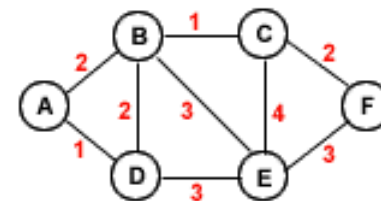




Effiziente Algorithmen

Hartmut Klauck
Universität Frankfurt
SS 06

4.7.





Decrease_Key und Delete

- Wir vorher werden wir Delete durch Decrease_Key und ExtractMin implementieren: verringere Schlüssel zuerst unter das aktuelle Minimum
- Die Decrease_Key Operation zerstört die Eigenschaft, daß alle Bäume ungeordnet binomisch sind.
- Hier wird notwendig sein, zu zeigen, daß jeder Baum Knotengrad $O(\log n)$ hat, wenn n Schlüssel gespeichert sind



Decrease_Key

- Eingabe: Fheap H , Element x , Schlüsselwert k
 1. Wenn $k > \text{Key}(x)$: error
 2. Setze $\text{Key}(x) := k$, $y := \pi(x)$
 3. Wenn $y \neq \text{NIL}$ und $\text{Key}(x) < \text{Key}(y)$
 1. $\text{Cut}(H, x, y)$
 2. $\text{CascadingCut}(H, y)$
 4. Wenn $\text{Key}(x) < \text{Key}(\text{Min}(H))$: $\text{Min}(H) := x$



Cut

- Eingabe: H, x, y
- Entferne x aus der Kinderliste von y , verringere $\text{degree}(y)$, wenn $\text{child}(y)$ auf x zeigt lasse $\text{child}(y)$ auf $\text{left}(x)$ zeigen
- Füge x in Wurzelliste ein
- $\pi(x) := \text{NIL}$
- $\text{mark}(x) := 0$



CascadingCut

- Eingabe: Fheap H , Knoten y
- $z := \pi(y)$
- Wenn $z \neq \text{NIL}$ und $\text{mark}(y) = 0$:
 - $\text{mark}(y) := 1$
- Wenn $z \neq \text{NIL}$ und $\text{mark}(y) = 1$:
 - $\text{Cut}(H, y, z)$
 - $\text{CascadingCut}(H, z)$



Was passiert?

- Wenn die Heapordnung erhalten bleibt: nichts
- Sonst: x wird als neue Wurzel eingefügt und aus seinem Baum entfernt
- Problem: Bäume nicht mehr binomisch
- „Beschneide Bäume“
- Folge Pfad von x zur Wurzel und schneide jeden Knoten, der:
 - irgendwann mal Wurzel war, dann Kind wurde
 - und zwei seiner Kinder „verloren“ hat
- Solche Knoten werden neue Wurzeln
- Damit gilt: Jeder innere Knoten in einem Baum hat höchstens 1 Kind verloren, daher sind wir „fast“ ein binomischer Baum
- $\text{mark}(y)=1$ gdw: y ist keine Wurzel und y hat genau ein Kind verloren, seit es an z angehängt wurde



Laufzeit

- Annahme: k mal wird `CascadingCut` aufgerufen
- Dann ist die tatsächliche Laufzeit $ck = O(k)$ für ein konstantes c
- Anzahl der Bäume steigt um k , also $t(H) + k$ nachher
- Anzahl markierter Knoten sinkt um $k - 2$:
 $k - 1$ mal wird ein Knoten unmarkiert, einmal eventuell markiert, also $m(H) - k + 2$ nachher
- Potential nachher also $t(H) + k + 2m(H) - 2k + 4 = t(H) + 2m(H) - k + 4$
- Damit Potentialänderung $-k + 4$
- Durch Skalierung der Potentialfunktion mit c ergibt sich, daß die amortisierte Laufzeit

$$ck + c(-k + 4) = O(1)$$



Der maximale Knotengrad

- Die Laufzeit von ExtractMin hängt vom maximalen Knotengrad im Fheap ab
- Wenn kein DecreaseKey, sind alle Bäume binomisch und B_k hat 2^k Knoten und Grad k
- Wir zeigen: maximaler Knotengrad ist $O(\log n)$ bei n gespeicherten Schlüsseln
- Analyse klärt den Namen: Fibonacci Heaps



Ein Lemma

- Sei x ein Knoten in einem Fheap mit Grad k .
- Seien y_1, \dots, y_k die Kinder in der Reihenfolge, in der sie an x gehängt wurden
- Dann gilt:
 - $\text{degree}(y_1) \geq 0$ [trivial]
 - $\text{degree}(y_i) \geq i-2$ sonst



Beweis

- Wenn y_i angehängt wird sind y_1, \dots, y_{i-1} schon Kinder
- $\text{degree}(y_i) \geq i-1$ zu diesem Zeitpunkt, weil nur Bäume mit gleichem Wurzelgrad verschmolzen werden
- y_i kann nur ein Kind verlieren, sonst wird es geschnitten daher gilt $\text{degree}(y_i) \geq i-2$ solange es ein Kind von x ist



Fibonacci Zahlen

- $F(0)=0$
- $F(1)=1$
- $F(k)=F(k-1)+F(k-2)$

- FAKT:
 - $F(k+2)=1+\sum_{i=0}^k F(i)$
- FAKT:
 - $F(k+2) \geq \phi^k$
 - Wobei $\phi=(1+5^{1/2})/2=1.61803\dots$:
der goldene Schnitt



Lemma

- Sei x ein Knoten in einem Fheap mit n Elementen und $\text{degree}(x)=k$
- Dann ist $n \geq F(k+2) \geq \phi^k$
- Damit ist $k=O(\log n)$



Beweis

- Sei s_k die minimale Größe eines Teilbaums ab einem Knoten mit Grad k
- Klar: $s_0=1, s_1=2, s_2=3$
- Für jeden Knoten x mit Grad k ist die Größe seines Teilbaums mind. s_k
- Seien also y_1, \dots, y_k Kinder eines Knotens mit Grad k in Reihenfolge des Anhängens
- Damit ist hat der Teilbaum ab x mind. Größe
 - s_k
 - $\geq 2 + \sum_{i=2, \dots, k} s_{\text{degree}(y_i)}$
 - $\geq 2 + \sum_{i=2, \dots, k} s_{i-2}$
- Per Induktion zeigt man nun, dass $s_k \geq F(k+2)$



Beweis

- $k=0$: $s_0=1$ Knoten, $F(2)=1$
- $k=1$: $s_1=2$ Knoten, $F(3)=2$
- $s_k \geq 2 + \sum_{i=2, \dots, k} s_{i-2} \geq 2 + \sum_{i=2, \dots, k} F(i)$ per Induktion
 $\geq 1 + \sum_{i=1, \dots, k} F(i)$
 $= F(k+2)$