. The external arrival rate is  $r = 1$  customer per hour. The service rate  $\mu_1 = 4$  customers per hour.

The destination of a customer leaving station 1 is random: it may return to station 1 or go to station 2 or go to station 3. The probabilities are  $p_{11} = 0.3$ ,  $p_{12} = 0.2$ , and  $p_{13} = 0.5$ .

At station 2, the service rate  $\mu_2 = 4$  customers per hour. The destination of a customer leaving station 2 is again random: it may return to station 2 or go back to station 1. The probabilities are  $p_{21} = 0.2$ ,  $p_{22} = 0.8$ .

At station 3, the service rate  $\mu_3 = 2$  customers per hour. All customers leave the system after service at station 3.

Modeling the flow yields the following system of equations:

$$\lambda_1 = r + p_{11}\lambda_1 + p_{21}\lambda_2 \quad (1)$$

$$\lambda_2 = p_{12}\lambda_1 + p_{22}\lambda_2 \quad (2)$$

$$\lambda_3 = p_{13}\lambda_1 \quad (3)$$

The solution follows:

$$\lambda_1 = \frac{r}{p_{13}} \quad (4)$$

$$\lambda_2 = \frac{p_{12}r}{p_{21}p_{13}} \quad (5)$$

$$\lambda_3 = r \quad (6)$$

The utilization of station  $i$  is  $\rho_i = \lambda_i/\mu_i$ . In this example,  $\rho_1 = \rho_2 = \rho_3 = 0.5$ . The expected number of customers at each workstation equals 1, and the expected number in the whole system equals 3.

## 6.13 Complex Queueing Networks

For a complex system that has multiple single-stage queueing systems and multiple products (but no re-entrant flow), we can use the following approximation model [12].

The model requires the following data: For each workstation, the number of resources available, and the mean time to failure and mean time to repair a resource. For each existing product and the new product, the job size (number of parts) and the desired throughput (number of parts per hour of factory operation). The sequence of workstations that each job must visit. The mean setup time (per job) at each workstation and its variance. The mean processing time (per part) at each

workstation and its variance. The yield at each workstation that a job must visit (the ratio of good parts produced to parts that undergo processing).

- $I$  = the set of all products
- $T_i$  = desired throughput of product  $i$  (parts per hour)
- $B_i$  = job size of product  $i$  at release
- $c_i^r$  = SCV of job interarrival times for product  $i$
- $J$  = the set of all stations
- $n_j$  = the number of resources at station  $j$
- $m_j^f$  = mean time to failure for a resource at station  $j$
- $m_j^r$  = mean time to repair for a resource at station  $j$
- $R_i$  = the sequence of stations that product  $i$  must visit
- $R_{ij}$  = the subsequence that precedes station  $j$
- $t_{ij}$  = mean part process time of product  $i$  at station  $j$
- $c_{ij}^t$  = SCV of the part process time
- $s_{ij}$  = mean job setup time of product  $i$  at station  $j$
- $c_{ij}^s$  = SCV of the setup time
- $y_{ij}$  = yield of product  $i$  at station  $j$

The model first calculates, for each product, the processing time of each job at each station. It also calculates, for each station, the average processing time, weighted by each product's arrival rate. Finally, it modifies the aggregate processing times by adjusting for the resource availability.

- $Y_{ij}$  = cumulative yield of product  $i$  through  $R_{ij}$
- $Y_i$  = cumulative yield of product  $i$  through  $R_i$
- $x_i$  = release rate of product  $i$  (jobs per hour)
- $A_j$  = availability of a resource at station  $j$
- $V_j$  = the set of products that visit station  $j$
- $t_{ij}^+$  = total process time of product  $i$  at station  $j$
- $c_{ij}^+$  = SCV of the total process time
- $t_j^+$  = aggregate process time at station  $j$
- $c_j^+$  = SCV of the aggregate process time
- $t_j^*$  = modified aggregate process time at station  $j$
- $c_j^*$  = SCV of the modified aggregate process time

$$Y_{ij} = \prod_{k \in R_{ij}} y_{ij} \quad (7)$$

$$Y_i = \prod_{k \in R_i} y_{ij} \quad (8)$$

$$x_i = T_i / (B_i Y_i) \quad (9)$$

$$A_j = \frac{m_j^f}{m_j^f + m_j^r} \quad (10)$$

$$V_j = \{i \in I : j \in R_i\} \quad (11)$$

$$t_{ij}^+ = B_i Y_{ij} t_{ij} + s_{ij} \quad (12)$$

$$(t_{ij}^+)^2 c_{ij}^+ = B_i Y_{ij} t_{ij}^2 c_{ij}^t + s_{ij}^2 c_{ij}^s \quad (13)$$

This last equation, which is used to calculate  $c_{ij}^+$ , holds because the variance of the total process time is the sum of the variance of the part process times and the variance of the job setup time.

$$t_j^+ = \frac{\sum_{i \in V_j} x_i t_{ij}^+}{\sum_{i \in V_j} x_i} \quad (14)$$

$$(t_j^+)^2 (c_j^+ + 1) = \frac{\sum_{i \in V_j} x_i (t_{ij}^+)^2 (c_{ij}^+ + 1)}{\sum_{i \in V_j} x_i} \quad (15)$$

$$t_j^* = \frac{t_j^+}{A_j} \quad (16)$$

$$c_j^* = c_j^+ + 2A_j(1 - A_j) \frac{m_j^r}{t_j^+} \quad (17)$$

The arrival process at each station depends upon the products that visit the station. Some products are released directly to the station, while others arrive from other stations. The departure process depends upon the arrival process and the service process.

$V_{0j}$  = the set of products that visit station  $j$  first

$V_{hj}$  = the set of products that visit station  $h$  immediately before  $j$

$\lambda_j$  = total job arrival rate at station  $j$

$\lambda_{hj}$  = arrival rate at station  $j$  of jobs from station  $h$

$q_{hj}$  = proportion of jobs from station  $h$  that next visit station  $j$

$c_j^a$  = SCV of interarrival times at station  $j$

$c_j^d$  = SCV of interdeparture times at station  $j$

$$\lambda_j = \sum_{i \in V_j} x_i \quad (18)$$

$$\lambda_{hj} = \sum_{i \in V_{hj}} x_i \quad (19)$$

$$q_{hj} = \lambda_{hj} / \lambda_h \quad (20)$$

$$c_j^d = 1 + \frac{u_j^2}{\sqrt{n_j}}(c_j^* - 1) + (1 - u_j^2)(c_j^a - 1) \quad (21)$$

$$c_j^a = \sum_{h \in J} ((c_h^d - 1)q_{hj} + 1) \frac{\lambda_{hj}}{\lambda_j} + \sum_{i \in V_{0j}} c_i^r \frac{x_i}{\lambda_j} \quad (22)$$

Solving the above set of equations yields the complete set of  $c_j^a$  and  $c_j^d$  for all stations.

If the shop is a flow shop, and all products visit the same sequence of stations, then we can renumber the stations  $1, 2, \dots, J$ .  $V_j = I$  and  $V_{j-1,j} = I$  for all stations, and the last equation can be simplified as follows:

$$c_1^a = \frac{\sum_{i \in I} c_i^r x_i}{\sum_{i \in I} x_i} \quad (23)$$

$$c_j^a = c_{j-1}^d, 2 \leq j \leq J \quad (24)$$

The approximation calculates the utilization and the average cycle time ( $W_s$ ) at each workstation. The utilization incorporates the capacity requirements, while the average cycle time reflects the congestion and variability that are present.

$$\begin{aligned} u_j &= \text{the average resource utilization at station } j \\ CT_j^* &= \text{the average cycle time at station } j \end{aligned}$$

$$u_j = \frac{\lambda_j t_j^*}{n_j} \quad (25)$$

$$CT_j^* = \frac{c_j^a + c_j^*}{2} q(n_j, u_j) t_j^* + t_j^* \quad (26)$$

The term  $q(n_j, u_j)$  is a coefficient that is derived from an exact analytical model for the M/M/ $n_j$  queueing system. For instance,  $q(1, u_j) = u_j / (1 - u_j)$  and  $q(2, u_j) = u_j^2 / (1 - u_j^2)$ . For larger values of  $n_j$ , the formula is more complex, but it can be calculated exactly. See, for instance, Section 6.10 or Panico [13].

The cycle time of a product is the sum of the workstation cycle times.

$CT_i$  = the average cycle time of jobs of product  $i$   
 $W_i$  = the average work-in-process inventory of product  $i$  (in parts)

$$CT_i = \sum_{j \in R_i} CT_j^* \quad (27)$$

$$W_i = T_i CT_i \quad (28)$$

## References

- [1] Blanchard, Benjamin S., and Walter J. Fabrycky, *Systems Engineering and Analysis*, Third Edition, Prentice Hall, Inc., Upper Saddle River, New Jersey, 1998.
- [2] Bradley, Stephen P., Arnolde C. Hax, and Thomas L. Magnanti, *Applied Mathematical Programming*, Addison-Wesley Publishing Co., Reading, Massachusetts, 1977.
- [3] Buzacott, J.A., and J.G. Shanthikumar, *Stochastic Models of Manufacturing Systems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1993.
- [4] Cassandras, Christos G., *Discrete Event Systems: Modeling and Performance Analysis*, Richard D. Irwin, Inc., Homewood, Illinois, 1993.
- [5] Cassandras, Christos G., and Stephane Lafortune, *Introduction to Discrete Event Systems*, Kluwer Academic Publishers, Norwell, Massachusetts, 1999.
- [6] Hazelrigg, George A., *Systems Engineering: an Approach to Information-Based Design*, Prentice-Hall, Upper Saddle River, New Jersey, 1996.
- [7] Hopp, Wallace J., and Mark L. Spearman, *Factory Physics*, Irwin/McGraw-Hill, Boston, 1996.
- [8] Law, Averill M., and W. David Kelton, *Simulation Modeling and Analysis*, Second Edition, McGraw-Hill, Inc., New York, 1991.
- [9] Miser, Hugh J., and Edward S. Quade, *Handbook of Systems Analysis*, North-Holland, New York, 1985.
- [10] Palm, William J., *Modeling, Analysis, and Control of Dynamic Systems*, second edition, John Wiley and Sons, Inc., New York, 1999.
- [11] Hall, Randolph W., *Queueing Methods*, Prentice-Hall, 1991.

- [12] Herrmann, Jeffrey W., and Mandar Chincholkar, "Design for Production: A Tool for Reducing Manufacturing Cycle Time," paper DETC2000/DFM-14002 in CD-ROM Proceedings of DETC 2000, 2000 ASME Design Engineering Technical Conference, pp. 1-10, Baltimore, September 10-13, 2000.
- [13] Panico, J.A., *Queuing Theory*, Prentice-Hall Inc., New Jersey, 1969.
- [14] Sethi, S.P., C. Sriskandarajah, G. Sorger, J. Blazewicz, W. Kubiak, "Sequencing of parts and robot moves in a robotic cell," *International Journal of Flexible Manufacturing Systems*, Volume 4, pages 331-358, 1992.