CS 6210 Spring 2021 Test 2 - Soln

Name:____TAs Plus Kishore____GT Number:

Note:

1. **Write your name and GT number on each page.**
2. The test is **CLOSED BOOK** and **NOTES.**
3. Please provide the answers in the space provided. You can use scratch paper (provided by us) to figure things out (if needed) but you get credit **only** for what you put down in the space provided for each answer.
4. For conceptual questions, **concise bullets** (**not wordy sentences**) are preferred.
5. While it is NOT REQUIRED, where appropriate use figures to convey your points (a figure is worth a thousand words!)
6. **Illegible answers are wrong answers.**
7. **DON'T GET STUCK ON ANY SINGLE QUESTION…FIRST PASS: ANSWER QUESTIONS YOU CAN WITHOUT MUCH THINK TIME; SECOND PASS: DO THE REST.**

Good luck!

Question number	Points earned	Running total
1 (1 minute) (Max: 1 pts)		
2 (7 minutes) (Max: 10 pts)		
3 (7 minutes) (Max: 10 pts)		
4 (7 minutes) (Max: 10 pts)		
5 (7 minutes) (Max: 10 pts)		
6 (3 minutes) (Max: 5 pts)		
7 (3 minutes) (Max: 5 pts)		
8 (3 minutes) (Max: 5 pts)		
9 (10 minutes) (Max: 15 pts)		
10 (15 minutes) (Max: 20 pts)		
11 (7 minutes) (Max: 10 pts)		
Total (75 minutes) (Max: 101 pts)		

1. **(1 point, 1 minute)** (you get 1 point regardless of your answer)
Video hangout every week
(a) I attend all of them though I am not a fan
(b) I attend all of them and I love it
(c) I cannot attend them due to timing but watch every one of the recordings
(d) I cannot attend them due to timing but watch a few recordings for class participation points
(e) I cannot attend them due to timing and do not watch any of the recordings

CS 6210 Spring 2021 Test 2 - Soln

Name:____TAs Plus Kishore____GT Number:

Lesson 5: Distributed Systems

2. **(10 points, 7 minutes) (Lamport's happened before relation)**
A student has implemented a distributed algorithm using Lamport's happened-before relationship to timestamp events. She is in the middle of debugging the program. She observes the following activities in the program:

P1's activities	P2's activities	P3's activities
E1: local event	E6: msg-receipt(from P1)	E10: msg-receipt(from P2)
E2: local event	E7: local event	
E3: msg-send(to P2)	E8: msg-send(to P1)	
E4: local event	E9: msg-send(to P3)	
E5: msg-receipt(from P2)		

Please help her by giving the causal relationship between the following pairs of events with reasoning. (No credit without reasoning)

(a) (2 points) E1 and E6?

Ans: E1 -> E6. Reason – E6 will wait for msg from E3. E3 happens after E1.
Rubric: all or nothing

(b) (2 points) E4 and E6?

Ans: E4 || E6. Reason: E6 will execute after E3 (after msg receipt) and E4 will also execute after E3. Since, these two events are independent of each other, they will be concurrent.
Rubric: all or nothing

(c) (2 points) E6 and E8?

Ans: E6 -> E8. Reason: Since both the events are in P2 and E6 is higher up in the event execution order, E6 will happen before E8.
Rubric: all or nothing

(d) (2 points) E3 and E10

Ans: E3 -> E10. Reason: E10 will wait for msg from E9 which will happen after E6. E6 will wait for the msg from E3. Thus, E3 will happen before E10.
Rubric: all or nothing

(e) (2 points) E5 and E10

Ans: E5 || E10. E5 will execute after receiving msg from E8. E10 will execute after receiving msg from E9. E8 -> E9. But, the events they are sending messages to can happen in any order. Thus, E5 || E10.

Rubric: all or nothing

3. **(10 points, 7 minutes) (Lamport's M.E. Algorithm)**
Lamport's mutual exclusion algorithm for a distributed system is based on happened-before relationship. It also hinges on two assumptions: (a) messages between any two nodes go in order, and (b) there is no loss of messages.

(a) What additional machinery would you need to make sure the algorithm will correctly if the first assumption is relaxed?

Solution 1 using sequence numbers

- Every processor maintains a distinct sequence number (initialized to zero) for each of its peers.
- Message send from Pi to Pj: Modify the algorithm to include this sequence number in addition to the timestamp with every communication (lock, ack, unlock):
    - e.g., lock(L, ts, Sij) // Sij is the sequence number for Pi's message send events to Pj
    - Sij is incremented by 1 after each message send
- Message receipt by Pj from Pi: Pj remembers the last sequence number of message from each peer Pi (Li).
    - If the sequence number of the message received from Pi is NOT the next one expected (Li+1 != Sij) then buffer the message locally
    - If the sequence number of the message from Pi is the next one expected (Li+1 == Sij) then
        - process the message
        - increment Li

(+2 for mentioning sequence number like mechanisms for message sends to each peer; +1 for what happens on message send; +2 for what happens on message receipt) (Note: need not be as complete as above for full credit)
Note: alternate correct solutions that do not use sequence numbers will also get due credit like the one below

Solution 2 exploits the semantics of M.E. problem:

- Defer ACKs if the peer's lock request is later than mine and use the UNLOCK as an implicit ACK for a peer who is waiting for my ACK to get the lock.
- So at any point of time, a node ensures that even it if it has multiple messages in flight to the same destination (e.g., ACK followed by my own lock request; unlock followed by another lock request for the same lock), their reordering in the network is not going to result in an erroneous decision at the destination node.

(b) What additional machinery would you need to make sure the algorithm will correctly if the second assumption is relaxed?

Ans:

Solution using sequence numbers:

- Every processor maintains a distinct sequence number (initialized to zero) for each of its peers.
- Message send from Pi to Pj: Modify the algorithm to include this sequence number in addition to the timestamp with every communication (lock, ack, unlock):
    - e.g., lock(L, ts, Sij) // Sij is the sequence number for Pi's message send events to Pj
    - On the receipt of ACK for Sij, Sij is incremented by 1 after each message send
    - If ACK is not received within a timeout interval, Pi will resend the message to Pj
- Message receipt by Pj from Pi: Pj remembers the last sequence number of message from each peer Pi (Li).
    - If the sequence number of the message received from Pi is NOT the next one expected (Li+1 != Sij) then buffer the message locally
    - If the sequence number of the message from Pi is the next one expected (Li+1 == Sij) then
        - Send an ACK to Pi
        - process the message
        - increment Li

(Note: need not be as complete as above for full credit)
(-2 if the algorithm does not handle loss of messages correctly)

Note: alternate correct solutions that do not use sequence numbers will also get due credit like the one below

Alternate solution:

Define a timeout for process. When the process sends a message and does not receive an acknowledgement within the timeout interval, retransmit the message.

In the original ME algorithm, lock can be acquired in 2 cases:
1. When a process receives acknowledgements from all other processes
2. When a process sees that it has lock requests from other processes at a later timestamp.

Since the messages can be lost, we cannot rely on second case. So, we need to rely only on acknowledgements if we assume there is loss of messages.

4. **(10 points, 7 mins) (Latency Reduction in RPC)**
Your co-worker came up with a design for a Network Interface Card (NIC) that does scatter/gather DMA. That is, it is possible to give the DMA controller a tuple in the form: {(memory address, length), (memory address, length), (memory address, length), ...}. The NIC's DMA engine will then do the needful to transfer the packet on to the network.

You are using this DMA controller to implement an efficient RPC package at the user level. Give a sketch of your design that minimizes the number of copies for marshalling the arguments from the client. (It is sufficient to show client-side marshalling).

- Set up a shared descriptor between the client stub and kernel. Each element is of the form (memory address, length.
- During RPC, client updates the descriptor with the memory address and the length of arguments.
- This shared descriptor is passed on to NIC.
- NIC then copies the data from the memory address to its local buffer to send it out on wire.

This basically reduces the number of copies to 1.

(Rubric: +2.5 for each point)

5. **(10 points, 7 minutes) (Active Networks)**
One could argue that Software Defined Networking (SDN) is "old wine in a new bottle".
(a) Justify this argument with a few concise bullets.

- Like AN, SDN allows optimizing the routing per individual flow between {source, destination} pairs.
- Like AN, SDN allows each router along the way to decide on which outgoing link to send an incoming packet based on processing the packet header.

(+2.5 per point)

(b) With a few concise bullets, summarize the social and technology enablers that has made this wine drinkable now.

Strikes agains AN:

- AN was difficult to sell socially for two reasons: (a) vendors of network gear were loath to open the network for arbitrary code execution in the routers; (b) WAN for which AN was proposed is not owned by a single service provider so it is difficult to put the vision of AN into practice.
- AN was technologically infeasible in the WAN environment since software routing could never match line rate in the core of the network.

SDN is a resurgence of AN in a limited form:

- Social reason: A Datacenter on the other hand is owned by a single entity. Further, it is multi-tenant, and each tenant would want their network flows to be protected against sniffing by others for protecting their business interests. So, the social strike against AN went away.
- Technology reason: Advances in hardware (FPGAs) allowed route selection to be "programmed" into the switches (i.e., routers) at the beginning of the flow for the duration of the flow. So, while there is a one-time cost for setting up a unique route between source and destination to meet SLAs (service level agreements with the tenants), during the actual packet flow the routing decision is taken in hardware. Thus, the switching can match the line rate removing the technological infeasibility of AN.

(Kishore's note: I don't expect this elaborate an answer; +2.5 if datacenter is owned by single entity and needing to cater to multiple tenants who need their business logic to be protected is mentioned; +2.5 if emergence of hardware that allows switching at line rate is mentioned)

CS 6210 Spring 2021 Test 2 - Soln

Name:____TAs Plus Kishore____GT Number:

6. **(5 points, 3 minutes) (Ensemble/Nuprl)**
The authors cite features that justify the choice of OCaml as the systems programming language. Yet when it comes to optimization opportunities of the layered systems code, they point to many of these same features as the ones to circumvent to achieve good performance! Is there a contradiction? With a few concise bullets, explain why or why not.

Answer:
There is no contradiction if one considers the goals of the project:

- The top level goal of Ensemble/NuPrl is to mimic the VLSI methodology of composing complex software using basic building blocks. To accomplish this goal, they would have to choose a programming language that has well-defined semantics and no side effects on procedure calls. OCaml, being a functional language, fills the bill for this top-level goal.
- The secondary goal is that the building blocks must be fine-grained enough so that complex software could be built with just the functionalities needed and nothing more. Clearly, fine-grained objects would lead to inefficiencies due to the call-return (marshalling and unmarshalling the arguments and return values) inherent in achieving this level of modularity. This calls for optimization after the building blocks are assembled. Here again the well-defined interfaces of Ocaml and no side-effect are what makes it possible for them to optimize the code using the NuPrl theoretical framework.

(Kishore’s note: I don’t expect an elaborate answer as above.
+2.5 for need to use a language with no side effects on procedure calls;
+2.5 for having fine-grained objects to promote modularity which would imply post-composition optimization)

CS 6210 Spring 2021 Test 2 - Soln

Name:____TAs Plus Kishore____GT Number:

Lesson 6: Distributed Objects and Middleware

7. **(5 points, 3 minutes) (Spring Kernel)**
You are the implementor of the "subcontract" subsystem in the Spring kernel. On the client side, you have the following API calls available to the Client-side stub: Invoke; Marshall: Unmarshall

You want to optimize the "Marshall" call to exploit the location of the server. Assume the client and server are executing on different processors of a shared memory machine.

How would you optimize marshalling the arguments of the call?

Answer:

- The subcontract on the client-side will create a descriptor upon "marshall" call. Each element of the descriptor would be a tuple: {pointer to the argument, size} for each of the actual parameters in the "marshall" call.
- The descriptor will be passed (via shared memory) from the subcontract on the client-side to the subcontract on the server-side during the "invoke" call.
- When the server-side stub calls "umarshall", the subcontract on the server-side will use the descriptor populate the server stack with the actual parameters.

(Kishore's note: Students' answers could be different from the above; full credit if copying the arguments multiple times is avoided in the solution. -2 for each extra copy)

Name:____TAs Plus Kishore____GT Number:

8. **(5 points, 3 minutes) (EJB)**
You have a startup to implement a portal for hotel 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 requests
- 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:
(a) What functionalities would you put into the Web container (that interfaces with the client browsers)?

I would put distinct servlets with presentation logic for each client in the web container. This would help exploit parallelism across independent client request.

(+2.5 for above answer; all or nothing)

(b) What functionalities would you put into the EJB container?

I would put a session façade with business logic for each client in EJB container. This would prevent business logic from being exposed to the insecure WAN. Each session façade will internally communicate with entity beans for access to the database. Each entity bean would be responsible for fetching a group of rows from the database. This would help exploit parallelism within each client request.

(+1 for session façade for ensuring business logic not exposed; +1.5 for multiple entity beans to provide intra-client parallelism)

CS 6210 Spring 2021 Test 2 - Soln

Name:____TAs Plus Kishore____GT Number:

Lesson 7: Distributed Subsystems

9. **(15 points, 10 minutes) (GMS)**
(a) (5 points)
Due to memory pressure, Node N1 has paged out a **dirty** page X of a process P1 to Node N2. Node N2 is shut down for some reason. The process P1 on Node N1 page faults on X. What are the elements of the GMS design that ensures that Node N1 can satisfy the page fault for X?

Elements of GMS design that overcomes this problem:

- Only clean pages are kept in "Global".
- When N1 paged out X to N2, it would have first written to the disk to make the page clean.
- Thus, when N1 needs to get X to satisfy the page fault, it would go to the disk and fetch it even though N2 is shut down

(Kishore's note: full credit if the above sense is conveyed;
Partial credit: +2 for mentioning only clean pages in Global; +2 for writing a dirty page to disk before sending the page to a peer node; +1 for getting the page from disk)

(b) (5 points) (Answer True/False with justification. No credit without justification.)
The "geriatrics" algorithm decides the exact set of M pages that will be replaced in the upcoming epoch.

False. (+1 if there is some justification even if the justification is flawed)

The geriatrics algorithm computes minAge, which is the lower-bound for the age of the M pages that are likely to be replaced in the next epoch based on the ages of the pages in the epoch that just concluded. During the epoch, however, if a page X whose age was >= minAge in the previous epoch is accessed by a node, then X will no longer be a candidate for replacement for that node in this epoch.

(Kishore's note: +4 if the above sense is conveyed though not exactly how I have stated it;
Partial credit: +2 for what geriatrics algorithm actually computes; +2 for recognizing that a page deemed to be a replacement candidate is only a guess not an absolute)

(c) (5 points)
A new node N3 joins the GMS system which is currently in operation. What are the steps taken in GMS to integrate N3 into the system?
(Concise bullets please).

- Node N3 will notify the master node of the current GMS system of its arrival. (1 point)
- The master node will notify all the existing members about N3’s addition and will distribute new copies of the Page Ownership Directory (POD) to each node (including N3). (2 points)
- Each node will distribute portions of the Global Cache Directory (GCD) to N3. (2 points)

10. **(20 points, 15 minutes) (DSM/GMS)**
(a)
CS1: a critical section governed by lock L1
CS2: a critical section governed by lock L2
Memory consistency model: LRC

In the increasing time order shown below:

- Time T1: Process P1 in CS1 produces a diff for page X: Xd1
- Time T2: Process P2 in CS1 produces a diff for page X: Xd2
- Time T3: Process P3 in CS2 produces a diff for page X: Xd3

Process P4 wishes to execute CS1.
(i) (5 points) What should happen before P4 starts executing CS1? Justify your answer.
P4, at the point of acquiring the lock, it will invalidate the page X if it is locally present. Then it acquires the lock and starts executing CS1.

(+5 if the sense is conveyed)

(ii) (5 points) While inside CS1, P4 accesses page X. What should happen to ensure correct execution? Justify your answer.

- DSM will get the pristine copy of the page X from the owner for that page.
- DSM will get the diffs Xd1 and Xd2 from P1 and P2, respectively.
- DSM will apply the diffs in the order Xd1 followed by Xd2 to X to create an up-to-date copy of the page for use by P4.
- Note that change to X made by P3 in CS2 is not relevant for P4.

(Kishore’s note: full credit if the above sense is conveyed; -2 if the order of applying Xd1 and Xd2 not mentioned; -2 if the Xd3 is also applied to X)

(b) (5 points) (Answer True/False with justification. No credit without justification).
In Treadmarks, upon a page fault for a page X in Node N1, the DSM software on N1 broadcasts the virtual page number (VPN) of the faulting page to all the peer nodes.
False. (+1 if there is some justification even if flawed)

In Treadmarks, there is a manager node statically assigned for every VPN. The DSM software on N1 contacts the manager node for the faulting page X. (+4 if the above justification is conveyed)

(c) (5 points)
Inspired by the material they learned, alums of OMSCS 6210 decide to design a system that combines services offered by GMS and DSM in one integrated system. Give them your sketch of a design for integrating DSM and GMS in one unified system.
(Concise bullets please.)

(Kishore's note: This is an open-ended question, so no unique answer. If the answer seems reasonable then give full credit. Be generous on this question.)

Answer:
We will call our integrated design GMS++. It has the functionality of paging across the network and a simple page-based SC memory model for DSM using an invalidation-based coherence protocol.

- Start with a base design like GMS wherein the physical memory of each node is divided into local and global part and serves the paging needs to deal with memory pressure at a node.
- Incorporate the "geriatrics" algorithm and the integration of GMS++ into the OS for extracting age information for the pages.
- The system could be running simple processes and DSM processes.
- For DSM processes, like Treadmarks, divide the virtual address space and spread the management of the virtual pages evenly among all the nodes for maintenance of metadata relating to page sharing (single writer multiple readers).
- At process start-up, a DSM process will register its process-id with GMS++. "Thread-create" by a DSM process will be handled by GMS++ and will result in process creation on a remote node to represent the thread. GMS++ will register the process-id of the newly created process as a DSM process.
- Upon page fault, GMS++ will check if the faulting process is a DSM process. If not, the page fault will be handled in the way vanilla GMS does.

- Upon page fault by a DSM process, GSM++ will first contact the DSM manager for the page. The DSM manager will perform the necessary coherence action commensurate with the SC memory model and bring the page to the faulting node (informing the VM manager to make the faulting process runnable again).
- Page eviction for a non-DSM process will be handled the same way vanilla GMS does.
- Upon page eviction by a DSM process, GMS++ will contact the DSM software to check the state of the page. If the page is in read-mode and if there is at least one another actively read-sharing the page then the page will simply be dropped from the node updating the meta data in the manager. If the page is in write-mode, page eviction will be handled the same way as for non-DSM process.

(+1.5 for handling page faults; +1.5 for handling page evictions; +2 for handling coherence of shared pages)

11. **(10 points, 7 minutes) (DFS)**
Back in 1985, Sun Microsystems built the first Network File System and dubbed it NFS and that name has stood the test of time. To this day college campuses use derivatives of NFS for serving the file system needs of its clientele.

Answer True/False with justification for each of the following questions with reference to traditional NFS. No credit without justification.

(a)
Multiple network servers can provide file system service.

True: (+0.5 with some justification even if it is flawed)

NFS utilizes multiple network servers to store files and serve client requests, with each server being responsible for distinct and disjoint partitions of the entire file system.
(+2 or 0)

(b)
The network servers for the data (actual file content) and the metadata (information about client nodes that are using the file, etc.) for a given file are not necessarily the same.

False: (+0.5 with some justification even if it is flawed)
In NFS, the same server houses the file contents and the metadata related to that file. This is unlike xFS where metadata management is decoupled from the data management.
(+2 or 0)

(c)
Individual files are striped across the disks of multiple network servers on the Local Area Network (LAN).

False: (+0.5 with some justification even if it is flawed)
Each file is housed on a single server.
(+2 or 0)

(d)
A file cached at a client may be used to serve the needs of other network clients for the same file bypassing the network server that hosts the file on its disk.

False: (+0.5 with some justification even if it is flawed)
NFS does not keep track of client-side caching, and file caching is a feature of the server that hosts the file.
(+2 or 0)