Carnegie Mellon University in Qatar
15440 - Fall 2017

Problem Set 6

Out: November 20, 2017

Due: November 28, 2017

Question 1 [40 Points]

You have been nominated as the president and official web guru of DundeeAndMe.net, an unofficial fansite paying tribute to Paul Hogan and all things Crocodile Dundee. Due to the incredible growth of the site's popularity, you realized that having just a single web server can no longer handle the load. Hence, you decided to replicate the server, placing three replicas in different Australian cities (where the majority of the fan base resides), a single replica in southern Germany (where there is a small, but devoted cult), and a fifth replica in Boca Raton, Florida.

A. Assume you decided to use 2PC among the replicas for performing updates on site contents. List two advantages of using 2PC in this situation.

B. A freak tsunami destroyed the physical termination point of the main undersea cable connecting Australia and Asia. The Internet was effectively partitioned, separating Australia from the rest of the universe. Explain how this will affect users in Australia and in Germany, both when browsing the site as well as creating new blog entries.

C. Suppose you decided to use a Gifford-based voting scheme instead of 2PC to manage the replicas. In addition, assume a write quorum of 3 and an equal voting weight of 1 among all servers. Under the same partition conditions of part B, explain how users in Australia and in Germany will be affected, both when browsing the site as well as creating new blog entries.

D. Suppose you still wanted to employ Gifford voting with an equal weight of 1 across all servers, but now with a read quorum of 2. Again, under the same partition conditions of part B, explain how users in Australia and in Germany will be affected when browsing the site and creating new blog entries.

E. After some long daunting days, the undersea cable was finally restored, putting an end to this unpleasant network partition. During this time, however, the fortunate users who were able to continue using your site made several updates to the site's contents, which of course were not reflected on all replicas. Assuming a Gifford-based replica management scheme like the one discussed in part C, what recovery steps (if any) should you apply in order to ensure proper functioning of the replica set once network connectivity is restored?

Problem Set continues on the next pages

Question 2 [36 Points]

CNP bank uses a group of servers to coordinate and synchronize critical banking transactions. In particular, a Paxos-style protocol is employed on all the servers so as to achieve consensus on each operation. For any submitted operation, let $I \rightarrow J$ denote a successfully delivered message from server I to sever J. Recall the 4 types of messages that are used in Paxos, which can be written as follows:

- `Prepare(n)`, where `n` is a unique sequence (or round) number
- `Promise(n, (`n_k`, `a_k`))`, where n_k and a_k are the last sequence number and value, respectively accepted by Acceptor `k` (if any) and stored at its stable storage
- `PleaseAccept(n, v)`, where `v =` a_k of highest n_k seen among the promise responses, or any value if no promise response contained a past accepted proposal
- `Accept_OK()`

A. Assume 3 servers, S1, S2, and S3 are involved, and they all start out with no past accepted proposals (i.e., no sequence numbers and values are stored at their stable storages). A server can act as a Proposer, an Acceptor, or both. Suppose also that servers S1 and S2 submit proposals to quorums of Acceptors as shown in the Paxos communication trace below. What are the quorum sizes of S1 and S2, assuming that both will not submit prepare messages other than what is shown in the trace? Also, which proposal (or proposals) will achieve consensus? Explain your reasoning.

S1 → S1:	`Prepare(101)`
S1 → S1:	`Promise(101, null)`
S1 → S2:	`Prepare(101)`
S1 → S3:	`Prepare(101)`
S2 → S1:	`Promise(101, null)`
S2 → S3:	`Prepare(102)`
S3 → S2:	`Promise(102, null)`
S2 → S1:	`Prepare(102)`
	...

Question 2 continues on the next page

B. Assume now 5 servers, S1, S2, S3, S4, and S5 are involved, and also starting out with no past accepted proposals. As in part A, a server can act as a Proposer, an Acceptor, or both. Suppose also that servers S1 and S4 submit proposals to quorums of Acceptors as shown in the Paxos communication trace below. Which proposal (or proposals) will achieve a consensus in this case, assuming the [EVENT!] in the trace is S1 crashing? Explain your reasoning.

S1 → S1:	`Prepare(101)`
S1 → S1:	`Promise(101, null)`
S1 → S2:	`Prepare(101)`
S2 → S1:	`Promise(101, null)`
S4 → S4:	`Prepare(104)`
S4 → S5:	`Prepare(104)`
S4 → S4:	`Promise(104, null)`
S5 → S4:	`Promise(104, null)`
S1 → S3:	`Prepare(101)`
S3 → S1:	`Promise(101, null)`
S1 → S1:	`PleaseAccept(101, X)`
S1 → S1:	`Accept_OK()`
S4 → S1:	`prepare(104)`
S1 → S2:	`PleaseAccept(101, X)`
S1 → S3:	`PleaseAccept(101, X)`
	[EVENT!]
...	

Question 3 [24 Points]

Assume that the following four processes belong to the same group, with two different senders and a possible delivery order of messages under FIFO-ordered multi-casting:

Process P1	**Process P2**	**Process P3**	**Process P4**
Sends M1	Receives M1	Receives M3	Sends M3
Sends M2	Receives M3	Receives M1	Sends M4
Receives M2	Receives M2		
Receives M4	Receives M4		

A. How many delivery orderings are permissible with atomic multicasting?

B. Write down all the permissible delivery orderings with FIFO atomic multicasting.

C. Assume M1 causally precedes M4. Write down all the permissible delivery order-ings with causal atomic multicasting.