Konstrukcija i analiza algoritama
vežbe 7

Nina Radojičić
nina@matf.bg.ac.rs

Matematčki fakultet

novembar 2016.

Sadržaj

Sadržaj

Sadržaj

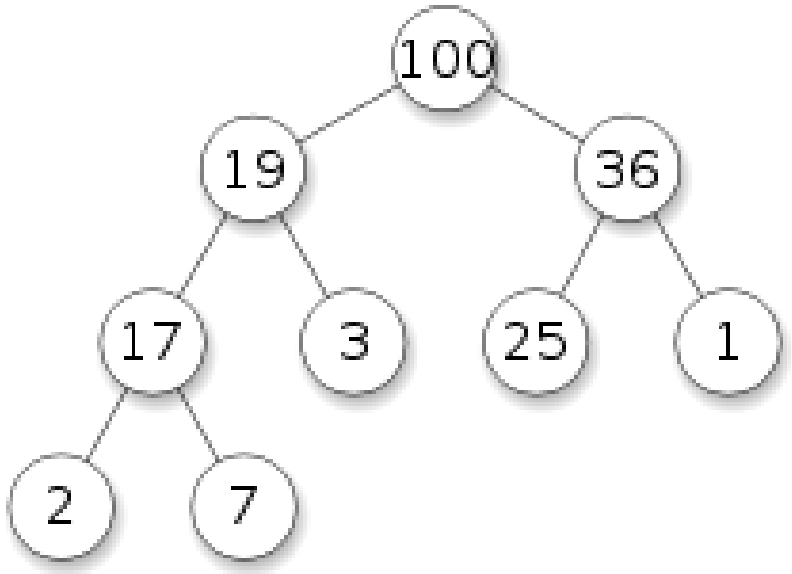
Sadržaj

Hip

Hip je binarno stablo koje zadovoljava uslov hipa: ključ svakog čvora veći je ili jednak od ključeva njegovih sinova.

Pored ključa, čvor obično sadrži i druge podatke.

Hip se još naziva i lista sa prioritetom.



Hip - predstavljanje u računaru

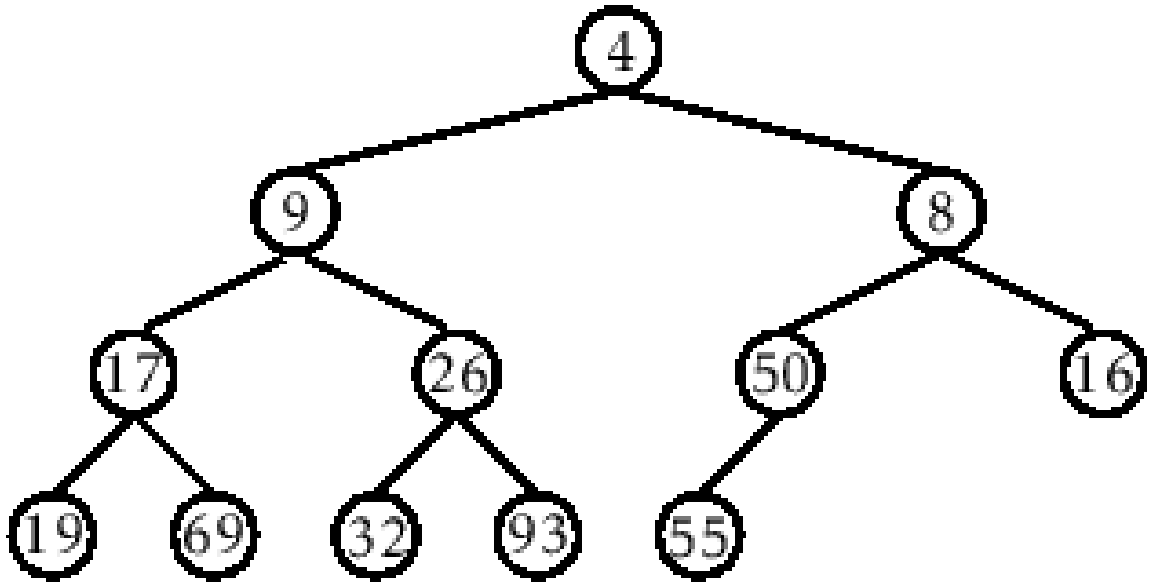
Hip se može realizovati implicitno i eksplicitno.

Algoritmi koji su ovde predstavljeni rade sa **implicitnom** realizacijom hipa.

Implicitna realizacija podrazumeva da ukoliko hip ima n elemenata, onda se za smeštanje elemenata koriste lokacije u nizu A sa indeksima $A[1..n]$, tako da ako element indeksa i predstavlja čvor stabla, tada
- element indeksa $2 * i$ predstavlja levo dete čvora
- element indeksa $2 * i + 1$ predstavlja desno dete čvora.

Na primer, deca čvora $A[1]$ su elementi $A[2], A[3]$. Gledano obratno, roditelj elementa sa indeksom j je čvor sa indeksom $\lfloor j/2 \rfloor$.

Hip - primer implicitna realizacija



4	9	8	17	26	50	16	19	69	32	93	55			

Uklanjanje najvećeg elementa iz hipa

Neka je dat hip A sa n elemenata. Treba ukloniti ključ $A[1]$, a potom transformisati niz A tako da opet zadovoljava svojstvo hipa. Najzgodnije je zameniti vrednosti elemenata $A[1]$ i $A[n]$ i smanjiti veličinu hipa za 1. Problem je što tada možda nije zadovoljeno pravilo hipa. Dovođenje niza u stanje koje zadovoljava pravilo hipa vrši se ponavljanjem sledeća 2 koraka:

1. Analizirati decu korena hipa koji se trenutno posmatra i odrediti koje dete je veće.
2. Ako je ključ korena veći od ključa većeg deteta postupak je završen; inače zameniti ključeve u korenu i većem detetu i preći na korak 1 ovog puta sa manjim hipom (to je hip u koji se prethodni koren spustio).

Uklanjanje najvećeg elementa iz hipa

Neka je dat hip A sa n elemenata.

- Ukloniti ključ $A[1]$.
- Transformisati niz A tako da opet zadovoljava svojstvo hipa. Najzgodnije je zameniti vrednosti elemenata $A[1]$ i $A[n]$ i smanjiti veličinu hipa za 1. Problem je što tada možda nije zadovoljeno pravilo hipa.
  Dovođenje niza u stanje koje zadovoljava pravilo hipa vrši se ponavljanjem sledeća 2 koraka:
  1. Analizirati decu korena hipa koji se trenutno posmatra i odrediti koje dete je veće.
  2. Ako je ključ korena veći od ključa većeg deteta postupak je završen; inače zameniti ključeve u korenu i većem detetu i preći na korak 1 ovog puta sa manjim hipom (to je hip u koji se prethodni koren spustio).

Uklanjanje najvećeg elementa iz hipa - primer

Postupak uklanjanja korena iz hipa koji je implicitno predstavljen nizom 8 4 6 3 1 2 5 je:

8 4 6 3 1 2 5

Uklanjanje najvećeg elementa iz hipa - primer

Postupak uklanjanja korena iz hipa koji je implicitno predstavljen nizom 8 4 6 3 1 2 5 je:

8 4 6 3 1 2 5

5 4 6 3 1 2 8

Uklanjanje najvećeg elementa iz hipa - primer

Postupak uklanjanja korena iz hipa koji je implicitno predstavljen nizom 8 4 6 3 1 2 5 je:

8 4 6 3 1 2 5
5 4 6 3 1 2 8
5 4 6 3 1 2

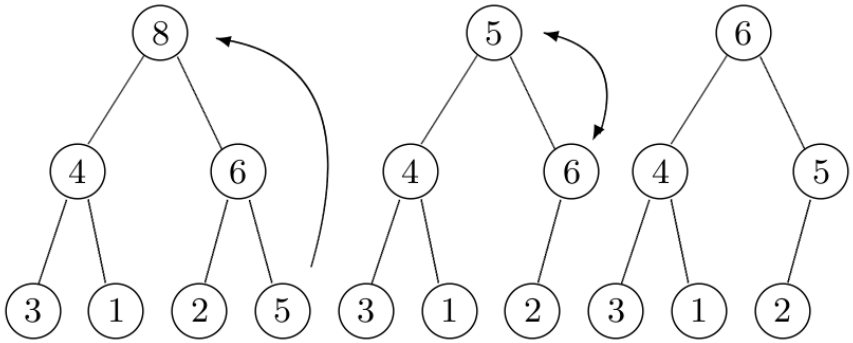
Uklanjanje najvećeg elementa iz hipa - primer

Postupak uklanjanja korena iz hipa koji je implicitno predstavljen nizom 8 4 6 3 1 2 5 je:

8 4 6 3 1 2 5
5 4 6 3 1 2 8
5 4 6 3 1 2
6 4 5 3 1 2

Uklanjanje najvećeg elementa iz hipa - primer

isti postupak može predstaviti i eksplicitno, tj. stablima:



Upis elementa u hip

Neka je dat hip A sa n elemenata i element x koji je potrebno dodati u hip. Ako želimo dodati element u prioritetni red možemo ga dodati na kraj niza A. To je ekvivalentno dodavanju krajnjeg desnog lista u stablu. Ta operacija ima za posledicu da se n uveća za 1, ali i da možda stablo ne zadovoljava pravilo hipa. Da bi se zadovoljilo pravilo hipa koriste se naredna 2 koraka:

1. Ako je ključ roditelja čvora koji sadrži element x manji od x, zamenjuje se sadržaj ova dva čvora.
2. Ako je ključ roditelja praznog čvora veći od elementa x, onda je postupak završen. Inače preći na korak 1.

Upis elementa u hip - primer

Prikaz ovog postupka sa stablom koje je implicitno zadato nizom 6 4 5 3 1 2, u koje se umeće broj 9 je:

6 4 5 3 1 2

Upis elementa u hip - primer

Prikaz ovog postupka sa stablom koje je implicitno zadato nizom 6 4 5 3 1 2, u koje se umeće broj 9 je:

6 4 5 3 1 2
6 4 5 3 1 2 9

Upis elementa u hip - primer

Prikaz ovog postupka sa stablom koje je implicitno zadato nizom 6 4 5 3 1 2, u koje se umeće broj 9 je:

6 4 5 3 1 2
6 4 5 3 1 2 9
6 4 9 3 1 2 5

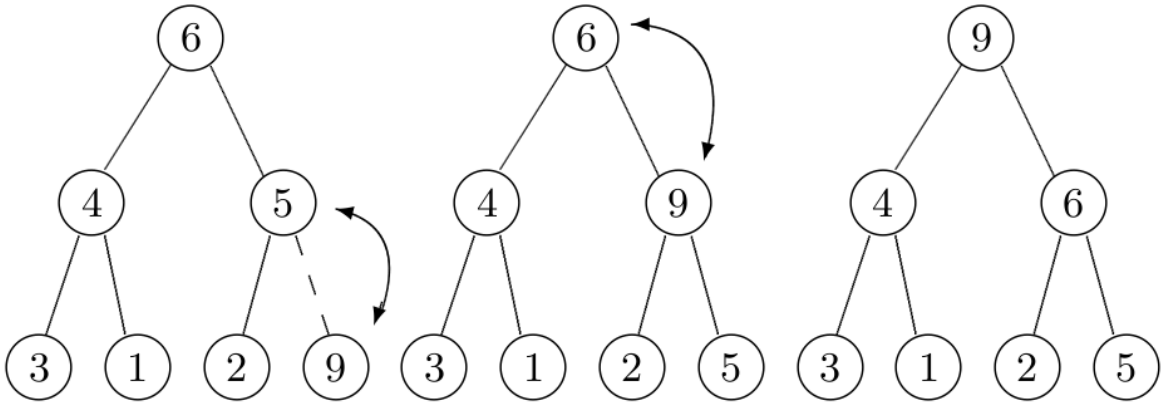
Upis elementa u hip - primer

Prikaz ovog postupka sa stablom koje je implicitno zadato nizom 6 4 5 3 1 2, u koje se umeće broj 9 je:

6 4 5 3 1 2
6 4 5 3 1 2 9
6 4 9 3 1 2 5
9 4 6 3 1 2 5

Upis elementa u hip - primer

Isti postupak se može predstaviti i eksplicitnim stablima:



Zadaci

Zadatak 1

Odrediti izgled implicitno predstavljenog hipa koji se dobija umetanjem redom brojeva 5, 1, 3, 9, 6, 4, 7, 8 polazeći od praznog hipa. Prikazati izgradnju hipa tabelom, čiji svaki red odgovara jednoj promeni hipa. Zatim odrediti izgled hipa dobijenog uklanjanjem najvećeg elementa.

Hipsort

Algoritam se sastoji iz dva dela:

1. Elementi iz niza dimenzije n se smeštaju u hip. Ovo se može uraditi na dva načina:
   1. pristupom odozgo-nadole čija je složenost $O(n \log n)$ (n dodavanja, svako složenosti $O(\log n)$).
   2. pristupom odozdo-nagore čija je složenost $O(n)$ (objašnjenje u profesorovoj knjizi).
2. Elementi se uklanjaju iz hipa, a svako uklanjanje ima složenost proporcionalnu visini hipa, tj. $O(\log n)$ pa pošto niz ima n elemenata ukupna složenost je $O(n \log n)$.

Zadaci

Zadatak 1

Koristeći pristupe odozgo-nadole i odozdo-nagore algoritmom hipsort sortirati niz brojeva 5 3 8 6 7 2 1 9 4.

REŠENJE (PRISTUP ODOZGO-NADOLE)

1. deo - pravljenje hipa odozgo-nadole (elementi se redom dodaju u hip gore opisanim algoritmom)

5
5 3
5 3 8
8 3 5
8 3 5 6
8 6 5 3
8 6 5 3 7
8 7 5 3 6
8 7 5 3 6 2

REŠENJE (PRISTUP ODOZGO-NADOLE)

1. deo - pravljenje hipa odozgo-nadole (elementi se redom dodaju u hip gore opisanim algoritmom)

8 7 5 3 6 2 1
8 7 5 3 6 2 1 9
8 7 5 9 6 2 1 3
8 9 5 7 6 2 1 3
9 8 5 7 6 2 1 3
9 8 5 7 6 2 1 3 4

2. deo (uklanjanje najvećeg elementa iz hipa dok se ne uklone svi elementi):

9 8 5 7 6 2 1 3 4
4 8 5 7 6 2 1 3|9
8 4 5 7 6 2 1 3|9
8 7 5 4 6 2 1 3|9
3 7 5 4 6 2 1|8 9
7 3 5 4 6 2 1|8 9
7 6 5 4 3 2 1|8 9
1 6 5 4 3 2|7 8 9
6 1 5 4 3 2|7 8 9
6 4 5 1 3 2|7 8 9

2. deo (uklanjanje najvećeg elementa iz hipa dok se ne uklone svi elementi):

2 4 5 1 3|6 7 8 9
5 4 2 1 3|6 7 8 9
3 4 2 1|5 6 7 8 9
4 3 2 1|5 6 7 8 9
1 3 2|4 5 6 7 8 9
3 1 2|4 5 6 7 8 9
2 1|3 4 5 6 7 8 9
1|2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9

Određivanje medijane

Medijana prestavlja srednji po veličini element u datom nizu brojeva. Ukoliko je broj elemenata niza paran, medijanom smatramo aritmetičku sredinu dva po veličini središnja elementa.

Zadatak je da se korišćenjem 2 hipa postigne da se određivanje medijane niza izvršava u konstantnom vremenu, pri čemu se u niz mogu dodavati novi elementi.

Određivanje medijane

Ovaj zadatak se rešava tako što se održavaju 2 hipa.
Ukoliko niz ima paran broj elemenata onda hipovi sadrže po polovinu ovih elemenata a ukoliko je broj elemenata neparan dozvoljeno je da jedan od hipova ima tačno jedan element više od drugog. **U prvi hip smeštamo manje, a u drugi veće brojeve.** Prvi hip je takav da je broj u svakom čvoru veći ili jednak od brojeva u sinovima a u drugom hipu je broj u čvoru uvek manji ili jednak od brojeva u sinovima. To znači da se u slučaju neparnog broja elemenata u hipu koji sadrži jedan element više nalazi medijana. U slučaju parnog broja elemenata, medijana se računa po formuli (koren1+koren2)/2.

Kako dodati novi element u niz a zadržati svojstva hipova? Razlikujemo više slučajeva:

1. ako je element manji od korena prvog hipa, dodati ga u prvi hip.
2. ako je element veći od korena drugog hipa, dodati ga u drugi hip.
3. ako je element veći od korena prvog hipa, a manji od korena drugog hipa: u slučaju različitog broja elemenata hipova, dodati ga u hip sa manje elemenata; u slučaju istog broja elemenata dodati ga u proizvoljan hip.

Ukoliko je posle dodavanja elementa nastala situacija da jedan hip ima 2 elementa više od drugog hipa, onda iz hipa sa više elemenata ukloniti koren i dodati ga u manji hip.

Heš tabele

Heš tabele se koriste kada je potrebno efikasno obavljati operacije dodavanja, pretrage i brisanja elemenata.

Ideja:

Koristićemo više prostora nego što nam je potrebno ali sa ciljem da imamo efikasne operacije.

Heš funkcija

Dobra heš funkcija treba da ravnomerno slika skup od n ključeva u skup slučajnih lokacija iz skupa $\{0, ..., m-1\}$.

Pri tome se smatra da broj lokacija nije mnogo veći (npr. duplo je veći) od broja ključeva koje unosimo (m nije mnogo veće od n).

3 heš metode:

1. L - *lančanje* ili *odvojeno nizanje* (chaining): funkcijom h(x) = h1(x)
2. LP - *linearno popunjavanje* (linear-probing) funkcijom
   h(x,i) = (h1(x)+i) % n, i=0..n-1
3. DH - *dvostruko heširanje* (double hashing) funkcijom h1 kao heš funkcijom i funkcijom g kao sekundarnom
   h(x,i) = (h1(x) + i·g(x)) % n, i=0..n-1

Rođendanski paradoks

Godina ima 365 dana. Ako je rođendan svake osobe odabran na slučajan način, onda je potrebno 23 ljudi da bi veroatnoća da među njima postoje dve osobe rođene na isti dan dostigla 50%.

Dakle, ukoliko imamo 365 pozicija u heš tabeli i 23 elementa koje treba da smestimo u tabelu, verovatnoća je veća od 50% da ćemo imati barem jednu koliziju.

Među 70 ljudi, verovatnoća da će među njima biti dve osobe rođene na isti dan je 99.99% (ovo bi odgovaralo popunjenosti od manje od 20% u heš tabeli).

Dakle, kolizije je u praksi nemoguće izbeći.

Zadatak

Skicirajte heš tabelu nakon smeštanja ključeva iz skupa 22, 1, 13, 11, 24, 33, 18, 42, 31 u zadatom poretku. Pretpostavite da je tabela veličine n=11, da je primarna heš funkcija h1(x) = x % 11, kao i da se za razrešavanje kolizija koristi lančanje, linearno popunjavanje i dvostruko heširanje sekundarnom heš funkcijom g(x) = 1+ x %10.

REŠENJE:

	L	LP	DH
0	$22 \leftarrow 11 \leftarrow 33$	22	22
1	1	1	1
2	$13 \leftarrow 24$	13	13
3		11	
4		24	11
5		33	18
6			31
7	18	18	24
8			33
9	$42 \leftarrow 31$	42	42
10		31	

Pretraživanje

Izvodi se tako što se pomoću heš funkcije dobije moguća pozicija elementa.

- Kod lančanja se zatim u listi traži dati element.
- Kod linearnog popunjavanja se pretražuju redom sledeće pozicije dok element ne bude pronađen ili dok se ne dođe do prazne pozicije.
- Kod duplog heširanja se pomoću koraka (vrednosti druge heš funkcije) prolazi kroz tabelu dok se element ne nađe ili se ne dođe do prazne pozicije.

Odabir heš funkcija

Najteži zadatak kod heširanja je odabir dobrih heš funkcija.

Potrebno da je da heš funkcija ravnomerno raspoređuje podatke po celoj tabeli kao i da bude laka za izračunavanje. Ne postoji neko opšte pravilo kako konstruisati dobru heš funkciju, već sve zavisi od slučaja do slučaja (tj. najviše od očekivanih podataka sa kojima će se raditi).

Kaže se da je dobar izbor heš funkcije "više umetnost nego nauka".

Najčešće se podaci preslikavaju u neki broj, a zatim se indeks smeštanja određuje nekom funkcijom koja vraća ostatak po modulu nekog prostog broja.

Složenost

Iako je vreme za sve operacije (unos, pretraga, brisanje) u najgorem slučaju $O(n)$, pri dobrom izboru heš funkcije se dobija da je prosečno vreme ovih operacija $O(1)$!!!

Primer

Struktura *korisnik* sadrži tri polja: ime, prezime i broj telefona korisnika. Potrebno je omogućiti brzu pretragu, unos i brisanje po imenu i prezimenu. Napisati C funkciju koja slika ime i prezime korisnika u prirodan broj između 0 i n. Ako niz a dimenzije k sadrži spojeno ime i prezime korisnika malim slovima, onda je potrebno tog korisnika preslikati u poziciju $(\sum_{i=0}^{n}(a[i] -' a'))\%n$ ($'a'$ je ASCII kod karaktera a). Napisati C funkciju koja smešta strukturu u heš tabelu veličine n pomoću heš funkcije $h(x) = x\%n$ a za razrešavanje kolizije koristi linearno popunjavanje.

Primer

Mana navedenog heširanja je da se dve osobe sa istim imenom neće razlikovati, izuzev po broju telefona. Da bi se razlikovale, potrebno je da informacije budu potpunije, npr. uz ime i prezime da bude prisutna i adresa osobe.

Brisanje

Uključivanje mogućnosti brisanja obično utiče na to da i ostale operacije (pretraga i dodavanje) budu komplikovanije.

Lančanje

Izvodi se tako što se prvo traži element, pa se briše ako je pronađen. Kod lančanja je to jednostavno, samo se obriše čvor.

Brisanje - Druga dva pristupa

Obično se stavlja indikator za svaku poziciju u tabeli: `puna/prazna/prazna_nakon_brisanja`. Ovo može podrazumevati dodatan niz indikatora. Pri pretrazi se prvo proverava indikator, i razlikuju se tri slučaja:

1. ako je pozicija `puna` proverava se broj u tabeli, ili je pronađen traženi broj i vraća se njegova pozicija ili je potrebno preći na narednu poziciju koju treba ispitati.
2. ako je pozicija `prazna` element nije nađen i pretraga se prekida.
3. ako je pozicija `prazna_nakon_brisanja` preći na narednu poziciju koju treba ispitati.

Pri dodavanju se samo proverava indikator i dok god je `puna`, prelazi se na narednu poziciju..
Pri brisanju prvo se vrši pretraga. Ako je element pronađen - dovoljno je indikator te pozicije promeniti da bude `prazna_nakon_brisanja`.
Kada broj pozicija `prazna_nakon_brisanja` postane veliki, može se izvršiti ponovno heširanje.