Hollow heap

Basics

- heap => element = (key, value)
  - Node ↑
- full nod: innehåller element
- tom nod: innehåller ej element (hollow)

(Dvs nod ≠ element)

flera rotnoder

- list of root nodes.
- All nodes have linked list of children nodes.
  - + pekare på föräldrar
    - ↳ Nya barn sätts in först. (minskande rank ordning)
- När ny element => Ny nod
  - ↳ kolla om minst.

*** Insert ***

1. Ny nod
2. kolla om den är minst
   - ↳ Uppdatera ~minst~

* Invariant för rank *

Case 2: En nod med rank r
- exakt r barn

case 1: hollow nod med rank ≥ 2
- exakt 2 barn
- De har rank r-1, r-2

- När ett element flyttas u → v

gör:
- rank v = max{ 0, rank(u) - 2 }
- alla barn till u med rank mindre än v
  ↪ flytta, tillsammans med alla sina barn. till v.

gör ev. enligt:
- om r ≥ 2
  - u behåller barnen med rank r-1 & r-2
- r = 1
  - u behåller sitt barn

0, 1, 1, 2, 3

Lemma: En nod med rank r har maximalt $f_{r+3} - 1$ ättlingar (tomma & fulla)

Bevis: Induktion

visa $r = 0$ eller $r = 1$ ⇒ Sant. Enligt invariant för rank. (finns 1 resp 2 noder)

antag $r \geq 2$ ⇒ $f_{r+3} - 1$ ättlingar

⇒ Noden själv, $r-1$, $r-2$ finns bland ättlingar.

ättling inkluderar sig själv

$$\underbrace{1}_{\text{noden}} + \underbrace{(f_{r+2} - 1)}_{(r-1)+3} + \underbrace{(f_{r+1} - 1)}_{(r-2)+3} = f_{r+3} - 1$$

Stämmer.

Lemma: Ranken för en nod i en hollow heap med N noder (fulla + tomma) är högst $\log_\phi N$

Bevis: Följer första lemmat

Pga $f_{r+3} - 1 \geq f_{r+2} \geq \phi^r$ för $r \geq 0$

hollow heap tidskomplexitet

link

- link då $r_1 = r_2$
  - ↳ ① barn (läggs) först parent lista
  - ② parent.rank ++

} konstant tid

decrease

- Barn flyttas till ny nod
  - ↳ Alla förutom första två flyttas (med pekare)

} konstant tid

delete

- Inte minsta : konstant
- minsta:
  - Hitta noder med samma rank
    - ↳ Använd fält som indexeras på rank

} konstant

- O(H+T) (dingört)
  - ↳ H: Antalet tomma rotnoder som förstörs
  - ↳ T: Antalet rotnoder som kvarstår efter tomma borttagna.

(Precis före link)

O(n)

- Efter link finns högst en nod per rank, dvs rank är max $\log_\Phi N$

$O(\log_\Phi N)$

⇒ minst $T - \log_\Phi N$ sammanslagning

* link operation *

↳ Compare keys of two nodes

⇒ Make smaller key parent

(Maintains, as such, heap structure)

* Decrease key *

case 1: Root!

Case 1: Är's min

① Decrease key

case 2: Not min (but root)

① Decrease key

② Potentiellt uppdatera min

case 2: Not root!

① Skapa en ny nod

② Flytta elementet till ny nod

③ Ursprunglig nod blir hollow

④ Vissa ättlingar flyttas.

↳ Se rank invariant ③.

* Delete *

case 1: Not root! -

① Make hollow

case 2: Root! -

① Make hollow

② Gå igenom hollow root nodes

⇒ Varje barn blir rootnode

③ Delete hollow rootnodes.

④ Link nodes of same rank

⇒ Resulterande parent får rank+=1