CS425 Fall 2024 – Homework 2

(a.k.a. "Once upon a time in Distributed Hollywood")

Out: Sep 19, 2024. Due: 2 pm US Central on Oct 6 (Sunday!), 2024

Topics: P2P Systems, Key-value Stores, Time and Ordering (Lectures 7-12)

Instructions:

1. **Attempt any 8 out of the 10 problems** in this homework (regardless of how many credits you're taking the course for). If you attempt more, we will grade only the first 8 solutions that appear in your homework (and ignore the rest). Choose wisely!
2. Please hand in **solutions that are typed** (you may use your favorite word processor. We will not accept handwritten solutions. Figures (e.g., timeline questions) and equations (if any) may be drawn by hand (and scanned).
3. **All students (On-campus and Online/Coursera)** – Please submit PDF only! Please submit on Gradescope. [https://www.gradescope.com/]
4. Please **start each problem on a fresh page**, and **type your name at the top of each page**. **And on Gradescope please tag each page with the problem number!**
5. Homeworks will be **due at time and date noted above. No extensions. For DRES students only:** once the solutions are posted (typically a few hours after the HW is due), subsequent submissions will get a zero**. All non-DRES students must submit by the deadline time+date.**
6. Each problem has the same grade value as the others (10 points each).
7. Unless otherwise specified in the question, the only resources you can avail of in your HWs are the provided course materials (slides, textbooks, etc.), and communication with instructor/TA via discussion forum and e-mail.
8. You can discuss lecture concepts and the questions on Piazza and with your friends, but you cannot discuss solutions or ideas on Piazza.

Prologue: You have just been made the technical head in a production company that is producing a new Hollywood movie. The movie is sure to be a blockbuster, with a lot of well-known actors and actresses hired to star in it. Amazingly many of them know distributed systems! You run into them every day on the set. Here is what ensues.

All characters and their actions used in this homework are meant to make the homework fun! Any resemblance of their actions or opinions to real events, or places, is purely coincidental. Any stories involving real actors or actresses are fictional.

Problems:

1. Walt Disney and Pokemon want to build a virtual theme park where young customers will all wear VR goggles and be connected via a Gnutella P2P system. They want to be very precise about their routing. They structure their Gnutella system as a perfect balanced binary tree with $N=2^m-1$ processes. The P2P system's nodes are arranged like a binary tree. All leaves are at the same level. Additionally, each node (at each level) knows how to "poke" one additional neighbor – each node has (apart from its 1 parent and 2 children neighbors) ONE additional sibling neighbor: its immediate "successor" node: for a left child, its successor is its immediate right sibling; while for a right child, its successor is its next "cousin" sibling to its immediate right. These sibling links DO NOT wrap around, i.e., the rightmost node (at any level of the tree) has no sibling neighbor. Note that sibling neighbors are bidirectional and exist at each of the m levels of the tree. You can assume m is quite large ($m > 20$). For any given m, answer the following questions:
    a. What is the minimal TTL required by a query originating at the root for all nodes to receive its query?
    b. What is the minimal TTL required by a query originating at the leftmost leaf for all nodes to receive its query?
    c. What is the minimal TTL required by a query originating at the rightmost leaf for all nodes to receive its query?

    We recommend that you try to solve the question with small values of m first, and then generalize. We need answers to the above questions in terms of the variable m.

2. You want to make a movie called USA @ Paris 2024. You have created the movie's core team. The core team is responsible for putting together the rest of the cast and crew. To facilitate the search, you want to build a peer-to-peer network to connect the team with aspiring artists. Inspired by the Olympic emblem of overlapping rings, one of your ideas is to build a new P2P DHT called "Olympic" based on the "OR" metric. Olympic contains TWO rings with ring R1 containing IDs 0 to 2^{m-1} -1 and ring R2 containing 2^{m-1} to 2^m -1 (m large). You design a routing algorithm but with a twist. Just like in Chord, a file is hashed on to a point in the range $[0, 2^m-1]$ and stored at the first node at or to the clockwise

of the ring. However, each node is hashed only to the range [0, 2^{m-1}-1]. Say a node is hashed to a point N in [0, 2^{m-1}-1], and this lies in R1. Then the same node (machine) also appears as a second node in the second ring R2 at ID N2=2^m-N-1. (Note that R2 essentially reverses the order of nodes from N1). Each of the nodes N and N2 maintain a successor and a predecessor routing tables, just like Chord does, in each respective ring R1 and R2 (So, R1 is a Chord ring with 2^{m-1} points, and R2 is a Chord ring 2^{m-1} points). When a query is originated, it is routed in R1 using Chord routing rules. However, the query is permitted one "short circuit" from ring R1 to ring R2: when a node N receives the query Q intended for destination D, then N can "forward" Q to N2 (essentially, to itself) and continue routing in R2. However, this short circuit is permitted only once for the lifetime of that query's routing (i.e., during its path). Routing in each of R1 and R2 occurs only clockwise using finger tables and successor (predecessor is not used for routing).

   a. Devise an algorithm to exploit the "dual ring" nature of R1-R2 to achieve a lower routing latency (in hops) than if only R1 had been present (i.e., vs. simple Chord in a ring of 2^{m-1} points).
   b. Calculate the asymptotic latency of your Olympic routing algorithm.
   c. Can you modify your algorithm so that your latency is asymptotically lower than Chord's in a ring of 2^{m-1} points? If so, show it. If not, argue why it is impossible to devise such a routing for the given R1-R2 setup.

3. Three wealthy industrialists, Lo Skum, Frank Bozo, and Sugar Mountain, all want to make a movie aboard a very unsafe spaceship called "Solargate" that travels to all 8 planets (go figure!). Anyway, you are responsible for managing the spaceship as long as it is active. Lo and behold, the different planets use BitTorrent! While working through an example, you find a case where a file has 6 shards: A, B, C, D, E, F. The querying node (peer, leecher) is on a spacecraft, connected to all the planets directly (the SBM group figured out a way to have high bandwidth, low latency communication that beats the speed of light!). The planet servers at the 8 planets each have the following shards: Mercury (AEF), Venus (BCDE), Earth (ABDEF), Mars (ABCDE), Jupiter (ABDE), Saturn (ABCD), Uranus (AB), Neptune (ADE).
   a. Which shard will be fetched *first*?
   b. What is the *order* in which shards will be fetched by the querying node? (assume that no shards are being fetched by any other nodes). If there are ties, you can use the alphabetically lower one.

c. If one introduced a 9th renegade planet called Pluto (and the spacecraft was connected to it), assign a set of just one shard to Pluto that changes the first fetched shard by the spacecraft (ties broken by lower letter)?

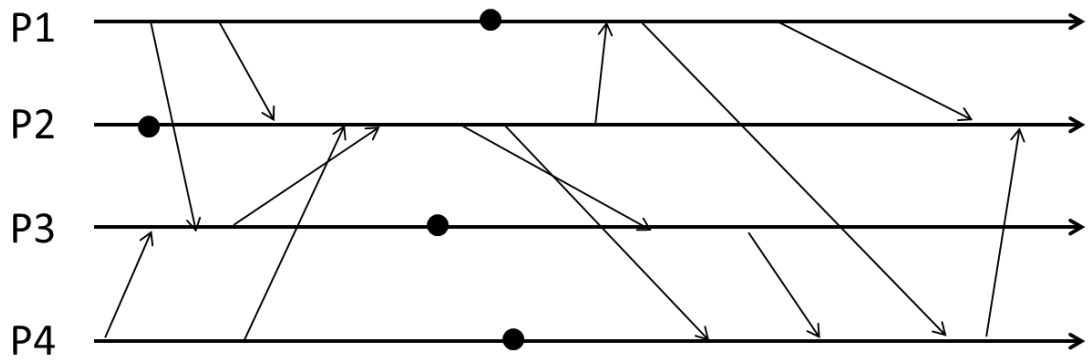
4. One of the Producers, Leo Bloom, and his star Orlando Bloom, both like Bloom filters. But being a producer, he wants to create a new Bloom filter-based data structure. A (regular) Bloom filter's false positive rate is given as $\left(1 - e^{-\frac{kn}{m}}\right)^k$ where k is the number of hash functions, n is the size of the input set and m is the size of the Bloom filter in bits. Leo says that instead of using a single Bloom filter B with 1024 bits and 4 hash functions, his new datastructure, called Leo Bloom filter, uses 4 Bloom filters B1, B2, B3, and B4, each with 256 bits, and each using 1 hash function (each hash function different from each other, and different from the above 4 hash functions). There are two variants: When checking for an item, it returns true only if the item is present in {Variant A: *all*, Variant B: *any*} of B1, B2, B3, and B4. When inserting an item, for both variants A and B, it is inserted into *all* of B1, B2, B3, and B4. Which of the above three approaches—original using Bloom filter B vs. Leo–any vs. Leo–all —gives the best (lowest) false positive rates? Answer this for two cases: (1) when there are typically 10 elements inserted into the datastructure, (2) when there are typically 1000 elements inserted into the datastructure. (We recommend though, that you solve the problem with the variables *k, n, m, etc.,* and then apply these values. But solving with only these two values of 10, 1000, would be ok as well.)

5. A big-budget movie is being directed by N directors. The N directors never seem to agree on anything about the movie. To solve this, the producer decides to implement a quorum approach with fixed-size quorums of size Q. The requirement is that any two arbitrary quorum sets must intersect in at least K directors with each other. There are N total directors. For each of the following cases, what should the minimum quorum size be in order to satisfy this requirement?
   a. N=100, K=1
   b. K=1 (any N)
   c. K=10 (any N)
   d. K=N/2
   e. K=3N/4
   f. K=N

6. The director, C. Nolan, likes to deal with time travel, which means he asks a lot of "What if?" questions. He asks you several questions about Cassandra. For each of these, give at least one disadvantage:
    a. What if Cassandra used finger tables (like Chord) for routing reads and writes?
    b. What if there were no Bloom filter (but the rest of the system works the same way)?
    c. What if Cassandra did not use Memtable at all, and instead wrote directly into the "latest" SSTable? (but the rest of the system works the same way)
    d. What if Cassandra didn't use tombstones for deletes and instead deleted records directly from Memtable and SStables?

7. An overly-religious person suddenly wants to fund the movie, and they only worship Cristian's algorithm. They find that the round-trip time for one round of synchronization messages is exactly 1.01 ms. They are trying to calculate the error for this run, and so they calculate some minimum delays. On the server side, they find that there is a delay of at least 13.7 microseconds for a packet to get from an application to the network interface, and the delay on the opposite path (network interface to application buffer) is at least 0.23 ms. On the client side, they measure that the minimum time to get from the network interface to the application buffer is at least 20 microseconds, and the minimum time on the reverse path is X microseconds. But they forget to measure X. What is the error, given the data just presented?

8. Spiderman and his doppelgangers in the metaverse are trying to figure out the communication among the different metaverses. They have the following timeline of messages exchanged, where each dot represents an instruction. Can you mark *Lamport timestamps* on each event? It is ok to print out this and hand-draw/write the timestamps (then scan or photograph your solution, and insert it into your solution doc).



9. Whoops, the simultaneous and sudden arrival of all of the concurrent villains (Doc Ock, Green Goblin, Electro, Sandman, and the others) means Lamport timestamps are unusable for distinguishing concurrent events from causal events. Can you mark *vector timestamps* for the same timeline as in the previous question? It is ok to print out this and hand-draw/write the timestamps (then scan or photograph your solution, and insert it into your solution doc).

10. The movie is a hit! The breakout star of the movie, Kristen, was so happy with your work that she has asked the production company to give you one last puzzle to solve before you can be paid the millions of $ you are owed. The puzzle concerns timestamps.
    a. Consider a modified version of the *Lamport timestamp* marking algorithm. Instead of setting the initial value of local clock to 0, each process initializes its clock to some *arbitrary* value. The processes then assign timestamps the same way as the original algorithm. Would the assigned timestamps still preserve all the properties of Lamport timestamps? Describe why or why not.
    b. Consider a modified version of the *vector timestamp* marking algorithm. Instead of initializing the vector of clocks to all 0s, each process initializes the vector clock to *a vector of arbitrary values*. The processes then assign timestamps the same way as the original algorithm. Would the assigned timestamps still preserve all the properties of vector timestamps? Describe why or why not.

====== END OF HOMEWORK 2 ====