Midterm Review Guide – TCSS 562

Version 0.1

Midterm Date: Wednesday November 7th

The midterm exam will test basic knowledge and awareness of core technologies and concepts relating to cloud computing through mostly short answer questions. If questions require writing, length will be limited.

The midterm exam is open book and open notes. Feel free to bring any paper-based resources for the exam. Calculators are allowed, but laptop computers and cell phones are not permitted. The midterm will be completed as individual work during the class session on Wednesday November 7th.

Content with be limited to material covered directly in the lecture slides and tutorials. The lectures are based in varying degrees from content from two textbooks: (1) Cloud Computing Concepts, Technology, and Architecture book, and (2) Cloud Computing: Theory and Practice. For Fall 2018, in class lecture slides relate to from textbook (1) chapters 1, 3, 4, and to a lesser extent 7. From textbook (2) from the 1st edition: chapters 1, and portions of 2, 3, 4. Chapter 11 provides content similar to our tutorials. With textbook (2) from the 2nd edition: chapters 1, 2, and 4 primarily. Questions relating to tutorials 1, 2, 3, 4, and 5 may be included. Tutorial 6 will not be covered on the midterm. Questions regarding various AWS cloud technology and pricing questions for example: Function-as-a-Service vs. Infrastructure-as-a-Service. In all cases the cloud pricing policy will be included in full on the exam. It is not necessary to memorize pricing policies, or collect notes on each specific policy. If a question involves a particular pricing policy, then the policy will be described on the midterm as described on the cloud provider's website(s).

List of review topics:

- Moore's Law
- Fine-grained parallelism
- Coarse-grained parallelism
- Message passing vs. shared memory for parallel computation
- Motivations for cloud computing
- What is Thread Level Parallelism (TLP)?
- Why is it important to measure the maximum TLP of an application when deploying it to the cloud?
- Data-Level Parallelism
- Bit-Level Parallelism
    - CPU pipelining
- Instruction-Level Parallelism
- What are CPU cores, CPU hyperthreads, and vCPUs?
- Control flow architecture (von Neuann architecture)
- Data flow architecture

    - Why have data flow architectures not panned out?
- Flynn's architectural taxonomy
    - Single Instruction, Single Data
    - Single Instruction, Multiple Data
    - Multiple Instruction, Multiple Data
- Arithmetic Intensity
- Roofline Model
- Speed-up from parallelization
- Amdahl's Law (perfect scaling)
- Gustafon's Law (scaled speed-up)
- What does it mean if a software system is highly available? (HA)
- Characteristics of Distributed Systems
    - Non-functional quality attributes
    - Availability, Reliability, Accessibility, Usability, Understandability, Scalability, Extensibility, Maintainability, Consistency
- Transparency properties of distributed systems
    - General goal of distributed systems is to hide the fact that they are actually "distributed"
    - Access transparency
    - Location transparency
    - Concurrency transparency
    - Replication transparency
    - Failure transparency
    - Migration transparency
    - Performance transparency
    - Scaling transparency
- Soft modularity – TRADITIONAL
    - Classic object oriented best practices
        - Minimize coupling between classes/modules
        - Maximize cohesion between functions within classes/module
        - These practices are thought to improve software maintainability, reusability, portability
        - Low coupling correlates with high cohesion
        - High coupling correlates with low cohesion
- Enforced modularity – CLOUD / WEB SERVICES
- What is Grid Computing?
- What is Cluster Computing?
- What is virtualization?
- What are virtual machines?
- What is horizontal scaling?
- What is vertical scaling?
- What is the difference between a cloud provider and a cloud consumer?
- Can a cloud consumer also be a provider?
- What factors (list a few) make ensuring service level agreements "on the cloud" difficult?
- What are some risks associated with cloud adoption?

- What is multitenancy?
- What is resource elasticity in the cloud?
- Types of clouds: Public, Private, Hybrid, Federated, Community
- What is an elastic block store volume?
- What is an ephemeral instance store volume?
- How are EBS volumes and instance store volumes different?
- For AWS, how does storage performance vary for local disks (instance store) vs. network disks (elastic block store)?
- Know about credit-based resource sharing models. Examples:
    - t2 series VM instances – the CPU is shared based on a “credits” system
    - General purpose 2, EBS volumes – I/O operations can burst at a higher rate until credits are exhausted
- Know about spot instances
- What’s the difference between a snapshot, an AMI, and an EBS volume?
- Cloud storage models
    - Elastic block storage
    - Blob/object storage (S3)
    - Local disk storage (ephemeral / instance storage)
- Overprovisioning of cloud resources
- Cloud Delivery Models
    - Infrastructure as a Service
    - Platform as a Service
    - Software as a Service
    - Function as a Service
    - Container as a Service
    - Database as a Service
- What are the motivations for serverless computing?
  Serverless platforms include Integrated support for:
    - high availability
    - fault tolerance
    - automatic scaling and load balancing
- Other advantages of serverless computing
    - No server configuration
    - Pay only for actual service execution time
- Know some of the challenges for leveraging serverless computing
    - obfuscated billing models
        - The pricing is hard to estimate
        - Reliance on so many external cloud services really “distributes” the cost across a variety of services, complicating cost accounting and bill reconcilation.
    - knowing how much “memory” to reserve
        - AWS Lambda & Google Cloud Functions: CPU power is coupled to memory reservation size producing approx. an order of magnitude (~10x) performance difference
    - Software composition into individual functions can impact hosting costs and performance

        - There is a case here for consolidating functions, which may be counter-intuitive to the recommended software engineering practices of building many light-weight independent microservices
        - Fewer functions (fewer package deployments) as in a switchboard architecture may reduce the overall cloud infrastructure footprint for a serverless application, reducing the volume of infrastructure that must freeze/thaw.
        - Composition of functions impacts how data flows in an application
            - Functions composed together can eliminate the need to transport some data over the network
            - Composing too many functions together may result in time outs
    - What is an Asynchronous service call?
    - What is a Synchronous service call?
    - What is double-billing as it relates to application flow control for a serverless application?
    - Provisioning variation of infrastructure can lead to performance variability
    - Infrastructure initialization provides 20x slower performance
- What is the difference between function-as-a-service and serverless computing?
    - Serverless computing refers to the abstraction of servers, and is not a specific cloud delivery model.
- What is a COLD call to a serverless computing platform (e.g. AWS Lambda)?
- What is a WARM call to a serverless computing platform (e.g. AWS Lambda)?
- Infrastructure Freeze/Thaw lifecycle of FaaS/serverless platforms
- When to use Infrastructure-as-a-Service cloud?
- When to use Platform-as-a-Service cloud?
- When to use Software-as-a-Service cloud?
- When to use Function-as-a-Service cloud? (Serverless Computing)
- When to use Database-as-a-Service cloud?
- When to use Container-as-a-Service cloud?

Lecture slides with questions could be adapted into midterm questions.

1 Change History

Version	Date	Change
0.1	11/02/2018	Original Version