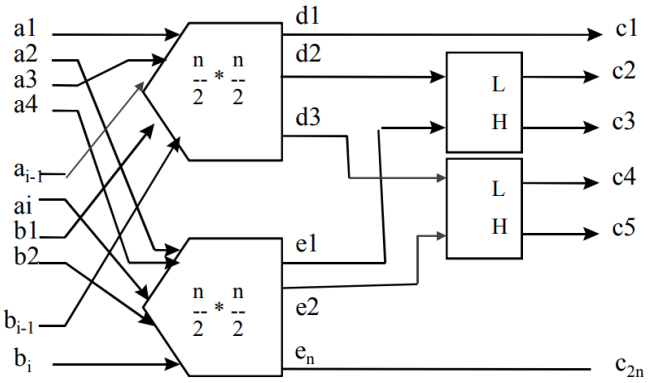
Implementace algoritmu Odd-Even Merge Sort

Jan Krejčí (xkrejc70)

1 Analýza algoritmu

Paralelní řadící algoritmus Odd-Even Merge Sort obecně pracuje na principu spojování dvou seřazených posloupností o délce *n* do jedné seřazené posloupnosti o délce *2n*. Pokud jsou na vstupu neseřazené hodnoty, začne algoritmus nejdříve vytvářet seřazené posloupnosti o délce 2, tedy začně spojovat dvě seřazené posloupnosti o délce jedna. Název *Odd-even* značí, že na vstup do každé další úrovně sítě se ze seřazené posloupnosti vyberou pouze lichá či sudá čísla, jak je uvedeno na obrázku 1. Takto algoritmus pokračuje dokud nejsou spojeny dvě seřazené posloupnosti o délce n/2 do finální seřazené posloupnosti. Podrobný rozbor algoritmu určeného k řazení 8 hodnot je popsán v implementační části.



Obrázek 1: Síť NxN. Převzato ze souhrnných materiálů PRL03-MNG ©Petr Hanáček, Model 2019.

1.1 Teoretická složitost

Jelikož se velikost nově seřazené posloupnosti zdvojnásobuje každým krokem v síti, celkový počet těchto kroků je $log\ n$. Čas potřebný na seřazení dvou sekvencí o délce 2^{i-1} v i-tém kroku sítě je roven $s(2^i)$, kde

$$s(2^i) = 1 \qquad pro\ i = 1,$$
$$s(2^i) = s(2^{i-1}) + 1 \qquad pro\ i > 1.$$

Na základě dvou předchozích vět je *časová složitost* tohoto algoritmu

$$t(n) = \sum_{i=1}^{logn} s(2^i) = O(log^2 n).$$

Počet procesorů potřebných na seřazení dvou sekvencí o délce 2^{i-1} v i-tém kroku sítě je roven $q(2^i)$, kde

$$q(2^i) = 1 \qquad pro\ i = 1,$$
$$q(2^i) = 2q(2^{i-1}) + 2^{i-1} - 1 \qquad pro\ i > 1.$$

Počet procesorů použitých v celé síti a tedy i *prostorová složitost* algoritmu je

$$p(n) = \sum_{i=1}^{logn} 2^{(logn)-i} q(2^i) = O(n\,log^2 n).$$

Celková *cena* řadícího algoritmu Odd-Even Merge Sort pro n vstupních prvků je vypočtena dle vzorce

$$c(n) = t(n) * p(n) = O(n\,log^4 n),$$

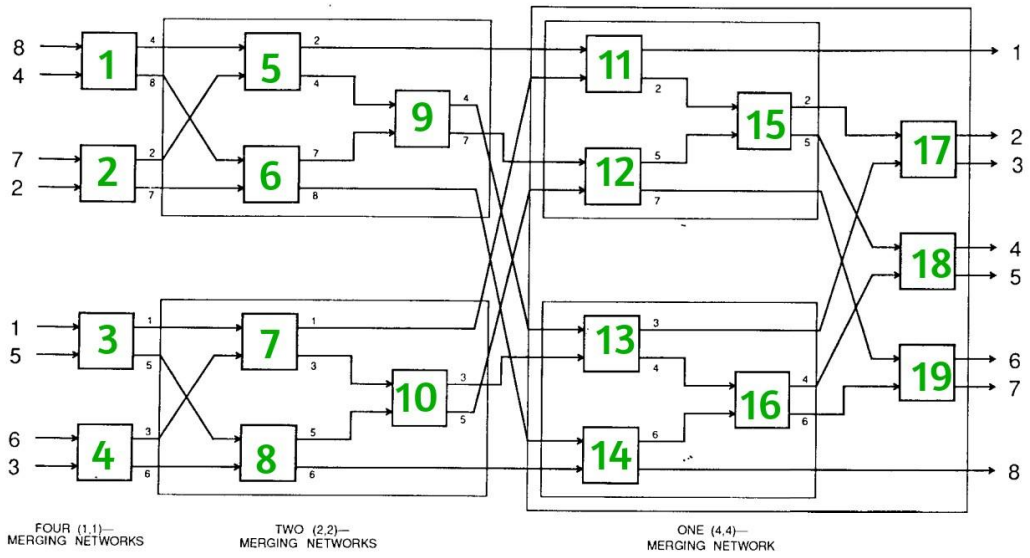
což není optimální.

2 Implementace

Algoritmus je implementován v jazyce C++ dle schématu řadící sítě na obrázku 2, který je složen z následujících řadících bloků:

- 4x 1x1 (procesy 1-4)
- 2x 2x2 (procesy 5-10)
- 1x 4x4 (procesy 11-19)

Jak je patrné z obrázku 2, síť řadící dvě posloupnosti o délce dva (2x2) je složena z tří CE 1x1. Obdobně je tomu u sítě 4x4. Každý blok (síť 1x1) obstarává jeden proces s konkrétním pořadovým číslem.



Obrázek 2: Odd-Even Merge Sort řadící síť pro osm vstupních hodnot. Převzato ze souhrnných materiálů PRL03-MNG ©Petr Hanáček, Model 2019.

Použitý způsob komunikace mezi procesy je obstarán knihovnou *Open MPI*. Je využita pouze asynchronní blokující komunikace. Jednotlivé procesy tedy před samotným provedením seřazení vyčkají na všechny posloupnosti, které potřebují na vstupu.

Vstupem je binární soubor obsahují neseřazenou posloupnost bajtů, která reprezentuje 8 celých čísel v rozsahu 0-255. Hodnoty jsou 1. procesem načteny ze souboru a vypsány v požadovaném formátu na standardní výstup. To obstarává funkce `loadAndPrintNumbers()`. Následně jsou hodnoty rozděleny na jednoprvková pole, která opět první proces rozešle dalším třem (s pořadovým číslem 2-4) pomocí `MPI_Send()`. První 4 procesy pak paralelně seřadí posloupnosti o délce 2 za použití sítě 1x1

implementované ve funkci `sort1x1()`. Tyto výsledky předají dalším procesům (5-10), které je přijmou pomocí funkce `MPI_recv()`. Dále je program implementován obdobně, dokud se nedojde na konec sítě, kde jsou výsledné hodnoty opět poslány 1. procesu, který je ve správném pořadí, tedy vzestupně seřazené, vypíše na standardní výstup.

3 Závěr

Výsledkem je funkční program odpovídající algoritmu *Odd-Even Merge Sort* pro 8 vstupních hodnot, který byl řádně otestován a nejsou známy žádné implementační nedostatky.

Možné rozšíření by zahrnovalo snížení celkového počtu procesů na 4, což je nejmenší možný počet pro zachování paralelního řazení.