Date: Aug 20, 2025

Binomial tree

1. A binomial tree of order k has the following properties:
   - It has exactly 2^k nodes.
   - It has depth as k.
   - There are exactly ${}^kC_i$ nodes as depth i, $i = 0, 1, 2, \ldots, k$.
   - The root has degree k and the children of root are themselves binomial trees of order $k-1, k-2, \ldots, 0$ from the left to the right.

Binomial heap

1. A binomial heap is implemented as a collection of binomial trees that satisfy that satisfy the binomial heap properties.
2. Each binomial tree in a heap obeys the minimum heap property, i.e. the key of a node is greater than or equal to the key of its parents.
3. This ensures the root of each binomial tree contains the smallest key in the tree, which applies to the entire heap.
4. There can only be either 1 or zero binomial trees for each order including zero order. This implies a binomial heap with n nodes consists of at most $log(n+1)$ binomial trees. By the number of nodes, n, the number and orders of these trees are uniquely determined.

Binary tree – merging

1. Two binomial trees of the same order can be merged.
2. The tree with higher root value is made the left most child of the root with minimum root value.

Binary heap – merging

1. Two binomial heaps can be merged by merging two binomial trees of the same order.
2. If only one of the heaps contain a tree of order j, then the tree is moved to the merged heap. If both heaps contain a tree of order j, the two trees are merged to one of order $j+1$ so that the minimum heap property is satisfied.

3. Note that it may later be necessary to merge this tree with some other tree of order $j+1$ present in one of the heaps.
4. In the course of the algorithm, we need to examine at most three trees of any order (2 from 2 heaps we merge and 1 composed of two smaller trees).

Binary heap – insertion

1. Create a new heap with the element to be inserted.
2. Merge the new heap with the existing heap.

Binary heap – minimum

1. Find minimum among the roots of the binomial trees.

Binary heap – deleting minimum

1. Find the minimum element.
2. Remove it from the binomial tree and obtain a list of its sub-trees.
3. Transform this list of sub-trees into a separate binomial heap by reordering them from smallest to largest.
4. Then merge this heap into the original heap.

Decrease key

1. After decreasing the key of an element, it may become smaller than the key of the parent violating the minimum heap property.
2. Then exchange the element with its parent, grandparent and so on until the minimum heap property is no longer violated.

Delete any element

1. To delete any element from the heap, decrease its key to $-\infty$ (i.e. some value lower than any element in the heap) and then delete the minimum element in the heap.