Trick: Add an artificial cost per operation!

Def \( E : \text{state} \rightarrow \mathbb{R} \) a potential

Def unit-cost = \( O_{pi} \) (unit-cost \( \leq \) real cost)

Def Amortized Cost = unit-cost + potential change

\[
\sum \text{AC}_i = \sum [O_{pi} + (E_i - E_{i-1})] = \sum O_{pi} + E_n - E_0 \leq TC + \Delta E
\]

if \( \Delta E \geq 0 \) then \( TC \leq \sum \text{AC}_i \)

Let \( AC = \max_{i} \text{AC}_i \) then

\( TC \leq n \cdot AC \) if \( \Delta E \geq 0 \)
Fibonacci Heaps

Goal: Modify Binomial Heaps so that
O(1) Amortized decreaseKey

Back to Binomial Heaps

Lazy Meld = Only link during delete min

Claim AC is still O(log n)

Claim AC is still O(log n)

idea: decreaseKey(k, A)

1) disconnect k from its tree.
2) add subtree to trees of A.

Prob: Trees will become unbalanced!
O(1) AC cost for Decrease Key

Solution Each nonroot node can have at most one missing child.

Define Mark(k) if k has a missing child
   & k is not a root

Define rank(T) = \#children of root.

Cut(k) K a node in T
   1) Return if K is a root
   2) Let P = Parent(k)
   3) Remove subtree rooted at K
      a) move subtree to list of trees.
      b) unmark K.
      c) decrement rank of P.
      d) If P marked then Cut(P)
         else mark P.
AC decrease key

Potential Method: $\Phi(A) = \#\text{trees} + 2(\#\text{marks})$

Token Method: 1) token on each root
2) 2 tokens on each marked node

Claim: Amortized Cost of Cut \( \leq 4 \)

Proof: Consider cut(K)

Unit-cost = \# new trees formed \( \leq \# \text{nodes unmarked} + 1 \)
$$\Delta E = \#_{\text{new trees}} + 2 (1 - \#_{\text{unmarked}})$$

$$\leq \#_{\text{new trees}} + 2 (1 - (\#_{\text{new trees}} - 1))$$

$$\leq \#_{\text{new trees}} + 4$$

$$AC \leq \#_{\text{new trees}} - \#_{\text{new trees}} + 4 = 4$$
Fibonacci-DeleteMin

Fib-DeleteMin = Binomial-DeleteMin

Fib-DeleteMin (A = T_1, \ldots, T_x)

1) Links tree until at most one per rank.
2) Find tree whose root has min priority.
3) Remove root and add its subtrees to A.

Note: We need \#tree at step 2) = O(\log n).

i.e. We need the size of a tree to be exponential in its rank.

Def: The nth Fibonacci number

F_0 = 0, F_1 = 1 & F_{n+1} = F_n + F_{n-1}

Def: Let S_n = min size rank n Fib Tree

Thm: F_{n+2} \leq S_n (F_{n+2} = S_n)

- S_0 = 1, \quad S_1 = 2
- F_2 = 1, \quad F_3 = 2
As a warmup let's show that $S_n \leq F_{n+2}$.

Base case is OK.

Suppose $S_{n-1} \leq F_{n+1}$ & $S_n \leq F_{n+2}$.

Let $T_n$ be a min-size rank $n$ tree.

Consider $T = \text{link} (\overline{T_{n-1}}, \overline{T_n})$.

$\overline{T} = \text{link} (T, \overline{T_n})$.

$S_{n+1} \leq \text{size} \leq F_{n+1} + F_{n+2} = F_{n+3}$.
Size Lower Bounds for Fibonacci Trees

There are many different proofs the the size is bound below by the Fibonacci numbers. Here is yet another one.

Claim: All Fibonacci trees can be generated by the following set of rules.

1) The singleton tree is a Fib-tree
2) The link of 2 Fib-trees is a Fib-tree.
   The two tree must have the same rank and
   the rank of the new tree is one more than the children.
3) The removal of any child from the root generates a Fib-tree.
   The new rank is one less.
4) The removal of a child from an unmarked non-root parent is
   a Fib-tree. The parent is now marked.

Define the Fibonacci numbers to be

\[ F_0 = 0 \quad F_1 = 1 \quad F_(n+1) = F_n + F_(n-1) \]

Claim: \[ F_(n+2) <= | \text{Fib-tree of rank } n | \]

Proof:

The proof is by induction on the rank of a tree \( T \).

If the rank is 0 or 1 we are done by inspection. Let \( T \) be a
minimal size tree of rank \( n+1 \) generated by the rules above.

Consider a sequence \( S \) of operation generating \( T \).
Let \( T' = \text{Link}(T_1,T_2) \) be the last link in the sequence, \( T_2 \) being linked to \( T_1 \).
Assume that the rank of \( T' \) is of minimum size over all sequences
generating \( T \). In this case we claim that there will be
no rule-3's after \( T' \).

Proof of subclaim:

The proof is by contradiction.
Suppose there was a rule-3 after constructing \( T' \).

There are two case:
1) If we remove \( T_2 \) from the tree then we can find a new
   sequence which does not use this link at all.
2) Consider the case of removing a child of \( T_1 \) after \( T' \).
   Since \( T \) is of minimum size we will remove a child of \( T_2 \)
after $T'$. Thus we could have removed these two children before the link and then linked them. This contradicts our assumption that the rank of $T'$ was minimum.

Thus we now know that the rank of $T_1$ and $T_2$ are both $n$. WLOG all the rule-4s can be moved before $T'$ except for the removal of a child of $T_2$. Let $T'_2$ be the tree of rank $n-1$ with one child of the root removed.

$$|T| = |T_1| + |T'_2| \geq F_{n+2} + F_{n+1} = F_{n+3}.$$ 

QED
Low bounding $F_n$

Let's view Fibonacci as a matrix-vector product.

Let $( \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} ) ( \begin{pmatrix} f_{n-1} \\ f_n \end{pmatrix} ) = ( \begin{pmatrix} f_n \\ f_{n+1} \end{pmatrix} )$  $f_0 = 0$

Let $P_i = ( \begin{pmatrix} f_i \\ f_{i+1} \end{pmatrix} )$

\[
\begin{array}{cccccc}
0 & 1 & 2 & 3 & 5 \\
1 & 1 & 2 & 3 & 5 & 8 \\
P_0 & P_1 & P_2 & P_3 & P_4 & P_5
\end{array}
\]
What are the eigenvalues of $F = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$?

$$\text{roots}(\det(\lambda I - F)) = \text{roots}(\det(\begin{pmatrix} \lambda & 1 \\ -1 & \lambda - 1 \end{pmatrix}))$$

$$\text{roots}(\lambda^2 - \lambda - 1) \quad \lambda = \frac{1 \pm \sqrt{5}}{2}$$

Spectral Thm $\Rightarrow f_n \approx \left(\frac{1 + \sqrt{5}}{2}\right)^n$