Crveno-crna stabla

Seminarski rad u okviru kursa
Konstrukcija i analiza algoritama 2
Matematički fakultet

Nikola Dimitrijević, 1086/2017
nikoladim95@gmail.com

17. januar 2019

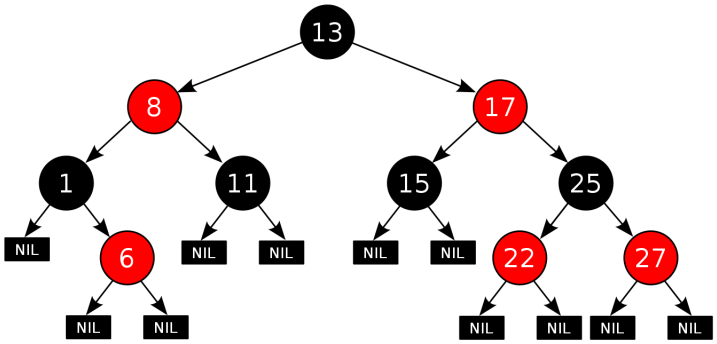
Sažetak

Crveno-crna stabla su vrsta binarnih uređenih samobalansirajućih stabala. To je struktura podataka koja garantuje brze operacije umetanja, pretrage i brisanja. Svaki čvor stabla ima boju: crnu ili crvenu. Uz određena pravila i načine na koje se menja stablo dobija se gornja granica za koliko stablo može biti nebalansirano.

Sadržaj

1 Osnovno

Crveno-crna stabla su struktura podataka koja omogućava brze operacije umetanja, brisanja i pretrage. Spadaju u binarna samobalansirajuća pretraživačka stabla. Svaki čvor ima dodatan bit (ili već neki tip podatka) koji reprezentuje boju čvora. Autori su se odlučili za crvenu i crnu boju jer je, pored klasične crne, crvena najbolje izgledala na tadašnjim laserskim štampačima [1]. Još jedan razlog je to što su imali crvene i crne hemijske pa su tako crtali ova stabla.



Slika 1: Primer crveno-crnog stabla.

Za razliku od mnogih stabala, u implementacijama crveno-crnih stabala se umesto uobičajenih null pokazivača koriste specijalni null čvorovi (na slici 1 su označeni sa NIL) koji ne sadrže nikakve podatke koji se unose u strukturu. Iako je moguće implementirati sve operacije crveno-crnih stabala korišćenjem običnih null pokazivača, ovako se uprošćava njihova implementacija. Da bi se uštedelo na memoriji, umesto da postoji mnogo različitih null čvorova, moguće je imati jedan takav čvor u memoriji, a da svi ostali koji žele da pokazuju na null čvor pokazuju baš na samo tog jednog. Crveno-crna stabla se koriste za skladištenje podataka koji imaju uređenost, pošto se na osnovu poretka određuje pozicija elemenata u stablu.

2 Svojstva

- Svaki čvor je ili crn ili crven.
- Koren je crn.
- Svi listovi su crni.
- Ako je čvor crven, onda su mu oba sina crna.
- Svaki put od korena do svih NIL listova ima isti broj crnih čvorova.

Na osnovu datih svojstava može se i intuitivno pokazati zašto ova stabla ostaju balansirana.

Svaki put od korena do lista ima isti broj crnih čvorova. Jedini način da se poveća broj čvorova na tom putu jeste da se ubace crveni čvorovi. Međutim, kako ne smeju biti dva crvena čvora jedan za drugim, to znači da se bilo koja putanja od korena do lista može udvostručiti ako bi se nakon svakog crnog čvora ubacio crven čvor. To bi u najgorem slučaju

udvostručilo visinu tog podstabla, ali je upravo to i najviše što stablo može biti udaljeno od potupne balansiranosti. Dakle u najgorem slučaju je jedno podstablo dva puta dublje od svog brata podstabla.

Crna dubina stabla je broj crnih čvorova od korena do listova, koji je za svaki put od korena do lista isti.

3 Operacije

3.1 Pretraga

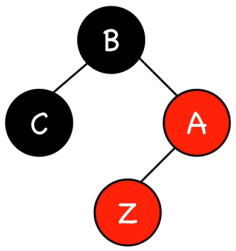
Počevsi od korena, ako je trenutni čvor trazen: gotovo, inače ako je traženi manji od trenutnog rekurzivno se pretražuje levo podstablo, inače se rekurzivno pretražuje desno podstablo.

3.2 Umetanje

Deda čvora je od čvorovog oca otac.

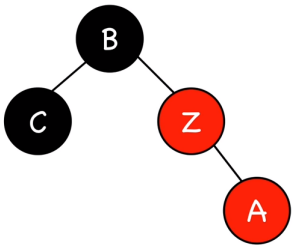
Ujak čvora je od dede čvora drugi sin, odnosno onaj sin koji nije roditelj čvoru.

Trougao formacija je kada je čvor levi sin oca, a otac desni sin dede, ili slučaj u ogledalu (videti primer na slici 2).



Slika 2: Trougao formacija.

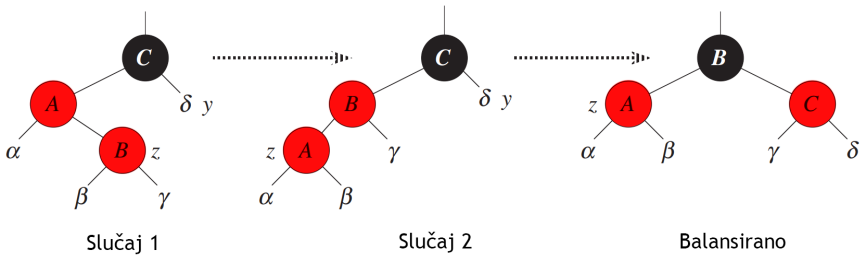
Linija formacija je kada je čvor levi sin oca i otac levi sin dede, ili slučaj u ogledalu (videti primer na slici 3).



Slika 3: Linija formacija.

Postupak umetanja:

- Kada se dodaje nova vrednost u stablo, uvek se prvo nov čvor oboji u crveno.
- Spušta se kroz stablo do listova, i onda se umeće kao potomak.
- Ako je roditelj crn, onda kraj
- Ako je novi čvor koren, oboji ga u crno.
- Ako mu je otac crn, onda se ne radi ništa, jer je stablo i dalje ispravno.
- Ako mu je ujak crven, onda oboji ujaka i oca u crno, a dedu u crveno. Obradi dedu rekurzivno.
- Ako mu je ujak crn i u trouglu su, onda se rotira otac čvora u suprotnu stranu od sina (videti slučaj 1 na slici 4).
- Ako mu je ujak crn i u liniji su, onda se deda čvora rotira u suprotnu stranu od unuka (videti slučaj 2 na slici 4).



Slika 4: Postupak rotiranja, preuzeto iz [4].

3.3 Brisanje

Brisanje je komplikovanije od pretrage i umetanja.

- Prvo se traži čvor koji se briše. Ako nije prisutan kraj.
- Ako nađeni čvor ima oba sina, onda prvo mora da se svede na prostiji slučaj, gde ima 0 ili 1 sina.
- Nađe se sledbenik trenutnog čvora u stablu, odnosno najmanji element u desnom podstablu.
- Čvor koji se briše zameni mesto sa njegovim sledbenikom.
- Njegov sledbenik ima najviše jednog potomka, pošto da je imao levog potomka, onda bi on bio sledbenik.
- Sada čvor koji se briše ima najviše jednog potomka.
- Ako nema nijednog potomka i crven je čvor, onda se samo obriše.
- Inače, pošto je čvor crn ne može samo da se obriše jer bi se smanjila crna dubina.
- Ako čvor ima jedno crveno dete, umesto sebe stavi dete i obriše sebe.
- Inače se prolazi kroz šest specifičnih i zamršenih koraka. Za detalje pogledati [3] [4].

4 Upoređivanje sa AVL stablima

Crveno-crna stabla su, kao i AVL stabla, samobalansirajuća i uređena. Oba pružaju $O(\log n)$ vreme pretrage, umetanja i brisanja. AVL stabla su bila popularna pre nego što su crveno-crna stabla postala poznata.

Za razliku od AVL stabla, teža su za implementaciju. Pre svega zbog svih detalja implementacije šest različitih slučajeva popravke nakon brisanja. Oba stabla imaju linearnu memorijsku složenost.

AVL stabla imaju jaču garanciju balansiranosti. Kod njih je razlika dubina levog i desnog podstabla najviše jedan, a crveno-crna garantuju da razlika nije više od duplo.

Prednost crveno-crnih stabla je što garantuju $O(1)$ rotacija po operaciji umetanja. Ta stvar zaista utiče na performanse u pravim implementacijama.

Crveno-crna stabla garantuju da jedna strana stabla nije više od duplo duža od druge.

AVL stabla imaju 4 rotacije. LR, LL, RR, RL. Crveno-crna stabla imaju samo levu i desnu rotaciju. Međutim, postoje neki slučajevi kada se briše element u kojima je potrebna dodatna obrada kako bi stablo ostalo balansirano. Ima 6 mogućih slucajeva. Ide se od prvog do poslednjeg, usput primenjujući pravila ako je moguće. Postoje tri terminirajuća pravila, tj. ako je njihov uslov zadovoljen, onda se odradi potrebno ažuriranje i prekida prolazak kroz ostale slučajeve.

Pošto su i crveno-crno stablo i AVL stablo stabla pretrage, na uobičajen način se vrši pretraga.

4.1 Zaključak

Crveno-crna stabla su danas najpopularniji izbor implementacije samobalansirajućih binarnih stabala.

Primeri korišćenja crveno-crnih stabala:

- Java: java.util.TreeMap, java.util.TreeSet
- C++ STL: map, multimap, multiset [5]
- Linux jezgro: Potpune fer raspoređivač, linux/rbtree.h
- U funkcionalnom programiranju za implementaciju postojanih struktura podataka. Tada svaka operacija brisanja i umetanja ima dodatnu memorijsku složenost $O(\log n)$ kako bi moglo da se rekonstruišu ranija stabla [2].

Na slici 5 je nacrtan grafik na kome se vidi vreme izvršavanja algoritma za različite ulaze. Ulaz označava koliko će se nasumičnih brojeva ubaciti u stablo. Vreme izvršavanja počinje da bude osetno tek za ulaze od oko 90000. Za veće ulaze se jasnije vidi kako raste potrebno vreme. Kada je u pitanju ulaz veličine pola miliona tada je potrebno 11 sekundi, a za milion 25 sekundi. To je tek nešto sporije od očekivanog odnosa vremena za algoritam linearne složenosti, što je razuman reuzltat s obzirom da je složenost ubacivanja n elemenata $O(nlogn)$.

Literatura

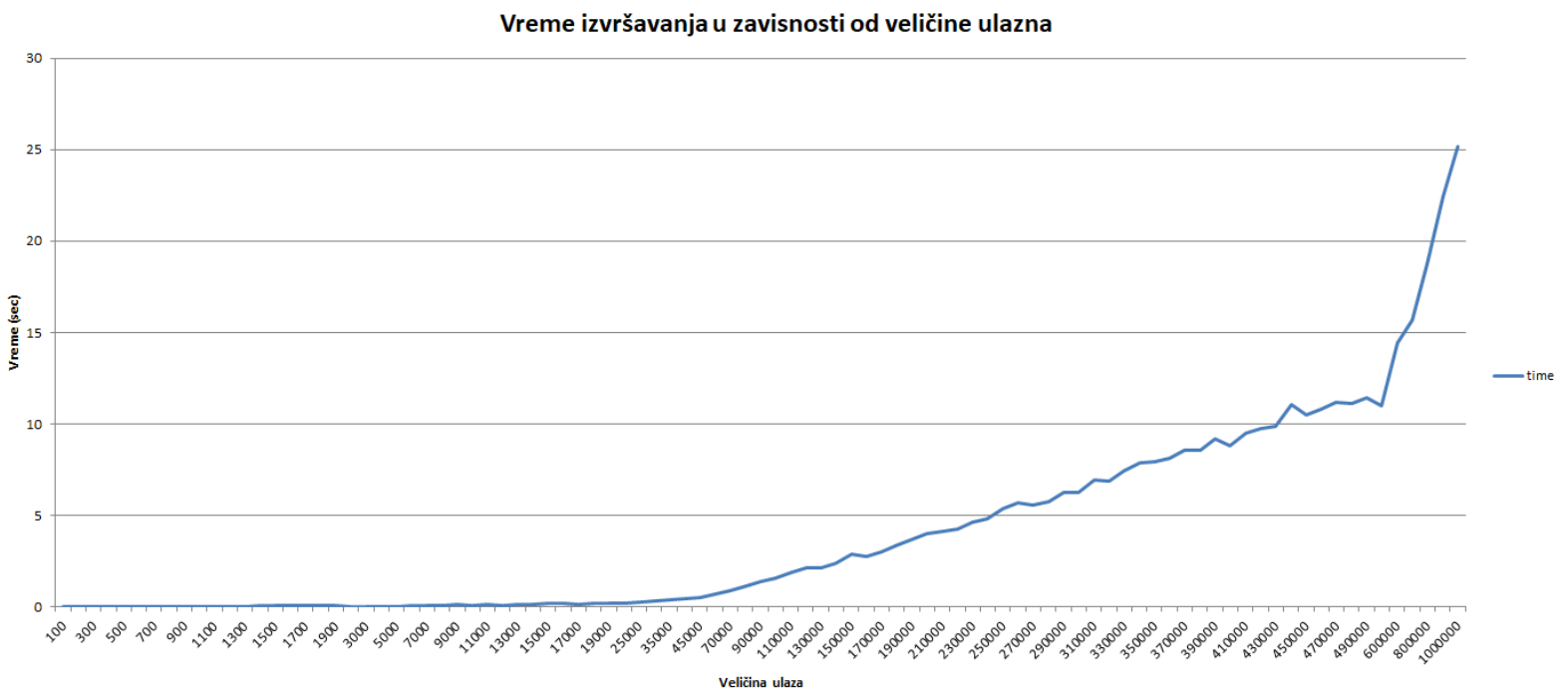
[1] Name origin. at: https://en.wikipedia.org/wiki/Red%E2%80%93black_tree.

[2] Persistent data structures. at: https://en.wikipedia.org/wiki/Persistent_data_structure.

[3] Red Black Tree Deletion. at: https://www.youtube.com/watch?v=CTvfzU_uNKE.

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

[5] Free Software Foundation. GNU gcc. at: http://gcc.gnu.org/.



Slika 5: Testiranje algoritma, skala x ose raste brže od linearno.