1 Úvod

Cílem projektu bylo za použití knihovny Open MPI a jazyka C/C++ implementovat paralelní řadící algoritmus Merge-splitting sort.

2 Rozbor a analýza algoritmu

Merge-splitting sort se řadí mezi paralelní řadící algoritmy. Algoritmus využívá pro řazení n čísel p procesorů, přičemž procesorů je méně než řazených čísel. Nejdříve procesor 0 načte posloupnost čísel a přibližně stejně velké části posloupnosti pošle ostatním procesorům. Všechny procesory danou část posloupnosti seřadí paralelně pomocí optimálního sekvenčního algoritmu. Všechny procesory poté paralelně dělají následující činnost, a to po $\lceil \frac{p}{2} \rceil + 1$ kroků. Každý sudý procesor načte posloupnost přiřazenou následníkovi (pokud existuje), setřídí svoji a následníkovu posloupnost, rozdělí ji na dvě části, spodní část si procesor ponechá a horní část odešle svému následníkovi. Liché procesory pracují stejně jako sudé (pokud mají následníka). Nakonec každý procesor odešle posloupnost procesoru 0, který jednotlivé posloupnosti seřadí a vypíše výslednou posloupnost čísel.

2.1 Odvození složitosti

Počet procesorů lze vyjádřit pomocí vztahu: $p(n) = p$, kde $p < n$.

- Načítání vstupní posloupnosti n čísel, její rozdělení a zaslání procesorům má časovou složitost $\mathcal{O}(p \cdot \frac{n}{p}) = \mathcal{O}(n)$.
- Řazení $\frac{n}{p}$ prvků pomocí optimálního sekvenčního algoritmu má časovou složitost: $\mathcal{O}((\frac{n}{p}) \log(\frac{n}{p}))$.
- Provedení následujících kroků $\lceil \frac{p}{2} \rceil + 1$ krát:
- Odeslání, respektive příjem $\frac{n}{p}$ prvků má časovou složitost $\mathcal{O}(\frac{n}{p})$.
- Setřídění dvou seřazených posloupností do jedné seřazené posloupnosti má časovou složitost $\frac{n}{p}$. Celková časová složitost je $(\lceil \frac{p}{2} \rceil + 1) \cdot \mathcal{O}(\frac{n}{p}) \cdot 3 = \mathcal{O}(n)$.
- Odeslání seřazené posloupnosti jednotlivými procesory procesoru 0 má časovou složitost: $\mathcal{O}(p \cdot \frac{n}{p}) = \mathcal{O}(n)$.

Časová složitost celého algoritmu je

$$t(n) = \mathcal{O}((\tfrac{n}{p}) \log(\tfrac{n}{p})) + \mathcal{O}(n) = \mathcal{O}(\tfrac{n \cdot \log n}{p}) + \mathcal{O}(n).$$

Cena algoritmu je:

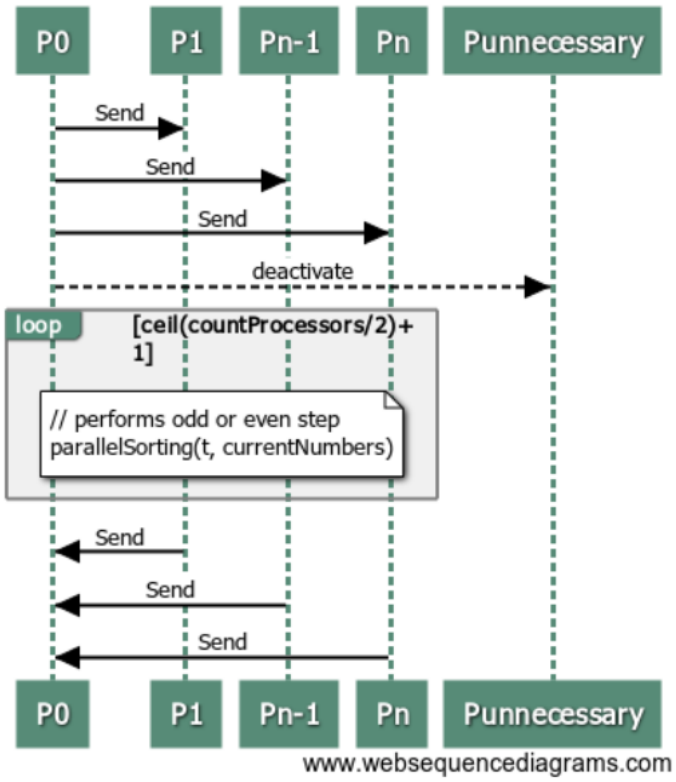
$$c(n) = t(n) \cdot p(n) = (\mathcal{O}(\tfrac{n \cdot \log n}{p}) + \mathcal{O}(n)) \cdot p(n) = \mathcal{O}(n \cdot \log n) + \mathcal{O}(n \cdot p).$$

Algoritmus je optimální, pokud platí $p \leq \log n$.

3 Implementace

Algoritmus byl implementován v jazyce C/C++ a byla použita knihovna Open MPI. Implementace algoritmu je téměř shodná s popisem, který byl uveden v rámci přednášek předmětu PRL. Rozdíl je však v číslování procesorů, kdy jsou procesory číslovány od 0 namísto od 1 (první procesor má přiřazen index 0). Úkolem procesoru 0 je načtení vstupní posloupnosti čísel ze souboru, její rozeslání ostatním procesorům a příjem seřazených posloupností zaslaných ostatními procesory. Dalším rozdílem proti původnímu algoritmu je počet iterací, kdy je posloupnost čísel seřazena po $(\lceil \frac{p}{2} \rceil + 1)$ iteracích. Důvodem jsou situace, kdy některé procesory mají přiřazen o jeden prvek méně než ostatní procesory. V případě, že je délka vstupní neseřazené posloupnosti n dělitelná počtem procesorů p, je každému procesoru přiděleno $\frac{n}{p}$ čísel. V opačném případě získá prvních $(n \mod p)$ procesorů $\frac{n}{p}$ hodnot a ostatních $(p - (n \mod p))$ procesorů $\frac{n}{p} - 1$ hodnot.

Komunikační protokol, který popisuje, jak jsou zprávy posílány mezi jednotlivými procesory, je znázorněn sekvenčním diagramem 1. Nejdříve procesor $P0$ pošle dané části posloupnosti čísel ostatním procesorům. Pokud jsou některé procesory nadbytečné, tak se deaktivují. Poté se provede hlavní část, která je tvořena dvěma kroky. V rámci prvního kroku sudé procesory setřídí svoji a následníkovu posloupnost. V druhém kroku provedou stejnou činnost liché procesory. Nakonec každý procesor pošle svou seřazenou část posloupnosti procesoru $P0$.

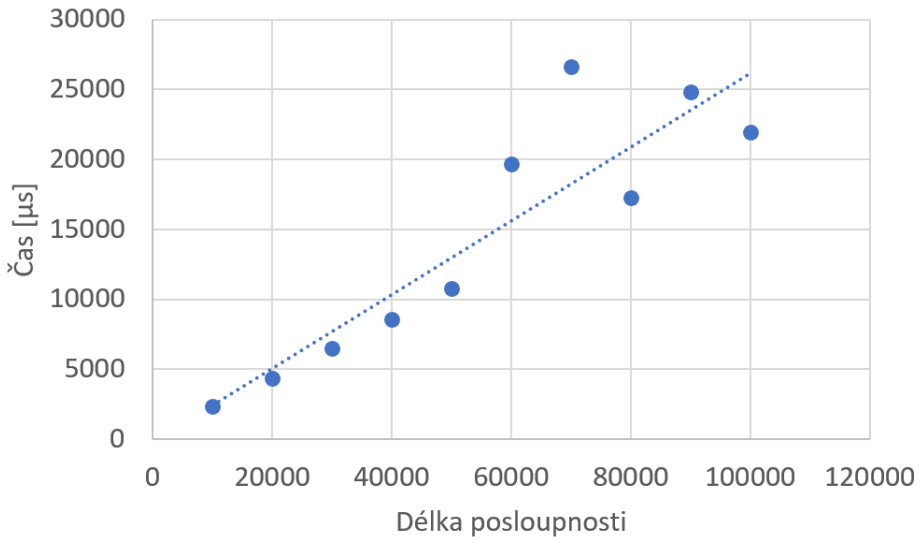


Obrázek 1: Komunikační protokol

4 Experimenty ověření implementace

Pro ověření časové složitosti byly provedeny experimenty. Sada s deseti testovacími vstupy byla tvořena neseřazenými posloupnostmi čísel v rozsahu od 10 000 do 100 000. Pro každou neseřazenou posloupnost daného počtu čísel byl program spuštěn desetkrát a následně byl vypočítán průměrný čas. Pro měření časové náročnosti výpočtu byla využita knihovna `chrono`. Zdrojový kód byl modifikován dvěma časovými razítky, kdy se jedno časové razítko vložilo za výpis neseřazené vstupní posloupnosti a druhé před výpis výsledku. Dobu výpočtu bylo poté možné získat pomocí rozdílu těchto dvou hodnot.

Závislost doby běhu algoritmu na délce vstupní posloupnosti je znázorněna na obrázku 2. Křivka má skoro lineární průběh, odpovídá teoretické časové složitosti algoritmu, která byla odvozena v kapitole Odvození složitosti.



Obrázek 2: Závislost doby běhu algoritmu na délce vstupní posloupnosti

5 Závěr

V rámci projektu byl implementován paralelní řadící algoritmus Merge-splitting sort, jehož časová složitost byla ověřena pomocí experimentů. V dokumentu byl algoritmus nejdříve analyzován, a to včetně své časové složitosti. Poté byla popsána implementace a komunikační protokol, pro jehož znázornění byl využit sekvenční diagram. Nakonec byly uvedeny výsledky experimentů, které ověřily časovou složitost implementovaného algoritmu.