VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Dokumentácia projektu
Predmet: **PRL** 2018/19
Projekt: **Bucket sort**
Autor: **Michael Halinár** xhalin01

1 Rozbor

Tento radiaci algoritmus pracuje pomocou binárneho stromu, potrebuje $(2.log(n)) - 1$ procesorov, kde n je počet radených čísel, m je počet listových uzlov stromu pre ktorý platí vzťah $n = 2^m$. Každý listový uzol sa pri tom stará o n/m čísel. Vygenerovaný strom musí byť úplný, preto sa m zaokrúhli tak aby $log_2 m$ bolo prirodzené číslo. Listovým uzlom sa potom rovnomerne rozložia radené čísla, ak by ich bolo menej ako kapacita všetkých listových uzlov tak sa do nich vložia čísla -1 a vyplnia sa tak prázdne miesta v listoch. Listy následne zoradia svoje čísla a pošlú ich rodičovskému uzlu, ktorý spojí a zoradí dve postupnosti čísel. Tento postup sa opakuje až kým koreň stromu dostane všetky čísla.

2 Analýza

Každý listový procesor prečíta $n/(log(n))$ prvkov. Pomocou `std::sort` so zložitosťou $O(n * (log(n)))$ zoradí každý list svoje prvky , tj. $(n/log(n) * log(n/log(n)) = O(n)$. Spájanie zoradených postupností rodičovských uzloch pomocou `std::merge` má zložitosť $O(n)$.

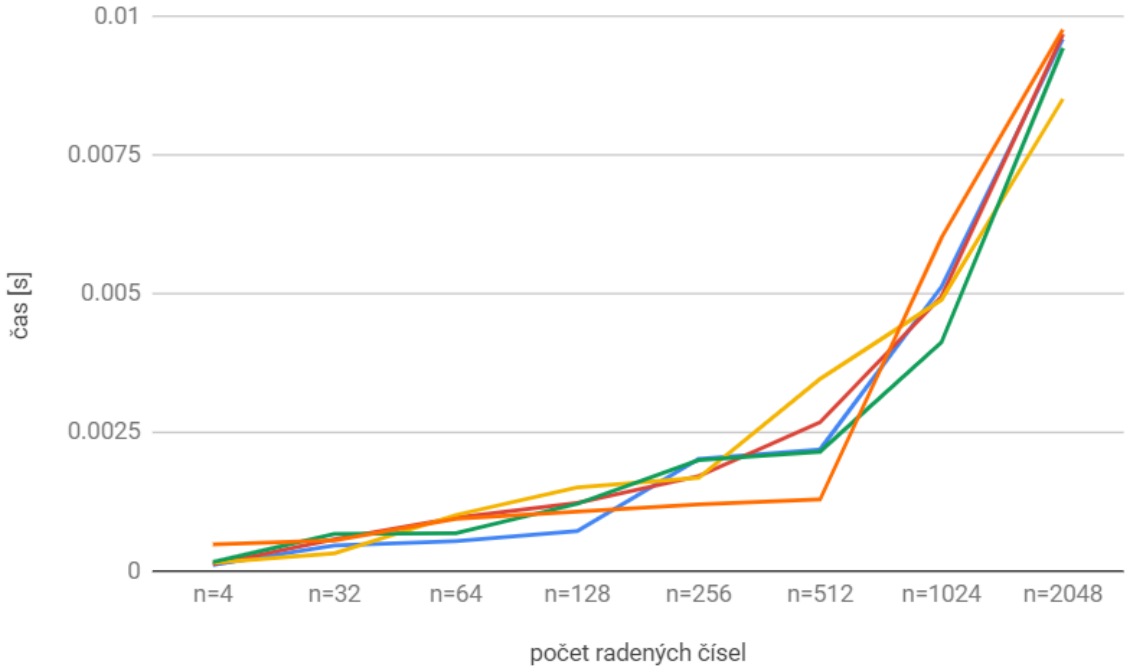
Celková cena: $c(n) = O(n * log(n))$, kde $t(n) = O(n)$,$p(n) = O(log(n))$,

3 Implementácia

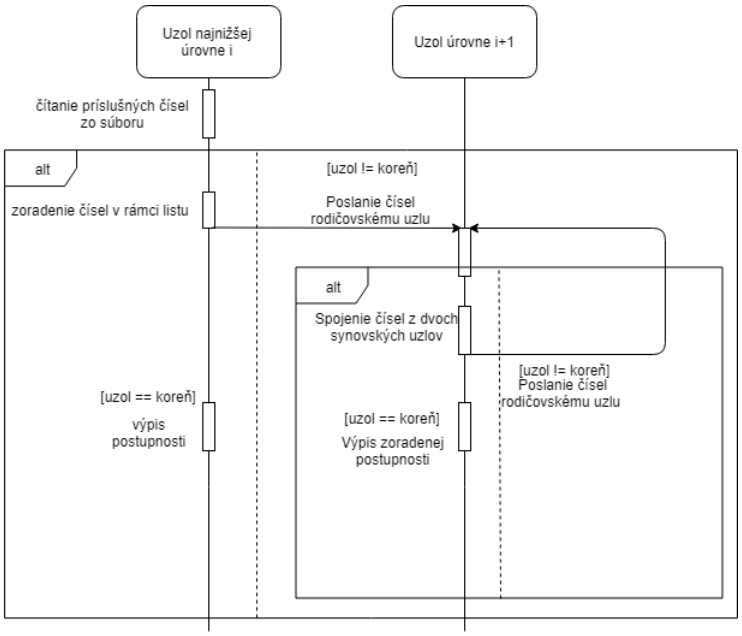
Bash skript vypočíta potrebný počet procesorov pre úplný strom bucket sortu. Následne každý procesor prečíta svoje čísla zo súboru. Keďže strom bude vždy úplný a vyvážený, nepotrebuje žiadny procesor explicitne dostať informáciu o tom ktorý uzol je jeho rodič a ktoré sú jeho synovské uzly, ale vie si túto informáciu dopočítať zo svojho identifikačného čísla. Aby každý listový procesor vedel koľko čísel bude čítať, urobí si distribučnú tabuľku všetkych listov. Distribučná tabuľka rovnomerne rozloží čísla pre listové procesory, aj v prípade že počet radených čísel nie je logaritmovatelný. V takejto situácii sa ostatné miesta v listoch vyplnia číslom -1 a pri výpise nebudú použité. Nasleduje radenie v listoch pomocou `std::sort`. Po zoradení začína séria cyklov kde synovské uzly pošlú svoje postupnosti do rodičovských uzlov a tie ich následne zoradia až kým koreň nemá komu poslať svoju postupnosť. Koreňový uzol potom už iba vypíše obsah súboru a aj zoradenú postupnosť čísel.

4 Experimenty

Algoritmus radenia bol použitý na rôzne počty čísel po 5 meraní. Na grafe vidíme približne exponenciálnu krivku. Vzhľadom na to že školský server Merlin nedovolí spustenie algoritmu pre väčší počet čísel, nedozvieme sa ako by krivka vyzerala pri väčších počtoch radených čísel.



5 Komunikačný protokol



6 Záver

Predpokladaná časová zložitosť O(n) týmito experimentami nebola dosiahnutá. Na grafe je vidieť exponenciálna krivka, pri väčších počtoch procesorov by sa krivka mohla vyvíjať inak, ďalšie problémy v rýchlosti mohli byť zapríčinené použitou kniznicou MPI, ktorá nie je stavaná pre paralelné výpočty.