OMSCS 6210 Spring 2023 Test 2 Rubric

Table of Contents

Instructions

- The test is closed everything except intellect.
- You can use only one scratch paper which should be visible when you perform the room scan on Honorlock.
- For True/False questions, no credit will be given without justification.
- If you face any issue, you can contact Honorlock support from Canvas while taking the exam.
- Clarifications will be posted on ed sometime after **Noon on April 8th**.
- We suggest you wait till we post clarifications before giving the exam.

[34 points] Distributed Systems

I. [6 points] Lamport's Logical Clock

A. [2 points]

Lamport's logical clock is an intellectually appealing strategy for maintaining the state of communicating processes in a distributed system. What is deficient about logical clocks that necessitates the physical clock formalism by Lamport in the same paper?

- Logical clock is a good abstraction for causal ordering of events through communication among the participating entities.
- Correctness of many distributed applications (e.g., ATM, Internet of things (IoT)) depend on temporal ordering of events with or without causal ordering (i.e., communication among cooperating processes), necessitating physical clocks.

B. [4 points] Assume a system where clocks on nodes do not drift and the network communication time is constant. Do we still need a Lamport's clock to determine "happened before" relations? Explain.

No. Since clocks do not drift, we just need the nodes to communicate with each other what clock time they start at. Since the network communication time is constant, two nodes can calculate this network time and then synchronize both the clocks. After this, all nodes can then use their local clock time as the system clock time thus globally determining the sequence of operations.

II. [12 points] Lamport's ME Algorithm

A. [5 points] Construct a simple ordering of events involving two processes (P1 and P2) where Lamport's algorithm violates mutual exclusion, where P2 acquires mutex when it has already been acquired by P1.

Give your answer in the form of a sequence of one of the following events. Show the state of the queues at P1 and P2 wherever necessary, and clearly indicate which messages were delivered out of order.

- SEND LOCK/ACK/UNLOCK P1 -> P2 (P1 sends lock/ack/unlock to P2)
- RECV LOCK/ACK/UNLOCK P1 -> P2 (P2 receives lock/ack/unlock sent by P1)
- ACQUIRE P1 (P1 acquires lock)

- RELEASE P1 (P1 releases lock)

P2: send LOCK to P1
P1: send LOCK to P2
P1: receives LOCK from P2 (P1: {P2}, P2: {})
P1: sends ACK to P2
P2: receives ACK from P1 (out of order!) (P1: {P1, P2}, P2: {P2} -> acquire)
P2: acquires LOCK (only it's lock is in its queue)
P2: receives LOCK from P1 (P1: {P1, P2}, P2: {P1, P2}- but P2 already started)
P2: sends ACK to P1
P1: acquires LOCK (its LOCK is the one with the lest timestamp in its queue)
Jist: An ACK sent after a LOCK message gets delivered before the LOCK message is delivered.

B. [3 points] Construct a simple example of a sequence of events involving two processes P1 and P2, where "progress" is violated, that is, in a situation where the mutex previously held by P1 has been released, but P2 is not able to acquire it.
Use the format shown in part (A) for the sequence of events, and clearly indicate which message was lost.

P1: sends LOCK to P2
P2: sends LOCK to P1
P2: receives LOCK from P1
P2: sends ACK to P1
P1: receives LOCK from P2
P1: receives ACK from P2
P1: acquires mutex (its first in its queue)
P1: releases mutex, sends UNLOCK to P2 -> this message is lost
P2 waits forever
Jist: Either an unlock message is lost or an ACK message is lost or a LOCK message is lost.

C. [2 points] The correctness of the basic Lamport's ME algorithm depends on no message loss. You want to relax this requirement and yet assure correctness. How would you do it?

Answer: TODO

D. [2 points] Your co-worker asserts that for the correctness of the basic Lamport's ME algorithm, messages from a given process P1 to all other processes in the entire distributed system must arrive in the order in which they are sent from P1. Is she right? You should justify your answer with an explanation.

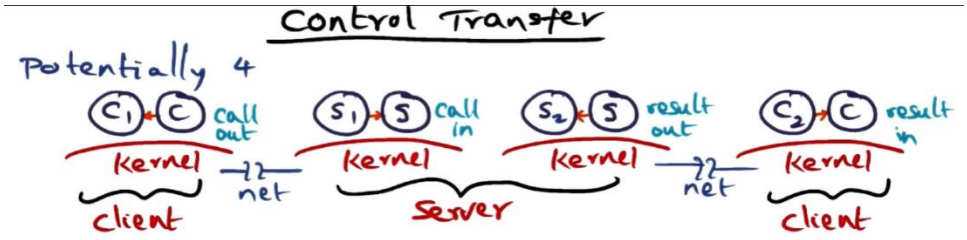
False. The correctness of the algorithm only demands that messages from Pi -> Pj must be delivered in the order in which they are sent. The order in which messages arrive at different destinations does not matter.

III. [10 points] Latency Reduction in RPC

A. [2 points] Thekkath and Levy suggest using a shared descriptor between the client stub and the kernel for marshalling arguments during an RPC call. Your friend argues that this does not result in reducing the copying overhead of an RPC call. How would you counter her argument?

Ans: I would argue that using a shared descriptor allows the kernel to assemble arguments into the kernel buffer without an extra copy by the client stub. (+1)
Each element of the shared descriptor contains: <start address, number of contiguous bytes> and client stub fills up the descriptor at point of call: <&arg1, length>, <&arg2, length>. The kernel can then assemble arguments into the buffer. (+1)

B. [4 points] Consider the following control transfers involved in an RPC call.



Out of these, identify the ones that are in the critical path. Why are the identified calls considered in the critical path while others are not?

Ans: Out of the 4 context switches, call in and result in are in the critical path while call out and result out are not in the critical path. (+2)
The reason for this is that both call in and result in are essential in processing a rpc request and response, whereas call out and result out are optimizations to ensure that the client and server are not underutilized.(+2)

C. [4 points] You have designed an RPC system which has the following parameters:

- T: Message transmission time in either direction for sending the arguments of the call or receiving the results (includes protocol processing, time on the wire, and interrupt processing)
- CSc = 3T: Context switch time at the client node
- CSs= T: Context switch time at the server node

You notice from the logs that the server procedure execution time is bimodal: it takes T units of time mostly but occasionally 20T units of time.

How would you optimize the latency for RPC calls for this client-server interaction?

Most of the time the RPC latency is 4T (CSs+server processing+communication times).

Context switch time at client (C1->C2->C1) is 6T.

Optimization:

- Spin for 4T units of time after sending the client request
- If response does not come back context switch to a different client

+2 if they mention spinning for a while and then context switching.

+2 if they list the exact duration for spinning as well as the proper reason.

IV. [6 points] Active Networks

A. In an Active Network, we expect the intermediate routers to execute code by looking at the "type" field of the capsule present in the incoming packet. For a "type" that it has not seen before, it requests the code from the node present in the "prev" field.

1. [2 points] Is it possible for the "prev" node to not have the code corresponding to the "type" of the incoming packet? Explain why.

   (+1) Yes.
   (+1) Size of soft cache is limited and as new network flows hit the node, type fields for old flows may get evicted.

2. [2 points] Given that the active store uses LRU to replace items from it, why would the "prev" node NOT have the code corresponding to the "type" field?

   While possible when there is heavy contention for the active store due to multiple parallel network flows, it is highly unlikely. It is a symptom of a flow that has long terminated but a capsule of that flow was meandering through the network (possibly through other non-active nodes) and finally hit an active node.
   (+2 if the above sense is conveyed)

3. [2 points] How is this situation handled in Active Networks? Why?

(+1) The capsule is dropped
(+1) A capsule is similar to an IP packet and higher level layers in the protocol stack would take care of packet loss and retransmission as deemed necessary.

[26 points] Distributed Objects and Middleware

V. [10 points] Spring OS

A. [2 points] What purpose does the memory object abstraction serve in the virtual memory subsystem of the Spring Kernel?

It allows the VM manager to flexibly map regions for virtual memory for a given process to files on the disk; it also allows memory sharing across different processes (i.e., the same memory object could be mapped to different regions of virtual memory in different processes).

+2 if the above sense is conveyed

B. [2 points] Your co-worker argues that Spring Kernel's memory management does not offer any extensibility features. How would you counter that argument?

User-defined pager object allows extending the default paging provided by Spring kernel.

+2 for any acceptable argument

C. [6 points] In Sun's Spring Kernel, a client/server model is used to provide network services like (e.g., NFS). However, the clients and servers are impervious as to whether they are collocated on the same machine or are interacting over a LAN. Describe in 4-5 concise bullet points how Sun achieves this location transparency for client-server interaction.

Spring uses strong interfaces. Contract between servers and clients is established using IDL. Only the service provided is made public and not its implementation.
Spring uses subcontract to hide runtime mechanisms from interfaces, and decouple the physical location of the client and server by providing server instance selection
Subcontract also helps in marshalling/unmarshalling of arguments/responses, and in transportation services between stubs.

Client-server interaction on the SAME machine is optimized by the subcontract mechanism utilizing shared memory.
Spring uses network proxies which are outside the kernel to make connections between clients and servers on the LAN (orchestrated as

needed by the subcontract mechanism), but these network proxies are invisible to both clients and servers.

-2 if subcontract is not mentioned
-2 if proxies are not mentioned
+6 if the above logic is conveyed

VI. [10 points] EJB

A. [6 points] You have a startup to implement a portal for airline reservations. The clients come to you over an insecure wide-area network. These are the objectives which are your "secret sauce" for the startup:

- You want to exploit parallelism across independent client request
- You want to exploit parallelism within each client request
- You want to protect your business logic from being exposed to the wide-area Internet

You are planning to use EJB for meeting these objectives. Your N-tier solution has a Web container, an EJB container, and a Database server. To meet the design objectives:

1. [2 points] What functionalities would you put into the Web container (that interfaces with the client browsers)?

Functionalities to put in the Web container:

(i) Servlet which corresponds to a session of a client (+1)

(ii) Presentation logic for each servlet (+1)

2. [4 points] What functionalities would you put into the EJB container? Justify

(i) A session façade that corresponds to each servlet in the web container (+1)

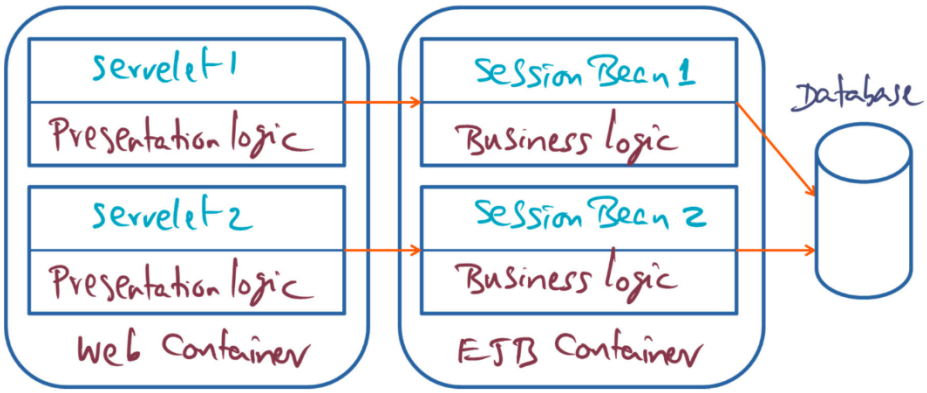
(ii) The business logic for each servlet (+1)

(iii) Entity beans which are the data access objects serving as an interface for the business logic layer to access the database. (+1)

(With this design, the business logic is not exposed and also parallelism across independent client requests and within the client requests can be exploited through the entity beans) (+1)

B. [4 points] Consider the following design alternative for constructing

enterprise software which uses coarse-grain session beans. In the figure, servelet1 and session bean 1 represent client 1; servelet2 and session bean 2 represent client 2.



1. [2 points] [True or False with justification] The above design exploits opportunities for parallelism within a single client.

False. Since one business logic tied to one session bean, there are no avenues of parallelism within a single business logic associated to a single client. Avenues of parallelism existed if there were multiple entity beans for a single client

+1 False

+1 if justification conveys above answer

2. [2 points] [True or False with justification] The above design is vulnerable to compromising the core business strategies of the enterprise.

False. Since one business logic tied to one session bean, there are no avenues of parallelism within a single business logic associated to a single client. Avenues of parallelism existed if there were multiple entity beans for a single client

False. Since the business logic is embedded in a session bean on an EJB container, it is only visible to the corporate network and not to the outside network.

+1 False
+1 if justification conveys above answer

VII. [6 points] Java RMI

A. [4 points] Java RMI evolved from the Spring Subcontract mechanism. Name one similarity and one difference in the implementation of the two systems.

+2 – Valid difference Ex: Spring uses IDL to describe remote objects whereas Java RMI uses pickling

+2 – Valid similarity Ex: Subcontract mechanism in Spring vs RMI.

B. [2 points] Java allows object references to be passed as parameters during object invocation. What is the difference in parameter passing (when a local object reference is passed as a parameter) while invoking a remote object using Java RMI?

+2 – Remote invocation stub – not affection original object.

[42 points] Distributed Subsystems

VIII. [16 points] GMS

A. [2 points] Is it possible for a page X to be present in the "local" part of two nodes N1 and N2 at the same time? If yes, explain how.

Answer: Yes. The page could be actively shared by a process running on N1 and another process running on N2

B. [4 points] N1 faults on page X; N1's global part is empty; N2 has the oldest page in the entire cluster in its global part; the missing page X is not in cluster memory. List the steps that will ensue to service this page fault.

Answer:

- X will be brought from the disk to N1

- N1 will send its oldest page (from local part since its global is empty) to N2

- N2 will discard its oldest page (which is in its global part and hence clean by design)

C. [8 points] Assume that we have a set of Nodes N1, N2 and N3 in a Global Memory System. The previous epoch of the geriatrics algorithm has just ended. Now each of the nodes send age information for each of their Local and Global pages to the initiator. The age information sent by each node is shown below:

Node N1: [LP1: 5, LP2: 7, LP3: 4, GP1: 11, GP2: 2]

Node N2: [LP1: 1, LP2: 8, GP1: 3, GP2: 3, GP3: 9]

Node N3: [LP1: 13, LP2: 15, LP3: 4, LP4: 1, LP5: 10]

The integers corresponding to each page denote its age (a page with age 10 is older than a page with age 5)

We choose the parameter for Max Page replacement M = 6 (assume that the parameter T for epoch duration does not play a role here)

1. [6 points] List down the response sent by the initiator to each of the nodes while clearly stating what each section of the response means. [Hint: each “weight” field in the response must be denoted as a percentage value or as a ratio]

Ans: the response should be in the format of minage, {weight vector}

Node N1: [8, {1/6, 2/6, 3/6}] Node N2: [8, {1/6, 2/6, 3/6}] Node N3: [8, {1/6, 2/6, 3/6}]

2. [2 point] Which node is selected as the initiator in the next epoch? Why?

Ans: Node N3 is picked as the initiator since it had the maximum weight in the previous epoch.

IX. [14 points] DSM

A. [6 points] Consider a page-based software DSM system that implements a single-writer multiple-reader coherence protocol. A process P on Node N1 wants to write to a page X. The page X is present in N1 but it is marked read-only. Node N3 is the owner of the page X which is currently read-shared by nodes N1, N2, and N3. List the steps involved in handling this situation to allow process P to be able to write to page X.

Answer:

- When node N1 wants write access to page X, the OS relays this to the DSM software residing on node N1 (+1)
- DSM finds the node that owns page X from the statically determined mapping (in this case, node N3) and contacts node N3 for write access to page X (since node N1 already has a fresh read-only copy of page X) (+2)
- Since node N1 wants to write to page X and this system follows single-writer coherence protocol, the owner node (node N3) also has to invalidate all the other copies of page X on other nodes, in this case nodes N2 and N3 (+2)
- Node N1 can now write to its own copy of page X after all the other read-only copes are invalidated (+1)

B. [6 points] In the Treadmarks DSM system, assume that nodes N1, N2 and N3 execute the following critical sections in the order shown below. Assume that the pages A and B are available on all three nodes, and that the computation begins with the code as shown below.

Node N1:

```
Lock(L1);
Write to page A;
```

```
Unlock(L1);
```

Node N2:

```
Lock(L1);
Write to page A;
Write to page B;
Unlock(L1);
```

Node N3:

```
Lock(L1);
Write to page B;
Unlock(L1);
```

1. [2 points] When node N2 acquires the lock L1, what actions are taken by the Treadmarks DSM system?

Answer: Page A is invalidated since it is associated with lock L1

+2 for above answer

2. [2 points] When node N3 acquires the lock L1, what actions are taken by the Treadmarks DSM system?

Answer: Pages A and B are invalidated since their writes are both associated with lock L1 after the actions on node N2

+1.5 Page A is invalidated

+0.5 Page B is invalidated

3. [2 points] When node N3 tries to write to page B, what actions are taken by the Treadmarks DSM?

Answer: Diffs for page B are applied onto its own copy of page B (+1), and a pristine twin page is created for comparing changes on release (+1)

C. [2 points] [True or False with justification] Multi-writer coherence protocol works with SC, RC and LRC memory models.

Ans: False. Only works with RC and LRC models since they can distinguish between synchronization variable accesses and normal data accesses.

+0 Incorrect

+1 False without valid justification

+2 False with valid justification

X. [12 points] DFS

A. [4 points] Consider xFS file system.
- Node Nf is the manager node for a file F1
- Assume that coherence is maintained at the granularity of individual files
- F1 is currently read-shared by nodes N1, N2, and N3

The following events happen:

- Time T1: Node N1 attempts to write to file Nf
- Time T2: Node N2 attempts to read the file Nf

1. [2 points] List the actions that would take place at time T1

   - The metadata manager Nf for F1 gets the write request from N1. [+0.5 points]
   - It then sends invalidation messages to all the other nodes (N2 and N3). [+0.5 point]
   - Once it gets the ACKs, it changes the metadata for F1 to indicate that N1 is the current writer. [+0.5 points]
   - It sends permission to write to N1 [+0.5 points]

2. [2 points] List the actions that would take place at time T2.

   - Metadata manager Nf for F1 gets the read request from N3. [+0.5 points]
   - It revokes the token for writing from the current writer (N1), and asks N1 to make its copy read-only. [+0.5 points]
   - Once it gets N1's ACK, it updates the metadata for F1 to indicate that both N1 and N2 have read access to the file. [+0.5 points]
   - It instructs N2 to get the file from N1's cache. [+0.5 points]

B. [2 points]

Why do file systems procrastinate writing a file to stable storage as soon as the process that is doing the write closes the file?

- Lifetime of temporary files are short, so save on I/O

C. [2 points]

In xFS, what purpose does the "log segment" data structure of the file system serve?

- Keep the changes to multiple files in the log segment to overcome the small write problem

D. [2 points] Distinguish between static and dynamic metadata management.

- Static: metadata manager decided at file creation time and does not change for the lifetime of the file
- Dynamic: metadata manager for a given file changes to adjust to "hotness" of files being used

E. [2 points][True or False with justification] In xFS the imap data structure is replicated globally.

False. The imap data structure is partitioned among the statically assigned metadata managers.