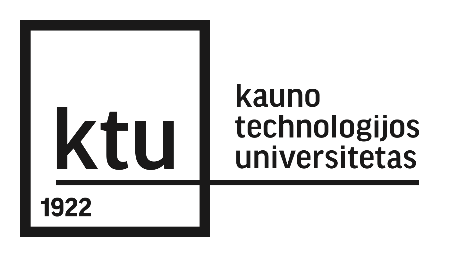
****

**Informatikos fakultetas**

**Duomenų struktūros**

Laboratorinis darbas Nr.2

Variantas Nr. 5

**Atliko:** IFF-1/1 gr. stud. Paulavičius Karolis

**Priėmė:** dėst. Budnikas Aurelijus

**KAUNAS, 2022**

**Testuojami metodai**

* Class BstSet: retainAll()
* Class AvlSet: retainAll()

**Asimptotinis sudėtingumas**

* BstSet<E>.retainAll() = O(n) \* O(log(n))
* AvlSet<E>.retainAll() = O(n) \* O(log(n)) amortizuotas

**Greitaveikos algoritmo aprašymas**

BstSet<E> ir AvlSet<E> ieškome po 5000, 10000, 20000, 40000, 80000 elementų, kuriuos reikia ištrinti naudojant sukurtą klasę „Car“. Tikriname kai ieškomas Set<E> turi 25%, 50%, 100% reikiamų elementų. Greitaveikos tikrinimo metu tikriname tik „retainAll“ algoritmo greitį. BstSet<E> ir AvlSet<E> naudoją paveldėtą BstSet<E> ir AvlSet<E> naudoją paveldėtą BstSet<E> „retainAll“ algoritmą.

Hipotezė: AvlSet<E>.retainAll() bus greitesnis dėl „Add“ algoritmo papildomo balansavimo algoritmo dalies, kuris pagreitiną asimptotinį greitį beieškant elemento iki O(Log(n)).

**Naudojamo kompiuterio duomenys**

Procesorius: Intel Core i7-6700K

Procesoriaus dažnis: 4.00GHz

Procesoriaus branduoliai: 4

Buferinė atmintis (Cache): 8 MB

Operatyvioji atmintis (RAM): Corsair Vengeance

Operatyviosios atminties tipas: DDR4

Atminties talpa: 16GB

Operatyviosios atminties taktinis dažnis: 2400MHz

Atminties ciklo uždelsimas: C14

Vaizdo plokštė: NVIDIA GeForce GTX 1060

Vaizdo plokštės atminties talpa: 6GB

Atminties taktinis dažnis: 8100MHz

**Testų tekstiniai duomenys naudojant JMH:**

|  |
| --- |
| Benchmark (elementCount) Mode Cnt Score Error Units  Benchmark.retainAllAvl100Percentage 5000 avgt 5 865.887 ± 29.543 us/op  Benchmark.retainAllAvl100Percentage 10000 avgt 5 1861.663 ± 160.992 us/op  Benchmark.retainAllAvl100Percentage 20000 avgt 5 3986.733 ± 837.227 us/op  Benchmark.retainAllAvl100Percentage 40000 avgt 5 8340.762 ± 862.164 us/op  Benchmark.retainAllAvl100Percentage 80000 avgt 5 18572.920 ± 4032.293 us/op  Benchmark.retainAllAvl25Percentage 5000 avgt 5 224.351 ± 5.245 us/op  Benchmark.retainAllAvl25Percentage 10000 avgt 5 486.127 ± 12.298 us/op  Benchmark.retainAllAvl25Percentage 20000 avgt 5 1047.777 ± 110.423 us/op  Benchmark.retainAllAvl25Percentage 40000 avgt 5 2465.694 ± 97.946 us/op  Benchmark.retainAllAvl25Percentage 80000 avgt 5 5576.461 ± 346.663 us/op  Benchmark.retainAllAvl50Percentage 5000 avgt 5 443.741 ± 14.791 us/op  Benchmark.retainAllAvl50Percentage 10000 avgt 5 950.469 ± 53.981 us/op  Benchmark.retainAllAvl50Percentage 20000 avgt 5 2128.161 ± 301.955 us/op  Benchmark.retainAllAvl50Percentage 40000 avgt 5 4758.582 ± 472.133 us/op  Benchmark.retainAllAvl50Percentage 80000 avgt 5 11126.626 ± 2094.617 us/op  Benchmark.retainAllBst100Percentage 5000 avgt 5 926.742 ± 39.579 us/op  Benchmark.retainAllBst100Percentage 10000 avgt 5 1990.037 ± 56.676 us/op  Benchmark.retainAllBst100Percentage 20000 avgt 5 4294.878 ± 459.831 us/op  Benchmark.retainAllBst100Percentage 40000 avgt 5 9319.155 ± 1578.002 us/op  Benchmark.retainAllBst100Percentage 80000 avgt 5 20481.420 ± 3846.844 us/op  Benchmark.retainAllBst25Percentage 5000 avgt 5 228.644 ± 8.228 us/op  Benchmark.retainAllBst25Percentage 10000 avgt 5 509.450 ± 42.475 us/op  Benchmark.retainAllBst25Percentage 20000 avgt 5 1140.500 ± 410.506 us/op  Benchmark.retainAllBst25Percentage 40000 avgt 5 2309.244 ± 339.279 us/op  Benchmark.retainAllBst25Percentage 80000 avgt 5 5130.663 ± 1171.048 us/op  Benchmark.retainAllBst50Percentage 5000 avgt 5 467.333 ± 15.221 us/op  Benchmark.retainAllBst50Percentage 10000 avgt 5 1009.487 ± 24.063 us/op  Benchmark.retainAllBst50Percentage 20000 avgt 5 2181.053 ± 45.951 us/op  Benchmark.retainAllBst50Percentage 40000 avgt 5 4711.900 ± 388.126 us/op  Benchmark.retainAllBst50Percentage 80000 avgt 5 10958.577 ± 1334.268 us/op |

**Testų laiko priklausomybė nuo elementų kiekio grafikas**

**Išvados**

Algoritmų asimptotinis sudėtingumas yra O(n) ieškant visų elementų yra n. AvlSet<E> yra greitesnis nes vieną elementą suranda log(n) greičiu palyginus su BstSet<E> log(n) amortizuotas atveju.

AvlSet<E> vykdymo laiko požiūriu yra geresnis.