# Class 4 – Operations Management – 2017-10-16

## Chapter 16 – Service Systems with Patient Customers

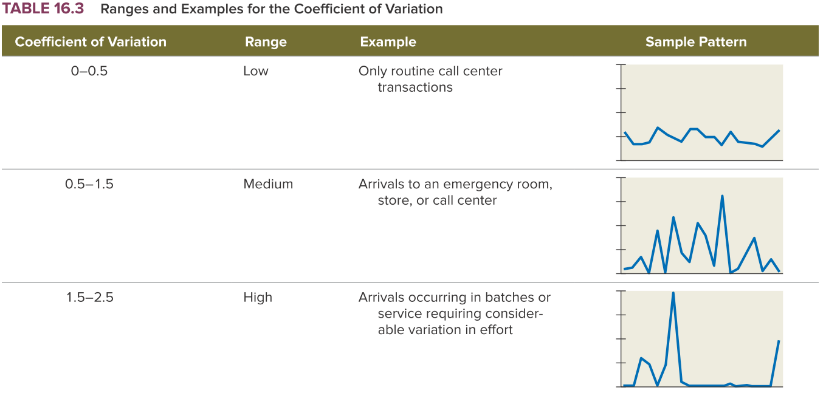
|  |
| --- |
| * **Inhibitors: Variability, Inflexibility** * LO1 Evaluate performance measures related to **length of the queue**, **average wait**, and **time to serve** for queue with **constant demand rate > service** **rate** * LO2 Evaluate performance measures influencing factors for queue with **variable interarrival and processing times on one server** * LO3 Evaluate performance measures influencing factors for queue with **variable interarrival and processing time and multiple servers** * LO4 Understand **Economies of Scale** n queuing systems, and understand pros and cons of **pooling** |

Introduction: Queues are the most visible forms of supply-demand mismatches. Manager needs to understand why queues form and how to better manage to improve service or lower cost, or both.

### 16.1 Queues when Demand Exceeds Supply (DR > CAP)

* Any interval of time where DR > Capacity, queue will grow at **Queue Growth Rate**
* After time T, the **Length of Queue** is
* Longest wait will be for the last unit arriving when DR>CAP; **Time to Serve Qth Person**
* **Time to Serve Person Arriving at Time T**
* **Average Wait Time to Serve a Unit**
* **Drivers of Average Wait Times**
  1. **Demand and Capacity (ie Utilization):** Therefore; higher utilization, longer customers wait
  2. **Duration of Busy Period, T**: Longer time T, longer queue & avg wait time will grow
  + ***Note Queues Do Not Grow Forever:*** *Demand will calm; limit to acceptable wait time*
* **Managing Peak Demand:** To improve situation (1) Reduce Demand or (2) Increase Capacity
  1. **Peak-Load Pricing:** Reducing demand using congestion pricing; higher prices offset loss in revenue by those who will not pay the premium
  2. **Off-Peak Discount:** Increase demand of during non-busy period; retain customers & spread out arrival time (better perceived as discount vs penalty)
  3. **Line Balancing:** Increase capacity by reallocating resources
  4. **Pre-Processing Strategy:** Increase capacity by reducing processing time during peak time by moving work to off-peak time

### 16.2 Queues When Demand and Service Rates are Variable – One Server

* When there is **Variability** in either arrival rate or processing time (or both) queues can form even if CAP > DR. We must understand arrival and service process.
* **Arrival Process**: Flow of customers arriving in the system
* **Interarrival Time**: time between customer arrivals to the system; **Average Interarrival Time, *a***
* **Service Process**: Flow of customers when they are being served
* **Processing Time**: Time which a customer spends with a server; **Average Processing Time, *p***
* **Coefficient of Variation:** Ratio of standard deviation to the average; to determine variability relative to average; common to observe values close to 1. CV < 1 ~ Consistency **vs** CV > 1 ~ Uncertainty

|  |
| --- |
| **Queuing Model with Single Server**   * Based on arrival and service process; how to estimate average wait time? * **Queuing Model:** Abstract representation of queue; allows for prediction of wait time and other performance measures * **Inputs:** 1 Server; 1 Queue; Customer arrives & waits until service complete; a, p , CVa, CVp * **Assumption: process time p < arrival time a**; on average capacity > demand, otherwise queue grows indefinitely * **Capacity** = 1/p; **Flow Rate** = 1/a * (ensure < 1)   + **Variability** does not impact utilization; only wait time * **Performance Measures**   + ***Tq = Average Time in Queue***   + ***Iq = Average # Customers in Queue*** *(Little’s Law)*   + ***Ip = Average # Customers in Service*** *(Little’s Law; Utilization)*   + ***T = Flow Time, Average Time in System*** *(Note: Time-in-queue is most frustrating to customers)* |

* **Key Drivers of Waiting Time**
  1. **Capacity Factor, p:** Capacity (1/p) Increases 🡪 Wait Decreases
  2. **Utilization Factor, Util/(1-Util):** Because p < a, factor is never negative. Utilization Increases 🡪 Wait Increases Exponentially. Trade off: Agents are idle with low wait vs busy with high wait.
  3. **Variability Factor, (CV2+CV2)/2**: Variability Increases 🡪 Wait Increases
* **Managing Wait Time:** (1) Increase Capacity; (2) Decrease Utilization; (3) Decrease Variability

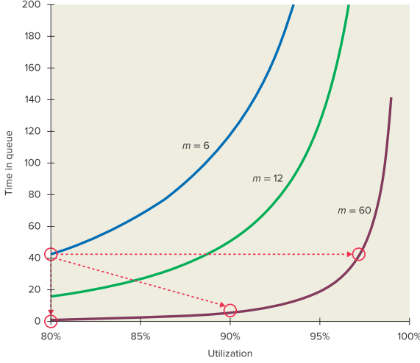
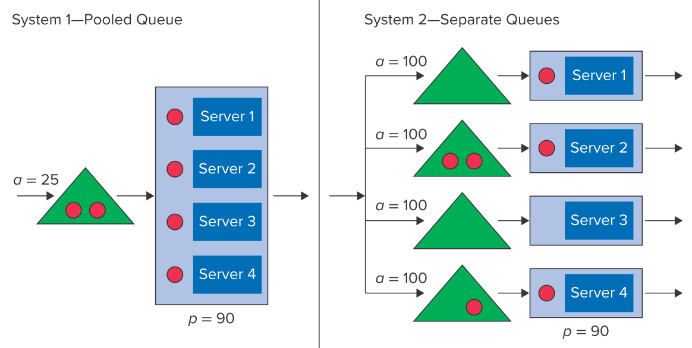
### 16.3 Queues When Demand and Service Rates are Variable – Multiple Servers

|  |
| --- |
| **Queuing Model with Multiple Servers**   * **Inputs:** *m* Servers; 1 Queue; Customer arrives & waits until service complete; 1 server/customer; a, p , CVa, CVp * **Assumption: system capacity m/p > flow rate 1/a or p/m < a**; on average capacity > demand, otherwise queue grows indefinitely * **Capacity** = m/p; **Flow Rate** = 1/a * (ensure < 1)   + **Variability** does not impact utilization; only wait time   + **Unstable Queue:** DR > CAP; # Servers < p/a   + **Stable Queue**: DR < CAP; # Servers > p/a * **Performance Measures**   + ***Tq = Average Time in Queue***   + ***Iq = Average # Customers in Queue*** *(Little’s Law)*   + ***Ip = Average # Customers in Service*** *(Little’s Law; Utilization)*   + ***T = Flow Time, Average Time in System*** *(Note: Time-in-queue is most frustrating to customers)* |

* **Managing Time in Queue**
  + Increase servers, *m* 🡪 *Utilization* falls 🡪 *Capacity* increases 🡪 Wait time, Tq, decreases 🡪 # customers in queue, Iq decreases
  + Note: Number of servers, *m*, has no influence on number of people in service Ip, because Little’s Law tells us # customers is dependent on Flow Rate and Process Time
* **Key Drivers of Time-In-Queue for Multi Serve Queue**

1. **Capacity Factor, p/m:** Process Time, p, Decreases or # Servers, m, Increases 🡪 Wait Decreases
2. **Utilization Factor, Util√/(1-Util):** Utilization Increases 🡪 Wait Increases Exponentially. Trade off: Agents are idle with low wait vs busy with high wait.
3. **Variability Factor, (CV2+CV2)/2**: Variability in either process Increases 🡪 Wait Increases

### 16.4 Queuing System Design – Economies of Scale and Pooling

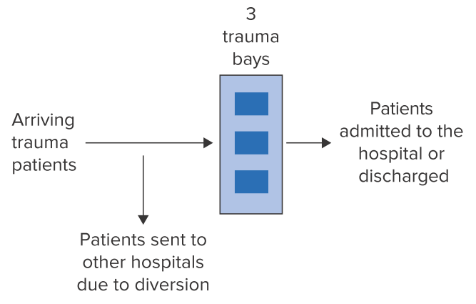
*  **Economies of Scale:** As service centres get bigger(*m* increases); become less sensitive to Utilization 🡪 Bigger queuing systems work better but need the demand!
  + **Three Options to Manage Increase in m:**
    - Keep Utilization constant but lower Tq
    - Keep Tq constant but increase Utilization
    - Increase Utilization and Reduce Tq
* **Power of Pooling:** EoS tells us it is better to operate a system with more demand; however, this may not be possible. When demand is fixed, what can we do to get best operation performance?
  + **Pooled Queue:** All demand is shared across server; customer is helped by first available server
  + **Separate Queue:** Demand is initially divided among servers and served by designated agent
  + **Pooled Queue** is much more effective than Separate Queue because: (1) **economies of scale** and (2) in pooled system workers are **busy at the same** time, in particular when there are many customers in the system.
  + ***Note:*** *Variability of arrivals, utilization, and inventory in process (Ip) do not change by pooling queues*
  + **Why not always Pools?**
    - (1) on going relationship between servers could lower processing time – *doctors office*
    - (2) more perceived value to customers – *priority check-in lines; is it costlier to have priority customers wait vs regular customers? If yes, separate queue*

|  |
| --- |
| **Chapter 16 Summary**   * **LO16-2 Capacity = 1/a; Flow Rate = 1/p** * **LO16-1 🡪 Unstable Queue** * **LO16-2,3 🡪 Stable Queue** * *Queues form when customers are patient enough to wait for service and DR > CAP* * *Queuing model allows managers to predict wait times & number of customers in the system based on (1) capacity (2) utilization, and (3) variation* * *Trade Off: High Utilization → High Wait Times* * *Performance Improves: (1) Economies of Scale and (2) Pooling* * *Avg Time in Queue, Tq; Avg # Customers in Queue, Iq; Avg # Customers in Service Ip; Flow Time/Time in System, T=Tq+p* * *Arrival Rate, λ = 1/Interarrival Time,a Capacity of Each Server, μ = 1/Processing Time,p*   **Questions: 1-4, 6, 9-12, 16, 19, 21** |

## Chapter 17 – Service Systems with Impatient Customers

|  |
| --- |
| * **System Inhibitors: Inflexibility and Variability 🡪 Loss of Demand and Capacity** * LO1 Use **Erlang Loss Model** to evaluate performance measures with **impatient customers** * LO2 Use **Erlang Loss Model** to understand value of **Economies of Scale, Pooling**, and **Buffers** with impatient customers * LO3 How can **Variability reduce Capacity** & how to **Restore Capacity** via **Standardized Work**, **Buffers**, or **Elimination of Sequential Work** |

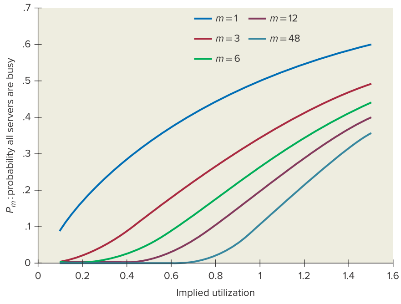
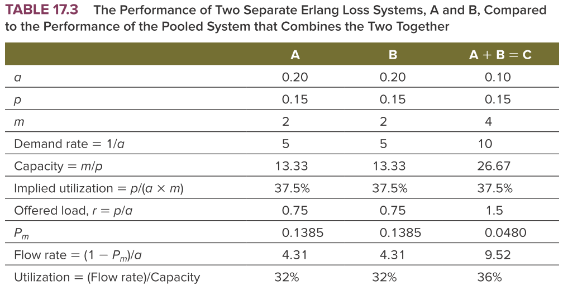
### 17.1 Lost Demand in Queues with No Buffers

* *Examine situations where customers do not wait (ie Emergency Room)*
* **Diversion**: Practice of sending demand away from a system when it cannot be served by the system in a sufficiently timely manner (*Example: Hospital diverting patients to another hospital) 🡪* Indication of inflexibility
* **Lost Demand**: Potential demand a system cannot serve; does not count towards process’ flow rate
* 2 Critical Parameters to understand Performance
  + **Average Interarrival Time, *a* & *CVa =1***
  + **Average Processing Time, *p***

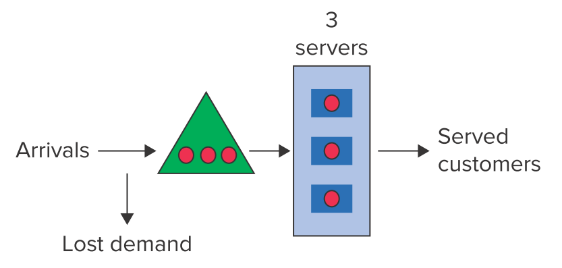
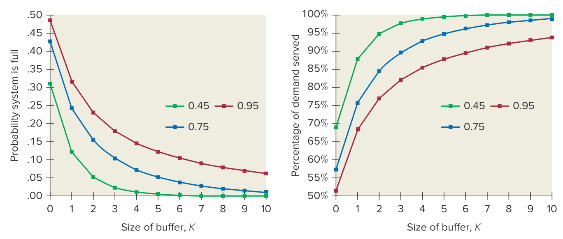
|  |
| --- |
| **Erlang Loss Model**   * **Erlang Loss Model:** Queuing model in which total number of flow units cannot exceed # of servers, m. If flow unit arrives and system is full, the flow unit is not served and becomes lost demand. *(Created by Agner Erlang when trying to determine how many trunk lines to build between cities for Cophenhagen Telephone Company)* * **Assumptions:** *m* servers, *a, p, CVa=1* (given value, not guaranteed but a fair assumption) and *CVp* not a factor*;* customers processed by single server, customers do not wait (Max Inv = m); * **Capacity and Utilization**   + *Note: When IU > 1 🡪 Lost Demand* * **Performance Measures**  1. **Probability of Diversion** 🡪 Service    * **Denial of Service Probability:** Probability that a flow unit in the ELM is not served and becomes part of the Lost Demand    * **Pm:** Probability all servers are occupied in ELM 🡪 Use Tables 17.1 based on *r* and *m (& P604-606)*    * **Offered Load:** ratio of demand to capacity of 1 server 🡪 the system only “accepts” a portion of the demand 2. **Flow Units of Demand Lost** 🡪 Flow Rate 3. **Flow Rate though System** 🡪 Revenue 4. **Average Utilization of Resources in the System** 🡪 Costs    * ***Utilization*** *(ratio of FR served/capacity)* ***≠ Implied Utilization*** *(ratio of potential demand/capacity) because there is demand loss even if IU<100%* 5. **On Average, How Many Resources are Busy** 🡪 Costs |
| * **Optimize Staffing vs Target Lost Demand**   + *How much capacity should we have to achieve to provide a desired level of service to customers? Ie: provide required level of service via a threshold level of diversion*     1. Identify Offered Load: r=p/a     2. Identify row in Table 17.1     3. Read across to find value < Target Pm     4. Identify # of servers required, m |

### 17.2 Managing a Queue with Impatient Customers: Economies of Scale, Pooling, and Buffers

*We see that to achieve a required level of service, utilization may be low. How can we improve this?*

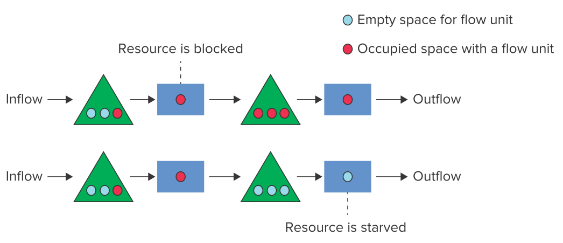
* **Economies of Scale**
  + Like waiting time problems, there is significant EoS in the Erlang Loss System 🡪 systems with more servers and proportionally the same amount of demand perform better than systems with few servers.
* **Pooling**
  + Create larger systems (EoS) by pooling demand and capacity of smaller systems.
  + **Costs to Pooling:** Trade off with capacity in fewer locations
  + **Declining Marginal Returns:** Incremental benefit of pooling decreases as the activity increases

|  |
| --- |
| **Combining Interarrival Times**   1. Add Demand together (1/a+1/a ….) to get DemandPool 2. Convert Pooled Demand to Interarrival Time (1/DemandPool) |

* **Buffers**
  + ELM can be improved by adding some limited waiting capacity
  + **Buffer, K**: Inventory within a process that helps to mitigate consequences of variability 🡪 Demand is only lost if m+k is full
    - *Buffer can provide a cheap, cost effective way to increase Demand Served vs increasing capacity*
    - *Note if K becomes very large 🡪 Ch16 Patient Customer Model*
  + Buffer is also subject to **Decreasing Marginal Returns**
  + **Limitations of Buffers** (*more applicable to large buffers*)
    1. Customers may not be able to wait
    2. **Balking:** customers may not choose to wait when they see a queue (ie no prevention of demand lost)
    3. **Reneging/Abandoning:** customers leaving queue

### 17.3 Lost Capacity due to Variability

Variability in demand/service can create queues of waiting customers, lose demand; and lose capacity. Due to randomness of processing time, synchronization between resources may not be possible.

* **Tandem** **Queue**: The output of one queue becomes input of another queue; due to variability problems arise
  + **Issue 1 – Blocking**: Resource completes a flow unit but prevent from working on a new flow unit because there is no room to store completed unit (output inventory full)
  + **Issue 2 – Starving**: Resource capable of working on a flow unit but there is no unit available to work on (input inventory empty)
  + *Blocking and starving destroy capacity 🡪 Flow rate is reduced*
* **Analyzing Tandem Queue Systems:** Complete systems that require **discrete event simulation** to track flow units through a process and random time flow *(ie Blood Bank 609)*
* **Improving Tandem Queues**
  + 1. **Reduce Variability 🡪 Increase Capacity:** Standard work procedures (ie accelerate learning curve)
    2. **Increase in Buffer Size 🡪 Increase in Capacity:** System’s capacity decreases without large buffer (“Buffer or Suffer”); downside is that it can hide problems (ie Toyota approach)
    3. **Avoid Sequential Process**: must outweigh the longer processing times due to less specialization

|  |
| --- |
| **Chapter 17 Summary**   * Variability & Inflexibility lead to Loss in Demand & Capacity * Capability to serve depends on Offered Load, r, and Number of Servers, m * *Question Type: Solving for m based on target Pm* * Improving Service   + Economies of Scale: Larger system 🡪 Better perf   + Pooling: Pool to create EoS     - *apool = 1 / Demandpool*   + Buffers: Add buffer to deal with variability * Improving Tandem Queue: Requires simulation analysis   + Increase in Variability 🡪 Reduction in Capacity   + Increase in Buffer Size 🡪 Increase in Capacity   + Avoid Sequential Process * **Patient Customers Queues** are evaluated based on **waiting** **time** and **Impatient Customers** are evaluated on **demand lost** * m > p/a to ensure a **stable system**   **Questions: 3-6** |