Section 1.1: Introduction

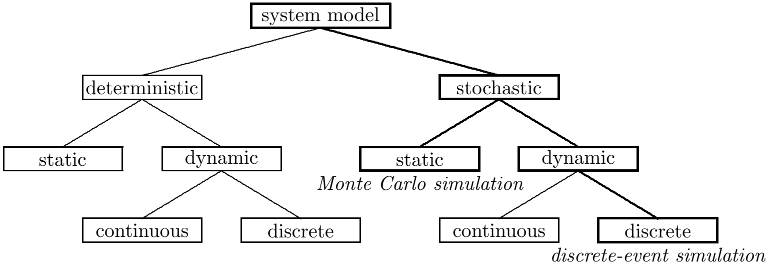
Discrete-Event Simulation: A First Course

0-13-142917-5

Introduction

- What is discrete-event simulation?
  - Modeling, simulation, and analyzing systems
  - Computation and mathematical techniques
- **Model**: conceptual framework describing a system
- **Simulate**: perform experiments using computer implementation of the model
- **Analyze**: draw conclusions from output

Characterizing a Model



system model

deterministic

stochastic

static

dynamic

continuous

discrete

static

Monte Carlo simulation

dynamic

continuous

discrete

discrete-event simulation

Characterizing a Model

- Deterministic or Stochastic
    - Does the model contain stochastic components?
    - Randomness is easy to add to a DES
- Static or Dynamic
    - Is time a significant variable?
- Continuous or Discrete
    - How does the system state evolve?
    - Continuous: classical mechanics
    - Discrete: queuing, inventory, machine shop models

Definitions

- Discrete-Event Simulation Model
    - Stochastic
    - Dynamic
    - Discrete-Event
- Monte Carlo Simulation
    - Stochastic
    - Static

DES Model Development

Algorithm 1.1.1 – How to develop a model:

1. Goals and objectives
2. Build a *conceptual* model
3. Convert into a *specification* model
4. Convert into a *computational* model
5. Verify
6. Validate

Typically an iterative process

Three Model Levels

- Conceptual
  - Very high level
  - How comprehensive should the model be?
  - What are the state variables?
- Specification
  - On paper
  - May involve equations, pseudocode, etc.
  - How will the model receive input?
- Computational
  - A computer program
  - General-purpose PL or simulation language?

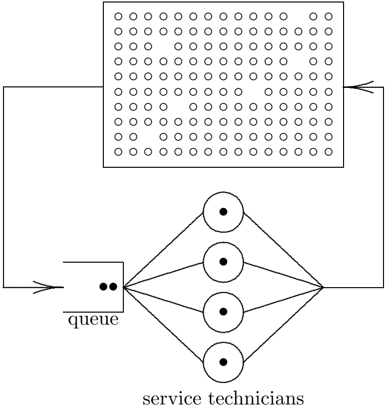
Verification vs. Validation

- Verification
  - Computational model should be consistent with specification model
  - Did we build the model right?
- Validation
  - Computational model should be consistent with the system being analyzed
  - Did we build the right model?
  - Can an expert distinguish simulation output from system output?
- Interactive graphics can prove valuable

A Machine Shop Model

- 150 identical machines:
    - Operate continuously, 8 hr/day, 250 days/yr
    - Operate independently
    - Repaired in the order of failure
    - Income: \$20/hr of operation
- Service technicians(s):
    - 2-year contract at \$52,000/yr
    - Each works 230 8-hr days/yr
- How many service technicians should be hired?

A Machine Shop Model: System Diagram



Algorithm 1.1.1 Applied

1. Goals and Objectives:
    - Find number of technicians for maximum profit
    - Extremes: one techie, one techie per machine
2. Conceptual Model:
    - State of each machine (failed, operational)
    - State of each techie (busy, idle)
    - Provides a high-level description of the system at any time
3. Specification Model:
    - What is known about time between failures?
    - What is the distribution of the repair times?
    - How will time evolution be simulated

Algorithm 1.1.1 Applied

4. Computational Model
    - Simulation clock data structure
    - Queue of failed machines
    - Queue of available techies
5. Verify
    - Software engineering activity
    - Usually done via extensive testing
6. Validate
    - Is the computational model a good approximation of the actual machine shop?
    - If operational, compare against the real thing
    - Otherwise, use consistency checks

Observations

- Make each model as simple as possible
    - Never simpler
    - Do not ignore relevant characteristics
    - Do not include extraneous characteristics
- Model development is not sequential
    - Steps are often iterated
    - For teams, steps may be in parallel
    - Do not merge verification and validation
- Develop models at three levels
    - Think a little, program a lot (and poorly);
    - Think a lot, program a little (and well).

Simulation Studies

Algorithm 1.1.2 Using the resulting model:

7. Design simulations experiments
    - What parameters should be varied?
    - perhaps many combinatoric possibilities
8. Make production runs
    - Record initial conditions, input parameters
    - Record statistical output
9. Analyze the output
    - Use common statistical analysis tools (Ch. 4)
10. Make decisions
11. Document the results

Algorithm 1.1.2 Applied

7. Design simulation experiments
    - Vary the number of technicians
    - What are the initial conditions?
    - How many replications are required?
8. Make production runs
    - Manage output wisely
    - Must be able to reproduce results exactly
9. Analyze the output
    - Observations are often correlated (not independent)
    - Take care not to derive erroneous conclusions

Algorithm 1.1.2 Applied

10. Make decisions
    - Graphical displays help
    - Implement the policy subject to external conditions
11. Document results
    - System diagram
    - Assumptions about failure and repair rates
    - Description of specification model
    - software
    - Tables and figures of output
    - Description of output analysis

DES can provide valuable insight about the system

Programming Languages

- General-purpose programming languages
    - Flexible and familiar
    - Well suited for learning DES principles and techniques
    - E.g.: C, C++, Java
- Special-purpose simulation Languages
    - Good for building models quickly
    - Provide built-in features (e.g., queue structures)
    - Graphics and animation provided
    - E.g.: Arena, Promodel

Terminology

Model vs. Simulation (noun)

- Model can be used with respect to conceptual, specification, or computational levels
- Simulation is rarely used to describe the conceptual or specification model
- Simulation is frequently used to refer to the computational model (program)

Model vs. Simulate (verb)

- To model can refer to development of the levels
- To simulate refers to the computational activity
- Do not merge verification and validation

Meaning should be obvious from the context