Počítač: Procesor AMD Ryzen 7 5800HS @ 3.20GHz, RAM 16GB.
Operační systém: Windows 10, Version 10.0.19044 Build 19044
Verze pythonu: 3.8.10
Seed: 94

1 Abstrakt

Představení a porovnání dvou implementací Splay stromů. Měří se počet rotací pro obě implementace a to v testech s náhodnými, sekvenčními a často opakujícími se daty. Výsledkem jsou grafy tyto hodnoty znázorňující a porovnávající.

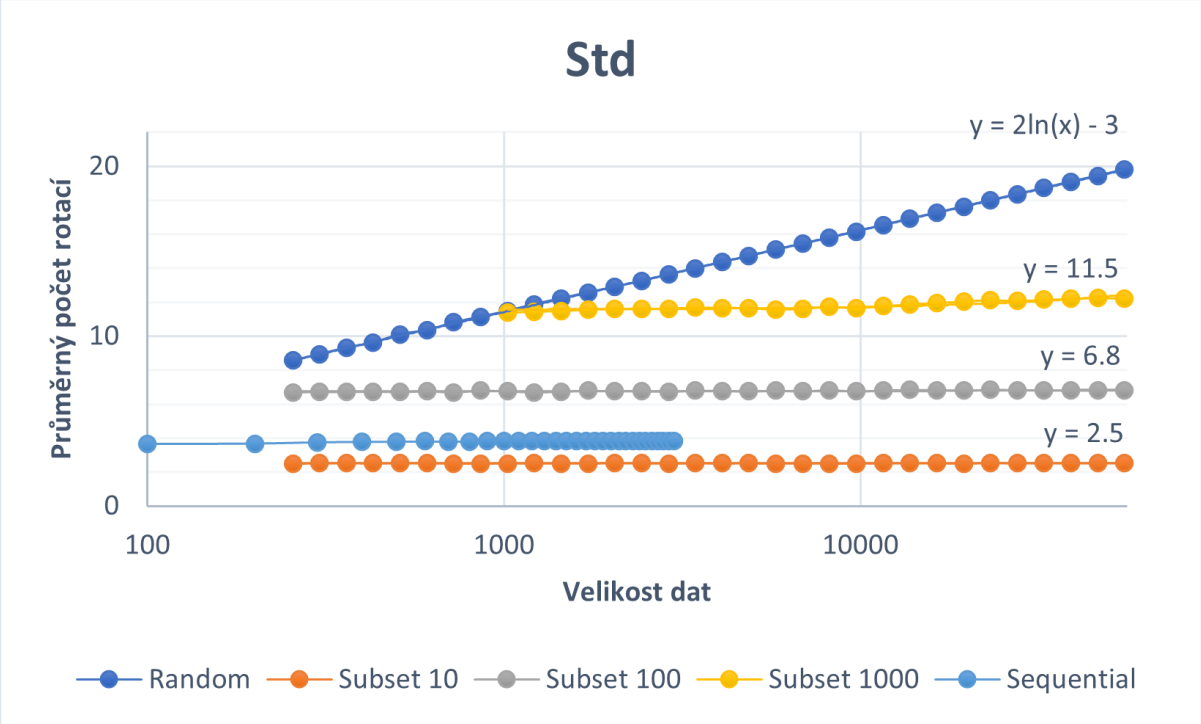
2 Testované implementace

2.1 Standardní implementace

První implementace je standardní a používá **dvojité rotace**.

Na grafu (Obrázek 1) můžeme pozorovat konstantní složitost pro data *sekvenční a často se opakující (Subset)* a logaritmickou složitost pro data *náhodná*.

Zatímco u sekvenčního a subset zápisu přistupujeme pouze k určitým konstantním datům, tak je i průměrná hloubka a počet rotací konstantní. V případě náhodného přístupu do dat musíme hledat data v celém rozsahu a od toho je odvozena i průměrná hloubka, neboli i počet rotací hledaných prvků, růst je logaritmický.



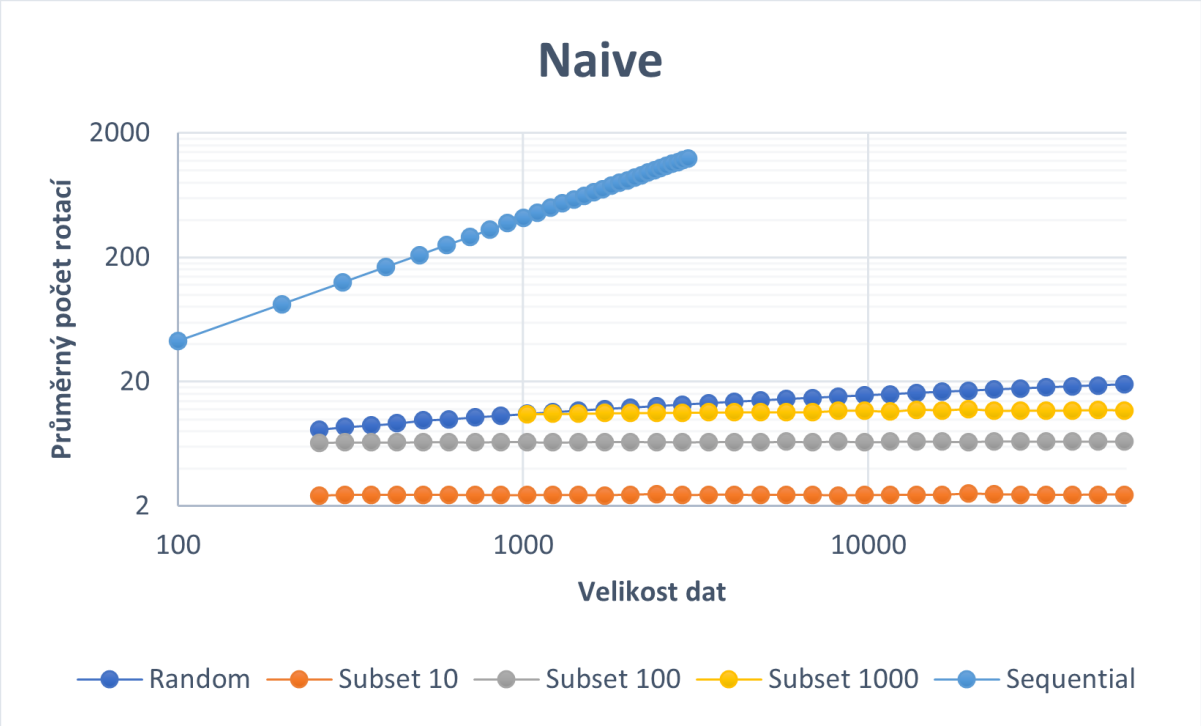
Obrázek 1: Správná implementace. Horizontální osa je logaritmická.

2.2 Naivní implementace

Druhá implementace je naivní a používá pouze **jednu rotaci**.

Na grafu (Obrázek 2) můžeme pozorovat kontstantní složitost pro *často se opakující (Subset)* data. Pro *náhodná* data je složitost logaritmická a pro *sekvenční* je složitost lineární.

Důvodem bude hledání prvku, jenž se jednoznačně nachází v hloubce rovné jeho hodnotě (za předpokladu, že data jsou všechna celá čísla ¡ N), tudíž amortizovaně máme cca 0.4 rotace za každý prvek v prohledávaných datech.



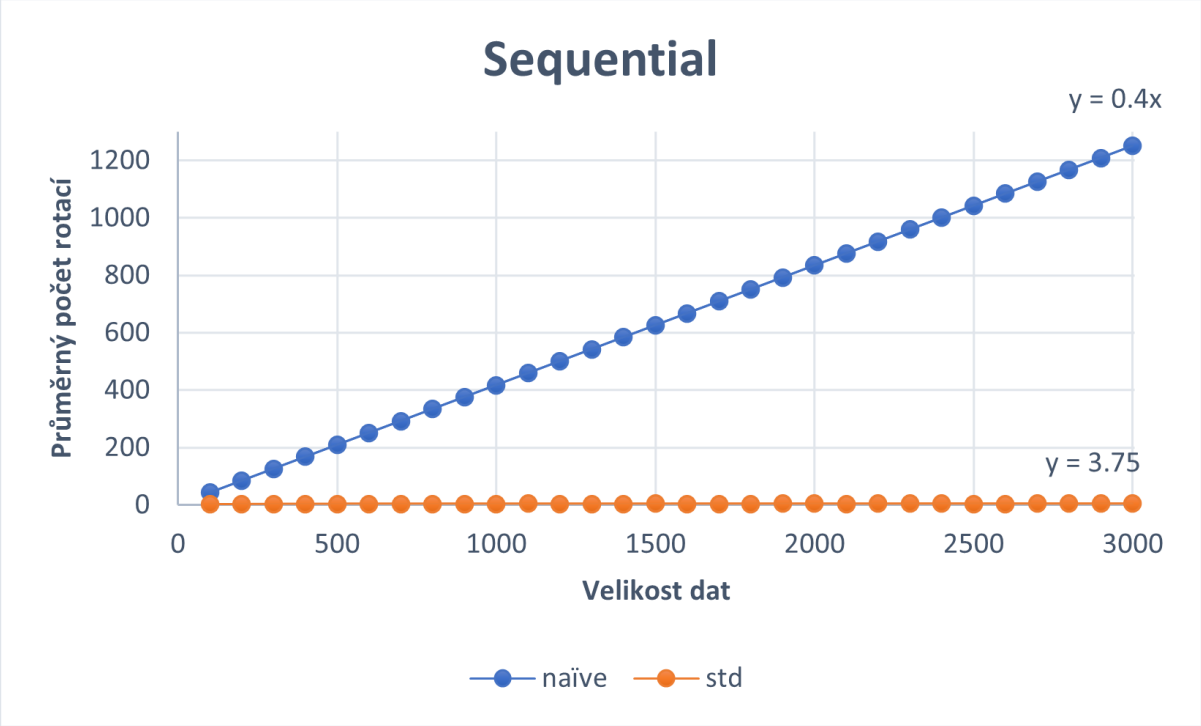
Obrázek 2: Naivní implementace spolu s křivkami odhadů v logaritmické škále. Obě osy jsou logaritmické.

3 Testy různých přístupů k datům

3.1 Sekvenční přístup

Na grafu (Obrázek 3) pozorujeme u standardního algoritmu konstatní počet rotací pro libovolně velká data.

Pro naivní implementaci však je počet rotací lineární k velikosti dat.

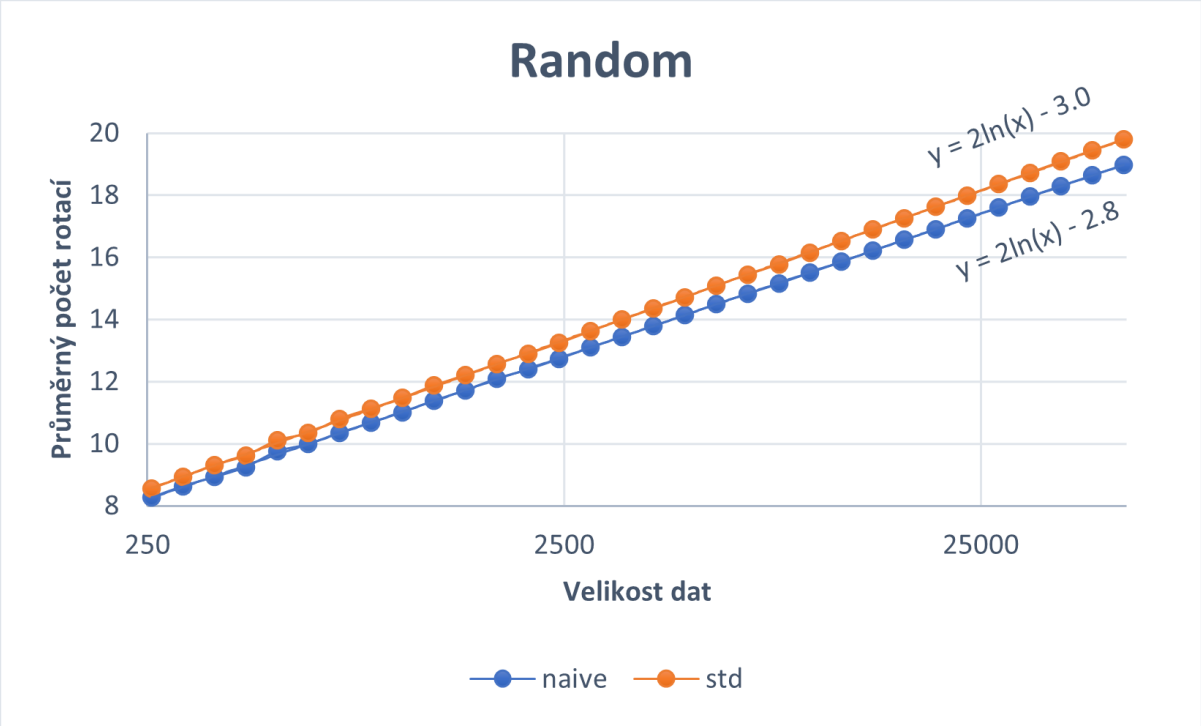


Obrázek 3: Naivní vs správná implementace. Vertikální osa je logaritmická.

3.2 Náhodný přístup

Na grafu (Obrázek 4) pozorujeme u standardního algoritmu logaritmický počet rotací vzhledem k velikosti dat.

Naivní implementace má řádově stejný počet rotací jako standardní implementace, jen o v průměru cca 10% lepší, což nepovažujeme za výrazný rozdíl.

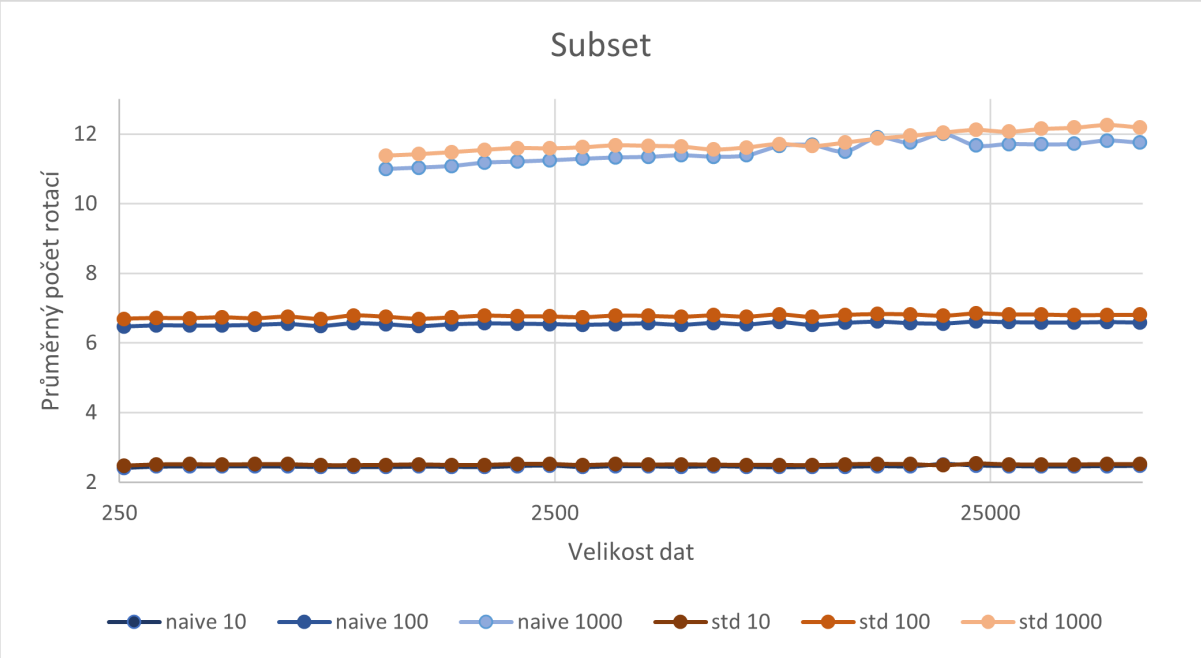


Obrázek 4: Naivní vs správná implementace. Horizontální osa je logaritmická.

3.3 Přístup k omezeně velkému výběru prvků

Na grafu (Obrázek 5) pozorujeme u obou implementací konstantní počet rotací vzhledem k velikosti často přistupavaných dat.

Takovýto přístup k datům lze srovnávat s nahodným přístupem do dat s konstantní velikostí.



Obrázek 5: Naivní vs správná implementace. Horizontální osa je logaritmická.

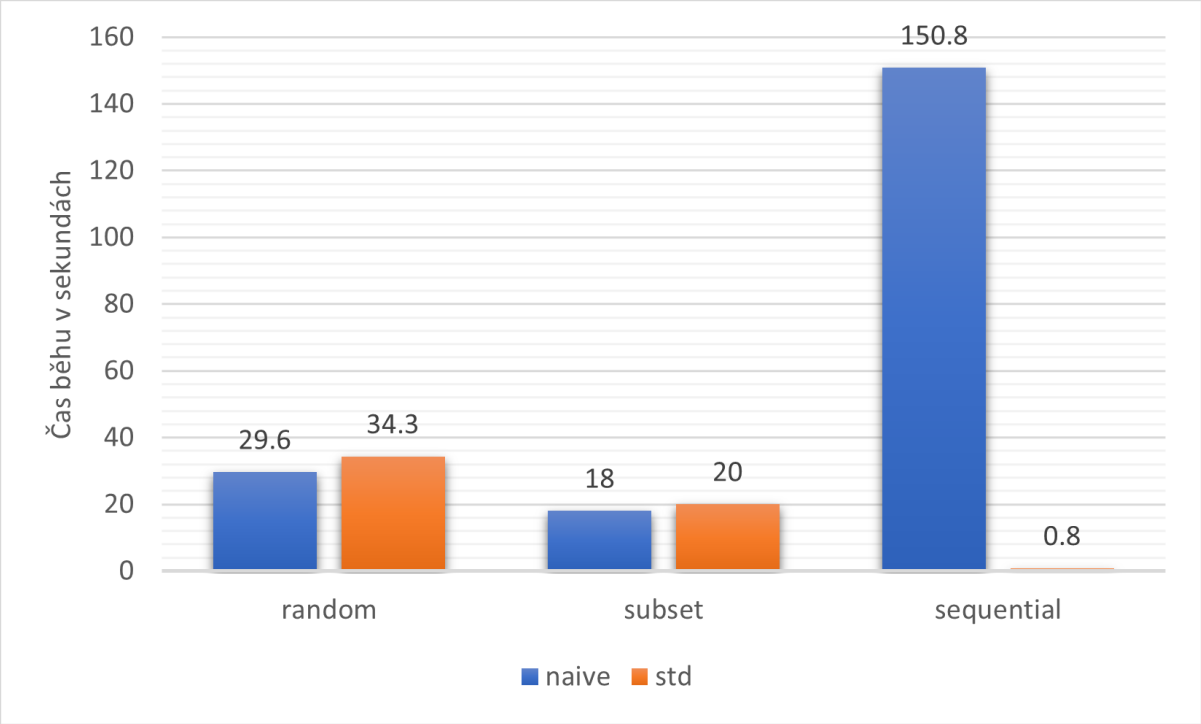
4 Měření času

Všechna měření probíhala na stejném zařízení a počet opakování každého testu byl 5.

Je důležité uvědomit si, že dataset sekvenčního čtení nebyl stejně velký jako ostatní dva datasety, proto jej nemůžme porovnat.

Náhodné čtení běželo cca dvakrát déle, než přístup do častých dat, což si lze vysvětlit velikostí datasetu, jenž k tomu není reprezentativní a pro větší datasety by rozdíl byl větší.

Sekvenční přístup k datům má očividný rozdíl způsobený porovnáním konstantního přístupu a přísstupu lineárního.



Obrázek 6: Čas běhu programu.

4.1 Shrnutí

V testech s náhodným přístupem a přístupem do opakujících se dat měly obě implementace řádově stejný počet rotací.

Jedinou výjimkou byl test sekvenčního hledání, kde standardní implementace měla konstatní počet rotací, avšak naivní lineární.