# CS 6423

# Scalable Computing for Big Data Analytics

## Lecture 1: Overview

Prof. Gregory Provan
Department of Computer Science
University College Cork

# Course Objectives

**Applying Modeling Methodologies to Cloud Computing/Distributed Algorithms and Architectures**

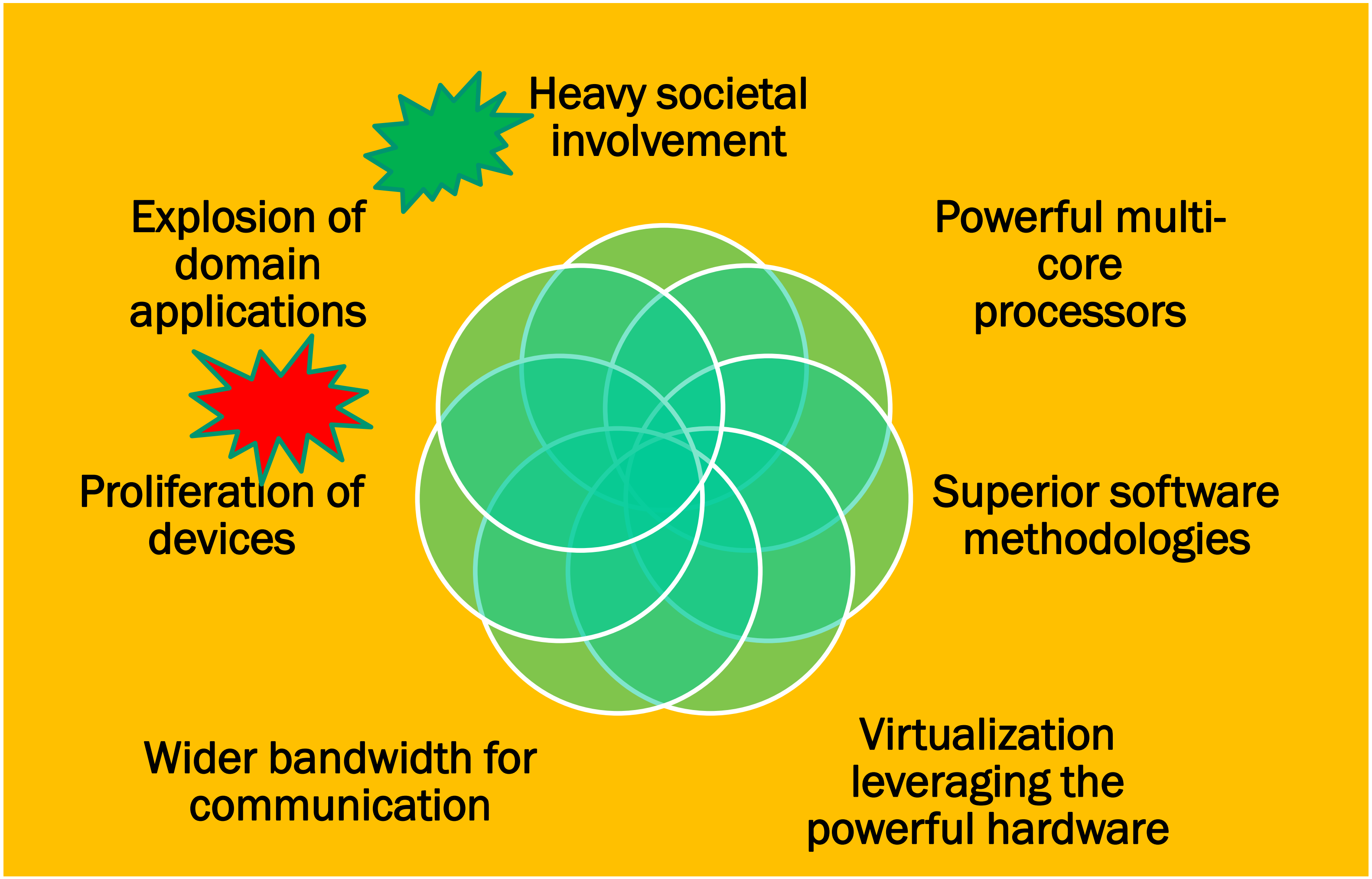
- Understand notion of real-world cloud networks
  - How to define such a system?
  - What do you want to do with such a system?
- Study algorithms used for performing inference on these systems
- Study approaches used for modeling the hardware for cloud systems

# Learning Outcomes

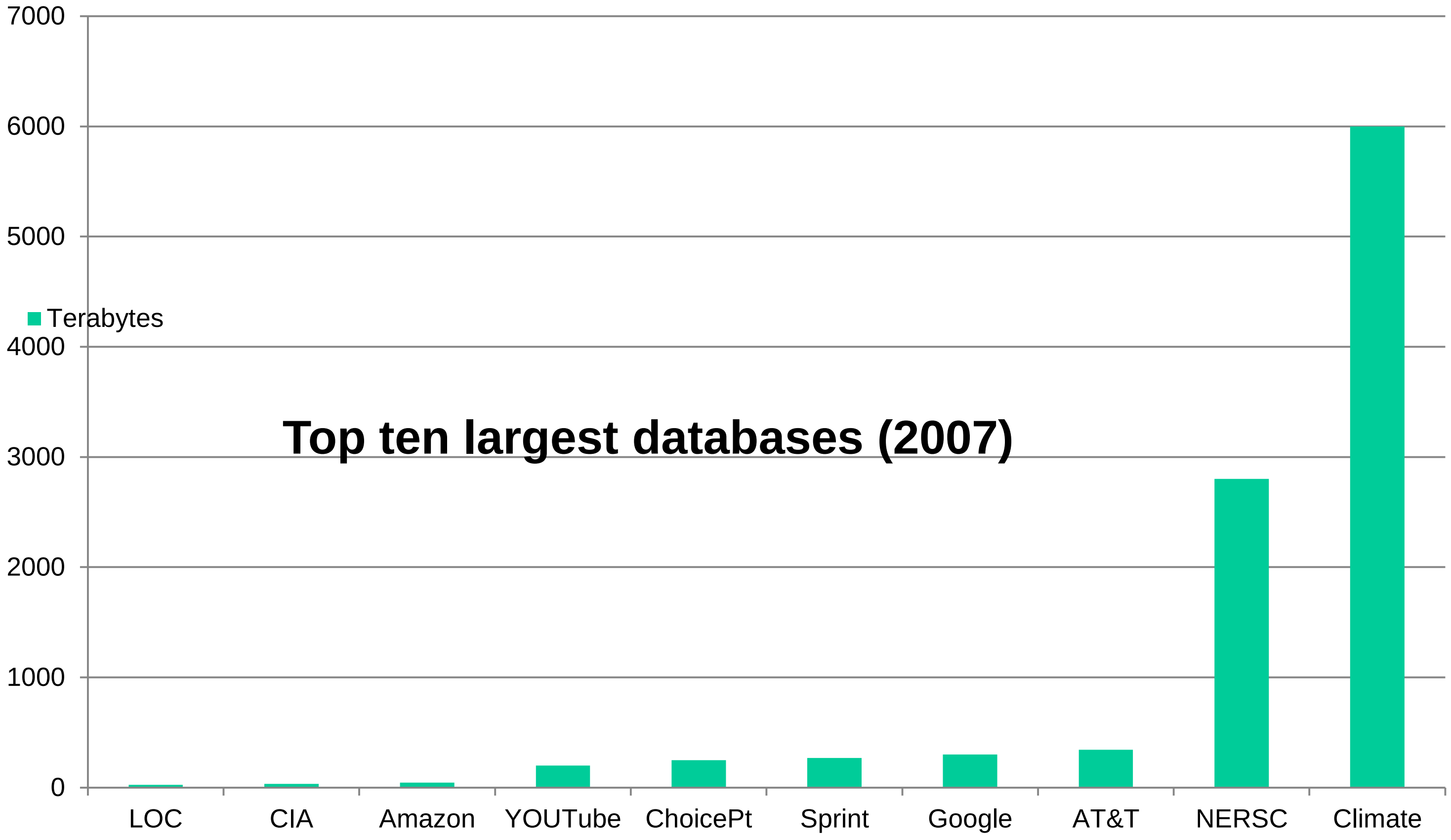
At the conclusion of this course, a participant will ...

- know the *basic concepts* underlying cloud systems modeling and performance analysis
- know how to
  - Develop mathematical models for cloud-system networks
  - Design algorithms using the MapReduce framework
  - Design network topologies to achieve particular performance objectives
  - conduct and evaluate a performance analysis using Queuing models

# A Golden Era in Computing

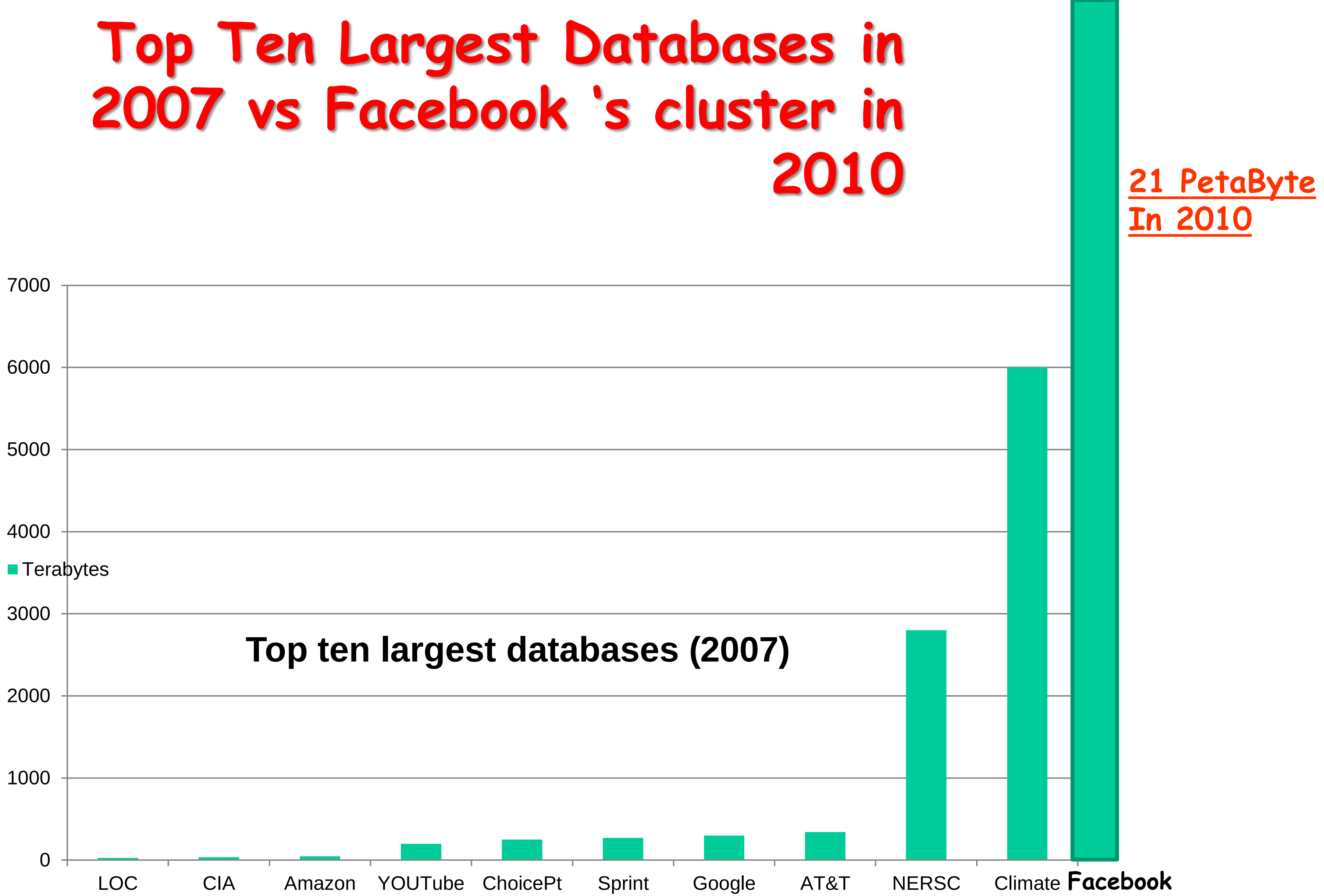


# Top Ten Largest Databases




**Ref:** http://www.comparebusinessproducts.com/fyi/10-largest-databases-in-the-world/

# Top Ten Largest Databases in 2007 vs Facebook 's cluster in 2010




**Ref:** http://www.comparebusinessproducts.com/fyi/10-largest-databases-in-the-world

# Computing Challenges

- Scalability issue: large scale data, high performance computing, automation, response time, rapid prototyping, and rapid time to production
- Need to effectively address (i) ever shortening cycle of obsolescence, (ii) heterogeneity and (iii) rapid changes in requirements
- Transform data from diverse sources into intelligence and deliver intelligence to right people/user/systems
- How to store the big-data? What new computing models are needed?
- What about providing all this in a cost-effective manner?
- How to make computing available and accessible as a public resource?
- Most of all how to store and process the data collected by the numerous devices and embedded system in our environment? What will serve as backend for these numerous devices?
- Democratization of computing: Let everyone (whether computer scientist, engineer or not) enjoy the benefits of technological advances

# Cloud Approach

- **Cloud computing** is Internet-based computing, whereby shared resources, software and information are provided to computers and other devices on-demand, like the electricity grid.
- The cloud computing is a culmination of numerous attempts at large scale computing with seamless access to virtually limitless resources.
  - on-demand computing, utility computing, ubiquitous computing, autonomic computing, platform computing, edge computing, elastic computing, **grid computing**, …
- Virtualization is the enabler technology for many offering of the cloud services

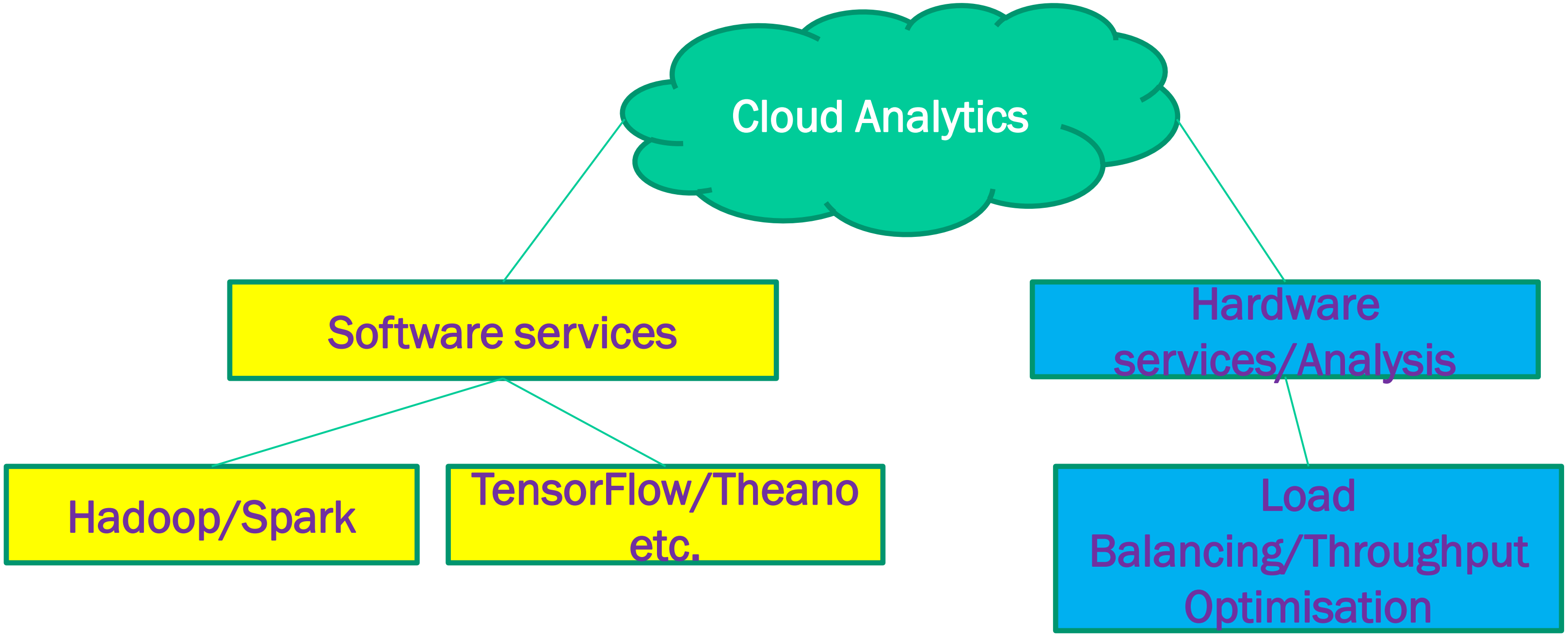
# Cloud Computing

- Cloud provides processor, software, operating systems, storage, monitoring, load balancing, clusters and other requirements as a service
- Pay as you go model of business
- When using a public cloud the model is similar to renting a property than owning one.
- An organization could also maintain a private cloud and/or use both.
- Cloud computing models:
  - platform (PaaS), Eg., Windows Azure
  - software (SaaS), Eg., Google App Engine/Google compute Engine
  - infrastructure (IaaS), Eg., Amazon AWS
  - Services-based application programming interface (API)

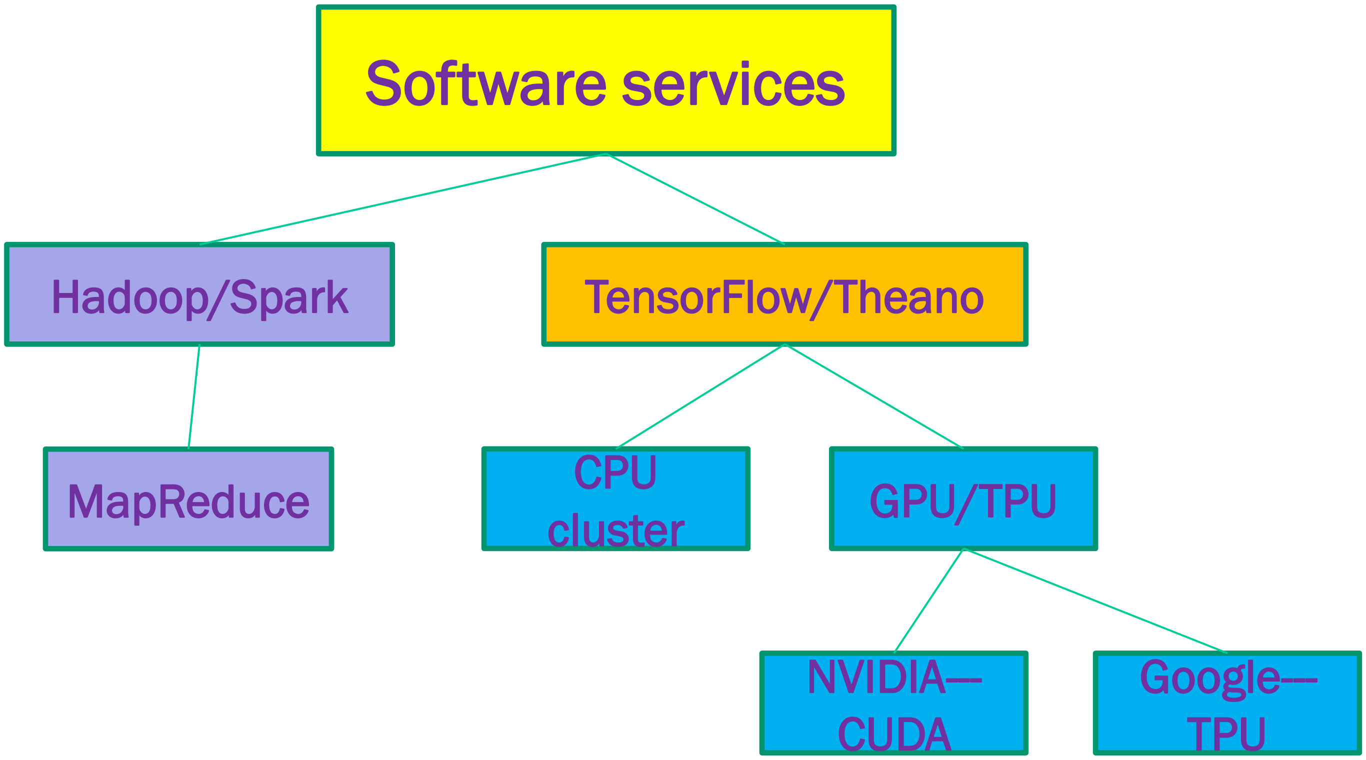
# Two Main Areas

- Software: Analytics Implementation for Scalable programming
  - MapReduce: general analytics
  - TensorFlow: learning
- Hardware and Architecture
  - How to scale in a reliable fashion
  - System reliability models

# Focus of Course



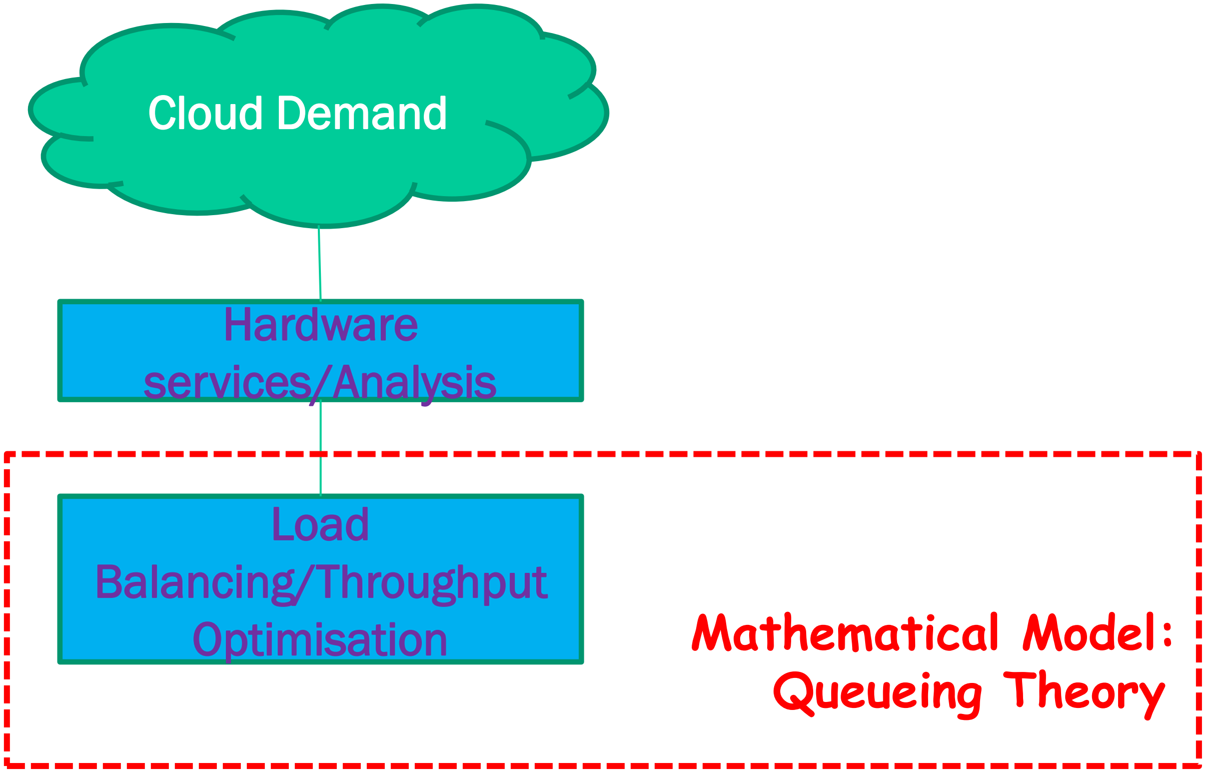
# Software Services



# Software Abstractions

- Issue
  - Parallel programming is extremely complex
- Software abstractions developed
  - MapReduce: for data centre cloud paradigm
  - Computation graph: for deep learning cloud paradigm
- Focus on theory, not tools
  - Tools will change, but not theory

# Hardware Focus



# Logistics

- Class time: Tues. 4-5; Fri. 3-4
- Location: WGB 1.06
- Instructor: Gregory Provan
  - Office: WGB 1-71
  - Tele: 420-5928
  - E-mail: g.provan@cs.ucc.ie
  - Office Hours:
    - Tuesday 11-1
    - By appointment

# 6423 Topics

- Introduction to Network Modeling
- Modeling tools (2)
- Cloud Computing (5)
  - Algorithms for Cloud Computing: MapReduce
- Performance Modeling (5)
  - Stochastic Models
  - Discrete-Event Models
- Reliability/Fault Modeling (2)
  - Model-Based Approaches
  - Reliability Models
- Integrated Modeling for Scalable Computing (3)

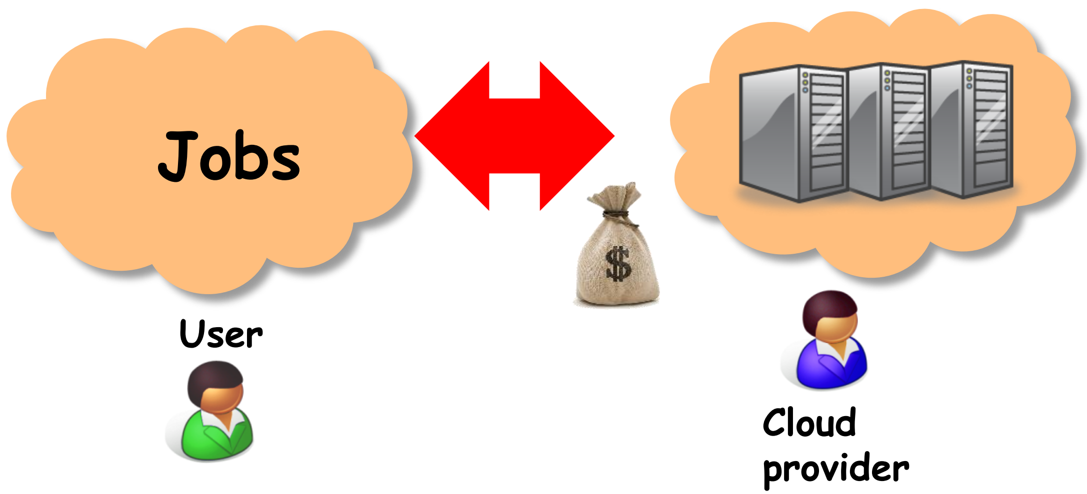
| Week | Date | Lecture | Section | Topic | Assignment |
|---|---|---|---|---|---|
| 1 | 15/01/2020 | 1 | **Introduction** | Course Objectives. Review: Graph Theory, Discrete Probability Theory | |
| | 17/01/2020 | 2 | | Review: Discrete Probability Theory | |
| | 22/01/2020 | 3 | **Distributed Network Inference: Cloud Computing** | Cloud Computing; MapReduce Framework | |
| 2 | 24/01/2020 | 4 | | MapReduce algorithms | |
| | 29/01/2020 | 5 | | Distributed Graph Algorithms I | |
| 3 | 31/01/2020 | 6 | | Distributed Graph Algorithms II | |
| | 05/02/2020 | 7 | | Other Distributed Algorithms | |
| 4 | 07/02/2020 | 8 | **Distributed Learning** | Deep learning computational tools | |
| | 12/02/2020 | 9 | | TensorFlow | |
| 5 | 14/02/2020 | 10 | | Caffe | |
| | 19/02/2020 | 11 | | Computation graphs and hardware implementation | |
| 6 | 21/02/2020 | 12 | **Network Performance** | Probability Review: Continuous | Practice Mid-Term |
| | | 13 | | Queueing Theory: M/M/1 | |
| 7 | | | **EXAM WEEK** | | |
| 8 | 04/03/2020 | 14 | | M/M/k models Queueing Networks | |
| | 06/03/2020 | 15 | | Priority Queues | |
| 9 | 11/03/2020 | 16 | **Network Performance Analysis (cont.)** | Performance Analysis Problem-Solving | |
| | 13/03/2020 | 17 | | | |
| 10 | 18/03/2020 | 18 | | Reliabilty Block Diagrams | |
| | 20/03/2020 | 19 | | Markov Models | |
| 11 | 25/03/2020 | 20 | **Network Reliability** | | |
| | 27/03/2020 | 21 | **Reliability Modeling** | Integrated Modeling | |
| 12 | 01/04/2020 | 22 | | Performance Networks | |
| | 03/04/2020 | 23 | | Integrated Modeling Problem-Solving | |
| | | | | | |
| | | | | | |
| 11 | 10/04/2020 | 23 | | | Problem set |
| | 12/04/2020 | 20 | | | |
| 12 | 17/04/2020 | 21 | | Course Review | Practice Final |
| | 19/04/2020 | | | Exam Problem-Solving | |
| | **TBD** | | **END-OF-TERM EXAM** | | |

# Course Evaluation

- Grading
  - Midterm Exam (week 7) 40%
  - Final exam 40%
  - Continuous Assessment 20%
- Mid-Term Exam
  - Covers the main principles introduced in class up to that point
- Final Exam
  - Covers the main principles introduced in class

# Mathematical Underpinning

- Modern computing is highly distributed
  - Need graphs to represent structures
- Jobs arrive stochastically
  - Need to use probability theory



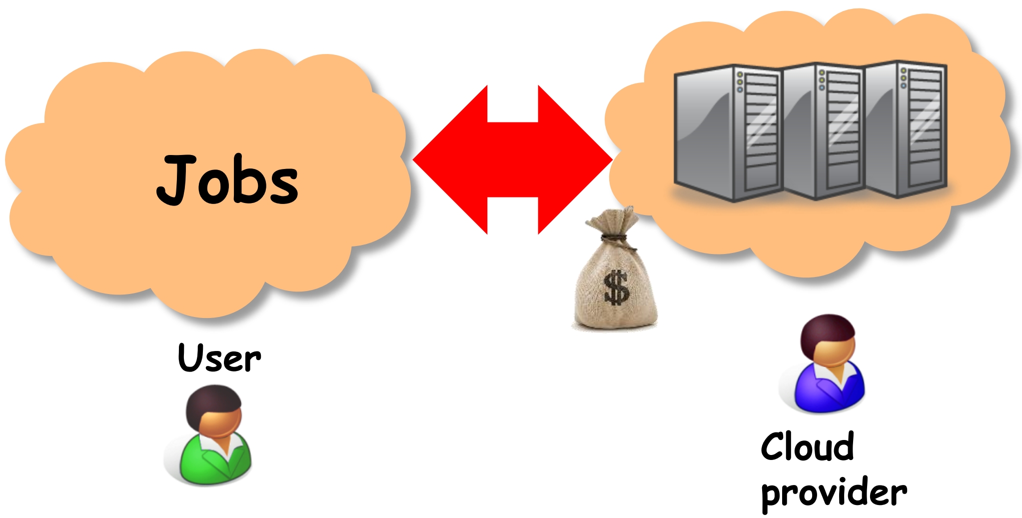
# Course Pre-Requisites

- Pre-requisites
  - Basic mathematics background necessary
    - Discrete mathematics
    - Probability theory
- Mathematical tools used
  - Graph theory
  - Probability theory
- We will review these tools

# Topic 1: Cloud Computing

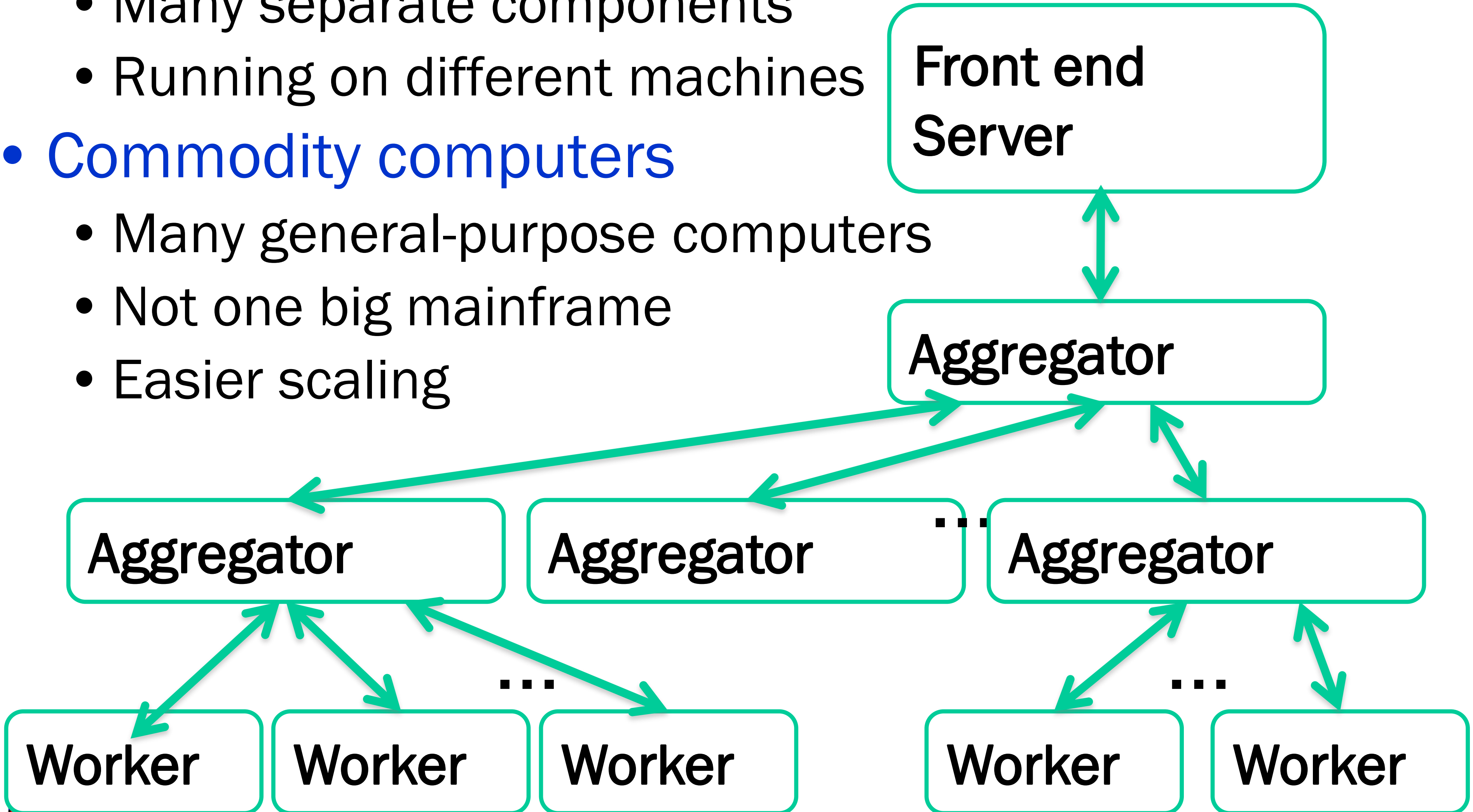
- What is the cloud computing paradigm
- How to characterise its behaviour
- Cloud computing software
    - MapReduce

# Cloud Computing Concept

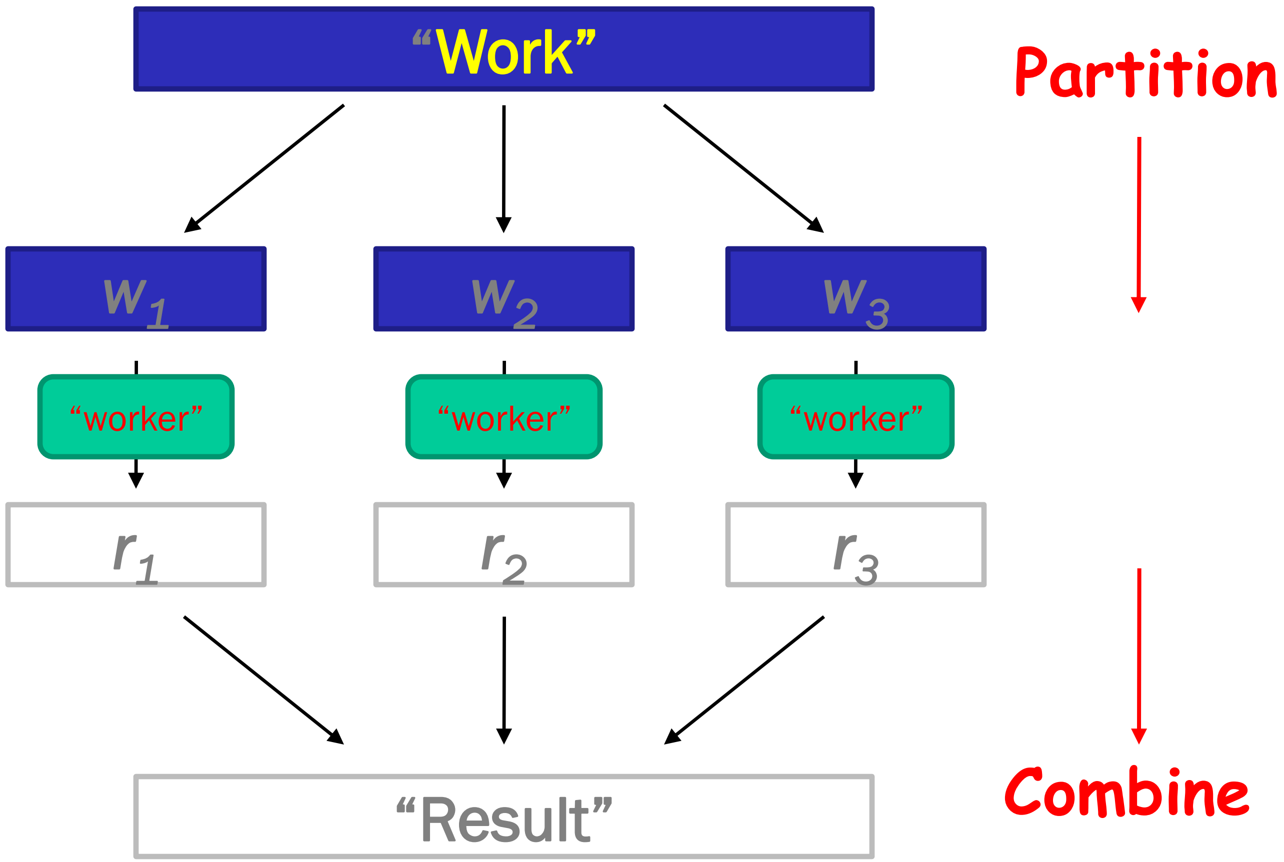


# Multi-Tier Applications

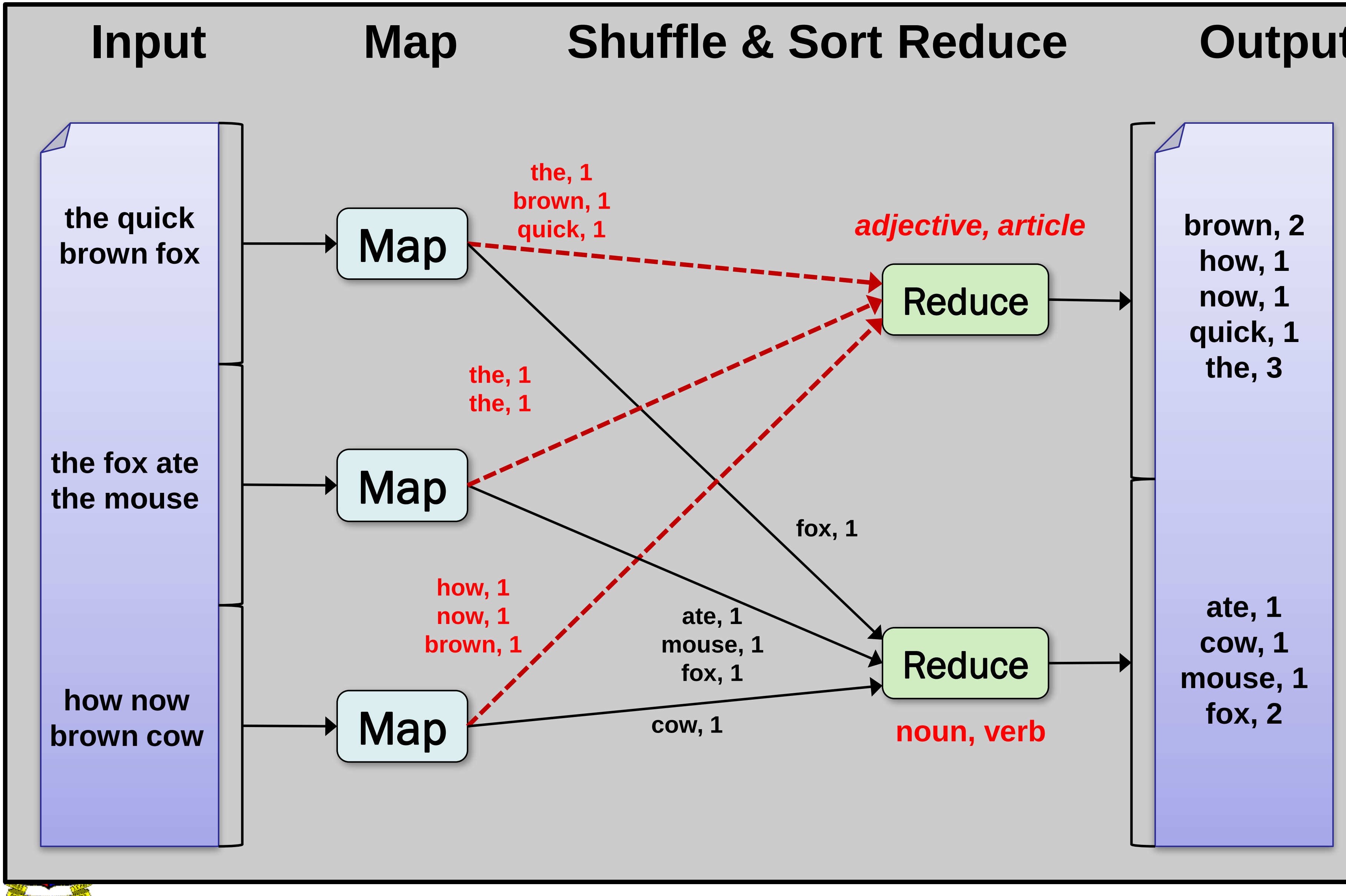
- Applications consist of tasks
  - Many separate components
  - Running on different machines
- Commodity computers
  - Many general-purpose computers
  - Not one big mainframe
  - Easier scaling



# Cloud Algorithm: MapReduce



# Word Count Example



# Parallelization Challenges

- How do we assign work units to workers?
- What if we have more work units than workers?
- What if workers need to share partial results?
- How do we aggregate partial results?
- How do we know all the workers have finished?
- What if workers die?

**What is the common theme of all of these problems?**

# MapReduce: Strengths and Weaknesses

- Works best for sequential operations
  - Mathematics: commutative monoid
- Not good for
  - Recursive operations
    - e.g., multi-step convergence
  - Non-commutative operations
    - e.g. matrix multiplication

# Topic 2: Scalable Machine Learning

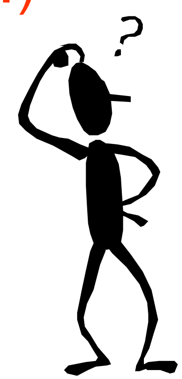
- MapReduce not great for learning tasks
- Scalable learning paradigm
  - Computation graph (TensorFlow)
- Define programming flow as a directed graph
  - Each operation uses a Tensor **Tensor**
  - Flow of data through the system **Flow**

# Topic 3: Scalable Systems for Network Performance

- Goal: analyse network performance (QoS metrics)
  - Throughput
  - Response time
    - Time for an internet query to be processed due to network traffic
- Outcome
  - Can design networks to achieve target QoS
    - Cloud, Internet, mobile phone network

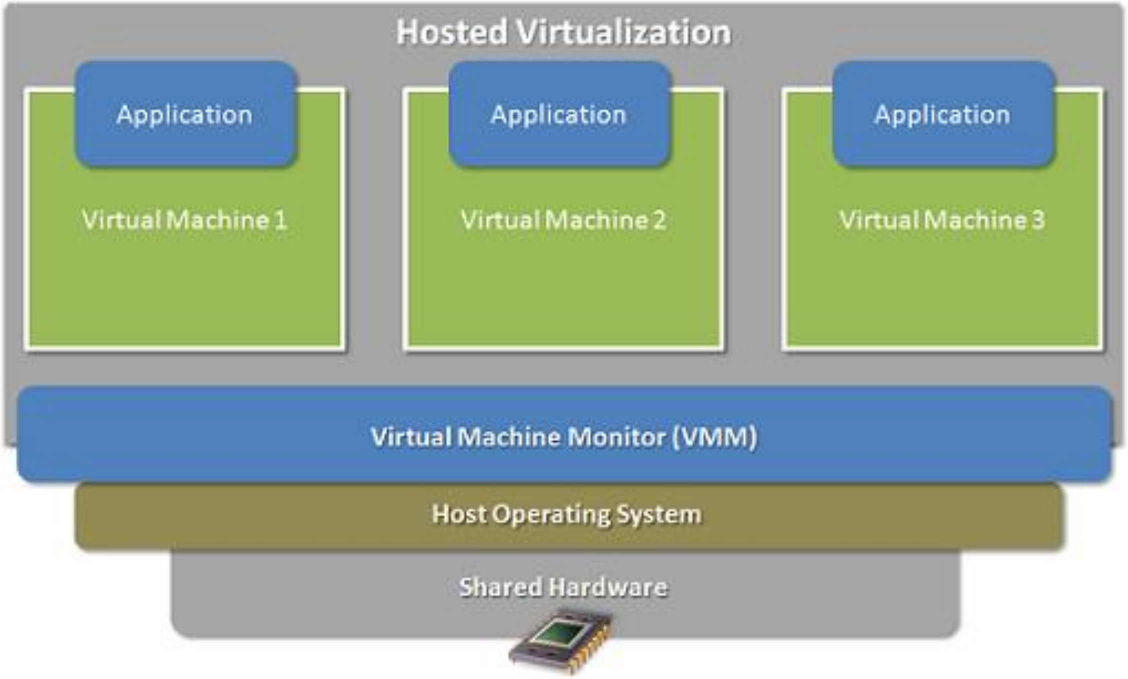
# Realizing the ‘Computer Utilities’ Vision: What Consumers and Providers Want?

- Consumers – minimize expenses, meet QoS
  - How do I express QoS requirements to meet my goals?
  - How do I assign valuation to my applications?
  - How do I discover services and map applications to meet QoS needs?
  - How do I manage multiple providers and get my work done?
  - How do I outperform other competing consumers?
  - …
- Providers – maximise Return On Investment (ROI)
  - How do I decide service pricing models?
  - How do I specify prices?
  - How do I translate prices into resource allocations?
  - How do I assign and enforce resource allocations?
  - How do I advertise and attract consumers?
  - How do I perform accounting and handle payments?
  - …




- Mechanisms, tools, and technologies
  - value expression, translation, and enforcement

# Enabling Technology: Virtualization




- Multiple virtual machines on one physical machine
- Applications run unmodified as on real machine
- VM can migrate from one computer to another

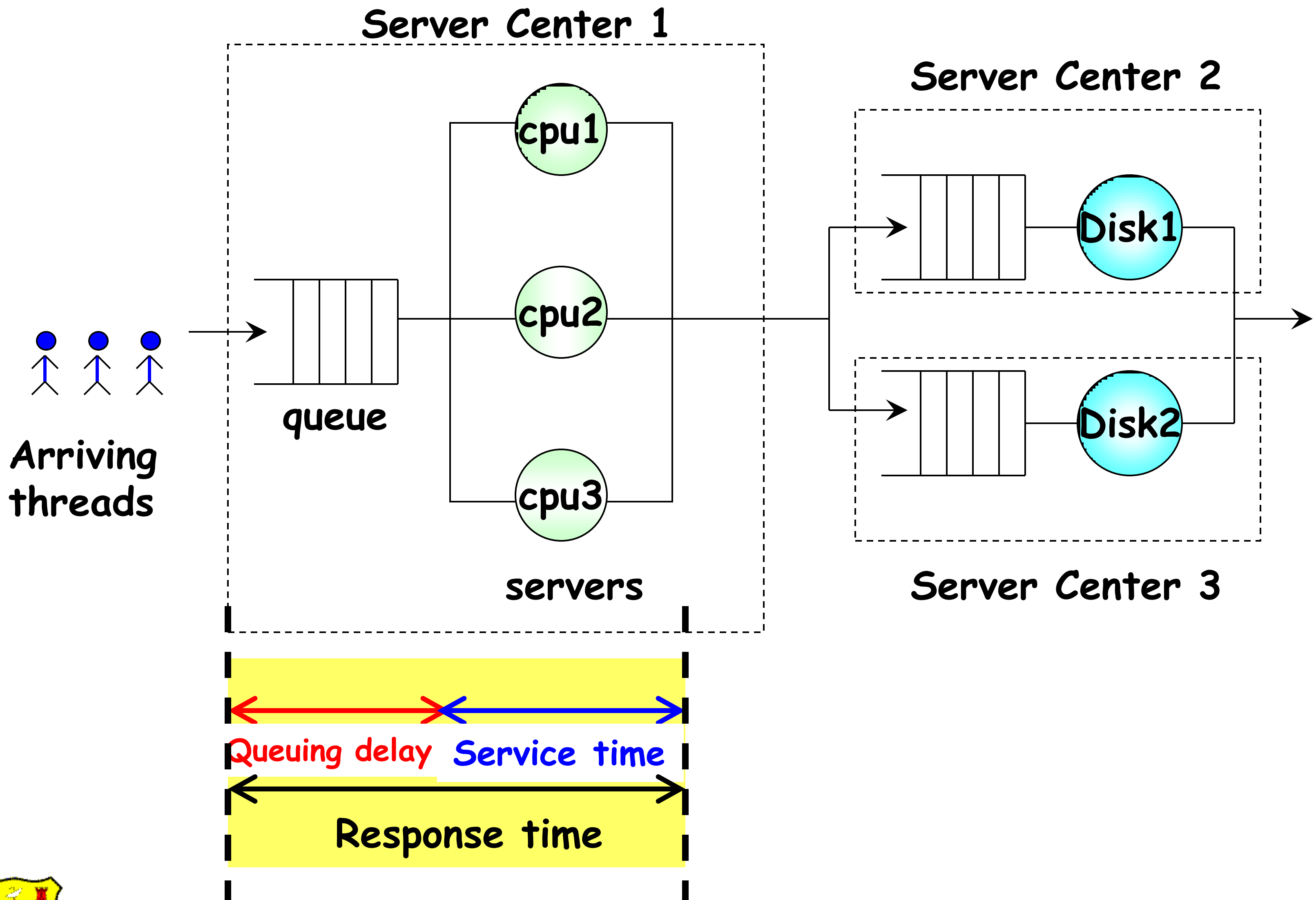
# How to Characterise Networks?

- Issues
  - *Stochastic behaviour:* randomness in traffic, faults, etc.
  - *Complexity*: measure of network “density”
  - *Performance*: what is target QoS?
  - *Dependability*:  how reliable is the network functionality
- Issues are in constant tension
- Examine how to define good tradeoffs

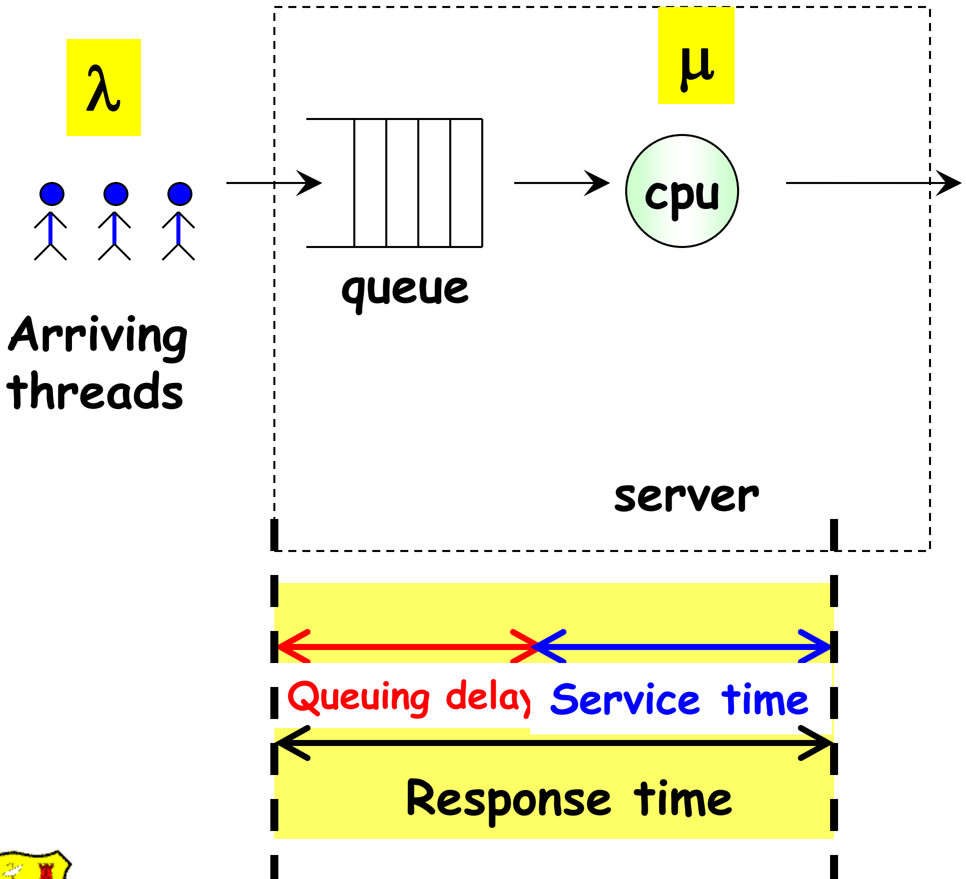
# Integrating "Performance" in Analysis of Complex Networks

- Two aspects to performance modeling
  - Optimal performance
  - Performance under degraded conditions
- Study issues separately and integrate them
  - Performance
  - System failure

# Computer-System Example



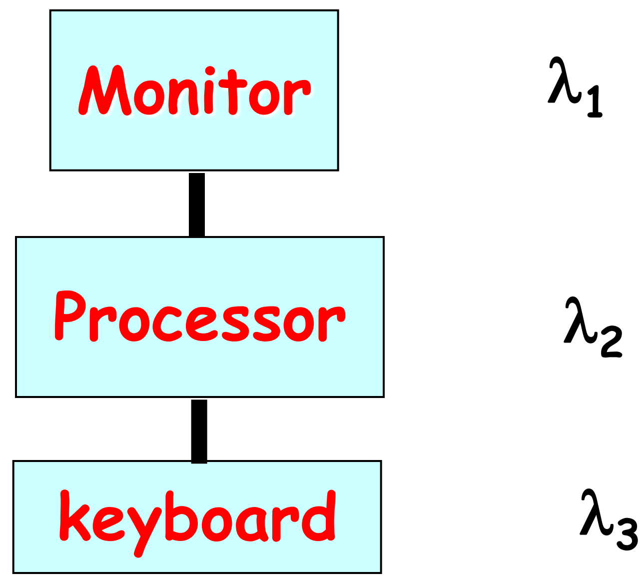
# Network Performance Analysis




- Mathematical tool: queueing theory
  - Probabilistic analysis
- Arrival rate: $\lambda$
- Service rate: $\mu$
- Response time: $\frac{1}{(\mu - \lambda)}$

# Topic 4: Network Failure Analysis

- What is the probability that this system functions normally?
  - Define failure rates for individual components

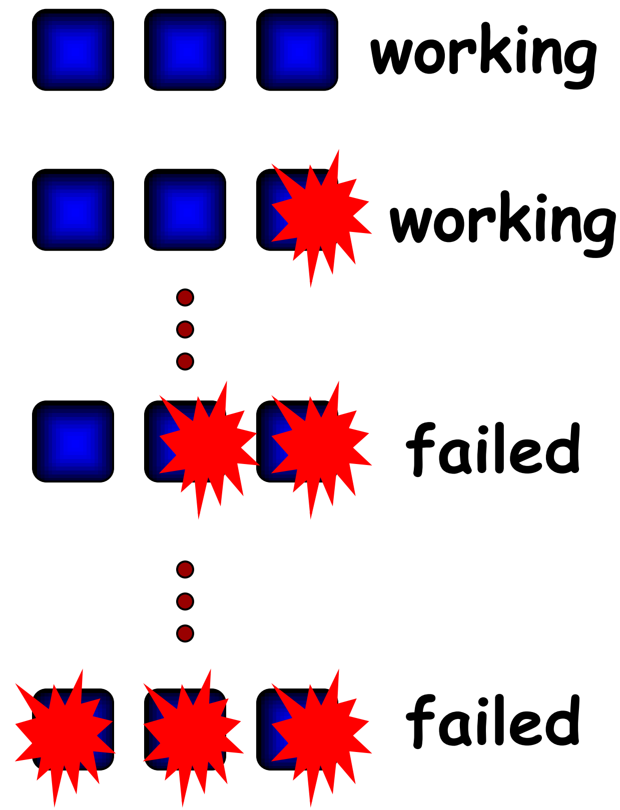


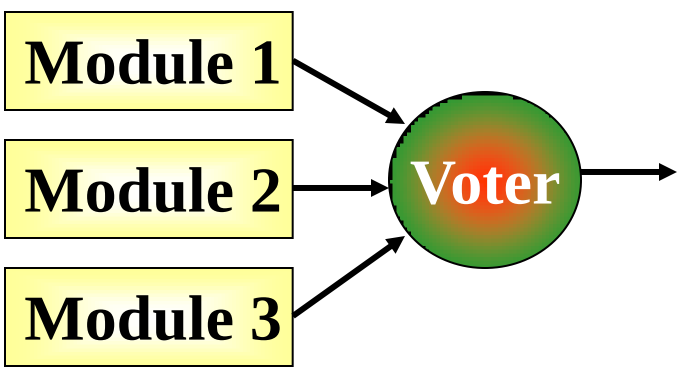
# Topic 5: Integrated Performance/Failure Analysis

- Compute network performance in the presence of faults
- Integrate 2 models
  - Performance model
  - Failure-rate model

# Integrated Model: Example

- Derive the Performability as a function of time
- The performance metric is considered only "working" or "failed". (Reliability)



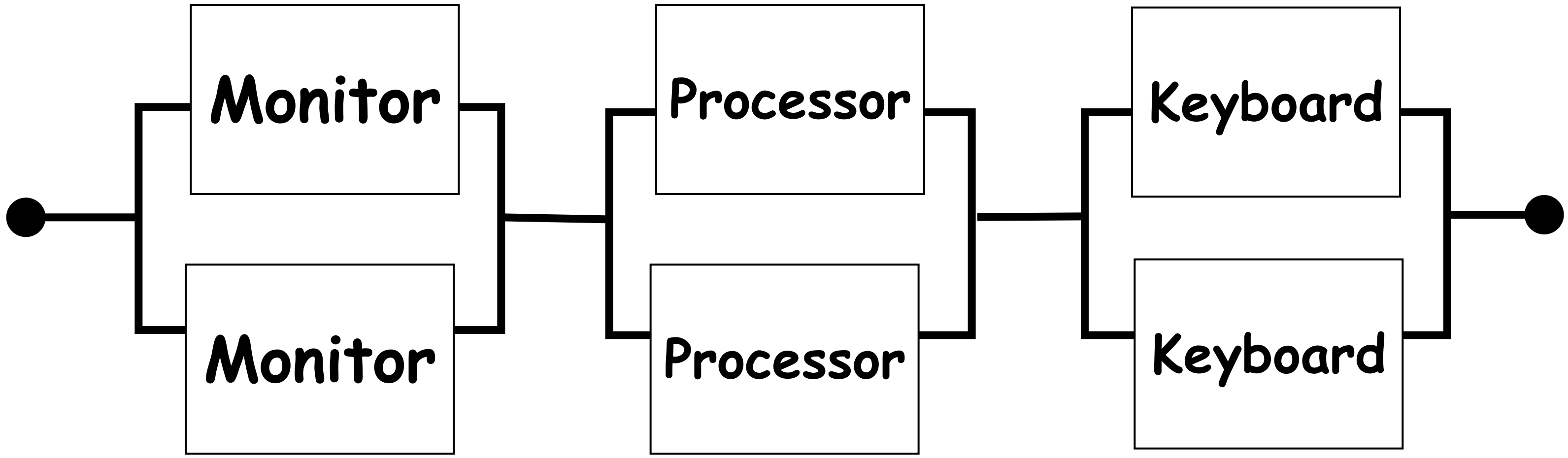



# Topic 6: Designs to Tolerate Faults—Fault-Tolerant Computing

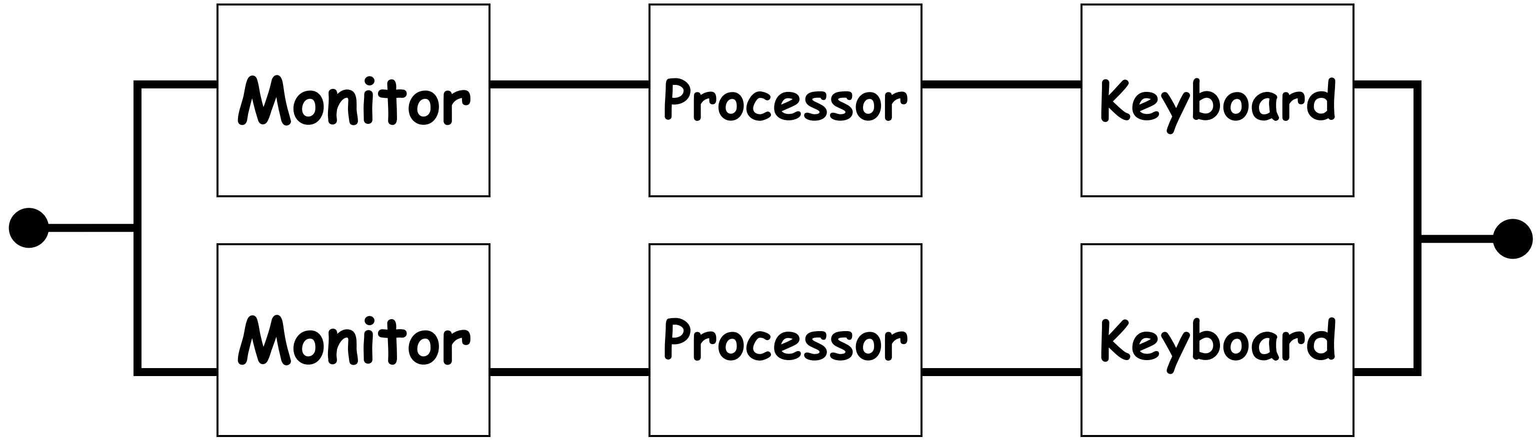
- Fault-tolerant computing is a generic term describing redundant design techniques with duplicate components or repeated computations enabling uninterrupted (tolerant) operation in response to component failure (faults).

# Comparing System Topologies

Assuming Buses are perfect




Which topology has better reliability?



# Text & References

Kishor S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, second edition, John Wiley & Sons, Inc. 2002, ISBN 0-471-33341-7.

Introduction to Wireless and Mobile Systems, by Dharma Prakash Agrawal and Qing-An Zeng, ISBN No. 0534-40851-6.

**References:**

[1] Robin A. Sahner, Kishor S. Trivedi, Antonio Puliafito, *Performance and Reliability Analysis of Computer Systems – An Example-Based Approach Using the SHARPE Software Package*, Kluwer Academic Publishers, 1996. ISBN 0-7923-9650-2.

[2]Martin L. Shooman, *Reliability of Computer Systems and Networks,Fault Tolerance, Analysis, and Design,* John Wiley & Sons, Inc., 2002. ISBN 0-471-29342-3.

[3] http://www.crhc.uiuc.edu/PERFORM/home.html

[4] http://www.eecs.umich.edu/~jfm/

[5] http://www.ee.duke.edu/~kst/