Binomial Heaps

are ordered

consist of a set of Binomial Heapsordered
trees.

\[ B_i : \text{Binomial tree of order } i \text{ having} \]
\[ 2^i \text{ nodes} \]

Given a set of \( n \) elements, we

can partition \( n \) into subsets \( J \)

size \( 2^i \), s.t. there is no more than

one subset of size \( 2^i \)

Binary representation of \( n \)

Example \( n = 11 \) \hspace{1cm} 1011

\[
\begin{align*}
\text{Root} & \quad B_3 \\
\text{left} & \quad B_1 \\
\text{right} & \quad B_0
\end{align*}
\]

\[
\begin{tikzpicture}
\coordinate (root) at (0,0);
\coordinate (left) at (-1,0);
\coordinate (right1) at (-0.5,0);
\coordinate (right2) at (0,0);
\coordinate (right3) at (0.5,0);
\coordinate (right4) at (1,0);

\draw (root) -- (left);
\draw (root) -- (right1);
\draw (root) -- (right2);
\draw (root) -- (right3);
\draw (root) -- (right4);
\end{tikzpicture}
\]

Obs: \# Binomial trees \( \leq \lceil \log n \rceil \)

since \( n \) has a \( \log n \) bit binary rep
Min: We can find min by searching the root list: log n
By maintaining an explicit reference to min: O(1) time
(Requiring not recompiling)

Merging of Binomial Heaps:
(Merging)

$H_1: B_1 B_3 B_5 B_6$

$H_2: B_0 B_1 B_2 B_5$

By simply linking $H_1, H_2$ we could notate

1. There may more than one tree of order $i$

2. # trees could be $> \log |H_1| + |H_2|$

Merging two root lists of Binomial Heaps is similar to binary addition
Procedure

Starting from the lower order trees
if there are two trees of order \(j\), we combine them (using the recursive defn) into \(B_{j+1}\)

\[
\begin{align*}
B_{j+1} & \quad \{ \quad B_j \quad \}
\end{align*}
\]

H1 \quad \to \quad B_j \quad \to \quad B_{j+1}

H2 \quad \to \quad B_j \quad \to \quad B_{j+2}

H3 \quad \to \quad B_{j+1}

Time : \quad O \left( \log |H_1| + \log |H_2| + 1 \right)

\[\vdots\]

\[\vdots\]

\[\vdots\]

\[\vdots\]

where \quad n = |H_1| + 1 + |H_2|
Extract min:

1. Delete root
2. Create a new root list from the children $H'$
3. Find new min

Time: $O(b \log n)$

Insert: a new element $x$ to $H$

Create a heap consisting of singletons $\{x\}$

Merge $\{x\}$ with $H$

Delete:

Delete $y$, build a root list $H$ its children and merge with the original root list
What is the total cost of n operations (E): \( O(n) \)?

Consider a sequence \( a_1, a_2, \ldots, a_n \). The cost of an operation \( i \) is \( c_i \).

Total cost = \( \sum_{i=1}^{n} c_i \).

You can maintain stacks under operations:

- push: \( \text{top} \), \( \text{empty queue skills} \), \( \text{element under top stack} \), \( \text{empty queue skills} \).
- pop: \( \text{top} \), \( \text{empty queue skills} \).

Decide key:

- If it is greater than the top, increment.
- If less, pop the top element.
The cost of an n x k is there are K elements in the stack

Aggregate analysis, known as "amortized analysis"

Example: Consider a n hit count

\[ \begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
1 & 1 & 1 & 1 & 1 & 1 \\
\end{array} \]

In a single cycle, how many total bit flips happen?

\[ \sum_{i=0}^{n-1} \frac{2^n}{2^i} \leq O(2^n) \text{ instead the simplistic } n \cdot 2^n \]