PROJEKTOVANJE I ANALIZA ALGORITAMA

Laboratorijska vežba 4
Ristovski Nikola 19347
Tehnologija: C#

Korišćeni resursi

Kako bih implementirao binomni heap, koristio sam prezentaciju sa CS-a, i konsultovao sajt https://www.geeksforgeeks.org/binomial-heap-2/.

Opis rešenja

Binomni heap je bolji u odnosu na običan jer svaki drugi put dodajemo čvor bez potrebe da išta menjamo u heap-u. Moja implementacija sadrži jedan bool flag koji označava je li reč o MIN ili MAX heap-u, kako bih izbegao redudantno pisanje 2 klase, jednu za MIN a jednu za MAX heap. U odnosu na taj flag imam i drugačije poređenje (implementirano u posebnoj funkciji Compare) u specifičnim funkcijama kao što su DecreaseIncreaseKey (f-ja koja decrease-uje ključ u MIN, a increase-uje ključ u MAX heap-u) i sl.

Moj zadatak je bio da snimim niz kao MAX heap, a da ga naknadno transformišem u MIN heap. Veoma je nepraktično iz Max heap-a izbacivati minimalne elemente jer bismo morali da prolazimo kroz praktično svaki element. Zato ja prvo implementiram min heap, potom generišem uz njegovu pomoć niz od N elemenata i na kraju taj min heap pretvaram u max heap. Kako mi zadatak glasi da MAX heap moram pretvoriti u MIN heap, ja opet taj max heap pretvaram u min heap. Malo je besmisleno ovo raditi, ali je i dobro da pokaze kako je veoma slično i gotovo jednako vreme potrebno da se iz MIN heap-a transformiše u MAX heap i obrnuto! Nadam se da nisam previse pogresio tematiku zadatka!

Rezultati izvršenja sa različitim ulazima

U tabeli prikazanoj na sledećoj stranici su dati rezultati u milisekundama izvršenja formiranja heap-a i naknadno pretvaranja MIN u MAX heap. U rezultatima u konzoli prikazano je i vreme potrebno da se opet iz MAX pretvori u MIN heap, ali s obzirom da pokazuje skoro identično vreme kao i prethodno, verujem da dolazi do neke vrste keširanja pa se zato gotovo instantno pretvori iz max u min heap.

Prikazano je dakle vreme potrebno od početka do kraja generisanja min heap-a veličine N, pa potom vreme od početka generisanja do kraja transformacije u MAX heap, pa je vreme potrebno samo za transformaciju jednako razlici ova dva vremena.

K \ N		1.000	10.000	100.000	1.000.000	10.000.000
10	GH	12.7737	72.6218	681.2003	5949.0128	41499.2597
	GH + mTM	15.5667	99.8272	1082.7187	10297.8941	76455.4773
	GH + mTM + MTm	16.2021	109.1883	1194.9225	11136.9985	85187.0409
20	GH	6.6396	25.1741	475.1289	3814.2584	32527.7634
	GH + mTM	8.2074	48.0907	900.985	8361.4718	71115.7577
	GH + mTM + MTm	8.8389	55.1313	1026.4178	9705.9214	80730.1121
50	GH	4.4055	20.5417	219.3457	3044.0993	25271.2581
	GH + mTM	6.2951	47.5313	610.499	7786.2154	66504.0016
	GH + mTM + MTm	6.9523	55.9664	747.9123	9146.7363	75956.2696
100	GH	1.5043	15.6774	181.9735	2116.4827	19585.157
	GH + mTM	3.155	42.7431	604.9045	7062.1665	60118.5179
	GH + mTM + MTm	3.9989	55.7238	717.2343	8503.4378	69692.9632

GH – Generacija heap-a;
mTM – transformacija min u max heap;
MTm– transformacija max u min heap;
*svi heap-ovi su generisani od integera iz opsega [1-10.000]

Zaključak

Kako je kompleksnost insert operacije kod binomnog heap-a u najgorem slučaju O(logn), možemo primetiti da rezultati testiranja mog koda otprilike prati tu zavisnost. Varijacije u vremenima su posledica upravo prirode heap-a, jer je O(logn) WORST CASE, a ne prosečni, a takođe i zbog drugih runtime uticaja. Pretvaranje min u max heap i obrnuto takođe uz varijacije zadržava ovu pravilnost.

Ristovski Nikola 19347

Elektronski fakultet, Niš, 2024