Discrete Event Systems

Simulation and Modeling (CSCI 3010U)

Faisal Z. Qureshi

http://vclab.science.ontariotechu.ca

Discrete Event Systems: Simulating a computer network

- Network traffic: number and types of packets
- Switch capacity
- Processing at each node (switch)
- Network performance: throughput and latency

Things to consider

- Packet generation
- Time it takes for a packet to travel between two nodes
- Probability of lost packets

Measuring performance

Throughput

- The number of packets (or bytes) that move through the network in a fixed period of time (usually second)

Latency

- The time for a packet to get from its source to its destination

Discrete Event Systems: Web services

- How many computers and their capacity
- Number and types of web requests
- Service times

Discrete Event Systems: Operation Research

- Discrete event simulations are often used in business and operations research
  - Banks, e.g., number of tellers and ATMs required at a bank branch
  - Manufacturing, e.g., the best configuration for a factory floor, the optimal mix of machines, etc.
  - Customer service, e.g., call centers, number of operators, etc.
  - Transit systems, e.g., bus routes, number of buses, length of routes, etc.

Discrete Event Simulations

Communicating ideas and presenting results to assist decision making

Unlike the simulations that we have seen so far (in this course):

- Questions tend to be more complex
- Emphasis on communicating results
- Explore alternate models

Example: Key Messages

- Bank customers will be unhappy
- This factory configuration will increase our profit margin
- Transit wait times are too long
- This process will result in an unreliable product
- We need this many more computers to service our South American market

Anatomy of a Discrete Event Simulations

- Entities
- Variables
- Resources
- Queues
- Statistics
- Events

Entities

- Entities: these flow through the simulation
  - Network packets, buses, bank customers, etc.
- Entities have *types* and *attributes*
  - Network packet, for example, is an entity type
  - An attribute might be the time a particular network packet has stayed in the simulation. Attributes are not shared between different instances of an entity.
- Entity creation and destruction
- Processing entities

Variables

- Current clocktime
- The average time required to process a customer request

Resources

- Resources are something that is required to process an entity
  - In the bank simulation, bank tellers are a resource
  - In the network simulation, the switches are resources
- When an entity is being processed it seizes one or more resources and doesn't release them until it is finished
- There may be more than one unit of a given resource, so several entities can be processed in parallel
  - There may be more than one unit of a given resource, so several entities can be processed in parallel

Queues

- Each resource, or group of similar resources will have a queue, this is where entities wait when there isn't a resource to process them
- Queues are typically one of the main things we are interested in, we want to know how long an entity has to wait for service
  - In the bank example, if the queue time is too long the customers will not be happy, so we need to accumulate the average queue length and maximum time in queue

Statistics

- A discrete event simulation will accumulate a large number of statistics
- For entities we are interested in
  - The time they spend in the system (average and maximum)
  - The time that is spent processing
  - Wait times
- For resources we are interested in
  - Utilization
  - The number of entities that they process

Events

- Events drive the simulation
  - Move the entities forward through the system
- Events are typically random numbers and they tend to be generated at random times
- For example an event can be used to generate a new entity, or the time the entity needs to wait to be serviced
- Events are also used to move the simulation clock forward
- The simulation maintains a list of events, in time order
- It picks the first event from the list, moves the clock forward to the event's time, and then processes the event
- Processing the event will typically generate more events that will occur in the future

Overall Structure

1. Entity Generation
   - A random process could be used to generate entities
   - This process generates a random time for the next entity to be created, and this event is placed on the event list
2. Entity processing
   - When the entity is generated it is moved to the first process in the simulation,
   - If the resource is busy it enters the resources queue and waits for it to be idle
   - Once the entity seizes the resource, another event is generated, the time when the entity is finished with the resource, this event is then placed on the event list

Overall Structure

3. Entity Destruction
   - This process continues while the entities move through the system and eventually reach the final step where they are destroyed
4. Statistics
   - At this point all of the summary statistics for the entity are accumulated

Python `simpy`

- A process-based discrete-event simulation framework based on standard Python.
- https://simpy.readthedocs.io/en/latest/

Example: Carwash Simulation

```
class Carwash(object):

    def __init__(self, env, num_wash_stations, washtimes):
        self.env = env
        self.wash_stations = simpy.Resource(env, num_wash_stations)
        self.washtimes = washtimes

    def wash(self, car):
        a, b = washtimes
        washtime =  random.randint(a, b)
        yield self.env.timeout(washtime)
```

Example: Carwash Simulation

```
def car(env, name, cw):
    print(f'{name} arrives at the carwash shop at {env.now}')

    with cw.wash_stations.request() as request:
        yield request
        print(f'{name} enters the carwash at {env.now}')
        yield env.process(cw.wash(name))
        print(f'{name} leaves the carwash at {env.now}')
```

Example: Carwash Simulation

```
def setup(env,
          num_wash_stations,
          wash_times,
          time_between_arrivals):
    carwash_shop = Carwash(env, num_wash_stations, time_between_arrivals)
    i = 0
    while True:
        a, b = time_between_arrivals
        arrival_time = random.randint(a, b)
        yield env.timeout( arrival_time )
        i += 1
        env.process( car(env, 'car %d' % i, carwash_shop) )
```

Example: Carwash Simulation

```
print('Carwash Simulation')

seed = 42
random.seed(seed)

num_wash_stations = 2
washtimes = (3,7)
time_between_arrivals = (1,2)
simulation_duration = 40

env = simpy.Environment()
env.process( setup(env, num_wash_stations, washtimes, time_between_arrivals) )
env.run(until=simulation_duration)
```

Data for DES

- A DES needs a lot of input data to function.
- This input data is in the form of random numbers drawn from a probability distribution

Questions

- Where does this data come from?
- How do we know that it is right?
- What impact does inaccurate data have on our simulation?

Data Gathering for DES

- Observe an existing system
- Collect the right data
- Ensure that data is not all *mixed up*

Example

- Mortgage application processing times at bank XYZ

Data Gathering for DES:

- Three scenarios:
  - We can collect data ourselves
  - We can observe the collection of data
  - We have no control over how data was collected
- We may get a sequence of observations x_i
  - Ideally the observer collected each observation individually with as much accuracy as possible
- Sometimes we get *pooled* data
  - A histogram of the observations and not the individual observations themselves
  - More difficult to work with

Why Pooled Data?

- Not enough accuracy was used in data collection
  - We still got individual observations but the rounding makes these the same as a histogram
- Privacy where individual observations might identify the people or company involved

We may have no choice but to use pooled data

Pooled Data for DES

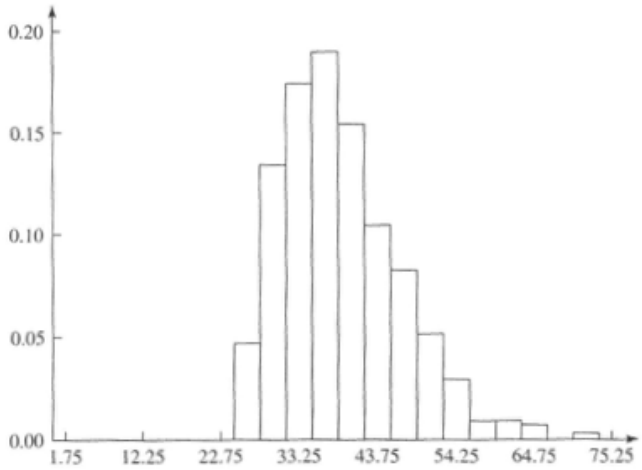


FIGURE 6.1
Histogram of 890 machine processing times for an automotive manufacturer.

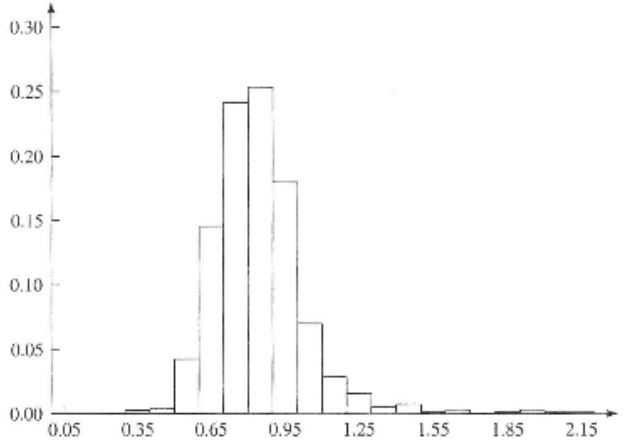


FIGURE 6.3
Histogram of 856 ship-loading times.

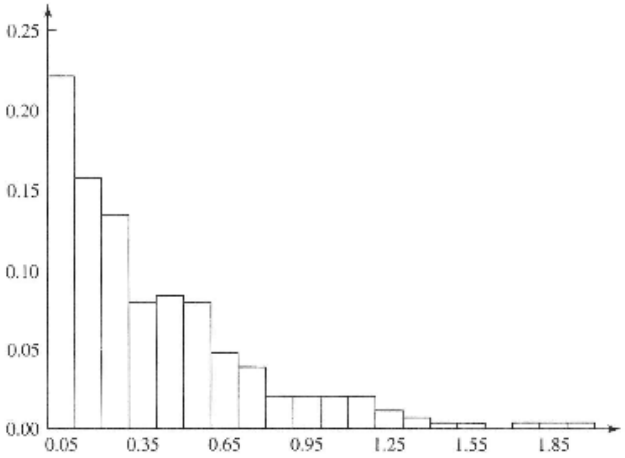


FIGURE 6.2
Histogram of 219 interarrival times to a drive-up bank.

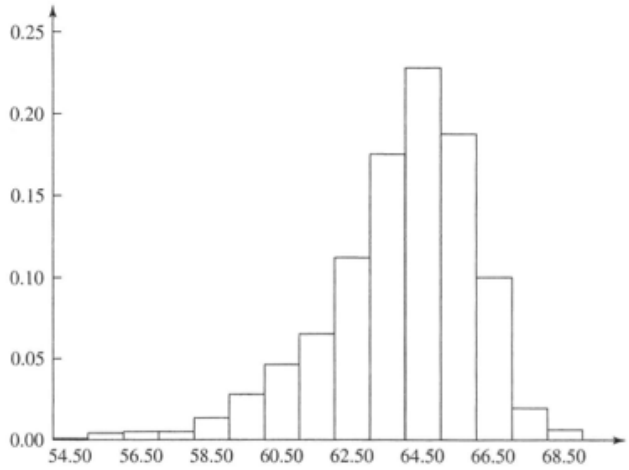


FIGURE 6.4
Histogram of the yardages for 1000 large rolls of paper for a household-products manufacturer.

Courtesy: Taken from Prof. M. Green Notes

Using Gathered Data for DES

- **Empirical data:** randomly choose from data
- **Empirical distribution:** construct a distribution from data and use it to generate random numbers
- **Theoretical distribution:** fit a theoretical distribution to the data and generate random numbers from that distribution

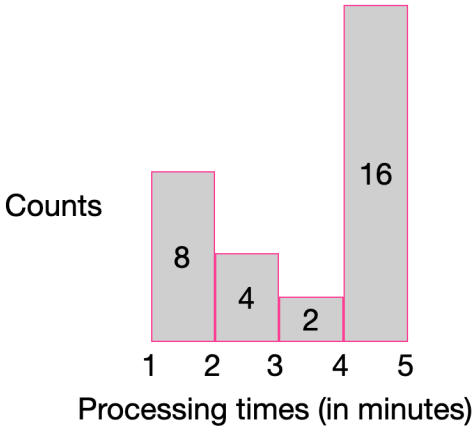
Empirical Data

- Each item in our data set is an observation of the real process, we can just use this data as it is as the random input to our system
- We can either play the data back in the order it was collected or randomly selected samples from the data
- Both of these techniques are very easy to implement

Pooled Data

- We could randomly *select one of the bins in the histogram* based on the number of items in the bin
- Then randomly generate a value from the bin's range

Example



Pros and Cons of Using Empirical Data

Pros

- One of the advantage of empirical data is that its real and we can use it to compare the results of our simulation to the real system
- We don't make any assumptions about the distribution or how the observations were generated

Cons

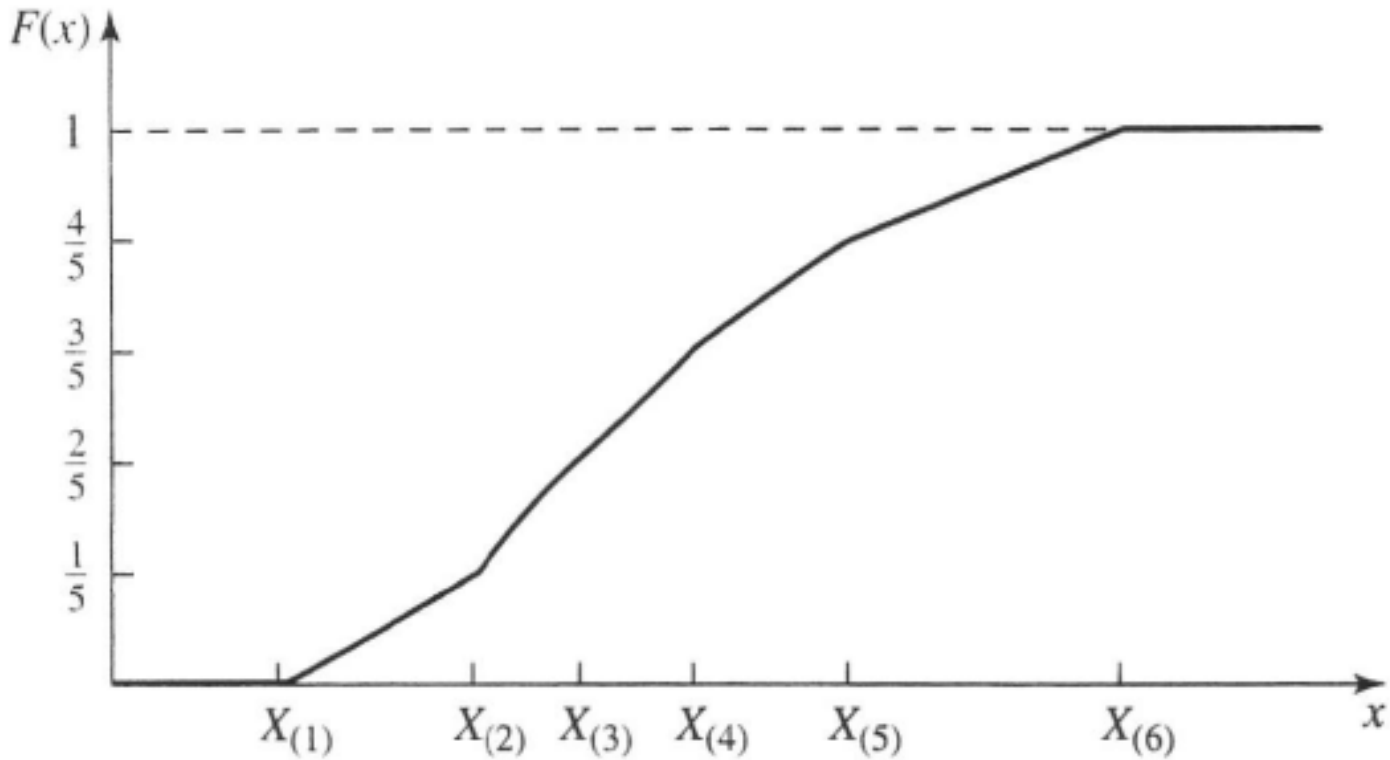
- We have a limited sample size
  - This can have an impact on the validity of our statistics
  - It severely limits the lengths of our simulations, a small integer multiple of the collection time
- We can't generate a value outside of the range of collected values, so extreme values won't occur

Empirical Distribution

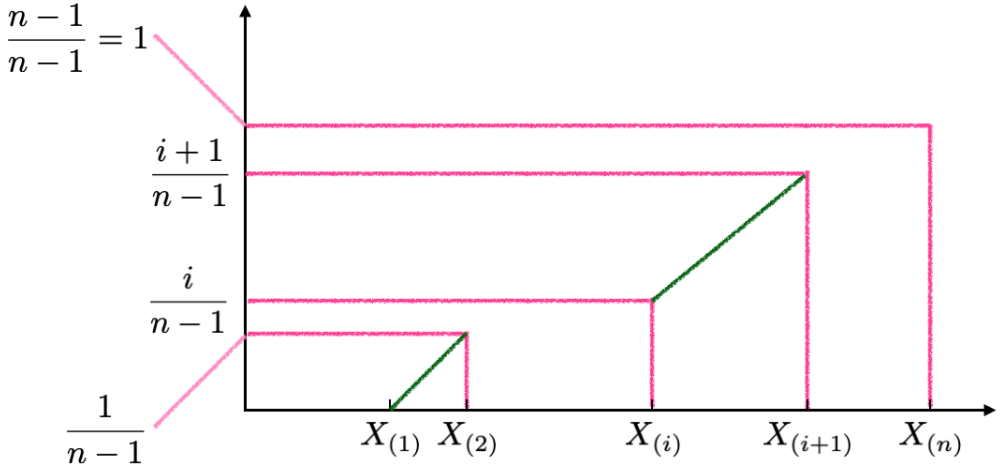
- Need a distribution to generate samples from
- Construct a distribution from the observations themselves
  - The resulting distribution is referred to as the empirical distribution
- Generate samples from this empirical distributions
  - Construct a cumulative probability distribution (CDF), so we may sample from this distribution

Constructing a CDF from Empirical Distribution

- Say $X_{(1)}, X_{(2)}, \cdots, X_{(n)}$ are (sorted) observations. $X_{(1)}$ being the smallest and $X_{(n)}$ being the largest.
- Goal is to construct the following piece-wise linear function



CDF from Empirical Distribution



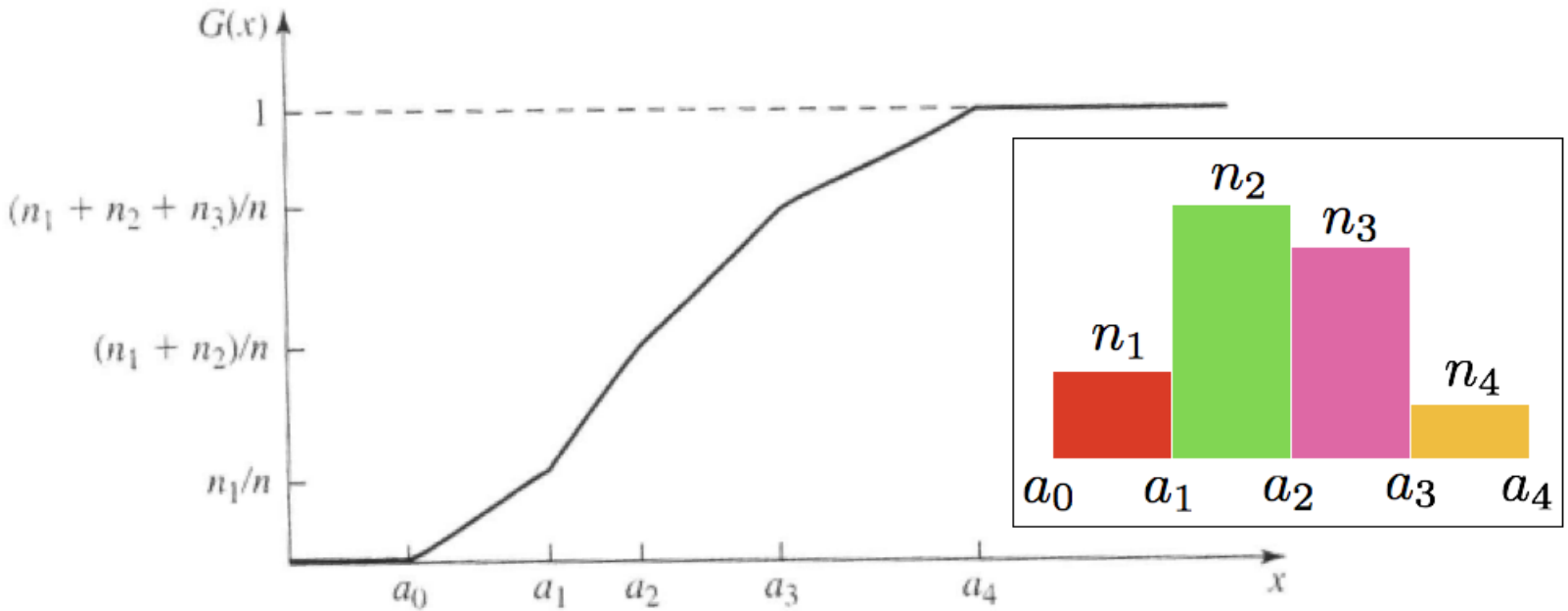
CDF from Empirical Distribution

- Given $X_{(1)}, X_{(2)}, \cdots, X_{(n)}$ (sorted) observations, the CDF is given by

$$F(x) = \begin{cases} 0 & \text{if } x < X_{(1)} \\ \frac{i-1}{n-1} + \frac{x - X_{(i)}}{(n-1)(X_{(i+1)} - X_{(i)})} & \text{if } X_{(i)} \leq x < X_{(i+1)} \\ 1 & \text{if } x > X_{(n)} \end{cases}$$

CDF from Empirical Distribution (Pooled Data)

- Given a histogram of k cells, where each cell has n_i data points (observations), where each cell covers the interval $[a_i, a_{i+1}]$.
- Define $G(x)$ at cell boundaries in the following way:
  - $G(a_0) = 0$
  - $G(a_i) = (n_1 + n_2 + \cdots + n_i)/n$, where n is total number obser
- Now construct the following piece-wise linear function



CDF from Empirical Distribution (Pooled Data)

- Given a histogram of k cells, where each cell has n_i data points (observations), where each cell covers the interval $[a_i, a_{i+1}]$.
- Define $G(x)$ at cell boundaries in the following way:
  - $G(a_0) = 0$
  - $G(a_i) = (n_1 + n_2 + \cdots + n_i)/n$, where n is total number observations
- Use the following formula to get $G(x)$

$$G(x) = \begin{cases} 0 & \text{if } x < a_0 \\ G(a_{j-1}) + \frac{x - a_{j-1}}{a_j - a_{j-1}}(G(a_j) - G(a_{j-1})) & \text{if } a_{j-1} \leq x < a_j \\ 1 & \text{if } a > a_k \end{cases}$$

Pros and Cons of Using Empirical Distributions

Pros

- We can now generate a much larger number of random numbers
- We can also generate numbers that are not in the original sample

Cons

- We cannot generate more extreme values
- All the numbers we generate are still within the range of the smallest to largest numbers in our sample
- A large number of samples requires a large amount of memory to store and longer processing time

Theoretical Distribution

- A theoretical distribution is a mathematical distribution that fits the observations
  - Compact representation
  - Ability to generate a wider range of random numbers

Types of Theoretical Distributions

- Discrete
- Continuous

Commonly Used Theoretical Distributions 1

- Uniform distributions
- Triangular distribution
- Exponential distribution
  - Inter-arrival times (of customers, parts, etc.)
  - Time-to-failure for a piece of equipment

Commonly Used Theoretical Distributions 2

- Gamma distribution
  - Time-to-complete a task (serving a customer, processing a part, etc.)
- Weibull distribution
  - Time-to-complete a task
  - Time-to-failure a part
- Normal (Gaussian) distribution
  - Error
  - Data that is a sum of a large number of random values, e.g., house prices, etc.

Commonly Used Theoretical Distributions 3

- Poisson distribution
  - Used for modelling the number events that occur in an interval of time when the events are occurring at a constant rate, e.g., nuclear decay, etc.
- Lognormal distribution
  - Similar to Gamma or Weibull distribution
- Beta distribution
  - Modeling random proportions, e.g., the number of defective parts in a shipment

Commonly Used Theoretical Distributions 4

- Bernoulli distribution
  - Simplest discrete distribution
  - Modeling coin tosses
- Binomial distribution
  - For number of successes in t independent Bernoulli trials
- Discrete uniform distribution
- Geometric distribution
  - Number of failures before the first success in a sequence of independent Bernoulli trials, e.g., the number of items inspected before finding the first defective item

Fitting a Theoretical Distribution to Data

- Estimate parameter values, e.g., mean and variance in the case of Normal (Gaussian) distribution
- Use Chi squared test to compare data sampled from the theoretical distribution to the observed data
- Many off-the-shelf packages to perform this task

Sensitivity Analysis

- Sensitivity analysis determines how the results of the simulation change in response to changes in its input
  - E.g., what is the effect of increasing the mean of the exponential distribution used for arrival times by 5%?
- Sensitivity analysis gives us a measure of the reliability of our results, do they change drastically in response to a small input change?
- Reasons for performing senstivity analysis
  - Impact of errors in input data, what is the impact of a bad guess or unreliable data?
  - Generality of the simulation, if there is a slight change in the system will the results be about the same?
  - Stability of the system/simulation, do small changes in the input have major impacts on the simulation?

Sensitivity Analysis

Changing an input by x%

- Case 1: simulation results change by less than x%
  - Input has little impact on the results
  - System has extra capacity, redesign?
  - Time spent on improving accuracy probably wasted
- Case 2: simulation results change by approximately x%
  - Need to be careful about accuracy of the data, since it has some impact on the results
- Case 3: simulation results change by much more than x%
  - This case should be investigate further
  - Need to be very careful about the accuracy of the data

Summary

- Anatomy of DES
- Data collection considerations
- Generating random numbers using the collected data
  - Raw data
  - Empirical distribution
  - Theoretical distribution
- Sensitivity analysis