Amortized Analysis

15-750
1/23/19

Trick: Add an artificial cost per operation!

Def Φ : state $\to \mathbb{R}$ a potential

Def unit-cost = Op_i (unit-cost $\approx$ real cost)

Def Amortized Cost = unit-cost + potential change

$$\sum AC_i = \sum\left[OP_i + (\Phi_i - \Phi_{i-1})\right] = \sum OP_i + \Phi_n - \Phi_0$$

$$= TC + \Delta\Phi$$

if $\Delta\Phi \geq 0$ then $TC \leq \sum AC_i$

Let $AC = \max_i AC_i$ then

$$TC \leq n \cdot AC \text{ if } \Delta\Phi \geq 0$$

1/23/19
15-750

Fibonacci Heaps

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

---

Back to Binomial Heaps

Lazy Meld $\equiv$ Only link during deletemin.

Claim AC is still $O(\log n)$ using $\Phi(A) = \#$ of trees

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 DecreaseKey

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

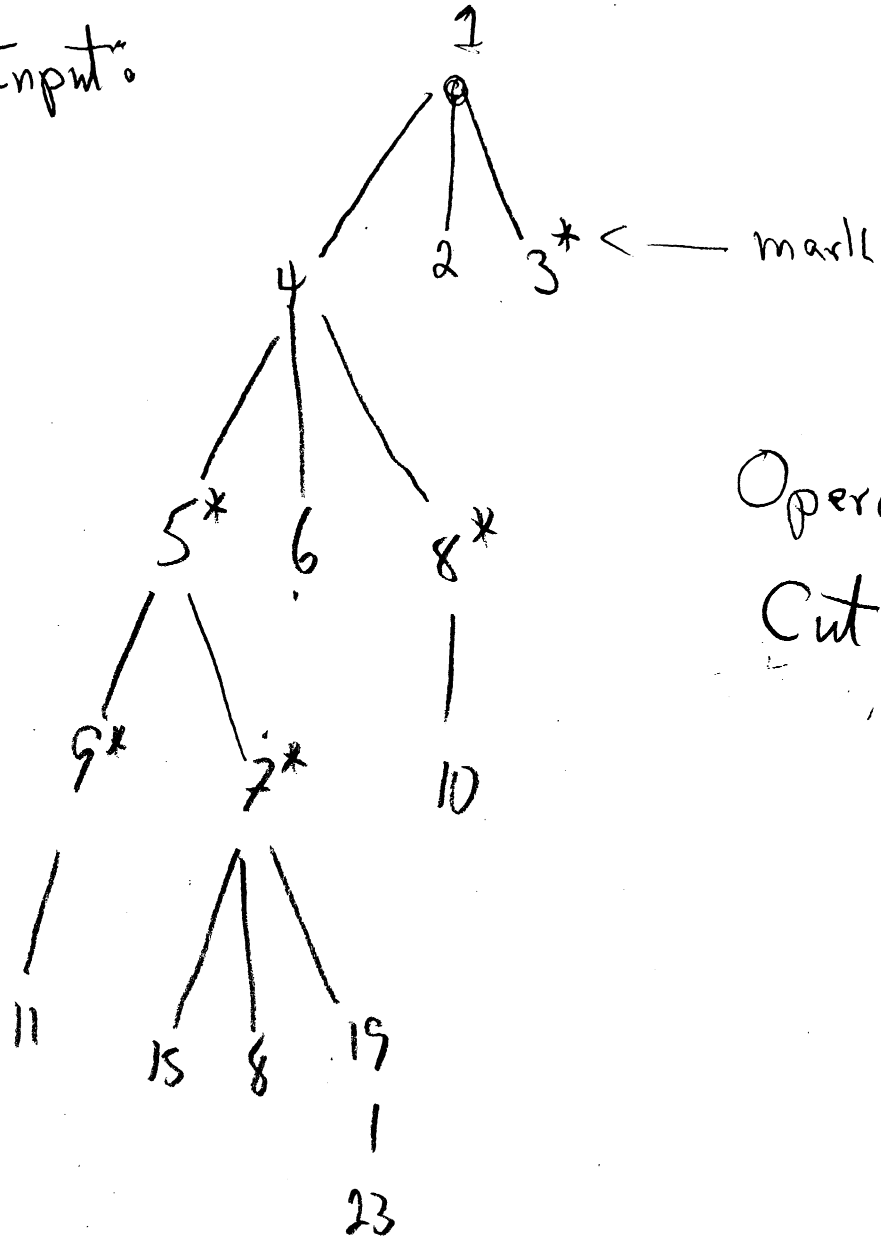
Def Mark(x) if x has a missing child & x is not a root

Def rank(T) ≡ # children of root.

Cut(x) x a node in T

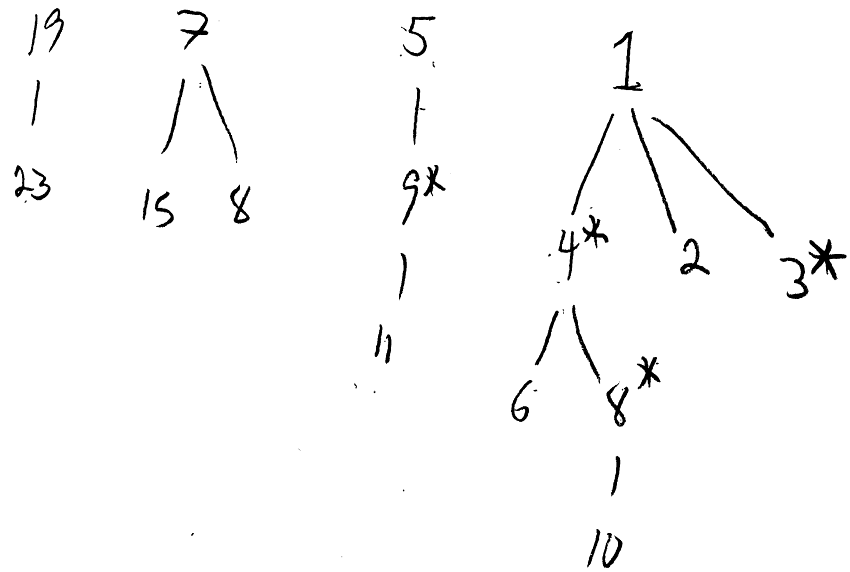
1) Return if x is a root
2) Let p = Parent(x)
3) Remove subtree rooted at x
   a) move subtree to list of trees.
   b) unmark x.
   c) decrement rank of p.
   d) If p marked then Cut(p) else mark p.

Input:



Operation:
Cut(19)

Output:



AC of decreaseKey

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 ≤ 4

pf consider cut(x)

Unit-cost $\equiv$ # new trees formed eg 4

$\leq$ # nodes unmarked +1

$$\Delta \Phi = \#newtrees + 2(\underbrace{1}_{\text{new mark}} - \#unmarked)$$

$$\leq \#newtrees + 2(1-(\#newtrees - 1))$$

$$\leq -\#newtrees + 4$$

$$AC \leq \#newtrees - \#newtrees + 4 = 4$$

Fibonacci-Deletemin

Fib-Deletemin $\equiv$ Binomial-Deletemin

Fib-Deletemin $(\bar{A} = T_1, \dots, T_x)$

1) Link tree until atmost one per rank.
2) Find tree whose root has min piority
3) remove root and add its subtrees to $\bar{A}$.

<u>Note</u> We need # tree at step 2) $= O(\log n)$.

i.e. we need the size of a tree to be exponenial in its rank.

<u>Def</u> The nth Fibonacci number

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

<u>Def</u> Let $S_n \equiv$ min size rank n Fib Tree

<u>Thm</u> $F_{n+2} \le S_n \quad (F_{n+2} = S_n)$

 $S_0 = 1$

$F_2 = 1$

 $S_1 = 2$

$F_3 = 2$

As a warmup lets show that $S_n \le F_{n+2}$

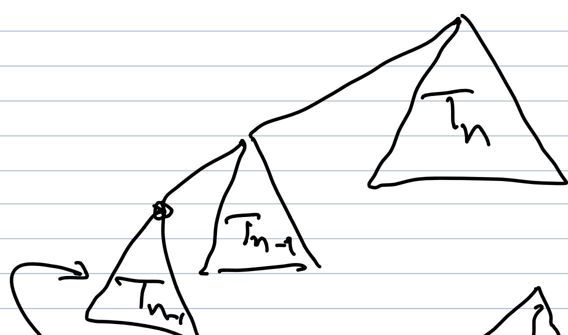
Base case is OK.

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

Let T_n be a min size rank n tree

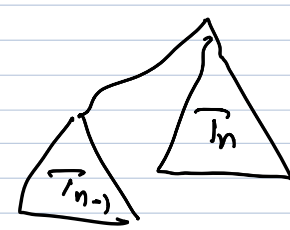
Consider $T = \text{link}(T_{n-1}, T_{n-1})$

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



Cut

Thus



is a rank $n+1$ Fib-Tree

$$S_{n+1} \le \text{Size} \le F_{n+1} + F_{n+2} = F_{n+3}$$

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

$F_0 = 0$ $F_1 = 1$ $F_{(n+1)} = F_n + F_{(n-1)}$

Claim: $F_{(n+2)} \le |\text{Fib-tree of rank } n|$

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' = 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.

$|T| = |T_1| + |T'_2| >= F_(n+2) + F_(n+1) = F_(n+3)$.

QED

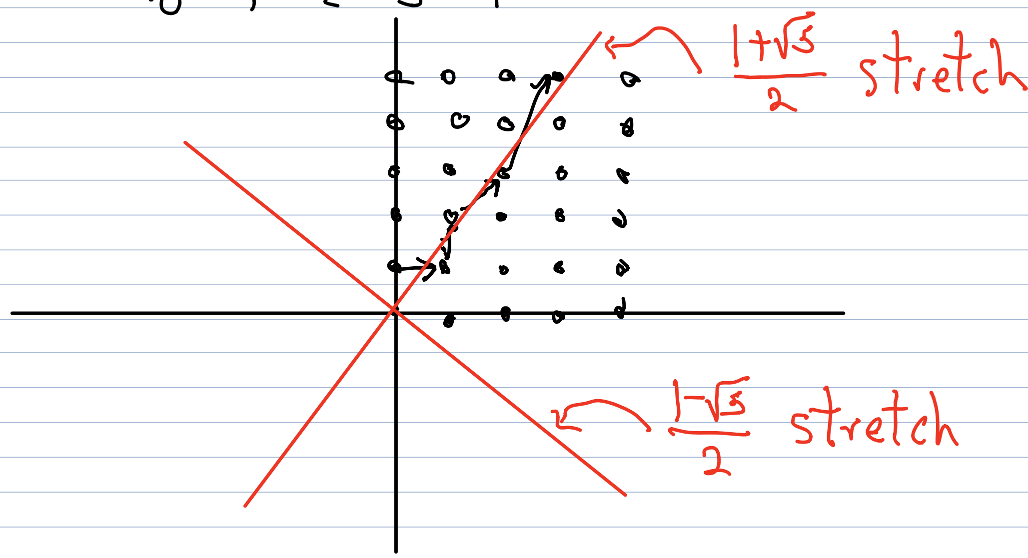
Low bounding F_n

Lets view Fibonacci as a matrix-vector product.

Let $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} f_{n-1} \\ f_n \end{pmatrix} = \begin{pmatrix} f_n \\ f_{n+1} \end{pmatrix}$ $\quad f_0 = 0$, $f_1 = 1$

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

0	1	1	2	3	5
1	1	2	3	5	8
P_0	P_1	P_2	P_3	P_4	P_5



$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} f_n \\ f_{n+1} \end{pmatrix}$$

What are the eigenvalues of $F = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$?

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

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

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