Advanced Cloud Computing
Service Models and Challenges

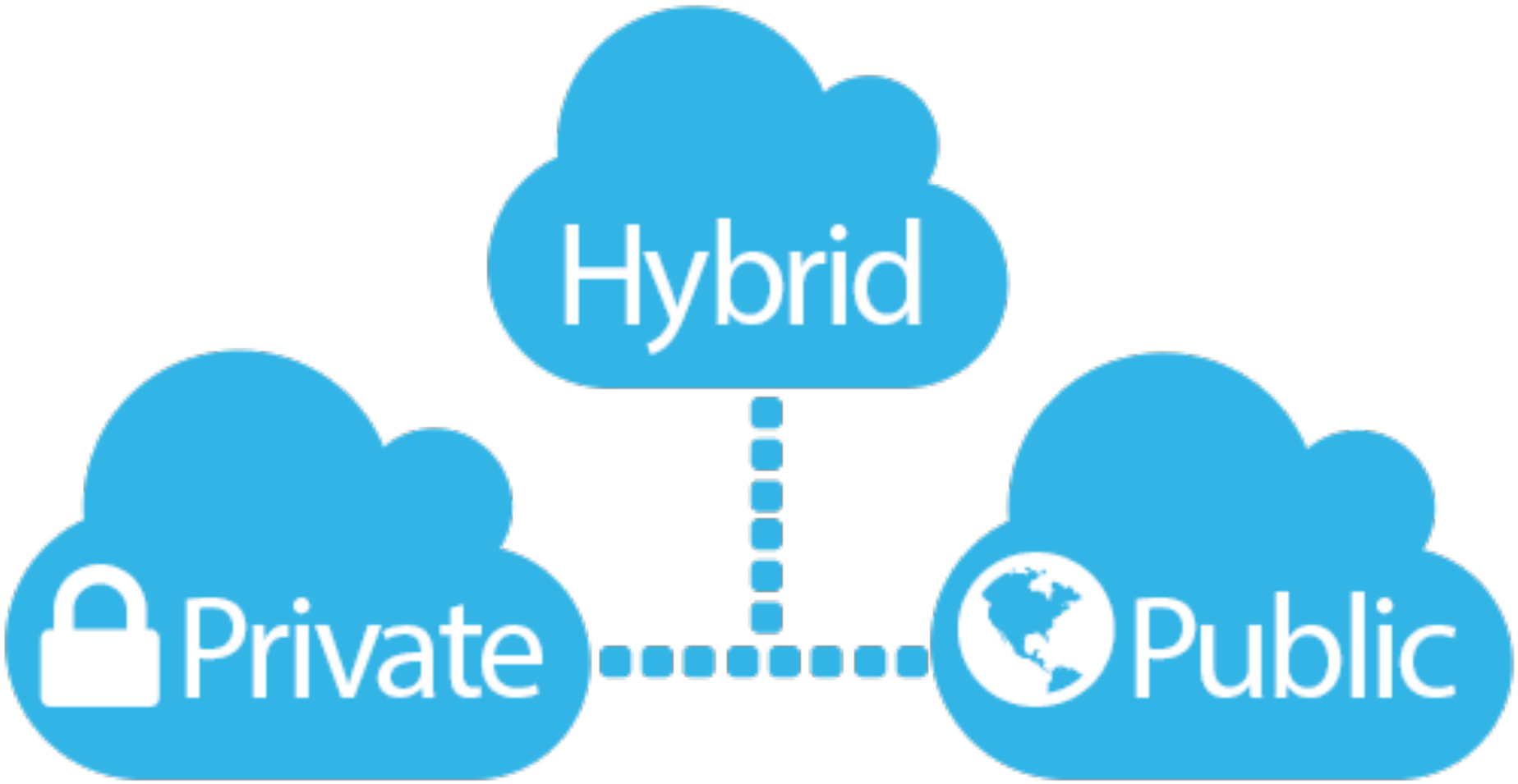
Wei Wang
CSE@HKUST
Spring 2022

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

Outline

- Cloud deployment models
- Service models
- Issues of Cloud
- Challenges

Cloud deployment models



Public Cloud

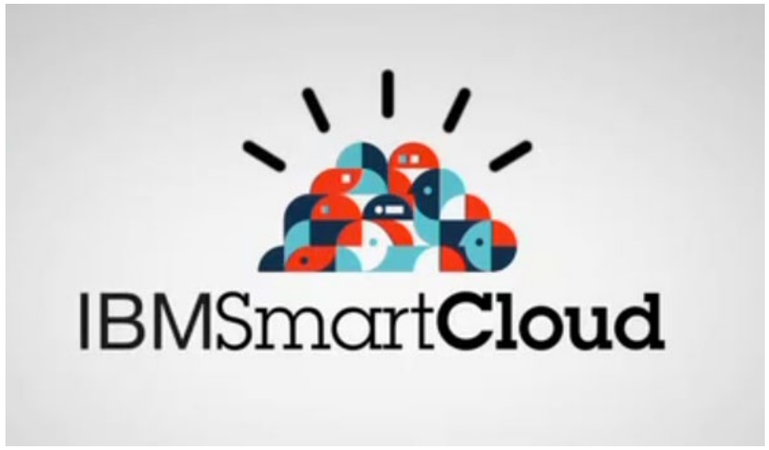
- Providers let clients access the cloud via Internet
- Made available to the general public

Public Cloud

- Multi-tenant virtualization, global-scale infrastructure
- Functions and pricing vary

Private Cloud

- The cloud is used solely by an organization (e.g. HKUST, Facebook, HSBC)
- May reside in-house or off-premise



Private Cloud

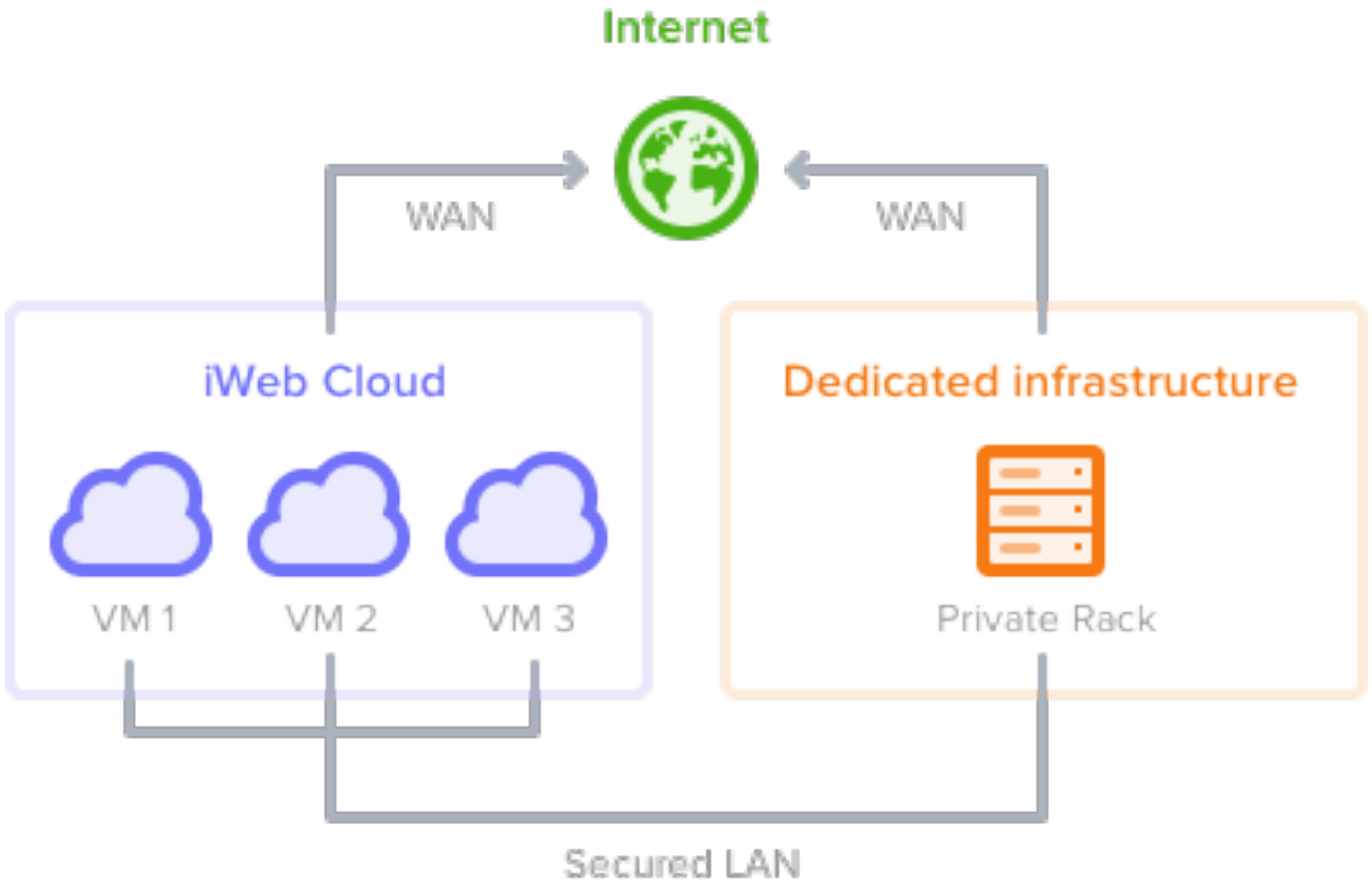
- Secure, dedicated infrastructure with the benefits of on-demand provisioning
- Not burdened by network bandwidth and availability issues and security threats associated with public clouds.
- Greater control, security, and resilience.

Hybrid Cloud

- Composed of multiple clouds (private, public, etc.) that remain independent entities, but interoperate using standard or proprietary protocols
- Banks, hospitals, government

Hybrid Cloud

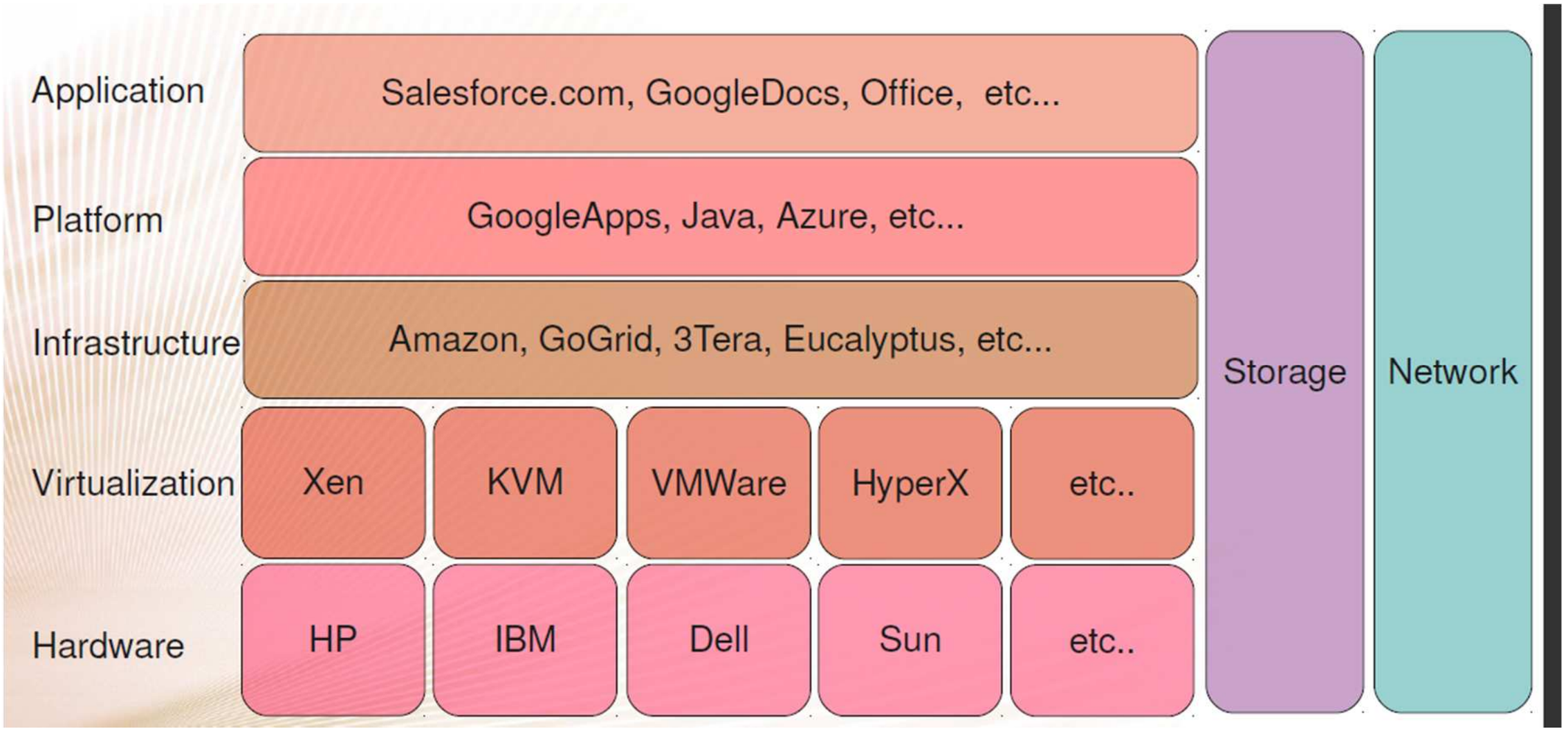
- Allows applications and data to flow across clouds



Copyright: iWeb

Cloud Service Models

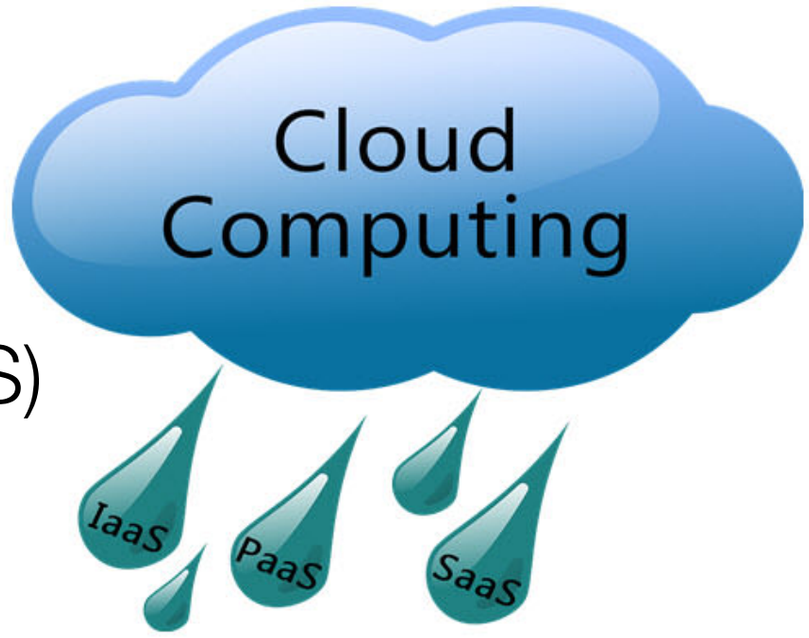
Cloud computing stack



By Nick Barcet, “What is Ubuntu Cloud”, Nov 2009

Cloud service models

- Infrastructure-as-a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)
- Other X-as-a-Service
  - Function-as-a-Service (FaaS)
  - Machine-Learning-as-a-Service (MLaaS)



Infrastructure-as-a-Service

- Providers give you the computing infrastructure made available as a service. You get “bare-metal” machines.
- Providers manage a large pool of resources, and use virtualization to dynamically allocate
- Customers “rent” these physical resources to customize their own infrastructure
- Full control of OS, storage, applications, and some networking components (e.g., firewalls)

Infrastructure-as-a-Service

Computation

Storage

Network

Amazon EC2

rackspace
the open cloud company

linode

IaaS use case

- Netflix rents thousands of servers, terabytes of storage from Amazon Web Services (AWS)
- Develop and deploy specialized software for transcoding, storage, streaming, analytics, etc. on top of it
- Is able to support tens of millions of connected devices, used by 40+ million users from 40+ countries

Platform-as-a-Service (PaaS)

- Providers give you a software platform, or middleware, where applications run
- You develop and maintain and deploy your own software on top of the platform
- The hardware needed for running the software is automatically managed by the platform. You can’t explicitly ask for resources.

PaaS

- You have automatic scalability, without having to respond to request load increase/decrease

- No control of OS, storage, or network, but can control the deployed applications and host environment

PaaS use case

- Best for web apps
- Language and API support: Python, Java, PHP, and Go

Google App Engine

Software-as-a-Service (SaaS)

- Providers give you a piece of software/application. They take care of updating, and maintaining it.
- You simply use the software through the Internet.

SaaS use case

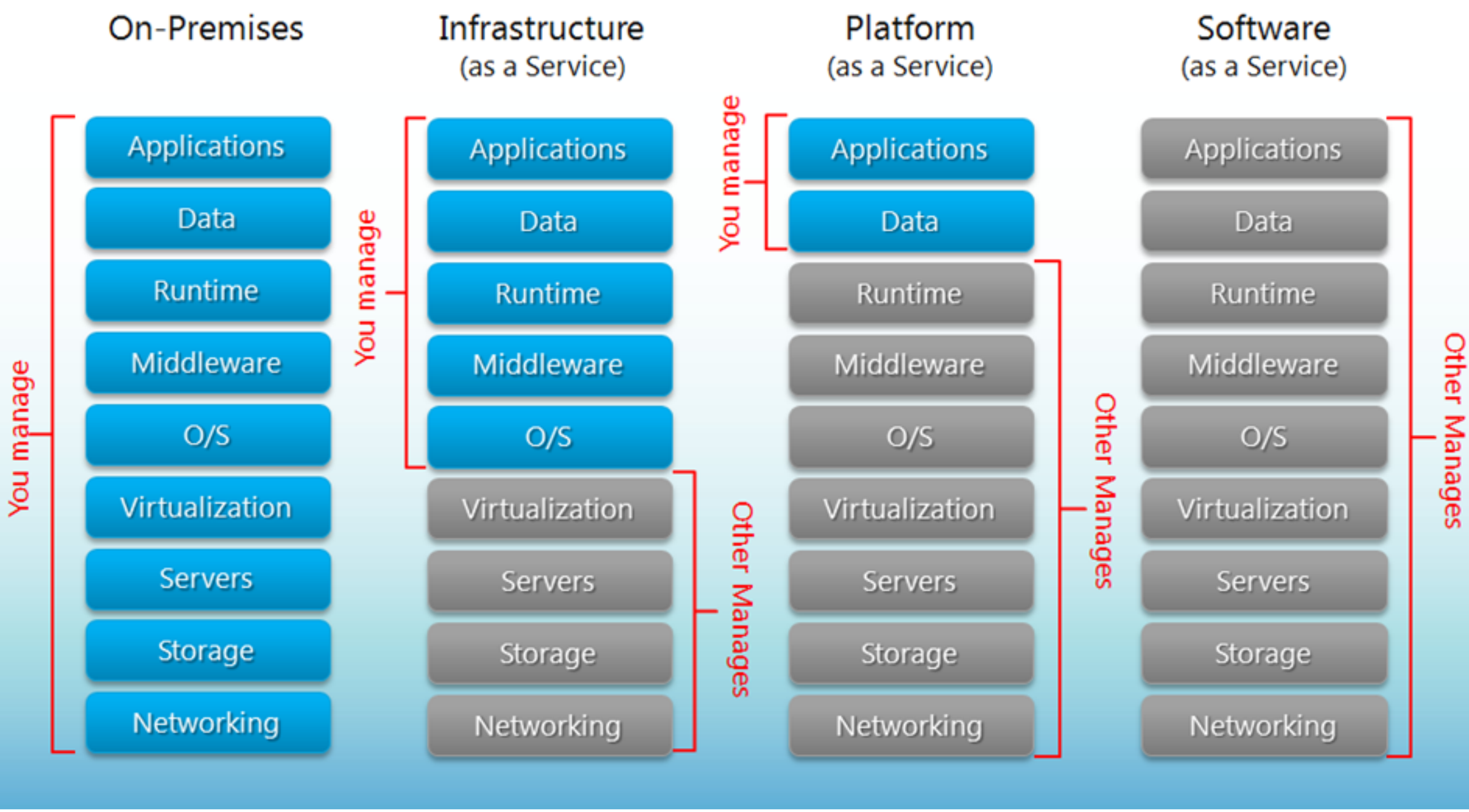
- HKUST uses Google Apps and Office 365 for student and staff email, calendar, etc.
- Don’t know how much they charge HKUST though…

ITSC INFORMATION TECHNOLOGY SERVICES CENTER

Gmail™ by Google

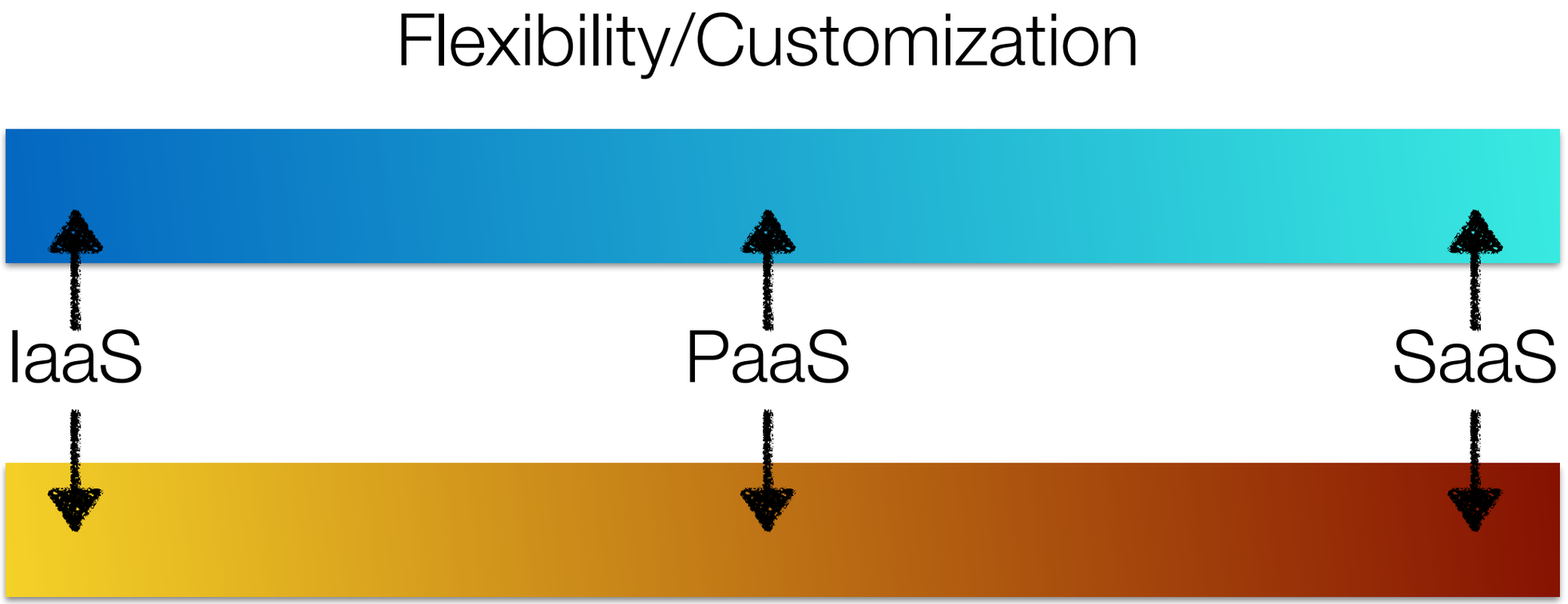
Office 365

Separation of Responsibilities



Source: K. Remde, “SaaS, PaaS, and IaaS.. Oh my!” TechNet Blog, 2011

A comparison



Convenience/Ease of management

Tradeoff between flexibility and “built-in” functionality

Other X-as-a-Service (XaaS)

Function-as-a-Service (FaaS)

- Users write applications in the form of "cloud functions"
- Users define the events that trigger the execution of those functions (e.g., HTTP requests, webhooks)
- Let the cloud platform to handle everything else, including resource provisioning, autoscaling, fault tolerance, etc.
- Users only pay for the CPU time used to run functions

Users manage no servers, hence termed "serverless computing"

Benefits of FaaS

- No server management
  - all handled by the cloud provider, not users
- Cost-effective
  - users only pay for the CPU time when functions are executed (no charge when code is not running)
- Flexible scaling
  - no need to set up autoscaling: it’s cloud provider’s problem
- Automated high availability and fault tolerance

IaaS vs. FaaS

- Configure an instance
- Update OS
- Install App platform
- Build and deploy App
- Configure autoscaling/load balancing
- Continuously secure and monitor instances
- Monitor and maintain apps

- Configure an instance
- Update OS
- Install App platform
- Build and deploy App
- Configure autoscaling/load balancing
- Continuously secure and monitor instances
- Monitor and maintain apps

Popular FaaS Platforms

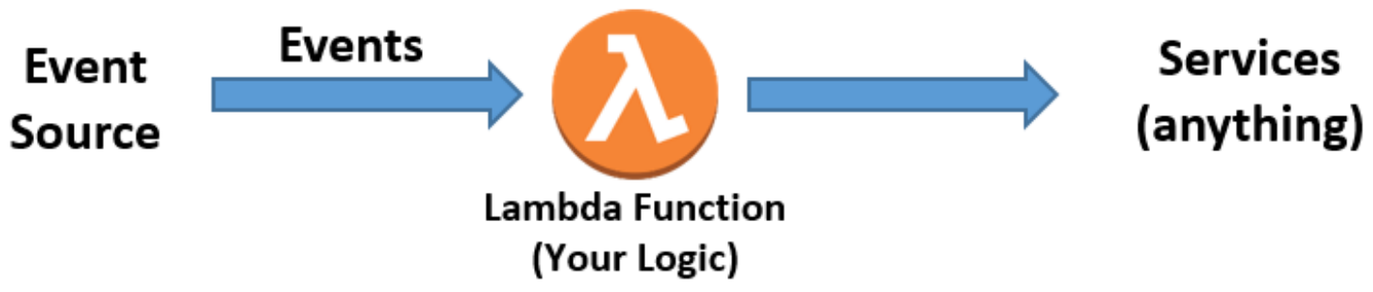
AWS Lambda

- Lets you run code without provisioning or managing servers
- Triggers on your behalf in response to events

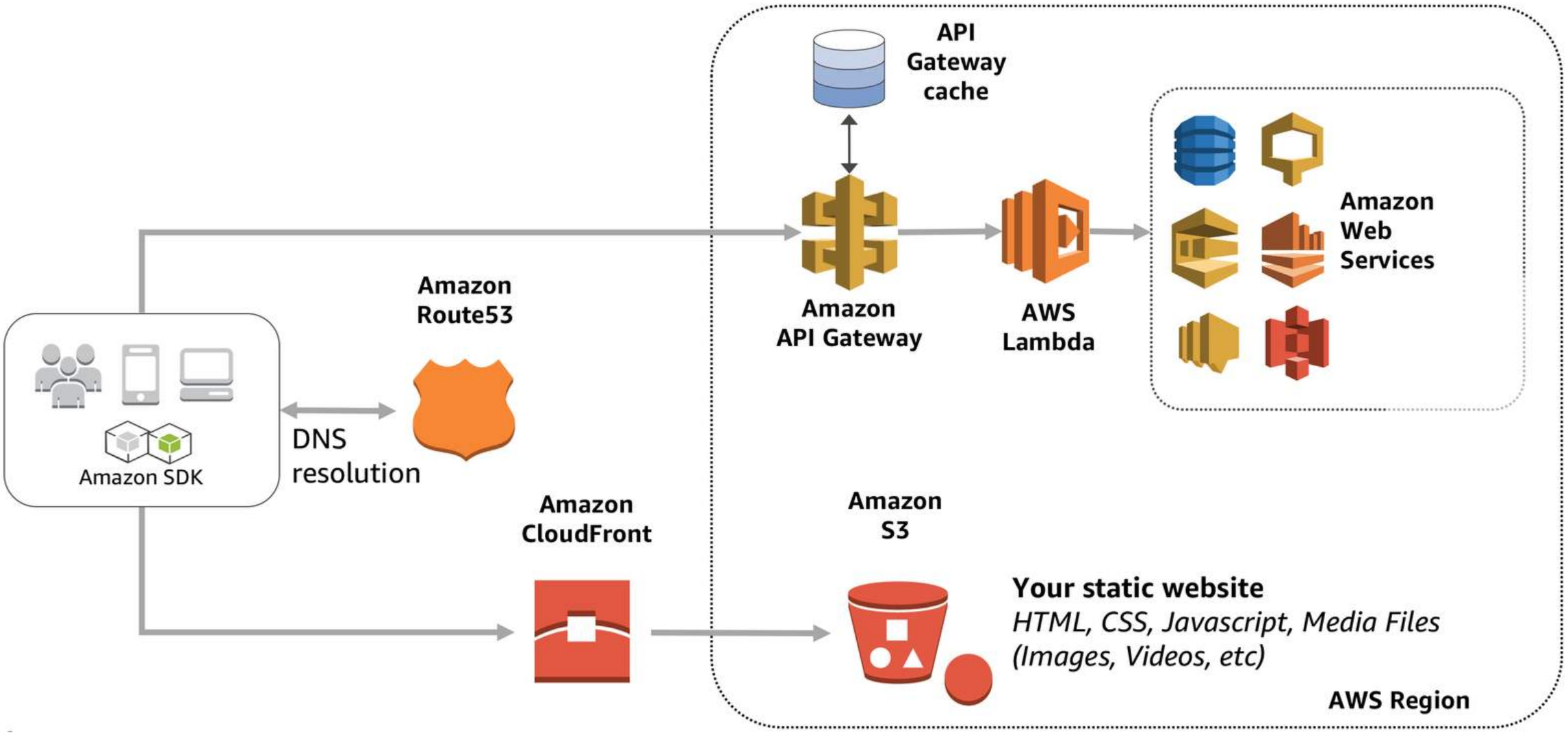
Google Cloud Functions

- Scales automatically
- Provides built-in code monitoring and logging via WebUI or CLI

Microsoft Azure



Example FaaS application



ML-as-a-Service

- An umbrella term for a set of cloud-based machine learning (ML) tools that cover most ML pipelines
  - e.g., data pre-processing, model training, model evaluation, and prediction serving
- Four key players in the MLaaS market
  - Amazon, Microsoft Azure, Google Cloud, IBM

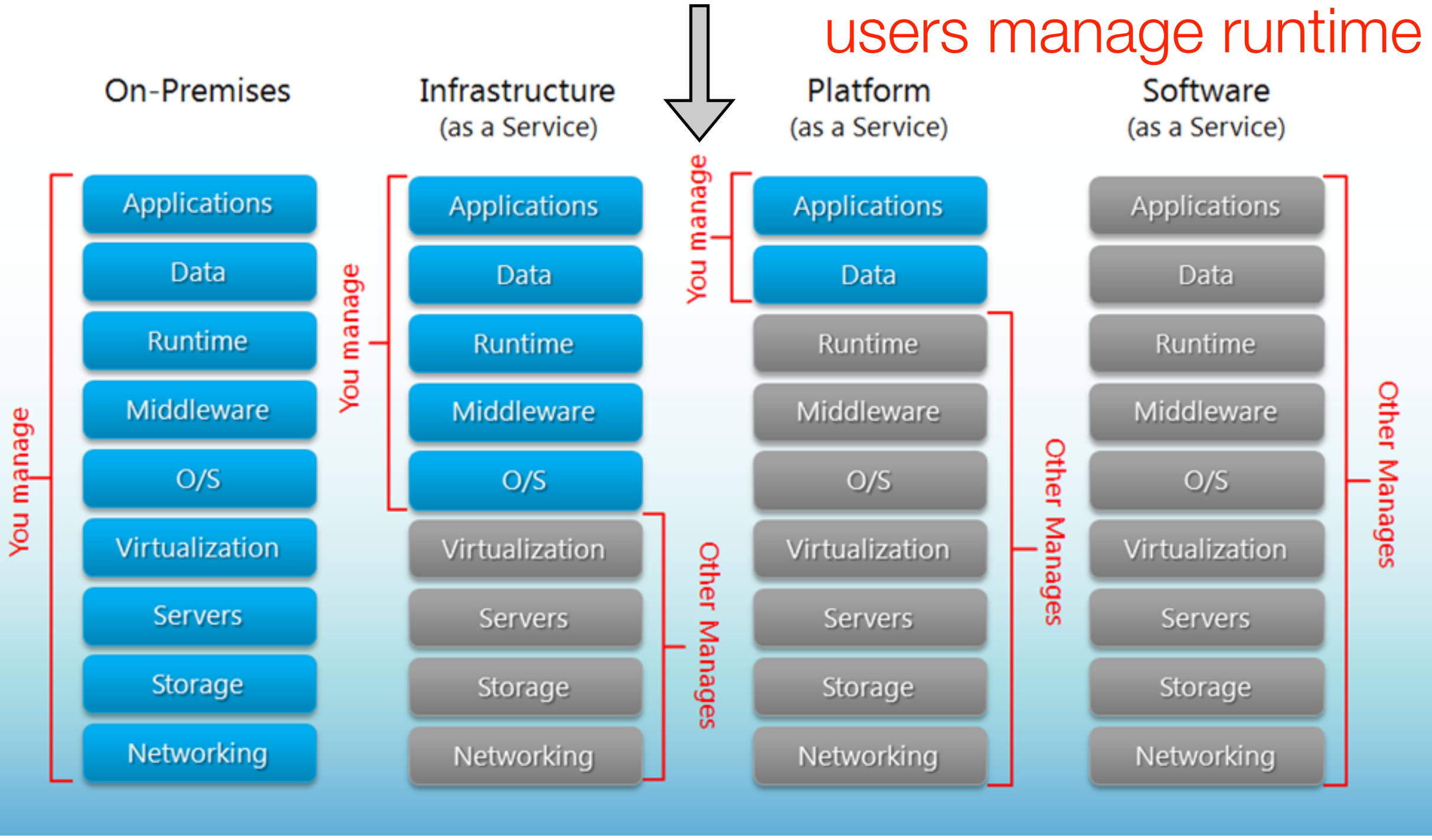
CLOUD MACHINE LEARNING SERVICES COMPARISON

	Amazon	Microsoft	Google	IBM
Automated and semi-automated ML services				
	Amazon ML	Microsoft Azure ML Studio	Cloud AutoML	IBM Watson ML Model Builder
Classification	✓	✓	✓	✓
Regression	✓	✓	✓	✓
Clustering	✓	✓	✗	✗
Anomaly detection	✗	✓	✗	✗
Recommendation	✗	✓	✓	✗
Ranking	✗	✓	✗	✗
Platforms for custom modeling				
	Amazon SageMaker	Azure ML Services	Google ML Engine	IBM Watson ML Studio
Built-in algorithms	✓	✗	✓	✓
Supported frameworks	TensorFlow, MXNet, Keras, Gluon, Pytorch, Caffe2, Chainer, Torch	TensorFlow, scikit-learn, Microsoft Cognitive Toolkit, Spark ML	TensorFlow, scikit-learn, XGBoost, Keras	TensorFlow, Spark MLlib, scikit-learn, XGBoost, PyTorch, IBM SPSS, PMML

altexsoft
software r&d engineering

FaaS & MLaaS are closer to PaaS than IaaS

users manage runtime



Source: K. Remde, “SaaS, PaaS, and IaaS.. Oh my!” TechNet Blog, 2011

We mainly focus on IaaS in this course, with some coverage of FaaS

Issues of Cloud

Issues of Cloud

- Availability
  - *AWS outage in August 2013, about an hour, takes down Vine, Instagram, Flipboard, etc.*
  - *Loss of sales: $1,100 USD per second*
- Data loss
- Vendor lock-in
  - Apps built on a cloud may not be transferred to another

Issues of Cloud

- Security:
  - Can an intruder/attacker get my data in the cloud?
  - *Twitter had a data breach due to an attack that exposed the usernames, email addresses, and encrypted passwords of 250,000 users in Feb. 2013.*

Issues of Cloud

- Privacy:
  - Will the provider look at my data in the cloud?
  - *Think about Google's targeted ads in your gmail*
  - Will the provider give my data to the government or other parties?
  - *Think about Mr. Snowden who fled and stayed at HK for a while*

Table 2. Top 10 obstacles to and opportunities for growth of cloud computing.

	Obstacle	Opportunity
1	Availability/Business Continuity	Use Multiple Cloud Providers
2	Data Lock-In	Standardize APIs; Compatible SW to enable Surge or Hybird Cloud Computing
3	Data Confidentiality and Auditability	Deploy Encryption, VLANs, Firewalls
4	Data Transfer Bottlenecks	FedExing Disks; Higher BW Switches
5	Performance Unpredictability	Improved VM Support; Flash Memory; Gang Schedule VMs
6	Scalable Storage	Invent Scalable Store
7	Bugs in Large Distributed Systems	Invent Debugger that relies on Distributed VMs
8	Scaling Quickly	Invent Auto-Scaler that relies on ML; Snapshots for Conservation
9	Reputation Fate Sharing	Offer reputation-guarding services like those for email
10	Software Licensing	Pay-for-use licenses

Challenges facing cloud providers

Storage

- Large dataset cannot fit into a local storage
- Persistent storage must be **distributed**
  - GFS, BigTable, HDFS, Cassandra, S3, etc.
- Local storage goes **volatile**
  - Cache for data being served
  - local logging and async copy to persistent storage

Scale

- Large cluster: able to host petabytes of data
- Extremely large cluster: at Google, the storage system pages a user *if there is only a few petabytes of spaces left available*!
- A 10k-node cluster is considered small- to medium-sized

Faults and failures

>1%	DRAM errors per year
2-10%	Annual failure rate of disk drive
2	# crashes per machine-year
2-6	# OS upgrades per machine-year
>1	Power utility events per year

Failure is a norm, not an exception!

- “A 2000-node cluster will have >10 machines crashing per day”

— Luiz Barroso

Source: J. Wilkes, “Cluster management at Google.”

Networking

- How can a cloud provide fast connections for hundreds of millions of clients coming from the entire globe to access their services?
- Inside a cloud, with hundreds of thousands of tenants, their apps, and servers, how to make sure the network is fast and robust enough to move bits from anywhere to anywhere?
- What about fairness of the bandwidth resources?

Machine heterogeneity

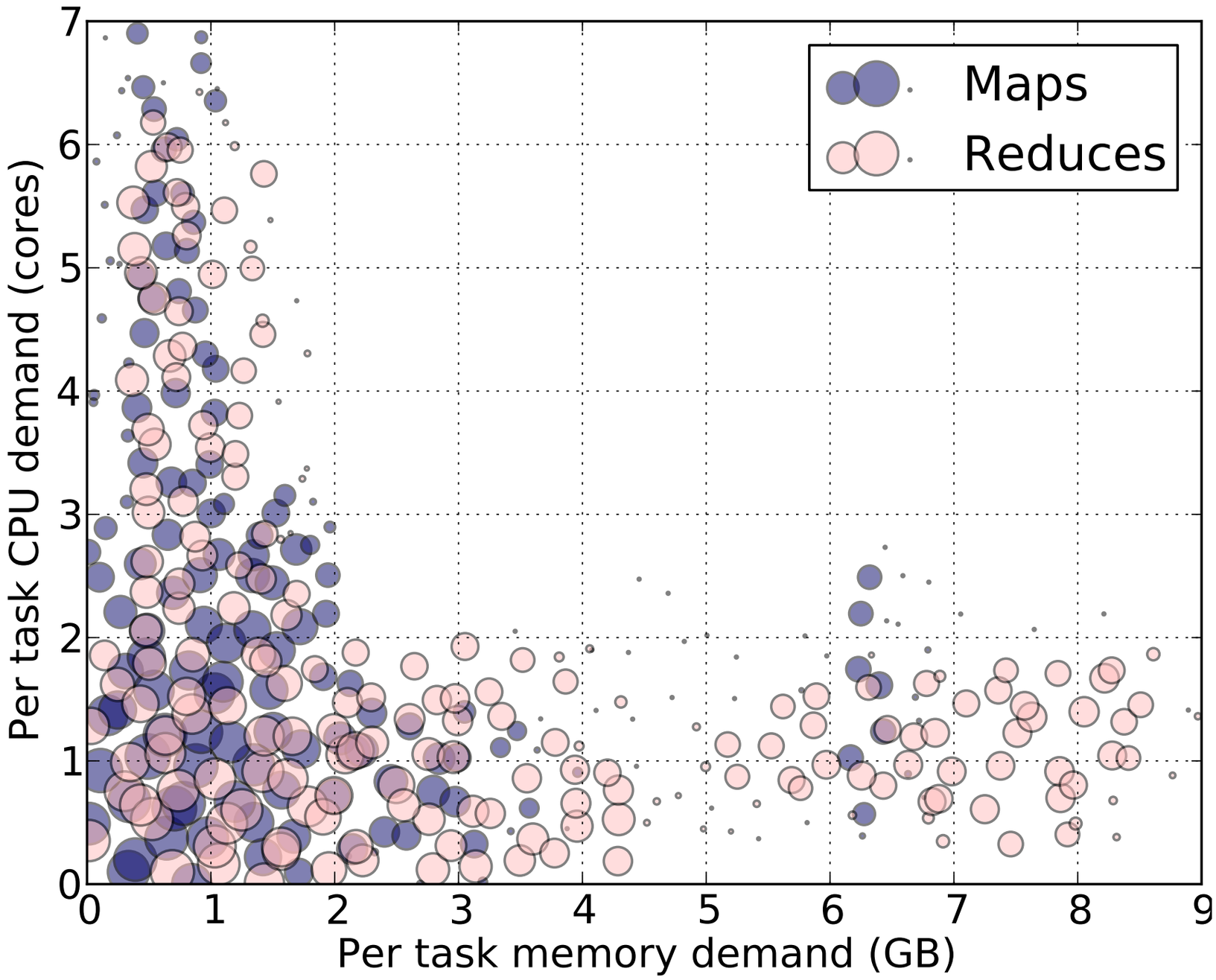
- Machines span multiple generations representing different points in the configuration space

System	#CPUs	Mem (GiB)	#GPUs	GPU type	#Nodes
PAI	64	512	2	P100	798
	96	512	2	T4	497
	96	512	8	Misc.	280
	96	384	8	V100M32†	135
	96	512/384	8	V100†	104
	96	512	0	N/A	83

Machine specs. of a GPU cluster in Alibaba Platform for AI (PAI)

Source: Q. Weng, “MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters,” in USENIX NSDI’22.

Workload heterogeneity



Source: A. Ghodsi et al., “Dominant resource fairness: fair allocation of multiple resource types,” USENIX NSDI’11.

Challenges due to heterogeneity

- Hard to provide predictable and consistent services
- Hard to monitor the system, identify the performance bottleneck, or reason about the stragglers
- Hard to achieve **fair sharing** among users

Nevertheless, we still want to achieve…

Objectives

- Able to run everything at scale
- Fault tolerance
- Predictable services
- High utilization
- Network with high bisection bandwidth

With the minimum human intervention!

Credit

- Some slides are adapted from Prof. Hong Xu's slides for CS 4296/5296 in CityU