Covers: lectures 13 - 24
20 multiple choice
2 free response- 1st like project 4 and 2nd like project 5
Can bring one double-sided normal-sized sheet of paper. No calculator.
Kloosterman slides
DeOrio slides

Lecture 13- Operating Systems and Parallelism

- A process is a program in execution; has its own address space
- Each CPU core can run one process at a time, every ~1-10 ms the OS can switch processes
- Threads are multiple functions in one process running simultaneously; each has its own stack (but same heap, code, etc.- lower overhead than processes)
- Thread.join() waits for Thread to finish executing before continuing to execute code in the thread that called Thread.join()
- lock.acquire() and lock.release() ensure lines of code in a particular thread are executed contiguously with no code from other threads executed in between (don't need to know this for 485 though)

Lecture 14- Networking

- Every internet device (servers, home routers, etc.) has an IP address
- Routers connect networks
- Messages broken up into packets and sent
- TCP ensures all packets received and re-arranged in order
- Packet sent. Client waits for Ack. If none received before timeout, re-sends.
- Flow control > Receiver window and congestion control > congestion window restrict packet flow -> TCP window = min(receiver window, congestion window)
- UDP lets application decide what to do about dropped or out-of-order packets
- Use TCP or UDP through sockets. Use multiple sockets to run multiple different network processes on one host. Tell sockets apart with their port number.

Lecture 15- Text Analysis for Web Search

- Define each document by a vector of booleans, each boolean corresponding to whether it contains a given term
- Query is a vector too. Rank results by how close results are to the query (vector dot product)
- Term frequency (TF): how many times term appears in a document
- Document frequency: how many times term appears in document collection
- Inverse document frequency (IDF): (total # of documents) / (# of documents with term)
- TF-IDF normalization (W): [TF * log(IDF)] / [sqrt of the sum of TF^2 * log(IDF)^2 for every document]
- Similarity of documents 1 and 2 is the sum of W_1 * W_2 for every term
- Precision: % of selected docs that are right, Recall: % of right docs selected

Lecture 16- Link Analysis for Web Search

- Make a graph with websites as nodes, and links to other websites as edges

- PageRank: start at random website, click random link on that website, repeat. Pages with more visits are more important
- More websites link to me > more important; important website links to me > more important
- HITS: authority score (value of content, based on others' hub scores) and hub score (how good at linking, based on others' authority scores)

Lecture 17- Scaling Search

- Make web crawlers to find lots of sites and generate an inverted index (map words to docs). About 10% of pages indexed; 90% are deep web and cannot be
- Check page has changed using page hash.
- Check for page similarity with jaccard similarity- percentage of shared words. To improve performance:
    - Compare word hashes (numbers faster than strings)
    - Only compare a few random words from each doc rather than every word
    - Compare the *k* smallest hashes
- For better accuracy, use shingles- groups of words so that order kind of matters
- Inverted index is huge; use parallel query processing

Lecture 18- Scaling Static Pages

- Infrastructure as a service- rent a computer in someone else's data center (AWS EC2)
- Platform as a service- same but they manage the software too (Microsoft Azure SQL)
- Software as a service- rent a web app built, hosted, and maintained by someone else (gmail, github)
- DNS translates website names to IP addresses
- CDNs- allow faster loading of content

Lecture 19- Scaling Dynamic Pages

- Multiple servers, with multiple cores, running multiple processes that are running multiple threads
- DNS > Load Balancer (there can be multiple of these) > one of many servers
- Hardware virtualization- one physical computer runs multiple Operating Systems
    - Advantages: efficiency, environment isolation, scaling (start more VMs if site gets high traffic)
    - Disadvantages: slow start, high overhead
- Containerization- isolate code and dependencies but share Operating System
    - Addresses slow start and high overhead, but all containers must run on the host Operating System
- Can combine these

Lecture 20- Scaling Storage

- Distributed database models: sharding and replication
- Sharding- store some data on one database, store other data on some other database
    - Easier consistency, but bottleneck concerns
- Replication- store all data on multiple databases
    - Moments of inconsistency, better throughput
- CAP theorem: cannot have all of Consistency, Availability, and Partition tolerance in a distributed database

        - Basically, when a network failure AKA partition, occurs, must return either an error or incorrect value
    - For media upload, either have dynamic page server act as a middleman or directly to distributed file system

Lecture 21- Recommender Systems

- Collect lots of data, use it to try to maximize something
- User-based filtering- recommend based on what others thought. Averaging vs. nearest neighbor
- Calculate nearest neighbor with euclidean distance, cosine similarity, pearson coefficient, tf-idf, etc.
- k-NN = select k closest neighbors, select most frequent score
- Content-based filtering- recommend items similar to what user has liked in the past
- Hybrid-filtering- both user and content based

Lecture 22- Ads and Auctions

- Ads sold to highest bidder
- On the web: auctions primarily used to sell just-in-time ads automatically
- Types: Ascending (English), Descending (Dutch), sealed-bid
- Can write sniping software- bot submits high bid just before auction end

Lecture 23- Blockchain

- Useful when database is shared with multiple writers, users don't trust each other or a central authority

Lecture 24- Dark Web

- Accessible only through an anonymous connection- Tor
- Encryption and other stuff solves issue of private message content, this solves issue of private message metadata
- With a VPN: strong encryption between you and VPN, not strong between VPN and destination. Eavesdropper can see you are talking with a VPN
- You must trust the VPN. Can't be sure they're honest. Vulnerable to traffic analysis
- Tor: route your traffic through a path of random volunteer nodes, each with its own encryption. Each node only knows the node before it and the node after it.
- Like VPNs, people can see you're using Tor