CS221: Algorithms and Data Structures Lecture #3.5 Sorting Takes Priority

Steve Wolfman
2014W1

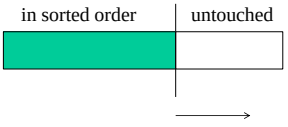
Today's Outline

- Sorting with Priority Queues, Three Ways

Quick Review of Sorts

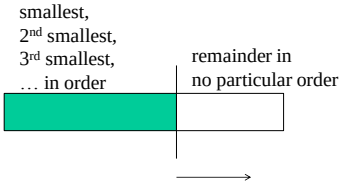
- Insertion Sort: Keep a list of already sorted elements. One by one, insert new elements into the right place in the sorted list.
- Selection Sort: Repeatedly find the smallest (or largest) element and put it in the next slot (where it belongs in the final sorted list).
- Merge Sort: Divide the list in half, sort the halves, merge them back together. (Base case: length ≤ 1.)

Insertion Sort Invariant



each iteration moves the line right one element

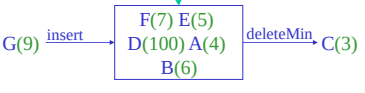
Selection Sort Invariant



each iteration moves the line right one element

How Do We Sort with a Priority Queue?

You have a bunch of data.
You want to sort by priority.
You have a priority queue.
WHAT DO YOU DO?



"PQSort"
(Super-Ridiculously Vague Pseudocode)

```
Sort(elts):
  pq = new PQ
  for each elt in elts:
    pq.insert(elt); // or all at once if that's easier
  sortedElts = new array of size elts.length
  for i = 0 to elts.length - 1:
    sortedElts[i] = pq.deleteMin  // or all at once
  return sortedElts
```

What sorting algorithm is this?

a. Insertion Sort
b. Selection Sort
c. Heap Sort
d. Merge Sort
e. None of these

Reminder: Naïve Priority Q Data Structures

- Unsorted array (or linked list):
  - *insert:* worst case O(1)
  - *deleteMin:* worst case O(n)
- Sorted array:
  - *insert:* worst case O(n)
  - *deleteMin:* worst case O(1)

"PQSort" deleteMaxes with Unsorted List **MAX**-PQ

0	1	2	3	4	5	6	7	8	9	10	11	12	13
5	9	4	8	1	6	10	12	13	2	3	14	20	7

How long does inserting all of these elements take?

And then the deletions…

"PQSort" deleteMaxes with Unsorted List MAX-PQ

0	1	2	3	4	5	6	7	8	9	10	11	12	13
5	9	4	8	1	6	10	12	13	2	3	14	20	7

PQ

"PQSort" deleteMaxes with Unsorted List MAX-PQ

0	1	2	3	4	5	6	7	8	9	10	11	12	13
5	9	4	8	1	6	10	12	13	2	3	14	20	7

PQ

5	9	4	8	1	6	10	12	13	2	3	14	7	20

PQ Result

"PQSort" deleteMaxes with Unsorted List MAX-PQ

0	1	2	3	4	5	6	7	8	9	10	11	12	13
5	9	4	8	1	6	10	12	13	2	3	14	20	7

PQ

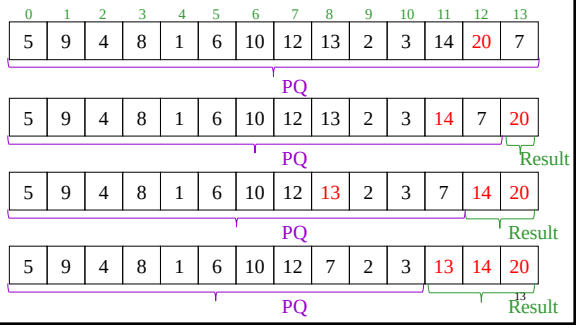
5	9	4	8	1	6	10	12	13	2	3	14	7	20

PQ Result

5	9	4	8	1	6	10	12	13	2	3	7	14	20

PQ Result

"PQSort" deleteMaxes with Unsorted List MAX-PQ



Two PQSort Tricks

1) Use the array to store both your results and your PQ. No extra memory needed!

2) Use a max-heap to sort in increasing order (or a min-heap to sort in decreasing order) so your heap doesn't "move" during deletions.

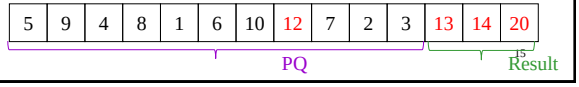
"PQSort" deleteMaxes with Unsorted List MAX-PQ

How long does "build" take? No time at all!

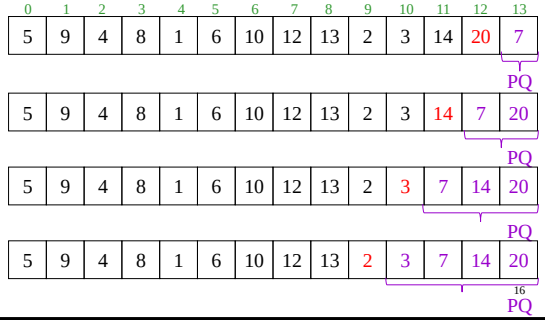
How long do the deletions take? Worst case: $O(n^2)$ ☹

What algorithm is this?

a. Insertion Sort
b. Selection Sort
c. Heap Sort
d. Merge Sort
e. None of these



"PQSort" insertions with Sorted List MAX-PQ



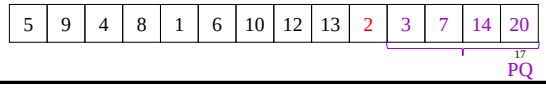
"PQSort" insertions with Sorted List MAX-PQ

How long does "build" take? Worst case: $O(n^2)$ ☹

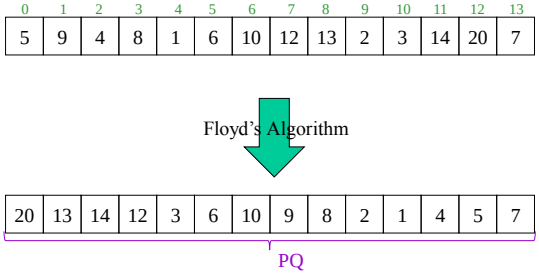
How long do the deletions take?

What algorithm is this?

a. Insertion Sort
b. Selection Sort
c. Heap Sort
d. Merge Sort
e. None of these



"PQSort" Build with Heap MAX-PQ



Takes only O(n) time!

"PQSort" deleteMaxes with Heap MAX-PQ

0	1	2	3	4	5	6	7	8	9	10	11	12	13
20	13	14	12	3	6	10	9	8	2	1	4	5	7

PQ

14	13	10	12	3	6	7	9	8	2	1	4	5	20

PQ

13	12	10	9	3	6	7	5	8	2	1	4	14	20

PQ

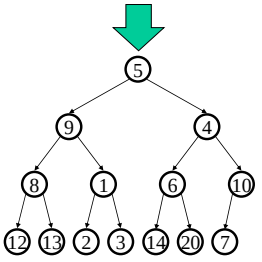
12	9	10	8	3	6	7	5	4	2	1	13	14	20

PQ

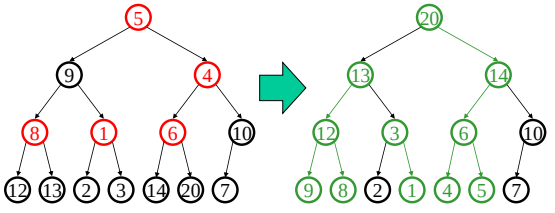
Totally incomprehensible as an array!

"PQSort" deleteMaxes with Heap MAX-PQ

5	9	4	8	1	6	10	12	13	2	3	14	20	7



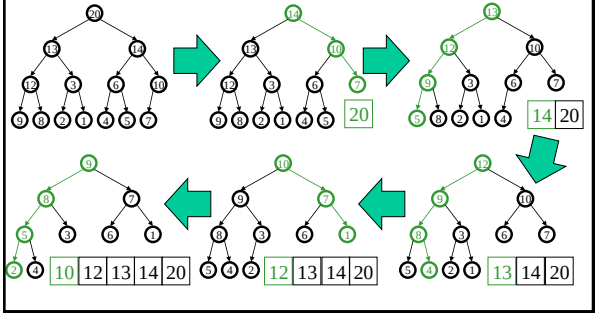
"PQSort" deleteMaxes with Heap MAX-PQ



Build Heap

Note: 9 ends up being perc'd down as well since its invariant is violated by the time we reach it.

"PQSort" deleteMaxes with Heap MAX-PQ



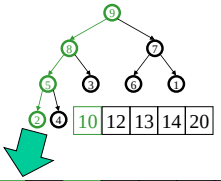
"PQSort" with Heap MAX-PQ

How long does "build" take? Worst case: O(n) ☺

How long do the deletions take? Worst case: O(n lg n) ☺

What algorithm is this?

a. Insertion Sort
b. Selection Sort
c. Heap Sort
d. Merge Sort
e. None of these



9	8	7	5	3	6	1	2	4	10	12	13	14	20

PQ | Result

"PQSort"

```
Sort(elts):
  pq = new PQ
  for each elt in elts:
    pq.insert(elt);
  sortedElts = new array of size elts.length
  for i = 0 to elements.length - 1:
    sortedElts[i] = pq.deleteMin
  return sortedElts
```

What sorting algorithm is this?

a. Insertion Sort
b. Selection Sort
c. Heap Sort
d. Merge Sort
e. None of these

To Do

- Read: Epp Section 9.5 and KW Section 10.1, 10.4, and 10.7-10.10

25

Coming Up

- More sorting!

26