Lecture 13: Anicard

CS 661: Simulation
Prof. Marvin Nakayama

Road Map

Objective: Analyze the ANICARD case problem.
- Understand relationship between different service objectives
- Performance of Dynamic routing

Road Map:
- Problem Description
- Performance Measures
- Initial size of operators pool using Erlang's formula
- Simulation results with and without dynamic routing
- Discussion of results and recommendations

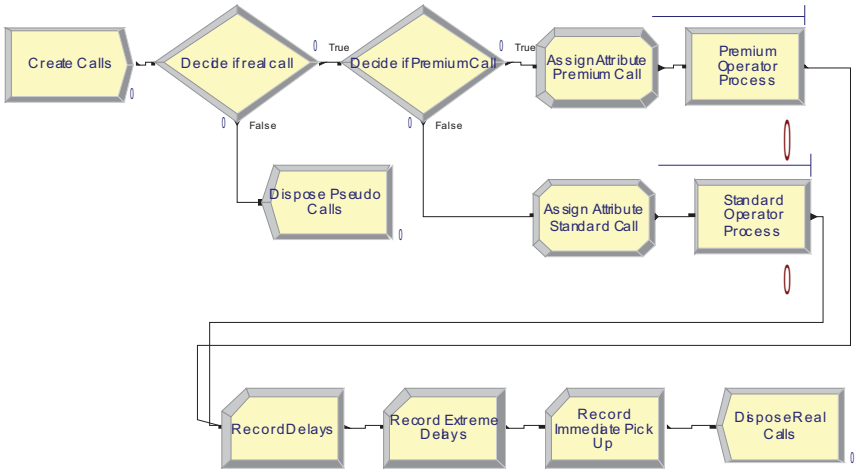
Problem Description

Problem: In early 1991, American North/East Interstate Bank Corp. decided to take full control of all its mail and telephone service operations.
- Design call center for its credit-card division.
- Success of center depends on quality of service.
- Main difficulties:
  - Set acceptable service objectives.
  - Translate service objectives to
    - number of lines
    - number of operators
  - Assess economic impact of objectives.
- Three major time scales: seasonal, weekly, daily.

Data for Problem:

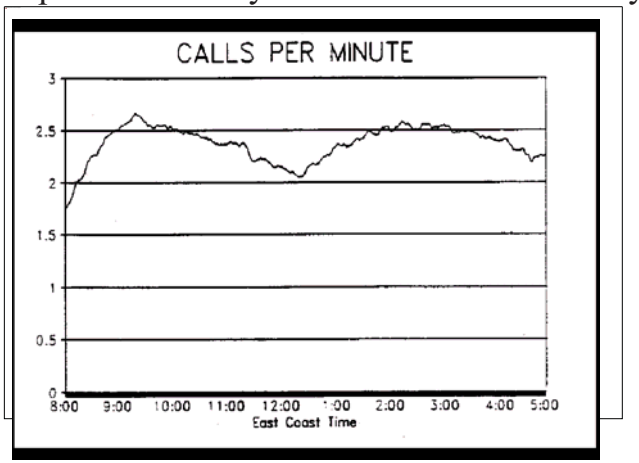
- *Two types of customers*
  - Standard 80%
  - Premium 20%
- *Peak average arrival rate* = 2.63 calls per minute.
- *Service time*
  - Same distribution for both classes
    - Mean of 1.5 min with probability 0.10
    - Mean of 6 min with probability 0.50
    - Mean of 4 min with probability 0.40

Arena Model



Time-Varying Arrival Rate

- Calls per minute vary as function of time of day

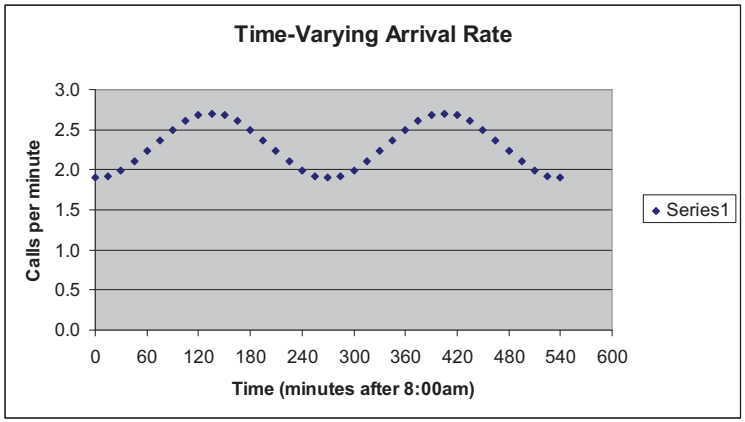


- How do we model this in Arena?

Time-Varying Arrival Rate (cont.)

- Arrival rate at time t (minutes after 8:00 am)

$$\lambda(t) = 2.23 - 0.4 \cos(2 \pi t / 270)$$



Approach 1 (Incorrect)

- Arrival rate at time t (minutes after 8:00 am)
  $\lambda(t) = 2.23 - 0.4 \cos(2 \pi t / 270)$
- In Create module, specify Time Between Arrivals as
  - *Type*: Random(Expo)
  - *Value*: 1/(2.23-(0.4*cos(6.2832*TNOW/270)))
- TNOW is variable defined by ARENA that keeps track of current value of simulation clock.
- This implementation in ARENA for interarrival times
  - is not quite correct (see pp. 7-123 to 7-127 of Lecture Notes 7),
  - but might be close enough in this context

Approach 2 (Correct): NHPP

- Arrival rate at time t (minutes after 8:00 am)
  $\lambda(t) = 2.23 - 0.4 \cos(2 \pi t / 270)$
- Define $\lambda_{max} = \max\{ \lambda(t) : t \geq 0\} = 2.63$
- Generate *homogeneous* Poisson process with constant rate λ_{max} ; i.e., EXPO($1/\lambda_{max}$) inter-event times.
- "Thinning" of arrivals:
  - If Poisson process has event at time t,
  - then accept the event as a "real" event with probability $\lambda(t)/\lambda_{max}$
- Reference: Lewis and Shedler (1979), "Simulation of Nonhomogeneous Poisson Processes by Thinning," Naval Research Logistics Quarterly, vol. 26, pp. 403-413.

Approach 2 (Correct): NHPP

- In *Create* module, specify Time Between Arrivals as
  - *Type*: Random(Expo)
  - *Value*: 1/2.63
- Use *Decide* module to "thin" arrivals:
  - *Type*: 2-way by Chance
  - *Percent True*: 100*(2.23-(0.4*cos(6.2832*TNow/270)))/2.63

Premium vs. Standard Calls

- 80% of calls are standard
- 20% of calls are premium
- Decide module:
  - *Name*: Decide if Premium Call
  - *Type*: 2-way by Chance
  - *Percent True*: 20
- Use Assign module to assign *attribute* CallType
  - 1 if premium call
  - 2 if standard call

Distribution for Call Duration

- Same distribution for both classes
  - Mean of 1.5 min with probability 0.10
  - Mean of 6 min with probability 0.50
  - Mean of 4 min with probability 0.40
- "Within each category, the standard deviation of call duration was found to be close to the mean."
- How do we model this?

Distribution for Call Duration

- Mixture of exponentials
  - Mean of 1.5 min with probability 0.10
  - Mean of 6 min with probability 0.50
  - Mean of 4 min with probability 0.40
- In Arena,

  EXPO(DISC(0.1, 1.5, 0.6, 6, 1.0, 4))

  or

  DISC(0.1, EXPO(1.5), 0.6, EXPO(6), 1.0, EXPO(4))

Service Standards

- *Service Objectives*
  - Average Delay
  - Immediate PickUp
  - Excessive Delay (initial threshold of 3 minutes)
- *Service Standards*

	Standard	***Premium***
Avg. Delay	1.5 min	0.25 min
IPU Rate	40%	90%
ED Rate	15%	1%

Tally Sets

- Use Record modules to collect
  - Separate statistics for Premium & Standard calls by using Sets
  - Think of Sets as arrays
  - Statistics on waiting times
- Set (from Basic Process template) is an Arena array.
  - Can define Set as collection of Tallies indexed by Set Index
  - Set Index can be an Attribute

Tally Sets (cont.)

- Record module:
  - *Name*: Record Delays
  - *Type*: Expression
  - *Value*: Entity.Waittime (Arena-defined Attribute)
  - *Record into Set*: Check
  - *Tally Set Name*: Delays
  - *Set Index*: CallType (which is defined as an Attribute in an Assign module; 1=prem, 2=std)

Tally Sets (cont.)

- Click on Set module (from Basic Process template)
- Spreadsheet appears in bottom right of screen
  - *Name*: Delays
  - *Type*: Tally
  - *Members*: Click on "0 rows" button
  - In new window that appears, double-click to add members:
    - Prem Delay
    - Std Delay
  - Note that we entered
    - Prem Delay first, and CallType=1 for Premium
    - Std Delay second, and CallType=2 for Standard

Excessive Delay and IPU Rates

In ARENA, do following to estimate ED and IPU Rates

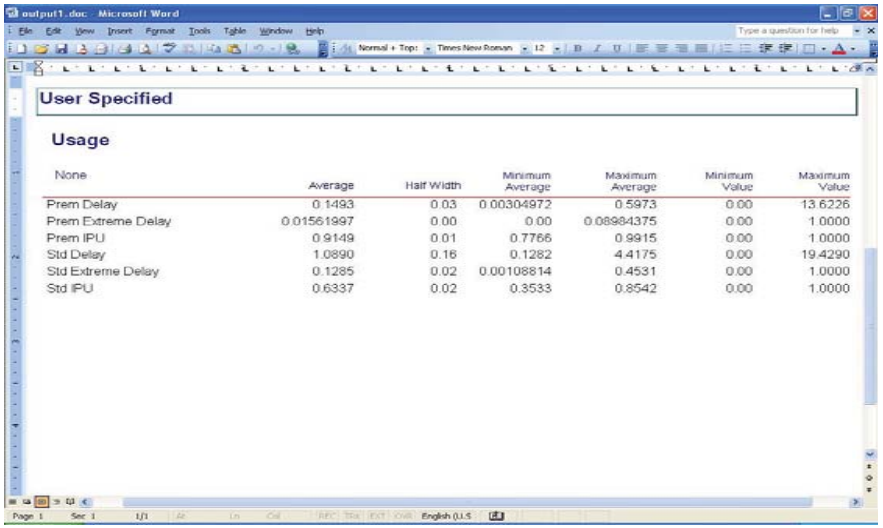
- In Record module,
  - *Name*: Record Extreme Delays
  - *Type*: Expression
  - *Value*: (ENTITY.WAITTIME > 3)
- In another Record module,
  - *Name*: Record Immediate Pick Up
  - *Type*: Expression
  - *Value*: (ENTITY.WAITTIME < 0.001)

Terminating or Steady State ?

Terminating simulation more appropriate

- Run/Run Setup menu, choose Replication Parameters
  - *Number of Replications*: 100
  - *Replication Length*: 540
  - *Time Units*: Minutes
  - *Base Time Units*: Minutes

Arena Output (5 prem operators, 11 std)



User Specified

Usage

None	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Prem Delay	0.1493	0.03	0.00304972	0.5973	0.00	13.6226
Prem Extreme Delay	0.01561997	0.00	0.00	0.08984375	0.00	1.0000
Prem IPU	0.9149	0.01	0.7766	0.9915	0.00	1.0000
Std Delay	1.0890	0.16	0.1282	4.4175	0.00	19.4290
Std Extreme Delay	0.1285	0.02	0.00108814	0.4531	0.00	1.0000
Std IPU	0.6337	0.02	0.3533	0.8542	0.00	1.0000

Proposed Design

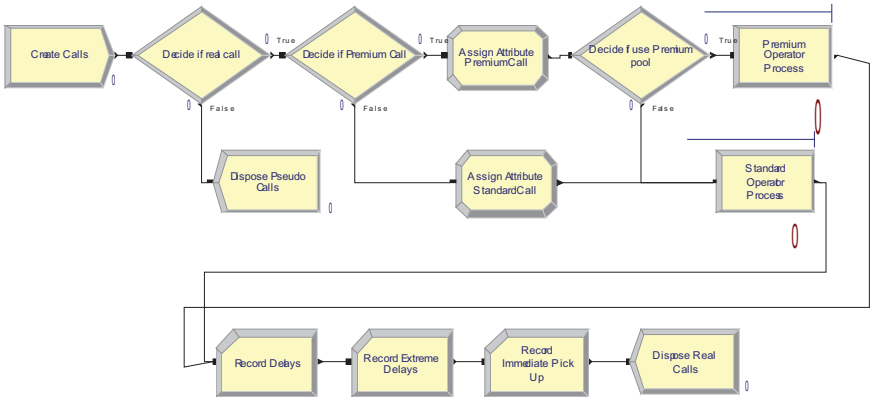
Two Ingredients:

- Two operator pools (standard and premium)
- Dynamic routing based on index:

$$\frac{\text{number of calls in process or on hold in pool}}{\text{number of operators in pool}}$$

  - *What is the interpretation of this index?*
  - *What is the rationale behind this index?*

Arena Model – Dynamic Routing



Dynamic Routing in Arena

- First define some Expressions by clicking on Expression module from Advanced Process template panel (attach by File/Template Panel/Attach…).
- In spreadsheet, enter
  - *Name*: Prem Index
  - Click on Expression Values "0 rows" button
  - *Expression Value*: (NR(Premium Operators) + NQ(Premium Operator Process.Queue)) / MR(Premium Operators)
    - NR( ) = number of busy units of the resource
    - NQ( ) = number of entities in the queue
    - MR( ) = resource capacity available

Dynamic Routing in Arena

- Similarly, define Expression for Standard Operators
  - *Name*: Std Index
  - *Expression Value*: (NR(Standard Operators) + NQ(Standard Operator Process.Queue)) / MR(Standard Operators)
- Decide module
  - *Type*: 2-way by Condition
  - *If*: Expression
  - *Value*: Prem Index < Std Index
- Give priority to premium customers
  - Click on Queue module
  - *Type:* Lowest Attribute Value
  - *Attribute Name:* CallType

Preliminary Analysis

Peak-Hour Analysis:

- *Two types of customers*
  - Standard 80%
  - Premium 20%
- *Average arrival rate* = 2.63 calls per minute.
- *Service time*
  - Same distribution for both classes
    - Mean of 1.5 min with probability 0.10
    - Mean of 6 min with probability 0.50
    - Mean of 4 min with probability 0.40

Initial Sizing of Operators Pools

- ***Erlang's Loss Formula:***
  - Unlimited queueing capacity.
  - Server capacity = m
  - Exponential interarrival times with mean $1/\lambda$ (rate λ)
  - Exponential service times with mean $1/\mu$ (rate μ)
  - $\rho = \lambda/(m\,\mu)$

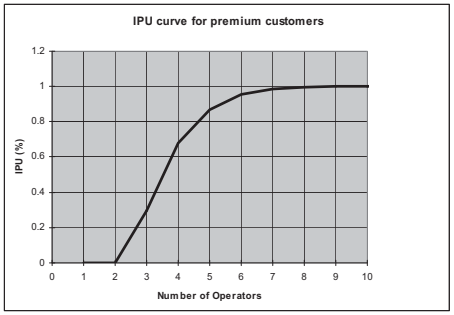
$$IPU = \frac{\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!}}{\left(\frac{(m\rho)^m}{m!}\right)\left(\frac{1}{1-\rho}\right) + \sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!}}$$

Can we apply this formula to the ANICARD problem?

Premium Pool

Basic Inputs:

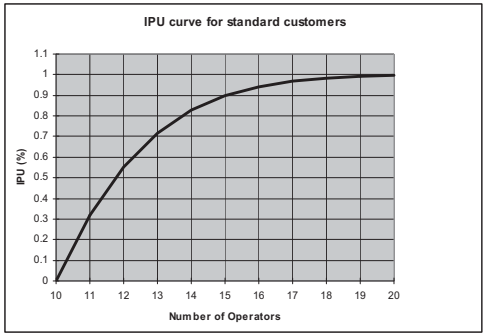
- Arrival Rate λ = 0.2 (2.63) = 0.526 calls/minute
- Service Rate μ = 1 / (Mean service time)
  - Mean Service Time = 0.10 (1.5) + 0.5 (6.0) + 0.4 (4.0) = 4.75 min
- Service Rate μ = 1/4.75 = 0.2105 per min



Standard Pool

Basic Inputs:

- Arrival Rate λ = 0.8 (2.63) = 2.104 calls/minute
- Service Rate μ = 1/ Mean service time
  - Mean Service Time = 0.10 (1.5) + 0.5 (6.0) + 0.4 (4.0) = 4.75 min
- Service Rate μ = 1/4.75 = 0.2105 per min



Simulation Results, 100 reps (95% C.I.)

	w/o Dynamic Routing		With Dynamic Routing				
(Std, Prem)	11, 5	12, 6	11, 5	12, 6	11, 6	12, 5	
Outcomes	Standard						Targets
Avg Delay	1.09 (0.16)	0.44 (0.07)	1.15 (0.2)	0.44 (0.05)	1.2 (0.18)	0.45 (0.06)	1.5 min
IPU (%)	63 (2)	76 (2)	60 (2)	75 (1)	60 (2)	75 (1)	40%
ED (%)	13 (3)	4.9 (1)	13 (2)	4.7 (1)	14 (2)	4.8 (1)	15%
	Premium						
Avg Delay	0.15 (0.03)	0.03 (0.01)	0.05 (0.01)	0.01 (0.00)	0.01 (0.00)	0.03 (0.01)	0.25 min
IPU (%)	91 (1)	97 (0)	97 (1)	99 (0)	99 (0)	98 (0)	90%
ED (%)	1.6 (0)	0.3 (0)	0.5 (0)	0.0 (0)	0.0 (0)	0.3 (0)	1%

Conclusions

- Dynamic routing allows effective sharing of resources between standard and premium operators.
  - Achieve very stringent criteria
  - Use a smaller number of operators than without dynamic routing.
- Dynamic routing is now used in
  - Call Centers
  - Communication and Computer Networks
  - Intelligent Vehicle Guidance Systems
- Simulation modeling provides insights about system performance and helps identify how to improve system designs.