

Vardas: Emilis

Pavardė: Keras

Grupė: Prif21/2

**Algoritmai ir duomenų struktūros**

**Savarankiškas darbas nr. 3. – rūšiavimo algoritmai.**

10 var.

Algoritmai:

* Šelo (angl. Shell sort) su originalia atstumų seka.
* Piramidės (angl. heapsort).

Tikslas – nustatyti ir analizuoti algoritmų rūšiavimo greitį įtakojančius veiksnius.

Testai:

1. Greičio testavimas atsitiktinių skaičių aibėje ir rezultatų analizė.
2. Greičio testavimas mažėjimo tvarka sugeneruotų atsitiktinių skaičių aibėje ir rezultatų analizė.

**Testavimo aplinka.**

Kompiuterio parametrai:

AMD Ryzen 9 5900HX 8 core Procesorius

32gb 3200 Mhz operativioji atmintis

1TB PCIe Gen3 x4 NVMe iki 2400MB/s

Programavimo kalba: Java

Duomenų struktūra, kurioje laikomi duomenys: masyvas

Kita:

**Testas nr. 1.**

**Greičio testavimas atsitiktinių skaičių aibėje ir rezultatų analizė.**

**Testavimo eiga.**

1. Pasirinktas pirmas duomenų kiekis – 6000000 Naudojant java.util.ramdom klasę buvo sugeneruotos 5 atsitiktinių skaičių aibės su galimomis reikšmemis: nuo -2147483648 iki 2147483647. Jos buvo saugomos masyvuose.
2. Kiekviena atsitiktinių skaičių aibė buvo paduota abiem rūšiavimo algoritmams 10 kartų. Aibių surūšiavimo laikai buvo išsaugoti, pateikti lentele ir buvo paskaičiuotas visų surūšiavimo laikų vidurkis.
3. 1 ir 2 punktai buvo pakartoti su duomen kiekiais: 12000000, 24000000, 48000000, 96000000
4. Bendri testavimo rezultatai buvo pateikti grafiku ir buvo atlikta rezultatų analizė.

**Rezultatai**.

Rezultatai pateikti lentelėmis ir grafiku.

**Rezultatų pateikimas lentele.**

1. lentelė. Rezultatai, kai n=6000000

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Testų laikai milisekundėmis | | | | | | | | | |  |
| Aibė 1 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 | Vid. |
| Shell Sort | 1413 | 1325 | 1323 | 1427 | 1395 | 1361 | 1386 | 1616 | 1395 | 1405 | 1405 |
| Heap Sort | 1827 | 1784 | 1798 | 1802 | 1740 | 1646 | 1581 | 1615 | 1603 | 1675 | 1707 |
| Aibė 2 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 1373 | 1397 | 1566 | 1442 | 1522 | 1387 | 1381 | 1467 | 1392 | 1416 | 1434 |
| Heap Sort | 1626 | 1650 | 1637 | 1665 | 1654 | 1683 | 1672 | 1581 | 1598 | 1620 | 1639 |
| Aibė 3 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 1302 | 1287 | 1312 | 1311 | 1300 | 1333 | 1306 | 1287 | 1285 | 1299 | 1302 |
| Heap Sort | 1656 | 1620 | 1606 | 1616 | 1605 | 1624 | 1670 | 1660 | 1720 | 1719 | 1650 |
| Aibė 4 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 1333 | 1488 | 1392 | 1415 | 1381 | 1290 | 1323 | 1322 | 1380 | 1403 | 1373 |
| Heap Sort | 1638 | 1583 | 1689 | 1601 | 1586 | 1624 | 1639 | 1637 | 1643 | 1712 | 1635 |
| Aibė 5 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 1378 | 1359 | 1425 | 1359 | 1398 | 1394 | 1439 | 1454 | 1437 | 1577 | 1422 |
| Heap Sort | 1681 | 1673 | 1660 | 1603 | 1632 | 1607 | 1664 | 1653 | 1650 | 1659 | 1648 |

2. lentelė. Bendri rezultatai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Laikai milisekundėmis, su kiekviena aibe atlikta 10 testų. | | | | |  |
| n=6000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 | Vidurkis |
| Shell Sort | 1405 | 1434 | 1302 | 1373 | 1422 | 1387.2 |
| Heap Sort | 1707 | 1639 | 1650 | 1635 | 1648 | 1655.8 |
| n=12000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 3092 | 2989 | 3177 | 2970 | 3016 | 3048.8 |
| Heap Sort | 3745 | 3792 | 3772 | 3722 | 3747 | 3755.6 |
| n=24000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 6690 | 6761 | 6843 | 6805 | 6747 | 6769.2 |
| Heap Sort | 8394 | 8394 | 8380 | 8412 | 8531 | 8422.2 |
| n=48000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 17787 | 17273 | 17506 | 16971 | 17371 | 17381.6 |
| Heap Sort | 20457 | 20383 | 20366 | 20343 | 20292 | 20368.2 |
| n=96000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 43373 | 43800 | 44596 | 43581 | 42944 | 43685.8 |
| Heap Sort | 45020 | 44991 | 44945 | 44879 | 41598 | 44286.6 |

**Rezultatų pateikimas grafiku.**

1 pav. Atsitiktinių skaičių rūšiavimo rezultatai.

**Rezultatų analizė.**

**1 teiginys – teoriniai įverčiai.**

Shell sort teoriniai įverčiai atitinka teorinius įverčius.

Heap sort teoriniai įverčiai atitinka teorinius įverčius.

Shell sort teorinis įvertis, kai duomenų aibė sudaryta iš atsitiktinių skaičių: : O(n log2n).

Duomenų kiekis buvo didinamas kiekvieną kartą du kartus, todėl:

Išvada: didinant duomenų kiekį du kartus, rūšiavimo laikas turėtų didėti apie 2 ± 1 kartus.

Tikriname:

Laikas visada kito ~2 kartus. Pateikta Shell sort algoritmo vidutinio laiko sudėtingumo formulė yra tiksli tik tada, kada yra mažas kiekis skaičių. Tačiau, kadangi visi skaičiai yra panašus, atitinka teorinį įvertį

Heap sort teorinis įvertis, kai duomenų aibė sudaryta iš atsitiktinių skaičių: O(n log2 n)

Duomenų kiekis buvo didinamas kiekvieną kartą du kartus, todėl:

Išvada: didinant duomenų kiekį du kartus, rūšiavimo laikas turėtų didėti apie 2 ± 1 kartus.

Tikriname:

Laikas visada kito ~2 kartus. Pateikta Heap Sort algoritmo vidutinio laiko sudėtingumo formulė yra tiksli. Visi skaičiai yra panašus, atitinka teorinį įvertį

**2 teiginys – greičių palyginimas.**

Rezultatai parodė, kad Shell sort yra greitesnis už Heap Sort, kai yra iki 75`000`000 skaičių. Kai yra daugiau skaičių rušiavimas, algoritmų laikai yra panašus ir skyriasi ±2 sekundes.

Pagrindimas.

*Shell sort ir Heap sort algoritmų vidutinis laiko sudėtingumas sutampa, kadangi pastebime, jog jų laiko sudėtingumo apskaičiavimas sutampa O(n log2 n ). Shell sort yra greitesnis, kai kiekis neviršiją (60 mln. – 75 mln) skaičių, kadangi shell sort algoritmui prireikia atlikti mažiau veiksmų.*

**3 teiginys – atminties valdymas.**

Heap sort rūšiavimas ir shell sort rūšiavimas yra rūšiavimo algoritmai, kuriuos galima naudoti duomenų sąrašams rūšiuoti. Tačiau jie skiriasi tuo, kaip rūšiavimo proceso metu valdo atmintį.

Heap sort rūšiavimas duomenims rūšiuoti naudoja duomenų struktūrą, vadinamą krūva (heap) . Krūva yra pilnas dvejetainis medis, atitinkantis krūvos savybę, kuri teigia, kad kiekvieno medžio mazgo vertė yra didesnė arba lygi jo vaikų reikšmėms. Ši savybė leidžia efektyviai rūšiuoti krūvą pakartotinai išimant didžiausią elementą iš krūvos ir įdedant jį į rūšiuojamo sąrašo pabaigą.

Kita vertus, shell sort rūšiavimas nenaudoja konkrečios duomenų struktūros, pavyzdžiui, krūvos. Vietoj to, jis lygina elementus, kurie yra tam tikru atstumu vienas nuo kito, ir sukeičia juos, jei jie nėra tinkama tvarka. Atstumas tarp lyginamų elementų vadinamas „tarpu“.

Kalbant apie atminties valdymą, heap sort rūšiavimo erdvės sudėtingumas yra O(1), nes jis rūšiuoja duomenis vietoje, nereikalaujant papildomos atminties. Kita vertus, shell sort rūšiavimo erdvės sudėtingumas yra O (n), nes norint išsaugoti tarpų seką ir lyginamuosius elementus, reikia papildomos atminties.

Apskritai, heap sort rūšiavimas yra efektyvesnis atminties valdymo požiūriu, palyginti su Shell rūšiavimu, nes duomenims rūšiuoti nereikia papildomos atminties.

**Testas nr. 2.**

**Greičio testavimas mažėjimo tvarka atsitiktinių skaičių aibėje ir rezultatų analizė**

**Testavimo eiga.**

1. Pasirinktas pirmas duomenų kiekis – 19000000 Naudojant java.util.ramdom klasę buvo sugeneruotos 5 atsitiktinių skaičių aibės su galimomis reikšmemis: nuo -2147483648 iki 2147483647. Surūšiuotos mažėjimo tvarka aibės buvo saugomos masyvuose.
2. Kiekviena atsitiktinių skaičių aibė buvo paduota abiem rūšiavimo algoritmams 10 kartų. Aibių surūšiavimo laikai buvo išsaugoti, pateikti lentele ir buvo paskaičiuotas visų surūšiavimo laikų vidurkis.
3. 1 ir 2 punktai buvo pakartoti su duomen kiekiais: 38000000, 76000000, 152000000, 304000000
4. Bendri testavimo rezultatai buvo pateikti grafiku ir buvo atlikta rezultatų analizė.

**Rezultatai**.

Rezultatai pateikti lentelėmis ir grafiku.

**Rezultatų pateikimas lentele.**

3. lentelė. Rezultatai, kai n=19000000

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Testų laikai milisekundėmis | | | | | | | | | |  |
| Aibė 1 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 | Vid. |
| Shell Sort | 157 | 151 | 173 | 175 | 147 | 168 | 147 | 158 | 158 | 159 | 159 |
| Heap Sort | 1992 | 2107 | 1946 | 2337 | 2511 | 1908 | 2109 | 1917 | 1953 | 1890 | 2067 |
| Aibė 2 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 194 | 201 | 212 | 234 | 211 | 184 | 153 | 145 | 152 | 183 | 187 |
| Heap Sort | 1812 | 1931 | 1934 | 1823 | 1810 | 1951 | 1946 | 1889 | 1995 | 1969 | 1906 |
| Aibė 3 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 162 | 157 | 157 | 154 | 153 | 163 | 151 | 157 | 159 | 181 | 159 |
| Heap Sort | 1945 | 1866 | 1853 | 2033 | 1959 | 1860 | 1946 | 2035 | 1762 | 1863 | 1912 |
| Aibė 4 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 149 | 162 | 170 | 166 | 153 | 147 | 149 | 145 | 144 | 149 | 153 |
| Heap Sort | 1807 | 1934 | 1970 | 1832 | 1811 | 2024 | 1902 | 1881 | 2025 | 1922 | 1911 |
| Aibė 5 | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |  |
| Shell Sort | 159 | 153 | 159 | 159 | 158 | 157 | 149 | 156 | 172 | 189 | 161 |
| Heap Sort | 2033 | 1838 | 1878 | 1954 | 1940 | 1852 | 2046 | 1924 | 1894 | 1888 | 1925 |

2. lentelė. Bendri rezultatai.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Laikai milisekundėmis, su kiekviena aibe atlikta 10 testų. | | | | |  |
| n=19000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 | Vidurkis |
| Shell Sort | 159 | 187 | 159 | 153 | 161 | 160,6 |
| Heap Sort | 2067 | 1906 | 1912 | 1911 | 1925 | 1944,2 |
| n=38000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 329 | 346 | 346 | 319 | 326 | 335,4 |
| Heap Sort | 3918 | 4120 | 3968 | 3927 | 3926 | 3979,6 |
| n=76000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 681 | 669 | 713 | 704 | 679 | 680 |
| Heap Sort | 8140 | 8241 | 8066 | 8048 | 8024 | 8043,8 |
| n=152000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 1410 | 1403 | 1424 | 1412 | 1625 | 1454,8 |
| Heap Sort | 16372 | 16532 | 16406 | 16454 | 16466 | 16458,4 |
| n=304000000 | Aibė 1 | Aibė 2 | Aibė 3 | Aibė 4 | Aibė 5 |  |
| Shell Sort | 3636 | 3511 | 3542 | 3458 | 3474 | 3524,2 |
| Heap Sort | 35253 | 35271 | 35009 | 35234 | 35293 | 35232 |

**Rezultatų pateikimas grafiku.**

**Rezultatų analizė.**

**1 teiginys – teoriniai įverčiai.**

Shell sort teoriniai įverčiai atitinka teorinius įverčius.

Heap sort teoriniai įverčiai atitinka teorinius įverčius.

Pagrindimas.

Shell sort ir heap sort teoriniai įverčiai sutampa su prieš tai pateiktais kiekias.

Tikriname:

Laikas visada kito ~2 kartus. Pateikta Shell sort algoritmo vidutinio laiko sudėtingumo formulė yra tiksli šiuo atvėju, kadangi šiuo metu, skaičiai surašyti tokia tvarka, kuria yra lengviausia suruošiuoti šio algoritmu. Tačiau, kadangi visi skaičiai yra panašus, atitinka teorinį įvertį

Laikas visada kito ~2 kartus. Pateikta Heap Sort algoritmo vidutinio laiko sudėtingumo formulė yra tiksli. Visi skaičiai yra panašus, atitinka teorinį įvertį

**2 teiginys – greičių palyginimas.**

Šiuo atvėju Shell sort rušiavimas yra daug greitesnis negu Heap sort. Shell sort yra daugiau nei 10 kartų efektyvesnis šiuo atvėju

Pagrindimas.

*Shell sort rušiavimo algoritmas yra greitesnis, kadangi Shell sort‘as, nenaudoja jam atitinkamos duomenų struktūros „kruvos“, o lygini iškart pagal atstuma. Taipogi, pagal Shell sort veikimo principą, kai duomenis bus lyginami, bus visada atliekamas keitimas, kadangi duodama, geriausio laiko sudėtingumo atvėjis.*

**3 teiginys – algoritmų greičių priklausomybė nuo aibės.**

Shell sort rūšiuoja greičiau mažėjančia tvarka sugernuotas aibės nei atsitiktine tvarka sugeneruotas.

Heap sort rūšiuoja greičiau mažėjančia tvarka sugernuotas aibės nei atsitiktine tvarka sugeneruotas.

Pagrindimas. *(0,3 balo)*

Krūvos (heap) rūšiavimas ir Shell sort rūšiavimas greičiau rūšiuoja atsitiktinių skaičių masyvą mažėjimo tvarka nei atsitiktinių skaičių masyvą didėjančia tvarka, nes atsitiktinių skaičių masyvas mažėjimo tvarka jau yra iš dalies surūšiuotas. Tai reiškia, kad reikia mažiau palyginimų ir apsikeitimų, kad elementai būtų išdėstyti teisinga tvarka, o tai gali lemti greitesnį rūšiavimo laiką. Priešingai, atsitiktinių skaičių masyvas didėjančia tvarka nėra iš dalies surūšiuotas, todėl reikia daugiau palyginimų ir apsikeitimų, kad elementai būtų išdėstyti teisinga tvarka.

**Bendros išvados.**

1. išvada.

Abu algoritmai rušiuoja greičiau mažėjančia tvarka sugeneruotas aibes negu atsitiktine tvarka.

1. išvada.

Didinant duomenų kiekį atsitiktinių skaičių aibėse, galime įžvelgti, jog, abu rušiavimo algoritmai veikia panašiu greičiu.

1. išvada.

Shell sort rušiavimo algoritmas atlieka savo funkcija su iš dalies surušiuota aibe greičiau, negu Heap sort

**PRIEDAI.**

Text

Description automatically generated with medium confidence

Text

Description automatically generated with medium confidence

A picture containing text

Description automatically generated

A picture containing calendar

Description automatically generated

A picture containing text, person, screenshot

Description automatically generated









