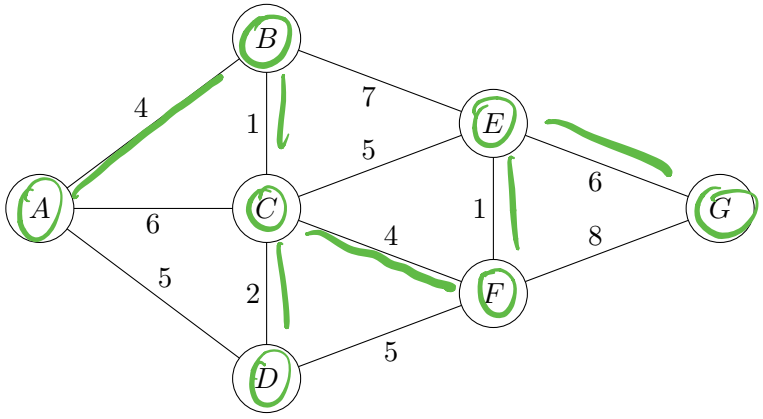
CSM 61B Fall 2020

Graphs Sorting

Mentoring 11: November 2, 2020

1 Networking

1.1 Consider the telephone network from last week. Construct a minimum spanning tree by running Prim's Algorithm from node A.



Cut property
- if you split V's into two groups, pick shortest edge between them
- ↑ crossing edge

Prim's

2 Sorting Overview

So far, we've learned a few different types of basic sorting algorithms. While sorting might seem like a simple idea, there are many real-world applications of sorting, and several different algorithms that we can use depending on the situation.

In the table below, fill out the best and worst-case runtimes for each of the sorting algorithms provided.

Sorting Algorithms ft. [1,4,3,5,2]

Selection Sort

- repeatedly find min of array

sorted | Unsorted

1. [1] } [4,3,5,2]
2. [1,2] } [4,3,5]
3. [1,2,3] } [4,5]
4. [1,2,3,4] } [5]
5. [1,2,3,4,5]

Insertion Sort

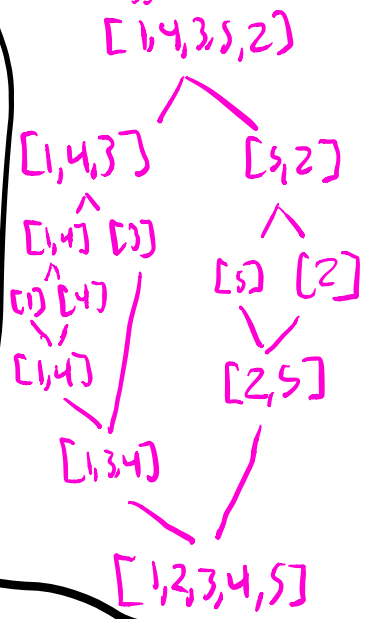
- swap items by inserting each item at its proper location in sorted portion of array

Sorted | Unsorted

1. [1] } [4,3,5,2]
2. [1,4] } [3,5,2]
3. [1,3,4] } [5,2]
4. [1,3,4,5] } [2]
5. [1,2,3,4,5]

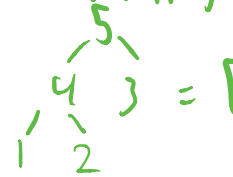
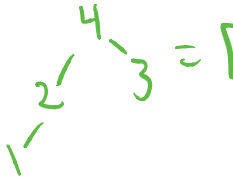
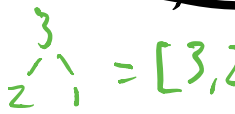
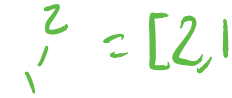
Merge Sort

- Divide & Conquer
- Recursively mergesort 2 halves of the array
- uses 'merge' operation that merges two sorted subarrays



Heap Sort

- Create a max-heap out of the items
- Continue popping off the largest element

1.  = [5,4,3,1,2]
2.  = [4,2,3,1] } [5]
3.  = [3,2,1] } [4,5]
4.  = [2,1] } [3,4,5]
5. 1 = [1] } [2,3,4,5]
6. [1,2,3,4,5]

Quicksort

- Pick a pivot (usually based on some pivot-picking strategy, usually given)
- Group the array into 3 parts: 1. < pivot
  2. == pivot
  3. > pivot
- Recursively quicksort groups #1. and #3.

* let the first element be our pivot

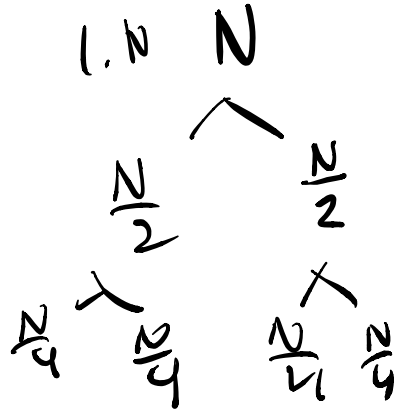
1. [ ] [0] [4, 3, 5, 2]

empty group ↗

2. [ ] [3, 2] [4] [5]

3. [ ] [2] [3] [4] [5]

	Algorithm	Best-case	Worst-case
RS	Selection Sort	$O(N^2)$	$O(N^2)$
JS	Insertion Sort	$O(N)$	$O(N^2)$ (already ordered) (reversed)
JK	Merge Sort	$O(N \log N)$	$O(N \log N)$
LQ	Heapsort	$O(N)$	$O(N \log N)$
SS	Quicksort	$O(N \log N)$	$O(N^2)$ based on pivot



2.1 Give a best and worst case input for insertion sort.

best: sorted
worst: reversed

2.2 Do you expect selection or insertion sort to run more quickly on a reverse list?

both $O(N^2)$

2.3 In Heapsort do we use a min-heap or max-heap? Why?

Max, do it 'in-place' w/out having to reverse output array

2.4 Sort the following array using Heap Sort. [3, 2, 1, 5, 6, 8, 7]

2.5 Run the quicksort algorithm. Assume we pick the middle element as the pivot; if there is no exact middle, pick the element to the right of the middle.

```
{ 1, 3, 8, 2, 6, 4, 5, 9 }
```

3 Stability

Stability is a property of some sorting algorithms. Stability essentially means that if we have two elements that are equal, then their relative ordering in the sorted list is the same as the ordering in the unsorted list. For instance, let's say that we had an array of integers.

```
{ 1, 2, 1, 3, 1, 2, 4 }
```

Since we have multiple 1 and 2s, let's label these.

```
{ 1A, 2A, 1B, 3, 1C, 2B, 4 }
```

A stable sort would result in the final list being

```
{ 1A, 1B, 1C, 2A, 2B, 3, 4 }
```

Why is this desirable? Say that we have an Excel spreadsheet where we are recording the names of people who log in to CSM Scheduler. The first column contains the timestamps, and the second column contains their username. The timestamps are already ordered in increasing order. If we wanted to sort the username, so that we could group the list to see when each username logs in, we would want that the timestamps maintain their relative order. This is precisely what a stable sort ensures.

3.1 Why does Java's built-in `Array.sort` method use quicksort for **`int`**, **`long`**, **`char`**, or other primitive arrays, but merge sort for all `Object` arrays?

4 *In'sort' Meme Here*

4.1 Each column below gives the contents of a list at some step during sorting. Match each column with its corresponding algorithm.

· Merge sort · Quicksort · Heap sort · LSD radix sort · MSD radix sort

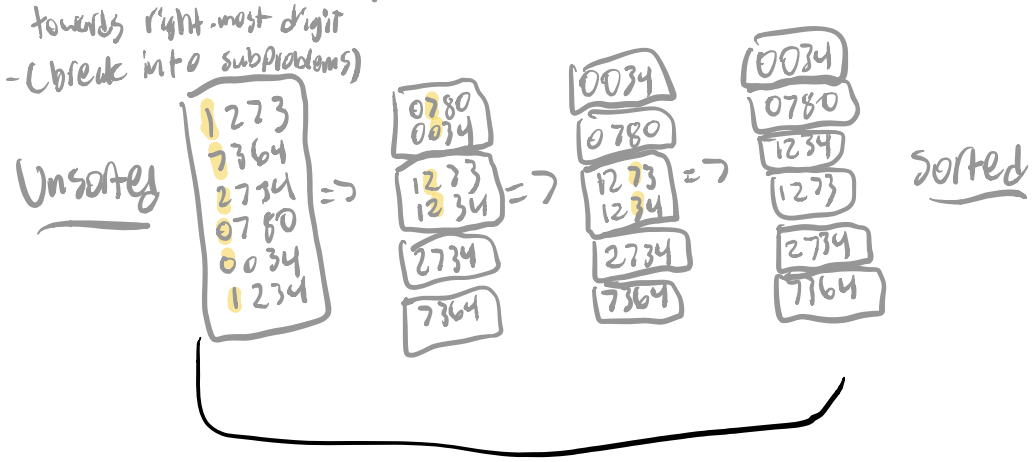
For quicksort, choose the topmost element as the pivot. Use the recursive (top-down) implementation of merge sort.

	Start	A	B	C	D	E	Sorted
1	4873	1876	1874	1626	9573	2212	1626
2	1874	1874	1626	1874	7121	8917	1874
3	8917	2212	1876	1876	9132	7121	1876
4	1626	1626	1897	4873	6973	1626	1897
5	4982	3492	2212	4982	4982	9132	2212
6	9132	1897	3492	8917	8917	6152	3492
7	9573	4873	4873	9132	6152	4873	4873
8	1876	9573	4982	9573	1876	9573	4982
9	6973	6973	6973	1897	1626	6973	6152
10	1897	9132	6152	3492	1897	1874	6973
11	9587	9587	7121	6973	1874	1876	7121
12	3492	4982	8917	9587	3492	9877	8917
13	9877	9877	9132	2212	4873	4982	9132
14	2212	8917	9573	6152	2212	9587	9573
15	6152	6152	9587	7121	9587	3492	9587
16	7121	7121	9877	9877	9877	1897	9877

Array = [1273, 7364, 2734, 780, 34, 1234]

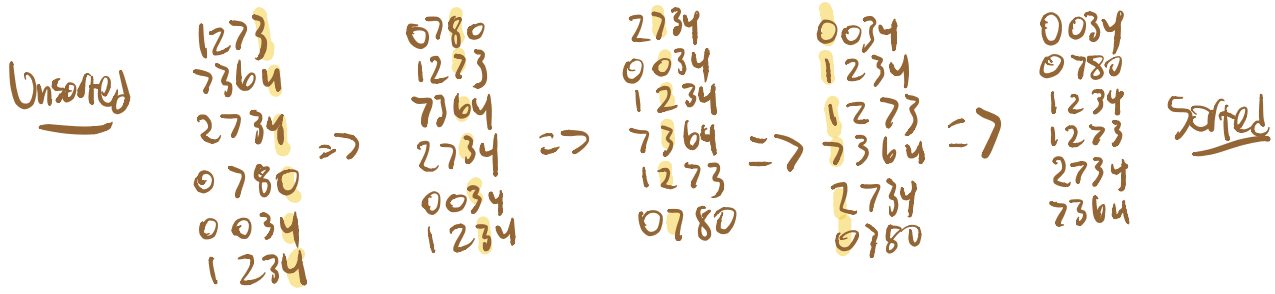
MSD Radix Sort

- Sort from left-most digit towards right-most digit
- (break into subproblems)



LSD Radix Sort

- Sort from right-most digit towards left-most digit
- Can't break into subproblems



5 Sorting Out My Head!

5.1 Web developers use many different sorts for the different types of lists that they might want to sort. For each of these, provide the best sorting algorithm amongst the following: Mergesort, Quicksort (with Hoare Partitioning), Insertion Sort, LSD Sort. Also, state the worst-case runtime.

(a) A list of N packets received by a server over time. Each packet has the timestamp at which the sender sent it. However, some packets may be dropped or arrive out-of-order due to the faulty network. Sort this list by that timestamp (sent time).

(b) A list of N websites. Each website has the number of total visitors. Sort this list by visitor count.

(c) After sorting by visitor count, we now want to sort by webpage file size. If websites have the same file size, they should be ordered by visitor count.

(d) A list of 20 names. Sort in alphabetical order.