IEMS 5780 / IERG 4080
Building and Deploying Scalable Machine Learning Services

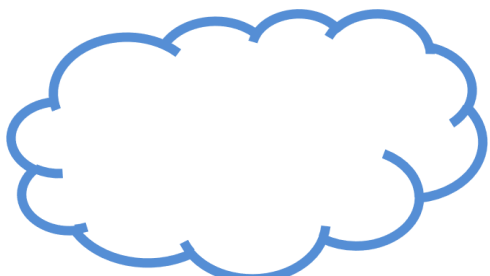
Lecture 13 - Deploying Machine Learning Applications

Albert Au Yeung
4th December, 2018

Cloud Computing

Cloud Computing

- What is **cloud computing**?



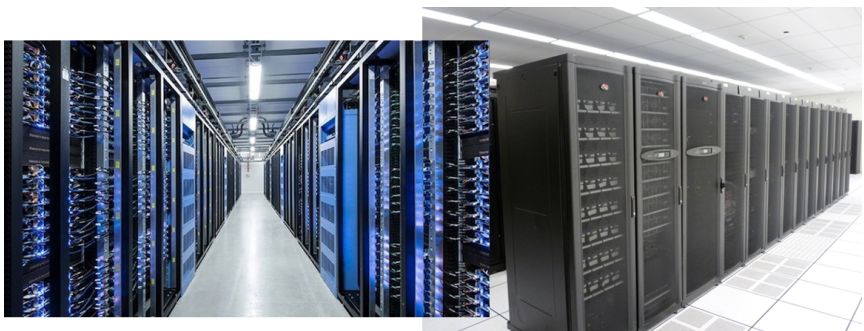
Servers

- What people do when they need to run a network application?

THe first Web server (a NeXT computer)

Data Centres

- What people do when they need to run a network application?



Data centres

Data Centre Services

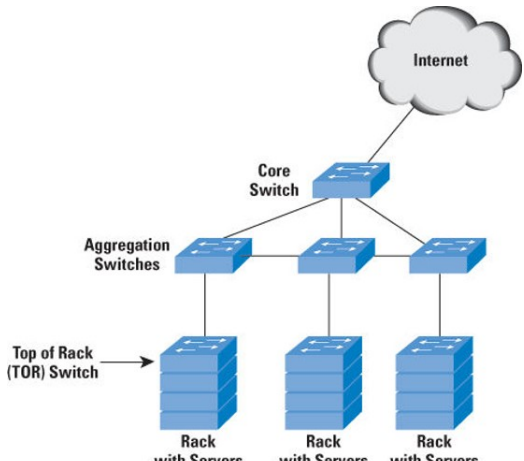


Figure from "Cloud Computing - A Primer - The Internet Protocol Journal", The Internet Protocol Journal, Volume 12, No.3

Data Centre Services

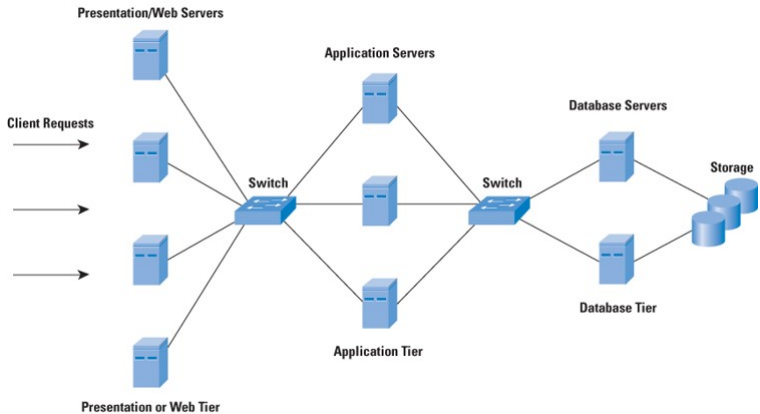


Figure from "Cloud Computing - A Primer - The Internet Protocol Journal", The Internet Protocol Journal, Volume 12, No.3

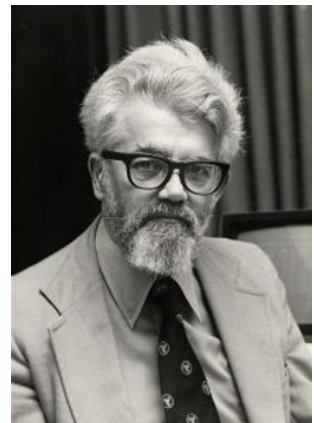
Cloud Computing

- **NIST** (National Institute of Standards and Technology)
  - “Cloud computing is a model for enabling **ubiquitous**, **convenient**, **on-demand** network access to a shared pool of configurable **computing resources** (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”
- Ref: http://www.nist.gov/itl/cloud/

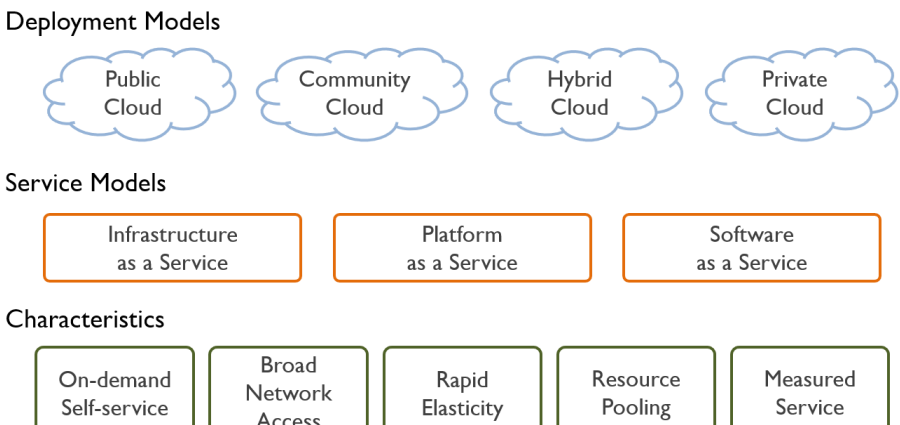
Cloud Computing

John McCarthy
(who invented the term "Artificial Intelligence")

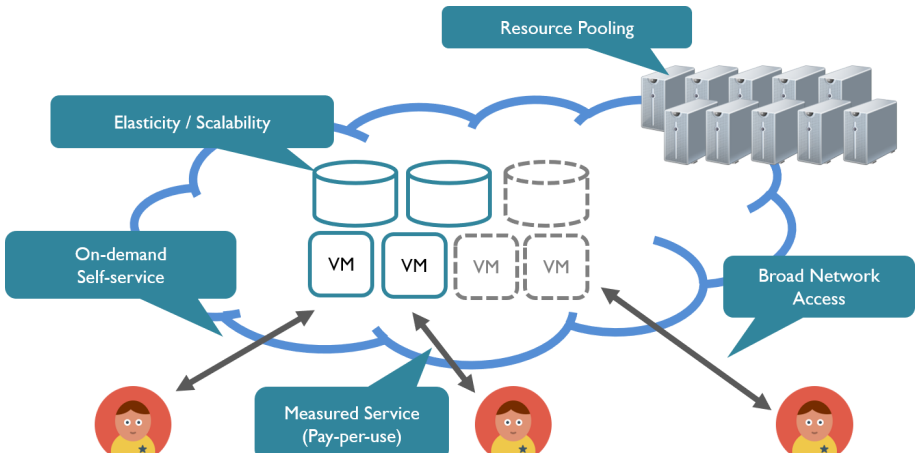
- The first to suggest publicly (in 1961 in a speech given to celebrate MIT's centennial) that computer time-sharing technology might result in a future in which computing power and even specific applications could be sold through the utility business model (like water or electricity).



Cloud Computing



Cloud Computing Characteristics



Infrastructure-as-a-Service (IaaS)

- To provision **processing**, **storage**, **networks**, and other fundamental **computing resources** where the consumer is able to deploy and run arbitrary software (e.g. virtual machines)
- Consumers do not manage or control the underlying cloud infrastructure but have control over **operating systems**, **storage**, and **deployed application**s; and possibly limited control of select **networking components**

Platform-as-a-Service (PaaS)

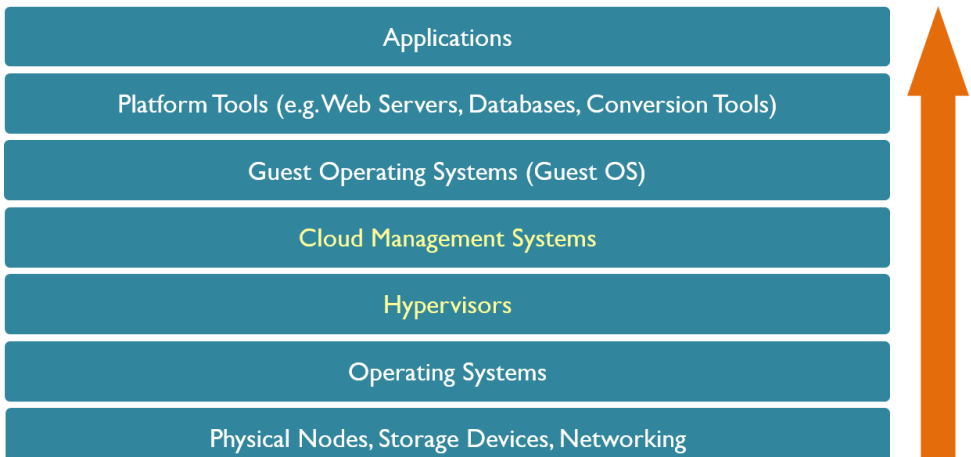
- To deploy onto the cloud infrastructure **consumer-created or acquired applications** created using programming languages, libraries, services, and tools supported by the provider.
- Consumers have control over the **deployed applications** and possibly **configuration settings** for the application-hosting environment.

Software-as-a-Service (SaaS)

- To use the provider's **applications** running on a cloud infrastructure, which are accessible from various client devices through either a thin client interface.
- Consumer do not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities

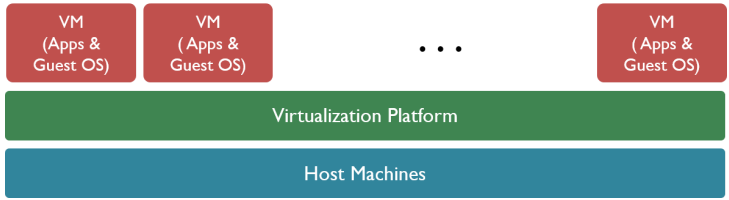
Enabling Technologies of Cloud Computing

What Makes Cloud Computing Possible?



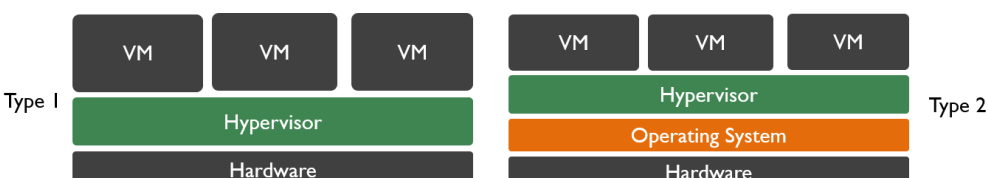
Virtualization Technologies

- **Virtualisation** divides the resources of a computer into multiple **isolated** execution environments, by applying one or more concepts or technologies such as hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, etc.



Hypervisors

- **Virtualisation** is enabled by software called **hypervisors**
- What does a hypervisor do?
  - Provide **isolated execution environment** for each VM
  - Manage **access of physical resources** by each VM
- Two types of hypervisors:

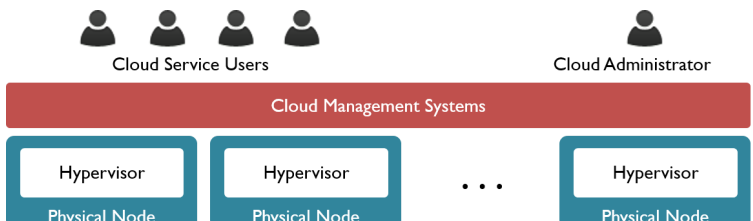


Hypervisors

- Hypervisors get its name because it is conceptually **one level higher than a supervisory program (part of an OS)**.
- **Type 1** hypervisors run directly on bare metal instead of within an operating system environment - provide the best performance, availability, and security of any form of hypervisor.
- **Type 2** hypervisors run within an operating system environment running on the host computer - typically referred to as hosted virtualization

Cloud Management Systems

- A layer above hypervisors
- **Manage** a cloud infrastructure, which may include many physical nodes, networking devices, storage devices, etc.



Cloud Management Systems

Some of the functions of a CMS:

- Resource allocation (Determine where to create a VM)
- Resource monitoring (Usage of physical resources)
- Enforcement of resource, security and configuration policies
- User management & billing
- VM image management
- User interface for on-demand self-service
- ...

Auto-scaling

Cloud & Scalability

Cloud computing allows you to **add** or **remove** computing resources more quickly and easily

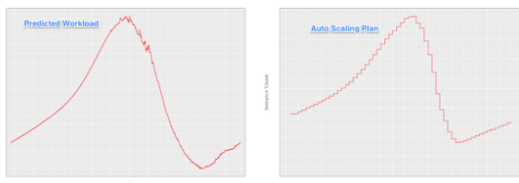
- Adding one more application server
- Adding new hard disk storage space
- Adding extra database servers
- ...

Auto-scaling

- Given that everything is **virtual** in the cloud, all changes to the configurations of the virtual machines, applications, firewalls, etc., can be automated
- **Auto-scaling** refers to the idea of automatically adding or removing computing resources in the system **based on the actual usage** in real-time
- Auto-scaling can be triggered by **different rules**, for example:
  - Based on a schedule (E.g. start three more application servers during Christmas)
  - Base on demand (work load)
    - Example 1: start one more server if the average CPU utilization rate is over 80% for 10 minutes)
    - Example 2: stop one server if the average CPU utilization rate is less than 20% for 10 minutes)

Case Study

- Netflix is an online video streaming company and it servers millions of users at different times.
- However, the load on the system can be different at different hours and on different days



- Imagine if you can predict the load on the application, you can auto-scale in advance
  - React to changes in demand more quickly
  - Save the amount of $$$ spent on unused resources
- Ref 1: http://techblog.netflix.com/2012/01/auto-scaling-in-amazon-cloud.html
- Ref 2: http://techblog.netflix.com/2013/11/scryer-netflixs-predictive-auto-scaling.html

Scaling using Cloud Services

- Take a look at how auto-scaling can be configured in **Amazon AWS**
  Auto Scaling EC2 With Custom Scaling Policy
  https://www.youtube.com/watch?v=5swEiz0i-kE
- Another video introducing similar function in **Google Cloud**
  Learn how to scale your applications with Google Compute Engine
  https://www.youtube.com/watch?v=TfbEwfYjKl4

Docker

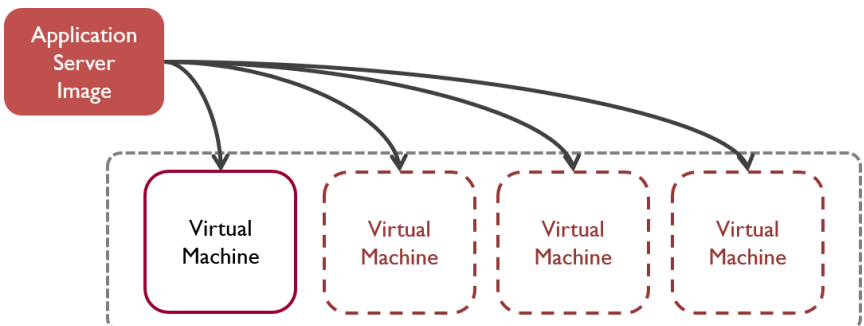
Deploying Applications

Deploy: to put your application into production

- How would you develop and deploy your application?
  - Set up a development environment to develop your app
  - Set up the production environment, and copy your application to the production machine
- **Problem?**
  - It takes time to set up the production environment

Deploying Applications

Does using **cloud** and **virtual machines** solve the problem?



Deploying Applications

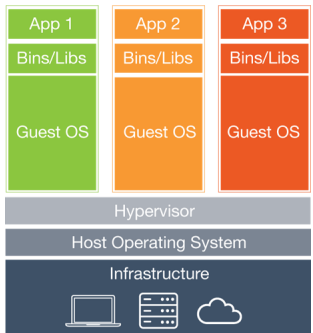
Limitations of using Virtual Machines (VMs)

- They run on hypervisor, and thus are not particularly efficient in using the resources of the machine
- The size of the image is usually very large (at the order of GB)
- Requires some time to boot and start up (at the order of minutes)
- One physical machine can only support at most tens of VMs

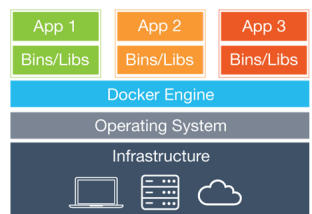
Docker

- Docker is a technology that provides virtualisation solution and allows a developer to package an application with all of its dependencies into a standardised unit for deployment.
- A relatively **lightweight** virtualisation compared to VMs
- Applications run inside **containers** and are **isolated** from each other
- A container can be started in **seconds**
- More efficient use of computing resources on the machine

Docker



Virtual Machines
Each VM contains a guest OS on which the binaries, libraries and applications are executed.

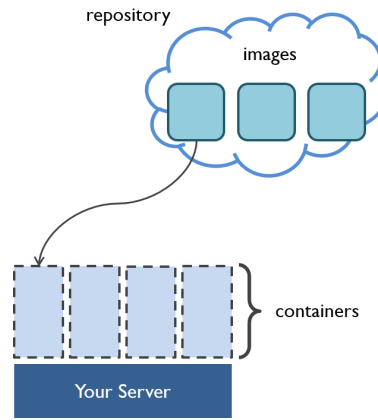


Containers
Containers include only the necessary binaries and libraries for running the application.

- See also: https://docs.docker.com/get-started/#containers-and-virtual-machines

Basic Concepts in Docker

- **Image**:
  a read-only file containing the application
- **Container**:
  an instance of an image for executing the application
- **Repository**:
  where images are stored, like GitHub
  - Can be private or public
  - The largest public repository is Docker Hub



Creating a Docker Image

- To create a docker container that runs your application, you need to:
  - Create a docker image (usually from a **base image**)
  - This is done by write a **Dockerfile**, which describes how the image is created/built
- Let's say we have a very simple Web application written in Flask

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def index():
    return "Hello!"

if __name__ == "__main__":
    app.run(host="localhost", port=5000)
```

Creating a Docker Image

- We can create a docker image for this application using the **`ubuntu:latest`** base image
- Our Dockerfile:

```
# Use the python3.6 base image
FROM python:3.6

# Set working directory to /app
WORKDIR /app

# Copy our application file into the image
COPY app.py /app

# Install our dependency (Flask in this case)
RUN pip install Flask

# Run app.py when the container is launched
CMD ["python", "app.py"]
```

Creating a Docker Image

- Building the **image** using the Dockerfile:
- (**`-t`** assigns a **tag** to the image created)

```
$ sudo docker build -t flask-app:1.0 .
```

- Creating a **container** running the app using the image built above

```
$ sudo docker run -p 5000:5000 flask-app:1.0
```

- Note: we **forward** connections on port 5000 of the host machine to the port 5000 of the docker container

End of Lecture 13