Priority Queue
- Heaps

Data structures

Spring 2017

Priority Queues

Two kinds of priority queues:

- Min priority queue.
- Max priority queue.

Min/Max 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/max priority
  - remove element with min/max 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

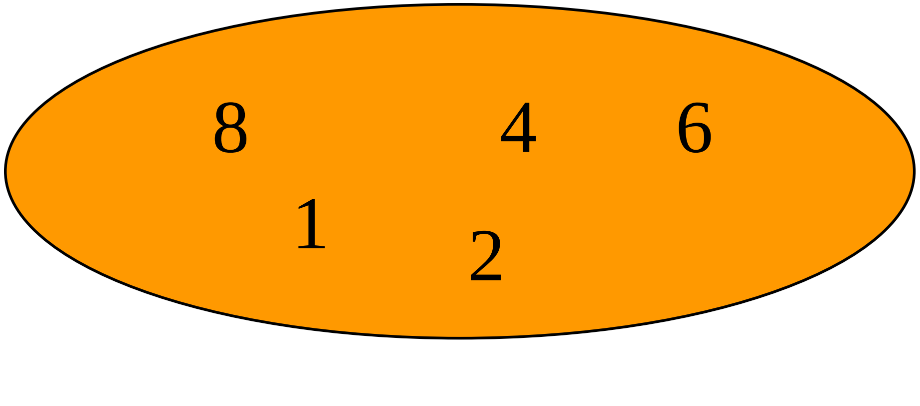
- use element key as priority
- put elements to be sorted into a priority queue
- extract elements in priority order
  - if a min priority queue is used, elements are extracted in ascending order of priority (or key)
  - if a max priority queue is used, elements are extracted in descending order of priority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

Sorting with Max Priority Queue



Max Priority Queue

1	2	4	6	8

Sorted Array

Complexity of Sorting

Sort n elements.

- n put operations => $O(n \log n)$ time.
- n remove max operations => $O(n \log n)$ 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

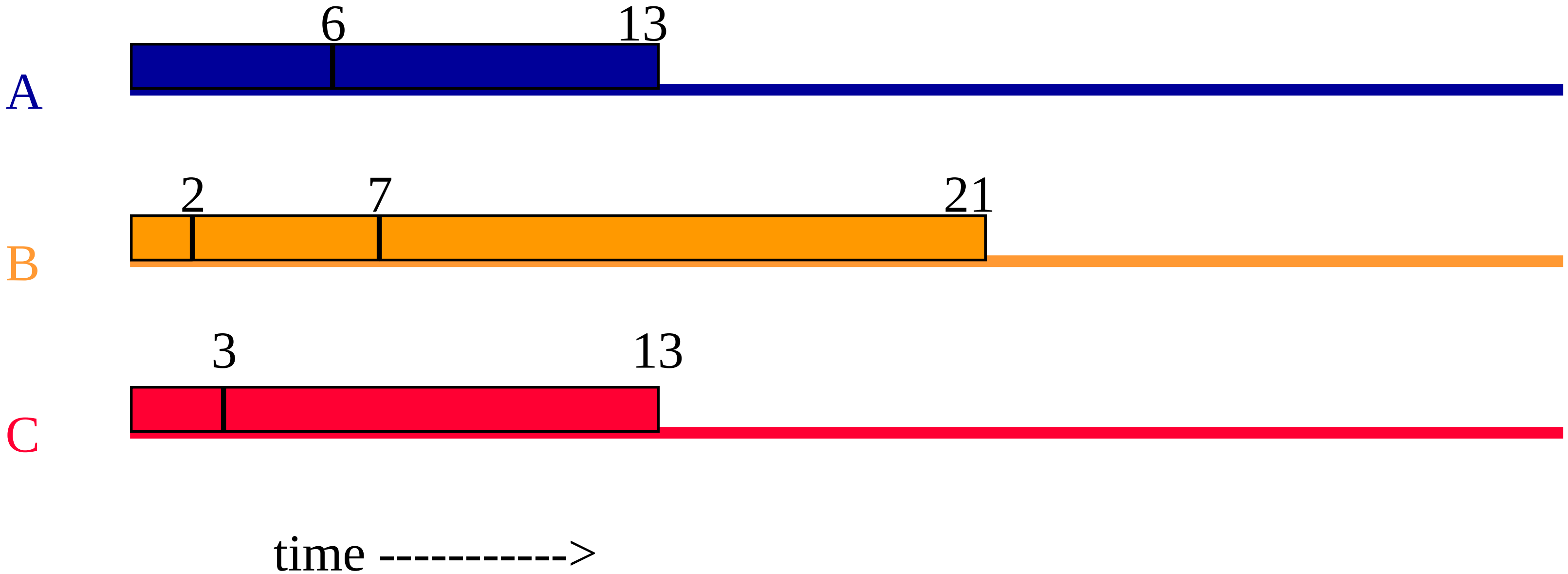
- m identical machines (drill press, cutter, sander, etc.)
- n jobs/tasks to be performed
- assign jobs to machines so that the time at which the last job completes is minimum

Machine Scheduling Example

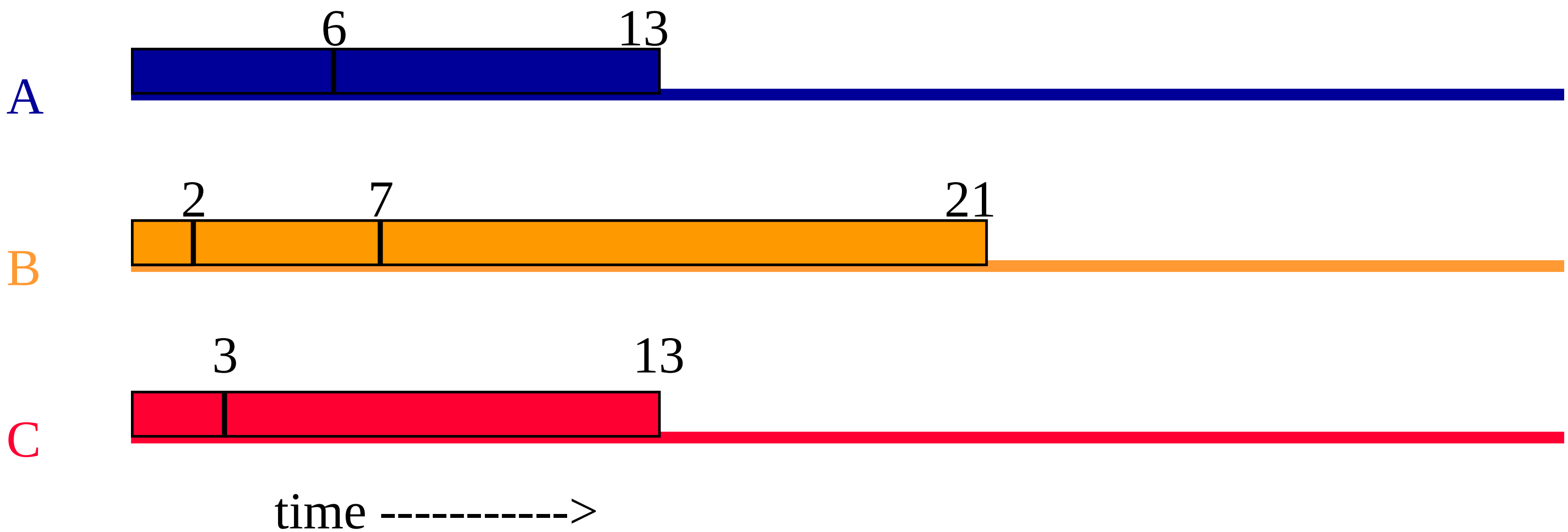
3 machines and 7 jobs

job times are [6, 2, 3, 5, 10, 7, 14]

possible schedule



Machine Scheduling Example



Finish time = 21

Objective: Find schedules with minimum finish time.

LPT Schedules

Longest Processing Time first.

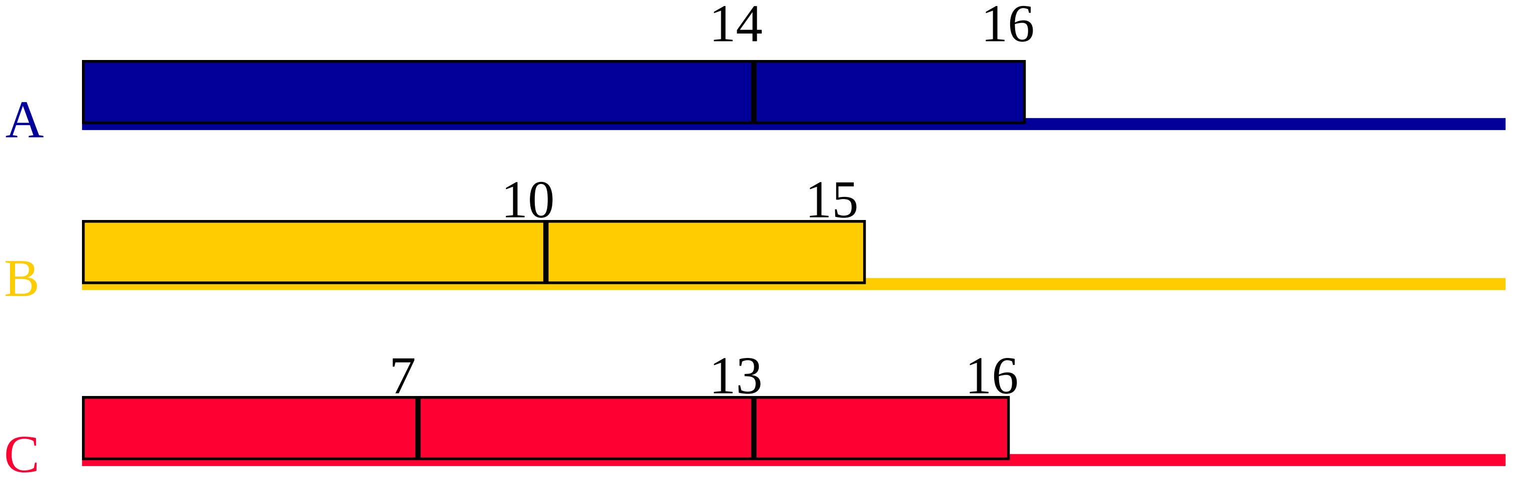
Jobs are scheduled in the order

14, 10, 7, 6, 5, 3, 2

Each job is scheduled on the machine on which it finishes earliest.

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 Finish Time)/(Minimum Finish Time) <= 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))

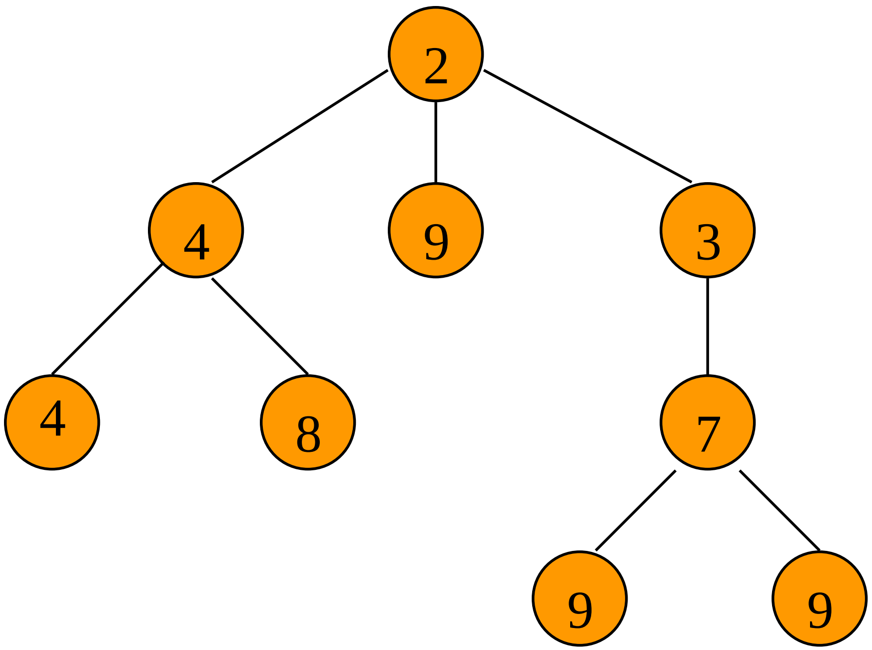
Min Tree Definition

Each tree node has a value.

Value in any node is the minimum value in the subtree for which that node is the root.

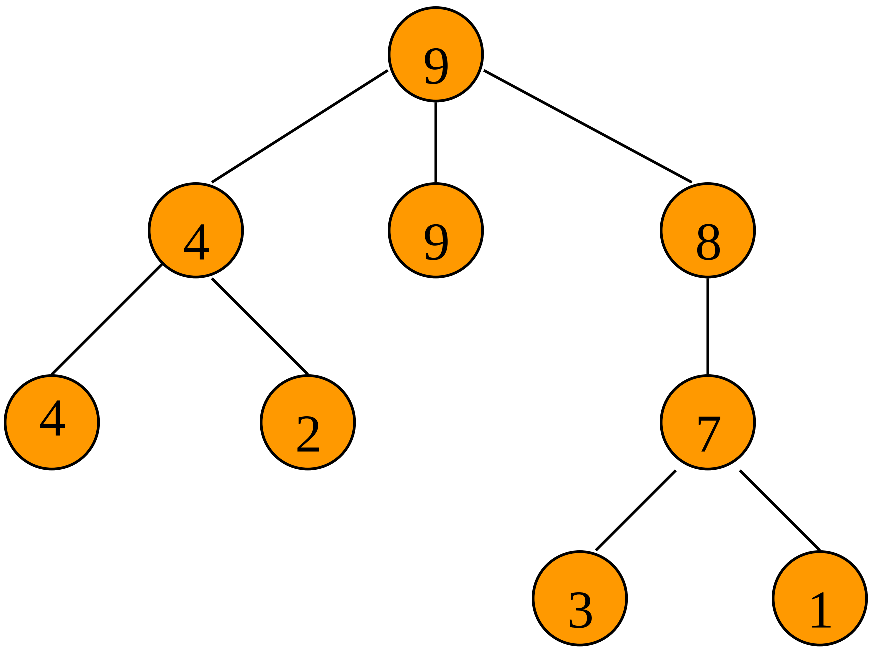
Equivalently, no descendent has a smaller value.

Min Tree Example



Root has minimum element.

Max Tree Example

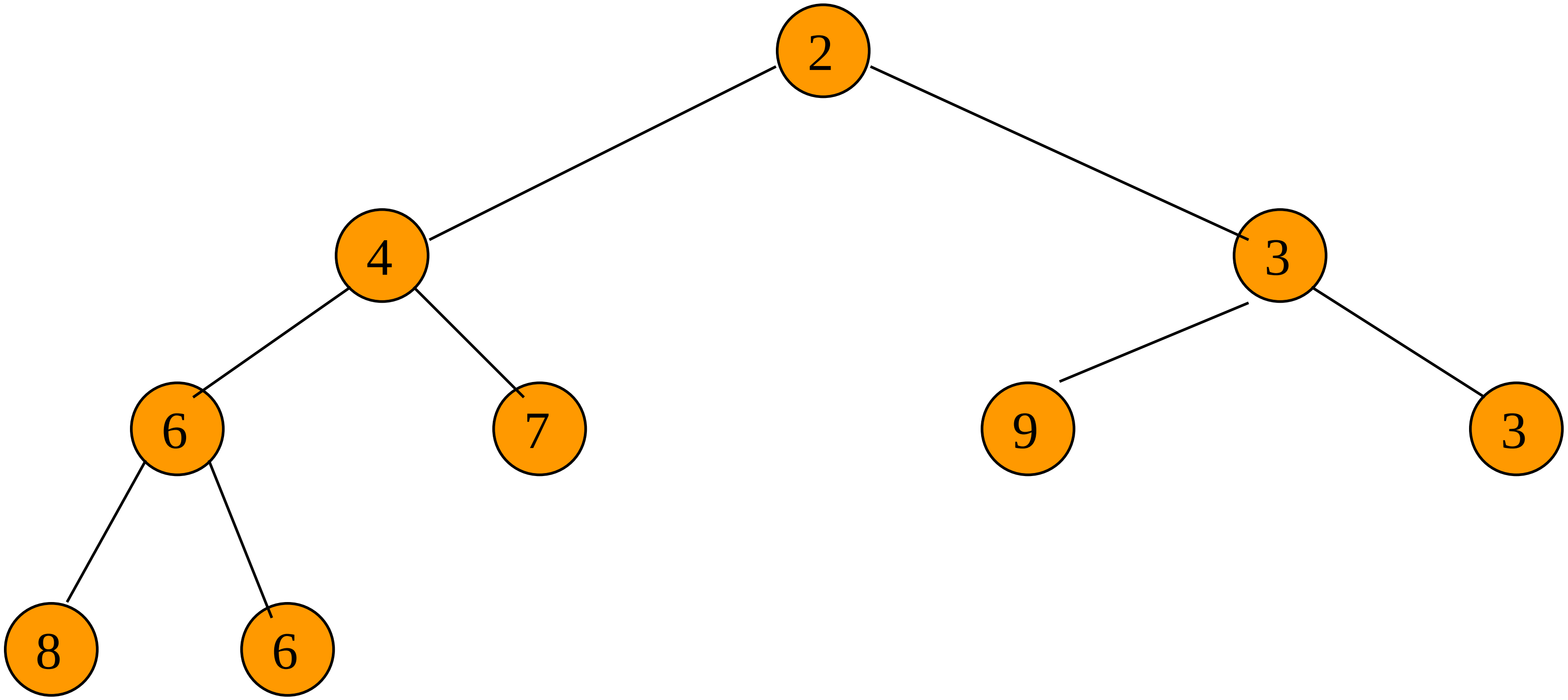


Root has maximum element.

Min Heap Definition

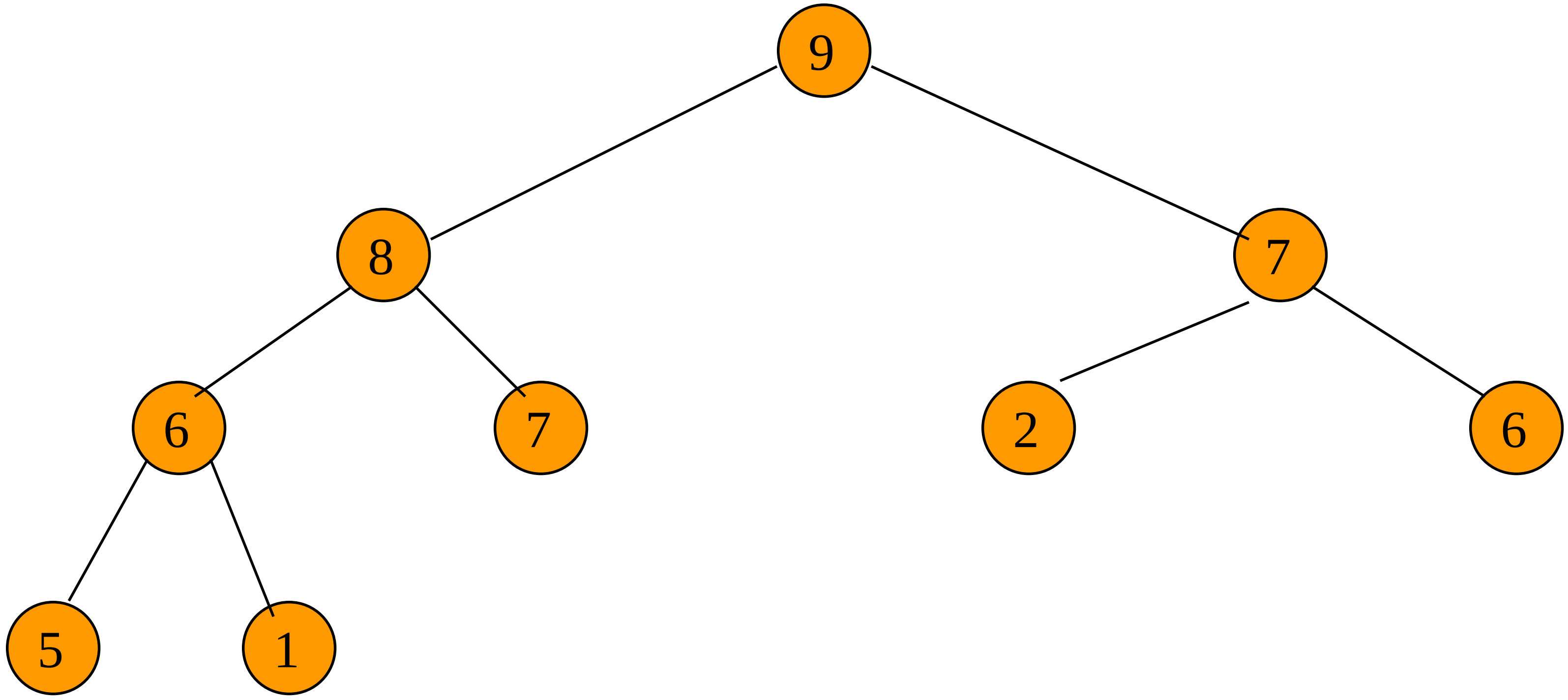
- complete binary tree
- min tree

Min Heap With 9 Nodes



Complete binary tree with 9 nodes that is also a min tree.

Max Heap With 9 Nodes

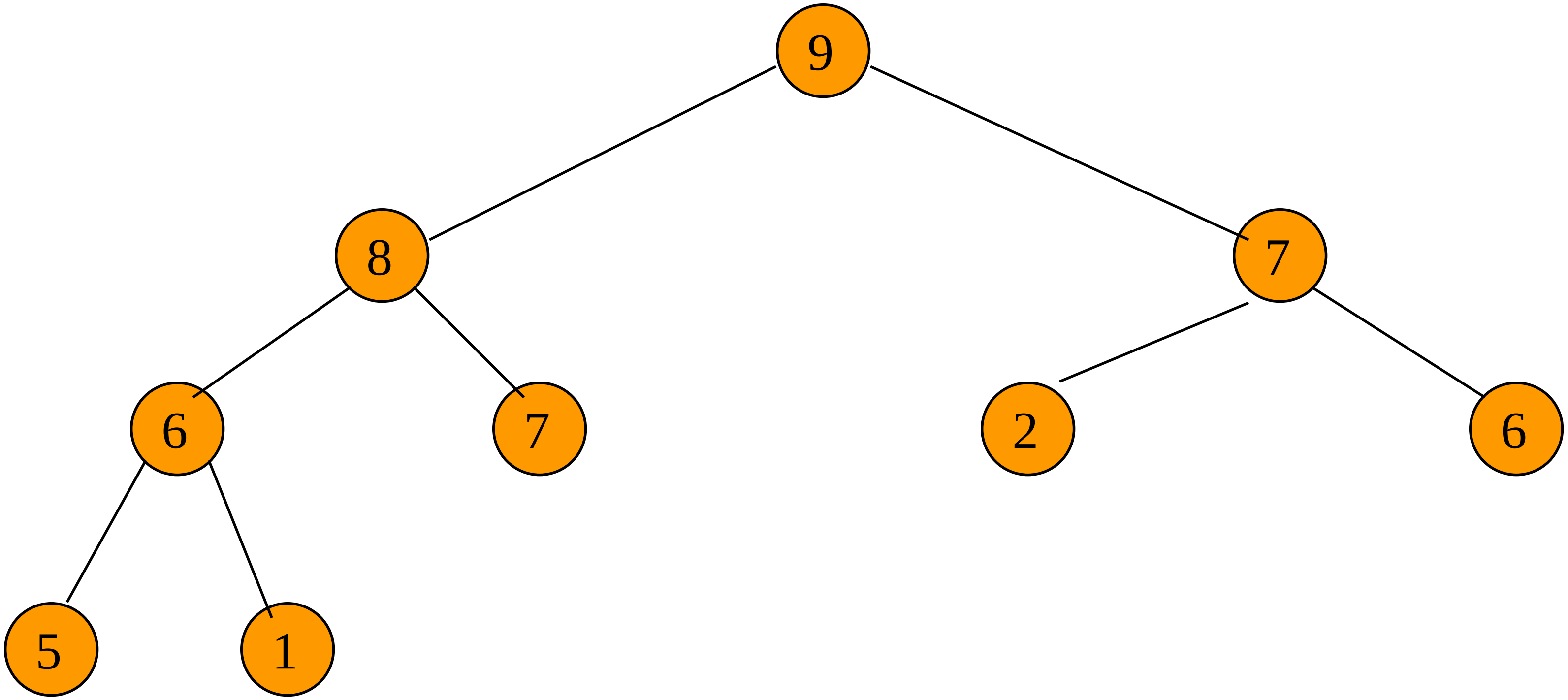


Complete binary tree with 9 nodes that is also a max tree.

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



	9	8	7	6	7	2	6	5	1							
0	1	2	3	4	5	6	7	8	9	10						

Moving Up and Down a Heap