Class 3 Outline

Dynamic Discrete-Event Simulation:

1. Discrete-Event Framework

2. Software Demo (Simul8)

3. Modeling Examples
A G/G/1 Queueing System

- Inter-arrival time distribution is A
- Service time distribution is S
- Jobs/Customers
- FIFO Queue
- Single Server

• How to simulate this system?
Example: G/G/1 Queue

- **system_state**: number of jobs in system (in service + waiting)
- **next_event** is either
  - **new_arrival** (n_a): a new job arrives in the system; or
  - **service_completion** (s_c): the job that was in service is completed

<table>
<thead>
<tr>
<th>simulated time</th>
<th>other variables</th>
<th>system</th>
</tr>
</thead>
<tbody>
<tr>
<td>sim_clock = 0</td>
<td>system_state = 0</td>
<td><img src="image" alt="SystemState0Diagram" /></td>
</tr>
<tr>
<td></td>
<td>event_list = {(n_a, 1.2)}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>next_event = (n_a, 1.2)</td>
<td></td>
</tr>
<tr>
<td>sim_clock = 1.2</td>
<td>system_state = 1</td>
<td><img src="image" alt="SystemState1Diagram" /></td>
</tr>
<tr>
<td></td>
<td>event_list = {(n_a, 1.8); (s_c, 2.2)}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>next_event = (n_a, 1.8)</td>
<td></td>
</tr>
</tbody>
</table>
## Example: G/G/1 Queue

<table>
<thead>
<tr>
<th>Simulated Time</th>
<th>Other Variables</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>sim_clock = 1.8</td>
<td>system_state = 2</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>event_list = {(n_a, 2.6); (s_c, 2.2)}</td>
<td>next_event = (s_c, 2.2)</td>
<td>n_a → n_a</td>
</tr>
<tr>
<td>sim_clock = 2.2</td>
<td>system_state = 1</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>event_list = {(n_a, 2.6); (s_c, 3.7)}</td>
<td>next_event = (n_a, 2.6)</td>
<td>n_a → s_c</td>
</tr>
<tr>
<td>sim_clock = 2.6</td>
<td>system_state = 2</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>event_list = {(n_a, 3.5); (s_c, 3.7)}</td>
<td>next_event = (n_a, 3.5)</td>
<td>n_a → n_a</td>
</tr>
<tr>
<td>sim_clock = 3.5</td>
<td>system_state = 3</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td>event_list = {(n_a, 4.4); (s_c, 3.7)}</td>
<td>next_event = (s_c, 3.7)</td>
<td>n_a → n_a</td>
</tr>
</tbody>
</table>
Example: G/G/1 Queue

- At the simulated time point $\text{sim\_clock} = 3.7 = t_6$, the performance collection variable $\text{stat\_counters}$ might look like:

<table>
<thead>
<tr>
<th>event number</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>sim_clock $(t_i)$</td>
<td>0</td>
<td>1.2</td>
<td>1.8</td>
<td>2.2</td>
<td>2.6</td>
<td>3.5</td>
<td>3.7</td>
</tr>
<tr>
<td>system_state $(S_i)$</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

- If called at that point $t_6$, the routine $\text{report\_generator()}$ might perform an operation like:

$$Y = \frac{1}{t_6-t_0} \sum_{i=0}^{5} (t_{i+1} - t_i) S_i$$
Discrete-Event Algorithm

Variables:
- `system_state`
- `sim_clock` current value of simulated time.
- `event_list` contains the next time when each type of event will occur.
- `next_event` is the type of the next event that will occur.
- `stat_counters` contains historical performance/behavior information.

Subroutines:
- `initialization()`
- `timing(event_list)` determines `next_event` from `event_list` and advances `sim_clock` to the time this event occurs.
- `event_occurrence(next_event)` updates all variables according to the model logic when each particular type of event occurs.
- `rv_generation()` generates samples of random variables according to the probability distributions specified in the model.
- `report_generator(stat_counters)` computes estimates of the performance variables of interest from the `stat_counters` variable.
Discrete-Event Algorithm

- **initialization()**
  - all variables are initialized

- **timing(event_list)**
  - next_event is determined;
  - sim_clock is updated.

- **event_occurence(next_event)**
  - system_state is updated;
  - stat_counters is updated;
  - rv_generation() is used to update event_list.

- **termination test?**
  - based on number of iterations, precision achieved, etc…

- **report_generator(stat_counters)**
  - statistical analysis, graph, etc…

Copyright 2002 © Jérémie Gallien
ClearPictures Production Model

Box / Sensor Board Assembly

Sensor Firmware Test

Inspection

Customer

TRIAN[5,10,15]

U[15,25]

U[13,24]

ST1

ST2

U[10,15]

N[9.5,4]

15%

85%

360

Copyright 2002 © Jérémie Gallien
Label-Based Distribution for ST1

Set-up:

- A work item label *Distribution ST1* is created
- A probability distribution named *Sensor Test ST1* and equal to U[13,24] is created
- A probability distribution named *Firmware Adjustment after Inspection* and equal to U[10,15] is created

Logic:

- The work entry point initially sets the label *Distribution ST1* to *Sensor Test ST1* for all items
- The service time on work center Sensor Testing ST1 is defined as a *label-based distribution* associated with the label *Distribution ST1*
- The work center “label-changing dummy” has 0 service time but sets the label *Distribution ST1* to the value *Firmware Adjustment after Inspection*
What is Kanban?

Kanban Board

Production Stage

WIP

Downstream Demand

Production Stage

WIP

Downstream Demand

Copyright 2002 © Jérémie Gallien
A Pull Production System

Kanban Board

New orders

Completed orders

Supplier

RM

stage 1

stage 2

FG

back orders

Copyright 2002 © Jérémie Gallien
Reorder Point Policy

• Assume now a component replenishment lead-time LT (random) and a continuous (r,Q) policy:

\[ \text{LT} = \text{Lead Time} \]
\[ \text{EDDLT} = \text{Expected Demand During Lead Time} \]
\[ \text{DDL} = \text{(Actual) Demand During Lead Time} \]

(Order 1 placed)

(Order 1 received)

(Adapted from material by Thomas Roemer)

Copyright 2002 © Jérémie Gallien
Class 3 Wrap-Up

Industrial decision-making is interdisciplinary

Industrial Problem

Contextual knowledge
- Operations
- Finance
- Accounting

Methodological skills
- Probability
- Optimization

Decision Model

Software Implementation
- Solver
- Crystal Ball
- Simul8...

Output Analysis
- Sensitivity Analysis
- Statistics