Systems Simulation
Chapter 4: Simulation Software

Fatih Cavdur
fatihcavdur@uludag.edu.tr

March 29, 2014

Introduction

- Software used to develop simulation models can be divided into three categories:
  - General-Purpose Programming Languages (C, C++, Java)
  - Simulation Programming Languages (GPSS/H, SIMAN V, SLAM II)
  - Simulation Environments
- How would/should you choose your software?
- Some criteria are cost, ease of learning, ease of use, power to model, animation capabilities etc.
- Here we will consider Java (first category), GPSS/H (second category) and some simulation software packages (third category).

History of Simulation Software

- The Period of Search (1955-1960): FORTRAN and others
- The Advert (1961-1965): GPSS, SIMSCRIPT, GASP
- The Formative Period (1966-1970): GPSS/360, SIMSCRIPT II
- The Expansion Period (1971-1978): GPSS/NORDEN, GASP IV
- The Period of Consolidation and Regeneration (1979-1986): SLAM II and SIMAN
- The period of Integrated Environments 1987-?

Selection of Simulation Software

- Do not focus on a single issue, such as ease of use.
- Execution speed is important.
- Beware of advertising claims and demonstrations.
- Ask the vendor to solve a small version of your problem.
- Beware of the "checklist" with "yes" and "no" as the entries.
- Simulation users ask whether the simulation model can link to and use code or routines written in external languages such as C, C++ or Java.
- There may be a significant trade-off between the graphical model-building environments and ones based on a simulation language.

Selection of Simulation Software-cont.

- Every two years, OR/MS Today publishes a simulation-software survey. The 2003 issue had 48 products, including simulation support packages such as input-data analyzers etc.
- There are many features that are relevant when selecting simulation software. Some of these are shown in Tables 4.1 to 4.5 in your text.
  - Model-Building Features
  - Runtime Environment
  - Animation and Layout Features
  - Output Features
  - Vendor Support and Product Documentation

The Checkout Counter (single-server queue)

- The system, a grocery checkout counter, is modeled as a single-server queue.
- The simulation will run until 1,000 customers have been served.
- Times between arrivals of customers are exponentially distributed with a mean of 4.5 minutes.
- Service times are normally distributed with a mean of 3.2 minutes and a standard deviation of 0.6 minutes.
- Simulation in Java, GPSS/H and SSF

Java Simulation-Main Components

- clock: a variable defining simulated time
- initialization method: a method to define system state at time 0
- event methods: for each event type, a method to update system state when that event occurs
- RV generators: methods to generate samples from probability distributions
- main program: to maintain overall control of the event-scheduling algorithm
- report generator: a method that computes summary stats and prints a report at the end of the simulation

Variables

Table : Variables

Type	Variable	Description
System State	`QueueLength`	number in queue
	`NumberInService`	number in service
Entity Attributes and Sets	Customers	FCFS queue
FEL	`FutureEventList`	priority-ordered list of pending events
Activity Durations	`MeanInterArrivalTime`	inter-arrival time between last two arrivals
	`MeanServiceTime`	service time of the most recent to begin service
Input Parameters	`MeanInterArrivalTime`	mean inter-arrival time (4.5 minutes)
	`MeanServiceTime`	mean service time (3.2 minutes)
	`SIGMA`	standard deviation of service time (0.6 minutes)
	`TotalCustomers`	the stopping criterion
Simulation Variables	`Clock`	current value of simulation time
	`LastEventTime`	time of occurrence of the last event
	`TotalBusy`	total busy time of server so far
	`MaxQueueLength`	max length of queue so far
	`SumResponseTime`	sum of response times for all departed so far
	`NumberOfDepartures`	number of departures far
Summary Stats	`RHO = BusyTime/Clock`	proportion of time server is busy
	`AVGR`	average response time

Functions and Methods

Table : Functions

Function	Description
`exponential(mu)`	function to generate exponential RVs
`normal(xmu, SIGMA)`	function to generate normal RVs

Table : Methods

Method	Description
`Initialization`	initialization
`ProcessArrival`	arrival event execution method
`ProcessDeparture`	departure event execution method
`ReportGeneration`	report generation

Classes

- Event Class [Click]
- EventList Class [Click]
- Queue Class [Click]
- Sim Class [Click]

Output

```
SINGLE SERVER QUEUE SIMULATION - GROCERY STORE CHECKOUT COUNTER

MEAN INTERARRIVAL TIME 4.5
MEAN SERVICE TIME 3.2
STANDARD DEVIATION OF SERVICE TIMES 0.6
NUMBER OF CUSTOMERS SERVED 1000
SERVER UTILIZATION 0.7154468565718423
MAXIMUM LINE LENGTH 9.0
AVERAGE RESPONSE TIME 6.943036600826695 MINUTES
PROPORTION WHO SPEND FOUR MINUTES OR MORE IN SYSTEM 0.667
SIMULATION RUNLENGTH 4474.645676201413 MINUTES
NUMBER OF DEPARTURES 1000
```


GPSS/H Program

```
      SIMULATE
      INTEGER          &LIMIT
      REAL             &IAT, &MEAN, &STDEV, &COUNT
      LET              &IAT = 4.5
      LET              &MEAN = 3.2
      LET              &STDEV = 0.6
      LET              &LIMIT = 100
*     ...
      GENERATE         RVEXPO(1, &IAT)
      QUEUE            SYSTIME
      QUEUE            LINE
      SEIZE            CHECKOUT
      DEPART           LINE
      ADVANCE          RVNORM(1, &MEAN, &STDEV)
      RELEASE          CHECKOUT
      DEPART           SYSTIME
      TERMINATE        1
*     ...
```

SSF Simulation

- The Scalable Simulation Framework (SSF) is an Application Program Interface (API) that describes a set of capabilities for object-oriented, process-view simulation.
- The API sparse and was designed to allow to achieve high performance (i.e., parallel processing)
- SSF API exist for both C++ and Java.
- SSF has a wide user base, particularly in network simulation by SSFNet.

SSF Simulation-cont.

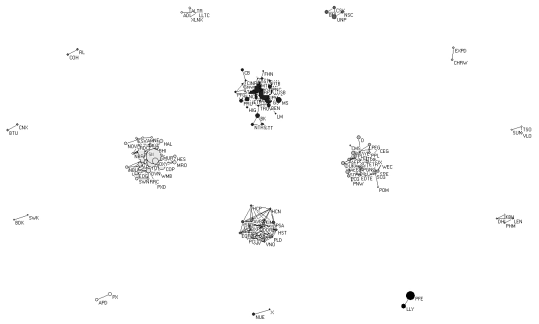
- The SSF API defines five base classes.
  - `process` is a class that implements threads of control
  - `entity` class is used to describe simulation objects
  - `inChannel` class is a communication endpoint
  - `outChannel` class is a communication endpoint
  - `Event` class defines messages sent between entities

About Network Simulation

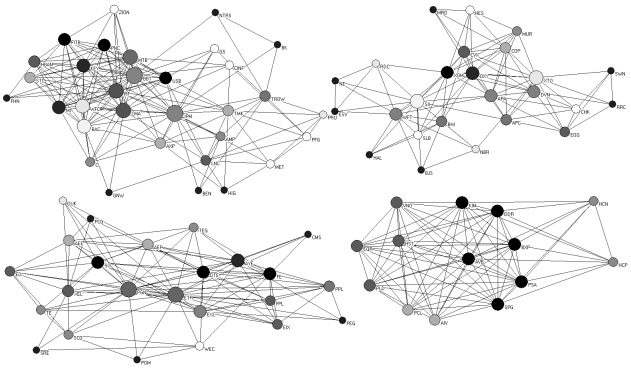
What can of simulations can you perform about networks?

- Network Reliability
  - Computer Networks
  - Wired-Wireless Communication Networks
  - Transportation Networks
- Propagation (Spread Process)
  - Virus Propagation in Wired-Wireless Communication/Computer Networks
  - Disease Spread in Human-Interaction Networks
  - News/Product Propagation in Social Networks

An Example Network Model



An Example Network Model-Closer Look



Simulation Software

- Arena
- AutoMod
- Extend
- Flexsim
- Micro Saint
- ProModel
- QUEST
- SIMUL8
- WITNESS

Arena

- We will use Arena for most applications for the class.
- The Basic Edition is targeted at modeling business processes and other systems in support of high-level analysis needs.
- The Standard Edition is designed for more detailed models of discrete and continuous systems.
- The Professional Edition enhances Arena SE with the capability to craft custom simulation objects that mirror components of the real system.
- At the heart of Arena is the SIMAN simulation language.
- Input, Output and Process Analyzer help with data analysis and comparison of design alternatives.

AutoMod

- It includes AutoMod simulation package, AutoStat for experimentation and analysis and AutoView for animation preparation.
- The main focus of manufacturing and material-handling systems.
- AutoMod's strength is in detailed, large models used for planning, operational decision support and control-systems testing.
- AutoStat works with AutoMod models to provide a complete environment for the user to define scenarios, conduct experimentation and perform analyses.

Extend

- Extend combines a block-diagram approach to model-building with a development environment for creating new blocks.
- Each Extend block has an icon and encapsulates code, parameters, user interface, animation etc.
- Extend includes a large set of elemental blocks; libraries of blocks for specific application areas.
- Models are built by placing and connecting blocks and entering the parameters on the block's dialog window.
- Extend has an open architecture; in most cases, the source code for blocks is available for custom development.

Flexsim

- Flexim is a discrete-event, object-oriented simulator developed in C++, using Open GL.
- It integrates Microsoft's Visual C++ IDE and compiler within a graphical 3-D click-and-drag simulation environment.
- Flexim is used to build models that behave like the actual physical or conceptual systems they represent.
- A simulation model of any flow system or process can be created in Flexim by using drag-and-drop model-building objects.

Micro Saint

- Micro Saint is a general-purpose, discrete-event network simulation-software package for building models to simulate real-life processes.
- It does not use the terminology or graphic representation of a specific industry.
- The terms that are used are defined by the user.
- In addition, the icons and backgrounds are customizable.
- It provides two views: Network Diagram and ActionView.
- OptQuest is included for optimization.

ProModel

- It is a simulation and animation tool to model manufacturing systems.
- ProModel's animation is automatically generated as the model is developed.
- ProModel has manufacturing-oriented modeling elements and rule-based decision logic.
- SimRunner can be used for optimization.
- OptQuest for ProModel is available as an add-on product.

QUEST

- QUEST (Queuing Event Simulation Tool) is a manufacturing-oriented simulation package.
- It combines an object-based, true 3-D simulation environment with a graphical user interface and material-flow modules.
- A QUEST model consists of elements from a number of element classes.
- Built-in element classes include AGVs and transporters, sub-resources, buffers, conveyors etc.
- QUEST Simulation Control Language (SCL) can be used for unique problems.

SIMUL8

- Models are created by drawing the flow of work, using a series of icons and arrows to represent the resources and queues in the system.
- Default values are provided for all properties of the icons, so that the animation can be viewed very early in the modeling process.
- The main focus of SIMUL8 is service industries where people are processing transactions.
- SIMUL8 saves model and data in XML format.
- It has a VBA interface and ActiveX/COM support.

WITNESS

- WITNESS has two separate versions for manufacturing and service industries. It contains many elements for discrete-part manufacturing and also contains elements for continuous processing.
- Models are based on template elements. These may be customized and combined into module elements and templates for reuse.
- It has object-model and ActiveX control for simulation embedding and includes direct data links to Microsoft Excel, MINITAB and any other OLEDB database source.

Common Features

- Virtually all simulation packages offer various degrees of support for statistical analysis of simulation outputs.
- Many packages have added optimization as an analysis tool.
- Optimization is used to find a "near-optimal" solution. The user defines an objective or a fitness function to optimize.
- Because of the random and non-linear structure, meta-heuristics can be useful for optimization (AI based approaches, neural networks, genetic algorithms etc.)

Products

- Arena's Output and Process Analyzer
- AutoStat
- OptQuest
- SimRunner

Summary

- Reading HW: Chapter 4.
- Chapter 4 Exercises.