Priority Queues

Two kinds of priority queues:
- Min priority queue.
- Max priority queue.

Min Priority Queue

- Collection of elements.
- Each element has a priority or key.
- Supports following operations:
  - isEmpty
  - size
  - add/put an element into the priority queue
  - get element with min priority
  - remove element with min priority

Complexity Of Operations

Two good implementations are heaps and leftist trees.

isEmpty, size, and get => O(1) time
put and remove => O(log n) time
where n is the size of the priority queue

Applications

Sorting
- use element key as priority
- put elements to be sorted into a priority queue
- extract elements in priority order
  - if a min priority queue is used, elements are extracted in ascending order of priority (or key)
  - if a max priority queue is used, elements are extracted in descending order of priority (or key)

Sorting Example

Sort five elements whose keys are 6, 8, 2, 4, 1 using a max priority queue.
- Put the five elements into a max priority queue.
- Do five remove max operations placing removed elements into the sorted array from right to left.
After Putting Into Max Priority Queue

Sorted Array

Max Priority Queue

8
4
6
1
2

After First Remove Max Operation

Sorted Array

Max Priority Queue

1
4
6
2

After Second Remove Max Operation

Sorted Array

Max Priority Queue

1
4
2

After Third Remove Max Operation

Sorted Array

Max Priority Queue

1
2

After Fourth Remove Max Operation

Sorted Array

Max Priority Queue

1

After Fifth Remove Max Operation

Sorted Array

Max Priority Queue

1
2
4
6
8
Complexity Of Sorting

Sort \( n \) elements.
- \( n \) put operations \( \Rightarrow O(n \log n) \) time.
- \( n \) remove max operations \( \Rightarrow O(n \log n) \) time.
- total time is \( O(n \log n) \).
- compare with \( O(n^2) \) for sort methods of Chapter 2.

Heap Sort

Uses a max priority queue that is implemented as a heap.

Initial put operations are replaced by a heap initialization step that takes \( O(n) \) time.

Machine Scheduling

- \( m \) identical machines (drill press, cutter, sander, etc.)
- \( n \) jobs/tasks to be performed
- assign jobs to machines so that the time at which the last job completes is minimum

Machine Scheduling Example

3 machines and 7 jobs
job times are [6, 2, 3, 5, 10, 7, 14]
possible schedule

LPT Schedules

Longest Processing Time first.
Jobs are scheduled in the order 14, 10, 7, 6, 5, 3, 2
Each job is scheduled on the machine on which it finishes earliest.

Machine Scheduling Example

Finish time = 21
Objective: Find schedules with minimum finish time.
LPT Schedule

\[ [14, 10, 7, 6, 5, 3, 2] \]

Finish time is 16!

LPT Schedule

- LPT rule does not guarantee minimum finish time schedules.
- \((LPT \text{ Finish Time})/(Minimum \text{ Finish Time}) \leq 4/3 - 1/(3m)\) where \(m\) is number of machines.
- Usually LPT finish time is much closer to minimum finish time.
- Minimum finish time scheduling is NP-hard.

NP-hard Problems

- Infamous class of problems for which no one has developed a polynomial time algorithm.
- That is, no algorithm whose complexity is \(O(n^k)\) for any constant \(k\) is known for any NP-hard problem.
- The class includes thousands of real-world problems.
- Highly unlikely that any NP-hard problem can be solved by a polynomial time algorithm.

NP-hard Problems

- Since even polynomial time algorithms with degree \(k > 3\) (say) are not practical for large \(n\), we must change our expectations of the algorithm that is used.
- Usually develop fast heuristics for NP-hard problems.
  - Algorithm that gives a solution close to best.
  - Runs in acceptable amount of time.
- LPT rule is good heuristic for minimum finish time scheduling.

Complexity Of LPT Scheduling

- Sort jobs into decreasing order of task time.
  - \(O(n \log n)\) time (\(n\) is number of jobs)
- Schedule jobs in this order.
  - assign job to machine that becomes available first
  - must find minimum of \(m\) (\(m\) is number of machines) finish times
  - takes \(O(m)\) time using simple strategy
  - so need \(O(mn)\) time to schedule all \(n\) jobs.

Using A Min Priority Queue

- Min priority queue has the finish times of the \(m\) machines.
- Initial finish times are all 0.
- To schedule a job remove machine with minimum finish time from the priority queue.
- Update the finish time of the selected machine and put the machine back into the priority queue.
Using A Min Priority Queue

- \( m \) put operations to initialize priority queue
- 1 remove min and 1 put to schedule each job
- each put and remove min operation takes \( O(\log m) \) time
- time to schedule is \( O(n \log m) \)
- overall time is \( O(n \log n + n \log m) = O(n \log (mn)) \)

Huffman Codes

Useful in lossless compression.
May be used in conjunction with LZW method.
Read from text.

Min Tree Definition

Each tree node has a value.
Value in any node is the minimum value in the subtree for which that node is the root.
Equivalently, no descendent has a smaller value.

Min Tree Example

```
2
4 9 3
4 8 7
9 9
```
Root has minimum element.

Max Tree Example

```
9
4 9 8
4 2 7
3 1
```
Root has maximum element.

Min Heap Definition

- complete binary tree
- min tree
Min Heap With 9 Nodes
Complete binary tree with 9 nodes.

Min Heap With 9 Nodes
Complete binary tree with 9 nodes that is also a min tree.

Max Heap With 9 Nodes
Complete binary tree with 9 nodes that is also a max tree.

Heap Height
Since a heap is a complete binary tree, the height of an $n$ node heap is $\log_2 (n+1)$.

A Heap Is Efficiently Represented As An Array

Moving Up And Down A Heap
Putting An Element Into A Max Heap

Complete binary tree with 10 nodes.

New element is 5.

New element is 20.

New element is 20.

New element is 20.
Putting An Element Into A Max Heap

Complete binary tree with 11 nodes.

Putting An Element Into A Max Heap

New element is 15.

Putting An Element Into A Max Heap

New element is 15.

Putting An Element Into A Max Heap

New element is 15.

Putting An Element Into A Max Heap

New element is 15.

Complexity Of Put

Complexity is $O(\log n)$, where $n$ is heap size.

Removing The Max Element

Max element is in the root.
Removing The Max Element After max element is removed.

Heap with 10 nodes. Reinsert 8 into the heap.

Reinsert 8 into the heap.

Reinsert 8 into the heap.

Reinsert 8 into the heap.

Max element is 15.
Removing The Max Element

After max element is removed.

Removing The Max Element

Heap with 9 nodes.

Removing The Max Element

Reinsert 7.

Removing The Max Element

Reinsert 7.

Removing The Max Element

Complexity Of Remove Max Element

Reinsert 7.

Complexity is $O(\log n)$. 