TCSS 462/562: (SOFTWARE ENGINEERING FOR) CLOUD COMPUTING

Cloud Computing – *How did we get here? – part IV,*

Introduction to Cloud Computing

Wes J. Lloyd
School of Engineering and Technology
University of Washington – Tacoma

TR 5:50-7:50 PM

1

OBJECTIVES – 10/10

- Questions from 10/10
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.2

2

OFFICE HOURS – FALL 2024

- **Tuesdays:**
  - 2:30 to 3:30 pm - CP 229 and Zoom
- **Fridays**
  - 1:00 pm to 2:00 pm – ONLINE via Zoom*
- Or email for appointment

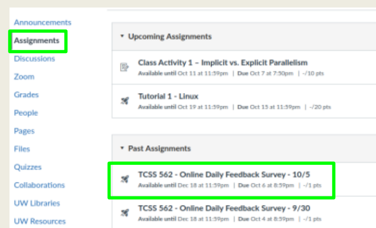
> *Office Hours set based on Student Demographics survey feedback*
> ** - Friday office hours may be adjusted due to faculty meeting conflicts or other obligations. The adjustments will be announced via Canvas.*

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.3

3

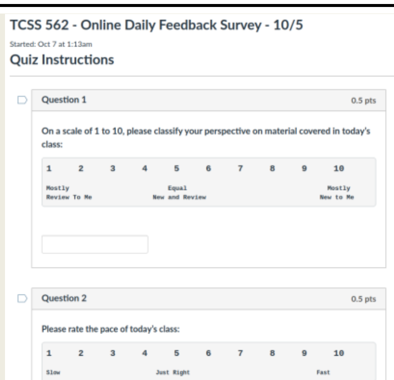
ONLINE DAILY FEEDBACK SURVEY

- Daily Feedback Quiz in Canvas – Take After Each Class
- Extra Credit for completing



October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.4

4



October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.5

5

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (46 respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.14 (↓ - *previous 6.27*)**

- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.32 (↓ - previous 5.40)**

- **Response rates:**
- TCSS 462: 32/42 – 76.2%
- TCSS 562: 14/20 – 70.0%

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.6

6

FEEDBACK FROM 10/10

- ***Performance bottlenecks are still unclear to me. If we are talking about a bottleneck existing within the CPU or the I/O?***
  - ***Does the I/O mean that the motherboard needs to be upgraded as the bus is integrated into the motherboard?***
- The bottleneck is the part of the program that limits performance.
- With the roofline model, the arithmetic intensity (compute intensity) of a program is below 100%, processing is limited by delays (I/O and memory communication).
- As compute intensity increases, performance hits a **ROOF**, where the system maxes out the CPU reaching peak output for the system.
  - Computer can't delivery more output/throughput without a CPU upgrade
  - When performance is limited by CPU, usually the I/O is not saturated
  - I/O limited (bound) performance: left of graph -OR- CPU limited (bound) performance: right of graph

7

AWS CLOUD CREDITS UPDATE

- Our AWS cloud credit distribution has started
- 22 requests have been fullfiled
- Credit codes are being "*securely*" exchanged
  - This can be in person (or by zoom), in the class during the breaks, after class, or during office hours
  - Credits can also be requested by email by sending an email with the subject "**AWS CREDIT REQUEST**" to **wlloyd@uw.edu**
    - Please use this exact subject so the email is not missed
    - Please see tutorial 0 for details

8

TUTORIAL 0

- Getting Started with AWS
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2024_tutorial_0.pdf
- Create an AWS account
- Create account credentials for working with the CLI
- Install awsconfig package
- Setup awsconfig for working with the AWS CLI

9

TUTORIAL 1 – DUE OCT 11

- **Introduction to Linux & the Command Line**
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2024_tutorial_1.pdf
- **Tutorial Sections:**
  1. The Command Line
  2. Basic Navigation
  3. More About Files
  4. Manual Pages
  5. File Manipulation
  6. VI – Text Editor
  7. Wildcards
  8. Permissions
  9. Filters
  10. Grep and regular expressions
  11. Piping and Redirection

10

TUTORIAL 2 – DUE OCT 19

- **Introduction to Bash Scripting**
- https://faculty.washington.edu/wlloyd/courses/tcss562/tutorials/TCSS462_562_f2024_tutorial_2.pdf
- Review tutorial sections:
- Create a BASH webservice client
  1. What is a BASH script?
  2. Variables
  3. Input
  4. Arithmetic
  5. If Statements
  6. Loops
  7. Functions
  8. User Interface
- Call service to obtain IP address & lat/long of computer
- Call weatherbit.io API to obtain weather forecast for lat/long

11

TUTORIAL 3 – TO BE POSTED

- Best Practices for Working with Virtual Machines on Amazon EC2
- To be posted
- Creating a spot VM
- Creating an image from a running VM
- Persistent spot request
- Stopping (pausing) VMs
- EBS volume types
- Ephemeral disks (local disks)
- Mounting and formatting a disk
- Disk performance testing with Bonnie++
- Cost Saving Best Practices

12

CATCH UP FROM – 10/8

- Questions from 10/3
- Tutorial 0, Tutorial 1, Tutorial 2
- Term Project Proposal
- Cloud Computing – How did we get here? - part III (Marinescu Ch. 2 - 1st edition, Ch. 4 - 2nd edition)
- **Graphics processing units**
- Speed-up, Amdahl's Law, Scaled Speedup
- Properties of distributed systems
- Modularity
- Introduction to Cloud Computing – loosely based on book #1: Cloud Computing Concepts, Technology & Architecture

13

GRAPHICAL PROCESSING UNITS (GPUs)

- GPU provides multiple SIMD processors
- Typically 7 to 15 SIMD processors each
- 32,768 total registers, divided into 16 lanes (2048 registers each)
- GPU programming model: single instruction, multiple thread
- Programmed using CUDA- C like programming language by NVIDIA for GPUs
- CUDA threads – single thread associated with each data element (e.g. vector or matrix)
- Thousands of threads run concurrently

14

CATCH UP FROM – 10/8

- Questions from 10/3
- Tutorial 0, Tutorial 1, Tutorial 2
- Term Project Proposal
- Cloud Computing – How did we get here? - part III (Marinescu Ch. 2 - 1st edition, Ch. 4 - 2nd edition)
- Graphics processing units
- **Speed-up, Amdahl's Law, Scaled Speedup**
- Properties of distributed systems
- Modularity
- Introduction to Cloud Computing – loosely based on book #1: Cloud Computing Concepts, Technology & Architecture

15

PARALLEL COMPUTING

- Parallel hardware and software systems allow:
  - Solve problems demanding resources not available on single system.
  - Reduce time required to obtain solution
- The *speed-up* (S) measures effectiveness of parallelization:

S(N) = T(1) / T(N)

T(1) → execution time of total sequential computation
T(N) → execution time for performing N parallel computations in parallel

16

SPEED-UP EXAMPLE

- Consider embarrassingly parallel image processing
- Eight images (multiple data)
- Apply image transformation (greyscale) in parallel
- 8-core CPU, 16 hyperthreads
- Sequential processing: perform transformations one at a time using a single program thread
  - 8 images, 3 seconds each: `T(1) = 24 seconds`
- Parallel processing
  - 8 images, 3 seconds each: `T(N) = 3 seconds`
- Speedup: `S(N) = 24 / 3 = 8x speedup`
- Called "perfect scaling"
- Must consider data transfer and computation setup time

17

AMDAHL'S LAW

- Amdahl's law is used to estimate the speed-up of a job using parallel computing

1. Divide job into two parts
2. Part A that will still be sequential
3. Part B that will be sped-up with parallel computing

- Portion of computation which cannot be parallelized will determine (i.e. limit) the overall speedup
- Amdahl's law assumes jobs are of a fixed size
- Also, Amdahl's assumes no overhead for distributing the work, and a perfectly even work distribution

18

AMDAHL'S LAW

Speed-up formula →

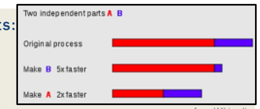
$$S = \frac{1}{(1-f) + \frac{f}{N}}$$

- S = theoretical speedup of the whole task
- f= fraction of work that is parallel (ex. 25% or 0.25)
- N= proposed speed up of the parallel part (ex. 5 times speedup)
- % improvement of task execution = 100 * (1 – (1 / S))
- **Using Amdahl's law, we can find the maximum possible speed-up (S) for a given scenario (e.g. ~8x) ...**

19

AMDAHL'S LAW EXAMPLE

- Program with two independent parts:
  - Part A is 75% of the execution time
  - Part B is 25% of the execution time
- Part B is made 5 times faster with parallel computing (N=5)
- Estimate the percent improvement of task execution
- Original Part A is 3 seconds, Part B is 1 second



from Wikipedia

- N=5 (speedup of part B)
- f=.25 (only 25% of the whole job (A+B) will be sped-up)
- S=1 / ((1-f) + f/N)
- S=1 / ((.75) + .25/5)
- S=1.25 (speed up is 1.25x faster)
- % improvement = 100 * (1 – 1/1.25) = **20%**

20

GUSTAFSON'S LAW

- Calculates the ***scaled speed-up*** using "N" processors
  S(N) = N + (1 - N) α

N: Number of processors

α: fraction of program run time which can't be parallelized (e.g. must run sequentially)

- *Can be used to estimate runtime of parallel portion of program*

21

GUSTAFSON'S LAW

- Calculates the ***scaled speed-up*** using "N" processors
  S(N) = N + (1 - N) α

N: Number of processors

α: fraction of program run time which can't be parallelized (e.g. must run sequentially)

- *Can be used to estimate runtime of parallel portion of program*
- Where α = σ / (π + σ)
- Where σ= sequential time, π =parallel time
- Our Amdahl's example: σ= 3s, π =1s, α =.75

22

GUSTAFSON'S LAW

- Calculates the ***scaled speed-up*** using "N" processors
  S(N) = N + (1 - N) α

N: Number of processors

α: fraction of program run time which can't be parallelized (e.g. must run sequentially)

- Example:
  Consider a program that is embarrassingly parallel, but 75% cannot be parallelized. α=.75
  QUESTION: ***If deploying the job on a 2-core CPU, what scaled speedup is possible assuming the use of two processes that run in parallel?***

23

GUSTAFSON'S EXAMPLE

- **QUESTION:**
  What is the maximum theoretical speed-up on a **2-core CPU** ?
  S(N) = N + (1 - N) α
  N=2, α=.75
  S(N) = 2 + (1 - 2) .75
  S(N) = ?
- What is the maximum theoretical speed-up on a 16-**core CPU**?
  S(N) = N + (1 - N) α
  N=16, α=.75
  S(N) = 16 + (1 - 16) .75
  S(N) = ?

24

GUSTAFSON'S EXAMPLE

- **QUESTION:**
  What is the maximum theoretical speed-up on a **2-core CPU** ?
  S(N) = N + (1 - N) α
  N=2, α=
  S(N) =
  S(N) = ?
- What is the maximum theoretical speed-up on a **16-core CPU**?
  S(N) = N + (1 - N) α
  N=16, α=.75
  S(N) = 16 + (1 - 16) .75
  S(N) = ?

For 2 CPUs, speed up is 1.25x

For 16 CPUs, speed up is 4.75x

October 8, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L4.25

25

MOORE'S LAW

- Transistors on a chip doubles approximately every 1.5 years
- CPUs nov
- Power di ... to heat removal
  - Transiti ... cores
- ***Symmetric core processor*** –multi-core CPU, all cores have the same computational resources and speed
- ***Asymmetric core processor*** – on a multi-core CPU, some cores have more resources and speed
- ***Dynamic core processor*** – processing resources and speed can be dynamically configured among cores
- ***Observation: asymmetric processors offer a higher speedup***

What kind of processor are modern Intel CPUs ?

October 8, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L4.26

26

OBJECTIVES – 10/10

- **Questions from 10/10**
- **Properties of Distributed Systems** Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.27

27

DISTRIBUTED SYSTEMS

- Collection of autonomous computers, connected through a network with distribution software called "middleware" that enables coordination of activities and sharing of resources
- **Key characteristics:**
- Users perceive system as a single, integrated computing facility.
- Compute nodes are autonomous
- Scheduling, resource management, and security implemented by every node
- Multiple points of control and failure
- Nodes may not be accessible at all times
- System can be scaled by adding additional nodes
- Availability at low levels of HW/software/network reliability

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.28

28

DISTRIBUTED SYSTEMS - 2

- Key non-functional attributes
  - Known as "ilities" in software engineering
- Availability – 24/7 access?
- Reliability - Fault tolerance
- Accessibility – reachable?
- Usability – user friendly
- Understandability – can under
- Scalability – responds to variable demand
- Extensibility – can be easily modified, extended
- Maintainability – can be easily fixed
- Consistency – data is replicated correctly in timely manner

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.29

29

TRANSPARENCY PROPERTIES OF DISTRIBUTED SYSTEMS

- **Access transparency**: local and remote objects accessed using identical operations
- **Location transparency**: objects accessed w/o knowledge of their location.
- **Concurrency transparency**: several processes run concurrently using shared objects w/o interference among them
- **Replication transparency**: multiple instances of objects are used to increase reliability
  - users are unaware if and how the system is replicated
- **Failure transparency**: concealment of faults
- **Migration transparency**: objects are moved w/o affecting operations performed on them
- **Performance transparency**: system can be reconfigured based on load and quality of service requirements
- **Scaling transparency**: system and applications can scale w/o change in system structure and w/o affecting applications

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.30

30

OBJECTIVES – 10/10

- **Questions from 10/10**
- Properties of Distributed Systems, **Modularity**
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

31

TYPES OF MODULARITY

- ***Soft modularity:*** TRADITIONAL
- Divide a program into modules (classes) that call each other and communicate with shared-memory
- A procedure calling convention is used (or method invocation)
- ***Enforced modularity:*** CLOUD COMPUTING
- Program is divided into modules that communicate only through message passing
- The ubiquitous client-server paradigm
- Clients and servers are independent decoupled modules
- System is more robust if servers are stateless
- May be scaled and deployed separately
- May also FAIL separately!

32

CLOUD COMPUTING – HOW DID WE GET HERE? SUMMARY OF KEY POINTS

- Multi-core CPU technology and hyper-threading
- What is a
  - Heterogeneous system?
  - Homogeneous system?
  - Autonomous or self-organizing system?
- **Fine grained vs. coarse grained parallelism**
- Parallel message passing code is easier to debug than shared memory (e.g. p-threads)
- Know your application's max/avg **Thread Level Parallelism** (***TLP***)
- **Data-level parallelism**: Map-Reduce, (SIMD) Single Instruction Multiple Data, Vector processing & GPUs

33

CLOUD COMPUTING – HOW DID WE GET HERE? SUMMARY OF KEY POINTS - 2

- **Bit-level parallelism**
- **Instruction-level parallelism** (CPU pipelining)
- **Flynn's taxonomy**: computer system architecture classification
  - **SISD** – Single Instruction, Single Data (modern core of a CPU)
  - **SIMD** – Single Instruction, Multiple Data (Data parallelism)
  - **MIMD** – Multiple Instruction, Multiple Data
  - MISD is RARE; application for fault tolerance…
- **Arithmetic intensity**: ratio of calculations vs memory RW
- **Roofline model:** Memory bottleneck with low arithmetic intensity
- **GPUs**: ideal for programs with high arithmetic intensity
  - SIMD and Vector processing supported by many large registers

34

CLOUD COMPUTING – HOW DID WE GET HERE? SUMMARY OF KEY POINTS - 3

- **Speed-up (S)**
  S(N) = T(1) / T(N)
- **Amdahl's law:**
  S=1 / ((1-f) + f/N)
  f= fraction of work that is parallel (e.g. 0.25)
  N= proposed speed up of the parallel part (e.g. 5x)
- **Gustafson's Scaled speedup with N processes:**
  S(N) = N + (1 - N) α
  N: Number of processors
  α: fraction of program run time which can't be parallelized
- Moore's Law
- Symmetric core, Asymmetric core, Dynamic core CPU
- Distributed Systems Non-function quality attributes
- Distributed Systems – Types of Transparency
- Types of modularity- Soft, Enforced

35

INTRODUCTION TO CLOUD COMPUTING

36

OBJECTIVES – 10/10

- Questions from 10/10
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.37

37

OBJECTIVES – 10/10

- Questions from 10/10
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.38

38

WHY STUDY CLOUD COMPUTING?

- LINKEDIN - TOP IT Skills from job app data
  - #1 Cloud and Distributed Computing
  - https://learning.linkedin.com/week-of-learning/top-skills
  - #2 Statistical Analysis and Data Mining
- FORBES Survey – 6 Tech Skills That'll Help You Earn More
  - #1 Data Science
  - #2 Cloud and Distributed Computing
  - http://www.forbes.com/sites/laurencebradford/2016/12/19/6-tech-skills-thatll-help-you-earn-more-in-2017/

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.39

39

WHY STUDY CLOUD COMPUTING? - 2

- Computerworld Magazine

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.40

40

OBJECTIVES – 10/10

- Questions from 10/10
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.41

41

A BRIEF HISTORY OF CLOUD COMPUTING

- John McCarthy, 1961
  - Turing award winner for contributions to AI
- "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility… The computer utility could become the basis of a new and important industry…"

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.42

42

CLOUD HISTORY - 2

- Internet based computer utilities
- Since the mid-1990s
- Search engines: Yahoo!, Google, Bing
- Email: Hotmail, Gmail

- 2000s
- Social networking platforms: MySpace, Facebook, LinkedIn
- Social media: Twitter, YouTube

- Popularized core concepts
- Formed basis of cloud computing

43

CLOUD HISTORY: SERVICES - 1

- Late 1990s – Early Software-as-a-Service (SaaS)
  - Salesforce: Remotely provisioned services for the enterprise

- 2002 -
  - Amazon Web Services (AWS) platform: Enterprise oriented services for remotely provisioned storage, computing resources, and business functionality

- 2006 – Infrastructure-as-a-Service (IaaS)
  - Amazon launches Elastic Compute Cloud (EC2) service
  - Organization can “lease” computing capacity and processing power to host enterprise applications
  - Infrastructure

44

CLOUD HISTORY: SERVICES - 2

- 2006 – Software-as-a-Service (SaaS)
  - Google: Offers Google DOCS, “MS Office” like fully-web based application for online documentation creation and collaboration

- 2009 – Platform-as-a-Service (PaaS)
  - Google: Offers Google App Engine, publicly hosted platform for hosting scalable web applications on google-hosted datacenters

45

CLOUD COMPUTING NIST GENERAL DEFINITION

“Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (networks, servers, storage, applications and services) that can be rapidly provisioned and reused with minimal management effort or service provider interaction”…

46

MORE CONCISE DEFINITION

“Cloud computing is a specialized form of distributed computing that introduces utilization models for remotely provisioning scalable and measured resources.”

From Cloud Computing Concepts, Technology, and Architecture
Z. Mahmood, R. Puttini, Prentice Hall, 5th printing, 2015

47

OBJECTIVES – 10/10

- **Questions from 10/10**
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

48

BUSINESS DRIVERS FOR CLOUD COMPUTING

- Capacity planning
- Cost reduction
- Operational overhead
- Organizational agility

49

BUSINESS DRIVERS FOR CLOUD COMPUTING

- Capacity planning
  - Process of determining and fulfilling future demand for IT resources
  - Capacity vs. demand
  - Discrepancy between capacity of IT resources and actual demand
  - Over-provisioning: resource capacity exceeds demand
  - Under-provisioning: demand exceeds resource capacity
  - Capacity planning aims to minimize the discrepancy of available resources vs. demand

50

Dwight, The Office TV sitcom

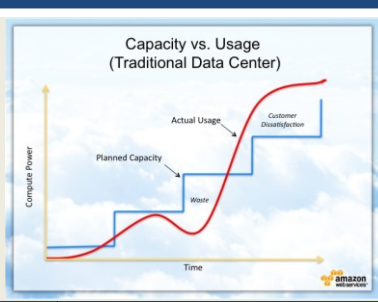
51

BUSINESS DRIVERS FOR CLOUD - 2

- Capacity planning
  - Over-provisioning: is costly due to too much infrastructure
  - Under-provisioning: is costly due to potential for business loss from poor quality of service
- Capacity planning strategies
  - <u>Lead strategy:</u> add capacity in anticipation of demand (pre-provisioning)
  - <u>Lag strategy:</u> add capacity when capacity is fully leveraged
  - <u>Match strategy:</u> add capacity in small increments as demand increases
- Load prediction
  - Capacity planning helps anticipate demand flucations

52

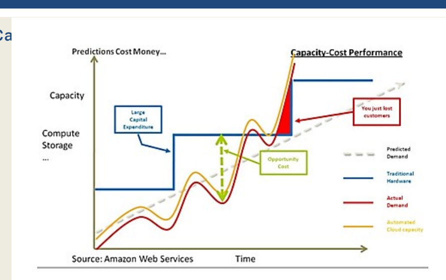
CAPACITY PLANNING



53

CAPACITY PLANNING - 2

- Ca



54

BUSINESS DRIVERS FOR CLOUD - 3

- Cost reduction
  - IT Infrastructure acquisition
  - IT Infrastructure maintenance
- Operational overhead
  - Technical personnel to maintain physical IT infrastructure
  - System upgrades, patches that add testing to deployment cycles
  - Utility bills, capital investments for power and cooling
  - Security and access control measures for server rooms
  - Admin and accounting staff to track licenses, support agreements, purchases

55

BUSINESS DRIVERS FOR CLOUD - 4

- Organizational agility
  - Ability to adapt and evolve infrastructure to face change from internal and external business factors
  - Funding constraints can lead to insufficient on premise IT
  - Cloud computing enables IT resources to scale with a lower financial commitment

56

OBJECTIVES – 10/10

- **Questions from 10/10**
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

57

TECHNOLOGY INNOVATIONS LEADING TO CLOUD

- Cluster computing
- Grid computing
- Virtualization
- Others

58

CLUSTER COMPUTING

- Cluster computing (clustering)
  - Cluster is a group of independent IT resources interconnected as a single system
  - Servers configured with homogeneous hardware and software
    - Identical or similar RAM, CPU, HDDs
  - Design emphasizes redundancy as server components are easily interchanged to keep overall system running
    - Example: if a RAID card fails on a key server, the card can be swapped from another redundant server
  - Enables warm replica servers
    - Duplication of key infrastructure servers to provide HW failover to ensure high availability (HA)

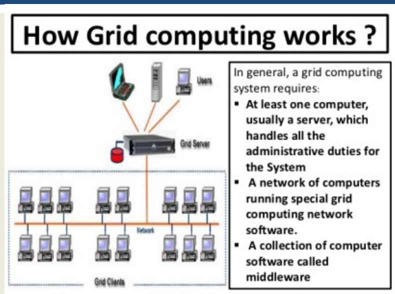
59

GRID COMPUTING

- On going research area since early 1990s
- Distributed heterogeneous computing resources organized into logical pools of loosely coupled resources
- For example: heterogeneous servers connected by the internet
- Resources are heterogeneous and geographically dispersed
- Grids use middleware software layer to support workload distribution and coordination functions
- Aspects: load balancing, failover control, autonomic configuration management
- Grids have influenced clouds contributing common features: networked access to machines, resource pooling, scalability, and resiliency

60

GRID COMPUTING - 2



October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.61

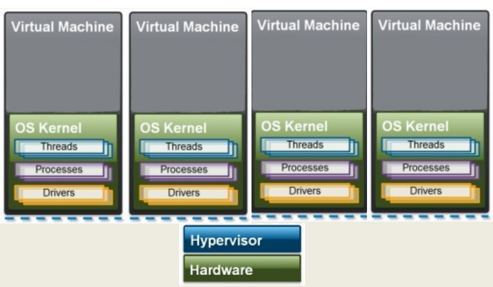
61

VIRTUALIZATION

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.62

62

VIRTUALIZATION



October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.63

63

VIRTUALIZATION

- Simulate physical hardware resources via software
  - The virtual machine (virtual computer)
  - Virtual local area network (VLAN)
  - Virtual hard disk
  - Virtual network attached storage array (NAS)
- Early incarnations featured significant performance, reliability, and scalability challenges
- CPU and other HW enhancements have minimized performance GAPs

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.64

64

OBJECTIVES – 10/10

- **Questions from 10/10**
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.65

65

KEY TERMINOLOGY

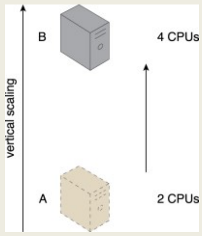
- **On-Premise Infrastructure**
  - Local server infrastructure not configured as a cloud
- **Cloud Provider**
  - Corporation or private organization responsible for maintaining cloud
- **Cloud Consumer**
  - User of cloud services
- **Scaling**
  - **Vertical scaling**
    - Scale up: increase resources of a single virtual server
    - Scale down: decrease resources of a single virtual server
  - **Horizontal scaling**
    - Scale out: increase number of virtual servers
    - Scale in: decrease number of virtual servers

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.66

66

VERTICAL SCALING

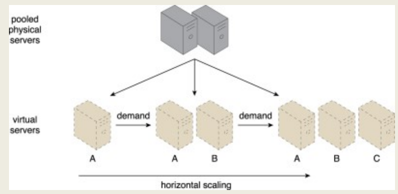
- Reconfigure virtual machine to have different resources:
  - CPU cores
  - RAM
  - HDD/SDD capacity
- May require VM migration if physical host machine resources are exceeded



67

HORIZONTAL SCALING

- Increase (scale-out) or decrease (scale-in) number of virtual servers based on demand



68

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers

69

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
IT resources instantly available	IT resources typically instantly available

70

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
IT resources instantly available	IT resources typically instantly available
Resource replication and automated scaling	Additional setup is normally needed

71

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
IT resources instantly available	IT resources typically instantly available
Resource replication and automated scaling	Additional setup is normally needed
Additional servers required	No additional servers required

72

HORIZONTAL VS VERTICAL SCALING

Horizontal Scaling	Vertical Scaling
Less expensive using commodity HW	Requires expensive high capacity servers
IT resources instantly available	IT resources typically instantly available
Resource replication and automated scaling	Additional setup is normally needed
Additional servers required	No additional servers required
Not limited by individual server capacity	Limited by individual server capacity

73

KEY TERMINOLOGY - 2

- Cloud services
  - Broad array of resources accessible "as-a-service"
  - Categorized as Infrastructure (IaaS), Platform (PaaS), Software (SaaS)
- Service-level-agreements (SLAs):
  - Establish expectations for: uptime, security, availability, reliability, and performance

74

OBJECTIVES – 10/10

- **Questions from 10/10**
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

75

GOALS AND BENEFITS

- **Cloud providers**
  - Leverage economies of scale through mass-acquisition and management of large-scale IT resources
  - Locate datacenters to optimize costs where electricity is low
- **Cloud consumers**
  - Key business/accounting difference:
  - **Cloud computing enables anticipated capital expenditures to be replaced with operational expenditures**
  - Operational expenditures always scale with the business
  - Eliminates need to invest in server infrastructure based on anticipated business needs
  - Businesses become more agile and lower their financial risks by eliminating large capital investments in physical infrastructure

76

CLOUD BENEFITS - 2

- On demand access to pay-as-you-go resources on a short-term basis (less commitment)
- Ability to acquire "unlimited" computing resources on demand when required for business needs
- Ability to add/remove IT resources at a fine-grained level
- Abstraction of server infrastructure so applications deployments are not dependent on specific locations, hardware, etc.
  - The cloud has made our software deployments more agile...

77

CLOUD BENEFITS - 3

- Example: Using 100 servers for 1 hour costs the same as using 1 server for 100 hours
- Rosetta Protein Folding: Working with a UW-Tacoma graduate student, we recently deployed this science model across 5,900 compute cores on Amazon for 2-days...
- ***What is the cost to purchase 5,900 compute cores?***
- Recent Dell Server purchase example: 20 cores on 2 servers for $4,478...
- Using this ratio 5,900 cores costs $1.3 million (purchase only)

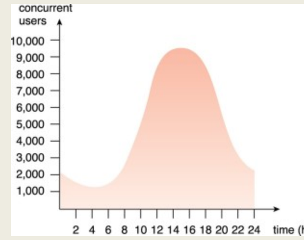
78

Gene Wilder, Charlie and the Chocolate Factory

79

CLOUD BENEFITS

- Increased scalability
  - Example demand over a 24-hour day →
- Increased availability
- Increased reliability



October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.80

80

OBJECTIVES – 10/10

- Questions from 10/10
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.81

81

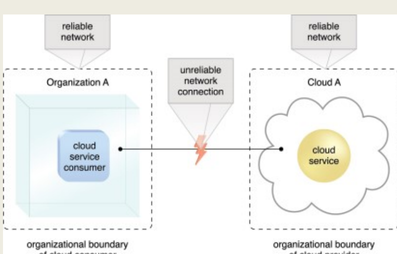
CLOUD ADOPTION RISKS

- Increased security vulnerabilities
  - Expansion of trust boundaries now include the external cloud
  - Security responsibility shared with cloud provider
- Reduced operational governance / control
  - Users have less control of physical hardware
  - Cloud user does not directly control resources to ensure quality-of-service
  - Infrastructure management is abstracted
  - Quality and stability of resources can vary
  - Network latency costs and variability

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.82

82

NETWORK LATENCY COSTS



October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.83

83

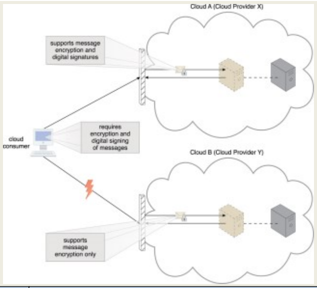
CLOUD RISKS - 2

- Performance monitoring of cloud applications
  - Cloud metrics (AWS cloudwatch) support monitoring cloud infrastructure (network load, CPU utilization, I/O)
  - Performance of cloud applications depends on the health of aggregated cloud resources working together
  - User must monitor this aggregate performance
- Limited portability among clouds
  - Early cloud systems have significant “vendor” lock-in
  - Common APIs and deployment models are slow to evolve
  - Operating system containers help make applications more portable, but containers still must be deployed
- Geographical issues
  - Abstraction of cloud location leads to legal challenges with respect to laws for data privacy and storage

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.84

84

CLOUD: VENDOR LOCK-IN



85

OBJECTIVES – 10/10

- Questions from 10/10
- Properties of Distributed Systems, Modularity
- Introduction to Cloud Computing –based on book #1: Cloud Computing Concepts, Technology & Architecture
  - Why study cloud computing?
  - History of cloud computing
  - Business drivers
  - Cloud enabling technologies
  - Terminology
  - Benefits of cloud adoption
  - Risks of cloud adoption
- Background on AWS Lambda for the Term Project

86

TCSS 462/562 TERM PROJECT

87

SERVERLESS – KEY CONCEPTS

- **Function-as-a-Service (FaaS) platform**
  - A platform where developers deploy "functions" written in common languages (e.g. Java, Python, Go, Node.js) that run as microservices
  - AWS Lambda is a FaaS platform
    - We will discuss platform limitations
- **Function instances**
  - This is an instantiation of a running function
  - A function instance is created when a client (user) calls the serverless function
  - Each concurrent (parallel) call to AWS Lambda to the same function will create a unique function instance to handle the request
  - The default maximum number of concurrently running function instances in your account is 10.
  - The default was originally 1,000 when the platform was introduced, and was dropped to 100, then 50, and is now just 10 in response to the growing popularity of AWS Lambda (they are running out of servers??)
  - You will want to request an increase in your AWS account's default concurrency. A minimum of 100 is recommended

88

AWS LAMBDA

- Lambda functions can be invoked by creating an HTTP REST endpoint that responds to HTTP POST requests
- A json object is provided as a request object to the function
- In the function code, the request object can be accessed to interpret how the user parameterized the function call
- The function generates a JSON response object
- AWS Lambda is introduced in detail in Tutorial 4

89

TYPES OF FUNCTION CALLS: SYNCHRONOUS

- **Serverless Computing:**
- AWS Lambda supports synchronous and asynchronous function calls
- Clients typically orchestrate synchronous calls and pipelines
- Asynchronous calls are often made via events

- **Synchronous web service:**
- Client calls service
- Client blocks (freezes) and waits for server to complete call
- Connection is maintained in the "OPEN" state
- Problematic if service runtime is long!
  - Connections are notoriously dropped
  - System timeouts reached
- Client can't do anything while waiting unless using threads

90

TYPES OF FUNCTION CALLS: ASYNCHRONOUS

- Asynchronous web service
- Client calls service
- Server responds to client with OK message
- Client closes connection
- Server performs the work associated with the service
- Server posts service result in an external data store
  - AWS: S3, SQS (queueing service), SNS (notification service)

91

AWS LAMBDA PLATFORM LIMITATIONS

- Maximum 10 GB memory per function instance
- Maximum 15-minutes execution per function instance
- 500 MB of /tmp disk space for local I/O (default)
- Up to 10 GB /tmp ephemeral storage (for additional charge)
  - https://aws.amazon.com/blogs/aws/aws-lambda-now-supports-up-to-10-gb-ephemeral-storage/
- Access up to 6 vCPUs depending on memory reservation size

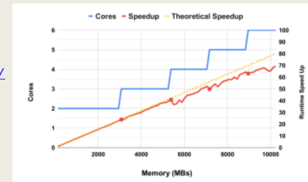


Figure 1: AWS Lambda Performance Speedup for Sysbench Prime Number Generation vs. Function Memory

92

AWS LAMBDA PLATFORM LIMITATIONS - 2

- 10 concurrent function executions inside account (default)
- Function payload: 6MB (synchronous), 256KB (asynchronous)
- Deployment package: 50MB (compressed), 250MB (unzipped)
- Container image size: 10 GB
- Processes/threads: 1024
- File descriptors: 1024
- Function instances run Amazon Linux 2
  - Pending upgrade to Amazon Linux 2023 ?
- See: https://docs.aws.amazon.com/lambda/latest/dg/gettingstarted-limits.html

93

CPUSTEAL

- *CpuSteal*: Metric that measures when a CPU core is ready to execute but the physical CPU core is busy and unavailable
- Symptom of over provisioning physical servers in the cloud
- Factors which cause *CpuSteal*: (x86 hyperthreading)
  1. Physical CPU is shared by too many busy VMs
  2. Hypervisor kernel is using the CPU
     - On AWS Lambda this would be the Firecracker MicroVM which is derived from the KVM hypervisor
  3. VM's CPU time share <100% for 1 or more cores, and 100% is needed for a CPU intensive workload.
- Man procfs – press "/" – type "proc/stat"
  - CpuSteal is the 8th column returned
  - Metric can be read using SAAF in tutorial #4

94

EFFECTS OF SCALING FUNCTION MEMORY ON CPU TIME SHARE

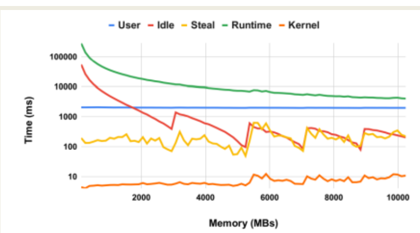


Figure 2: Linux CPU Utilization (log scale) vs. Function Memory for Sysbench Prime Number Generation

95

EFFECTS OF SCALING FUNCTION MEMORY ON CPU TIME SHARE

Key observations:

- Runtime decreases as vCPUs and CPU time share increase
- CPU user time remains constant for the prime number generation task – work doesn't change
- CPU idle time gradually decreases as memory and vCPUs increase (the idle time is becoming active time)
- When the 4th vCPU is added, cpuSteal tracks closely with cpuIdle time (hyperthreading effect)
- There is more cpu Kernel time after the 4th vCPU is added

96

FUNCTION INSTANCE LIFE CYCLES

- Function states:
- **COLD**: brand new function instance just initialized to run the request (more overhead)
  - Platform cold (first time ever run)
  - Host cold (function assets cached locally on servers)
- **WARM**: existing function instance that is reused
- All function instances persist for ~5 minutes before they begin to be “garbage collected” by the platform
  - 100% garbage collection may take up to ~30-40 minutes
- AWS Lambda appears to “recycle” infrastructure faster than other FaaS platforms
  - Presumably because of need, because the platform is busy

97

WARM VS COLD FUNCTION INSTANCES

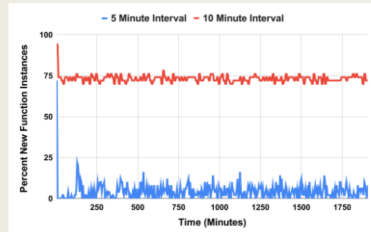


Figure 3: AWS Lambda Function Instance Replacement vs. Function Call Interval over 24-hours

98

CPU SCALING VS MEMORY: OTHER PLATFORMS: 0 TO 3 GB

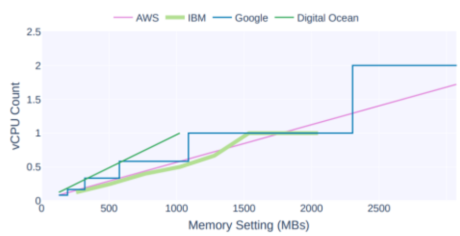
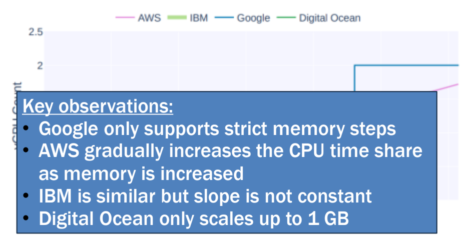


Fig. 2. Allocated vCPUs available at each memory setting on each platform.

99

CPU SCALING VS MEMORY: OTHER PLATFORMS: 0 TO 3 GB



Key observations:
- Google only supports strict memory steps
- AWS gradually increases the CPU time share as memory is increased
- IBM is similar but slope is not constant
- Digital Ocean only scales up to 1 GB

Fig. 2. Allocated vCPUs available at each memory setting on each platform.

100

ELASTIC FILE SYSTEM (AWS EFS)

- Traditionally AWS Lambda functions have been limited to 500MB of storage space
- Recently the Elastic File System (EFS) has been extended to support AWS Lambda
- The Elastic File System supports the creation of a shared volume like a shared disk (or folder)
  - EFS is similar to NFS (network file share)
  - Multiple AWS Lambda functions and/or EC2 VMs can mount and share the same EFS volume
  - Provides a shared R/W disk
  - Breaks the 500MB capacity barrier on AWS Lambda
- *Downside: EFS is expensive: ~30 ¢/GB/month*
- **Project**: EFS performance & scalability evaluation on Lambda

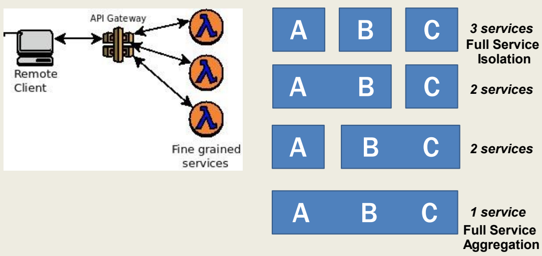
101

SERVERLESS FILE STORAGE COMPARISON PROJECT

- Elastic File System (EFS): Performance, Cost, and Scalability Evaluation in the context of AWS Lambda / Serverless Computing
  - EFS provides a file system that can be shared with multiple Lambda function instances in parallel
- Using a common use case, compare performance and cost of extended storage options on AWS Lambda:
  - Docker container support (up to 10 GB) – read only
  - Emphemeral /tmp (up to 10 GB) – read/write
  - EFS (unlimited, but costly) – read/write
  - image integration with AWS Lambda – performance & scalability

102

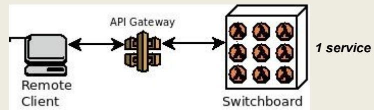
SERVICE COMPOSITION



Other possible compositions: group by library, functional cohesion, etc.

103

SWITCH-BOARD ARCHITECTURE



Single deployment package with consolidated codebase (Java: one JAR file)

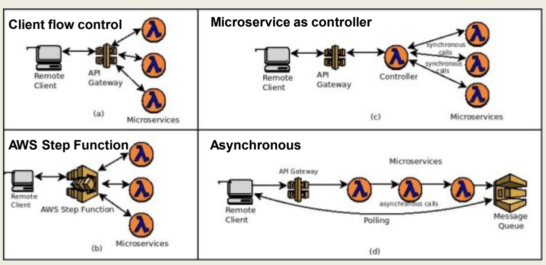
Entry method contains "switchboard" logic
Case statement that route calls to proper service

Routing is based on data payload
Check if specific parameters exist, route call accordingly

Goal: reduce # of COLD starts to improve performance

104

APPLICATION FLOW CONTROL - 3



105

PROGRAMMING LANGUAGE COMPARISON

- FaaS platforms support hosting code in multiple languages
- AWS Lambda- common: Java, Node.js, Python
  - Plus others: Go, PowerShell, C#, and Ruby
- Also Runtime API ("BASH") which allows deployment of binary executables from any programming language
- August 2020 – Our group's paper:
- https://tinyurl.com/y46eq6np
- If wanting to perform a language study either:
  - Implement in C#, Ruby, or multiple versions of Java, Node.js, Python
  - OR implement different app than TLQ (ETL) data processing pipeline

106

FAAS PLATFORMS

- Many commercial and open source FaaS platforms exist
- TCSS562 projects can choose to compare performance and cost implications of alternate platforms.
- Supported by SAAF:
- AWS Lambda
- Google Cloud Functions
- Azure Functions
- IBM Cloud Functions
- Apache OpenWhisk *(open source, deploy your own FaaS)*
- Open FaaS *(open source, deploy your own FaaS)*

107

DATA PROVISIONING

- Consider performance and cost implications of the data-tier design for the serverless application
- Use different tools as the relational datastore to support service #2 (LOAD) and service #3 (EXTRACT)
- **SQL / Relational:**
- Amazon Aurora (serverless cloud DB), Amazon RDS (cloud DB), DB on a VM (MySQL), DB inside Lambda function (SQLite, Derby)
- **NO SQL / Key/Value Store:**
- Dynamo DB, MongoDB, S3

108

PERFORMANCE VARIABILITY

- Cloud platforms exhibit performance variability which varies over time
- Goal of this case study is to measure performance variability (i.e. extent) for AWS Lambda services by hour, day, week to look for common patterns
- Can also examine performance variability by availability zone and region
  - Do some regions provide more stable performance?
  - Can services be switched to different regions during different times to leverage better performance?
- Remember that performance = cost
- If we make it faster, we make it cheaper…

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.109

109

CPU STEAL CASE STUDY

- On AWS Lambda (or other FaaS platforms), when we run functions, how much CpuSteal do we observe?
- How does CpuSteal vary for different workloads? (e.g. functions that have different resource requirements)
- How does CpuSteal vary over time hour, day, week, location?
- How does CpuSteal relate to function performance?

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.110

110

QUESTIONS

October 10, 2024 | TCSS462/562:(Software Engineering for) Cloud Computing [Fall 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.111

111