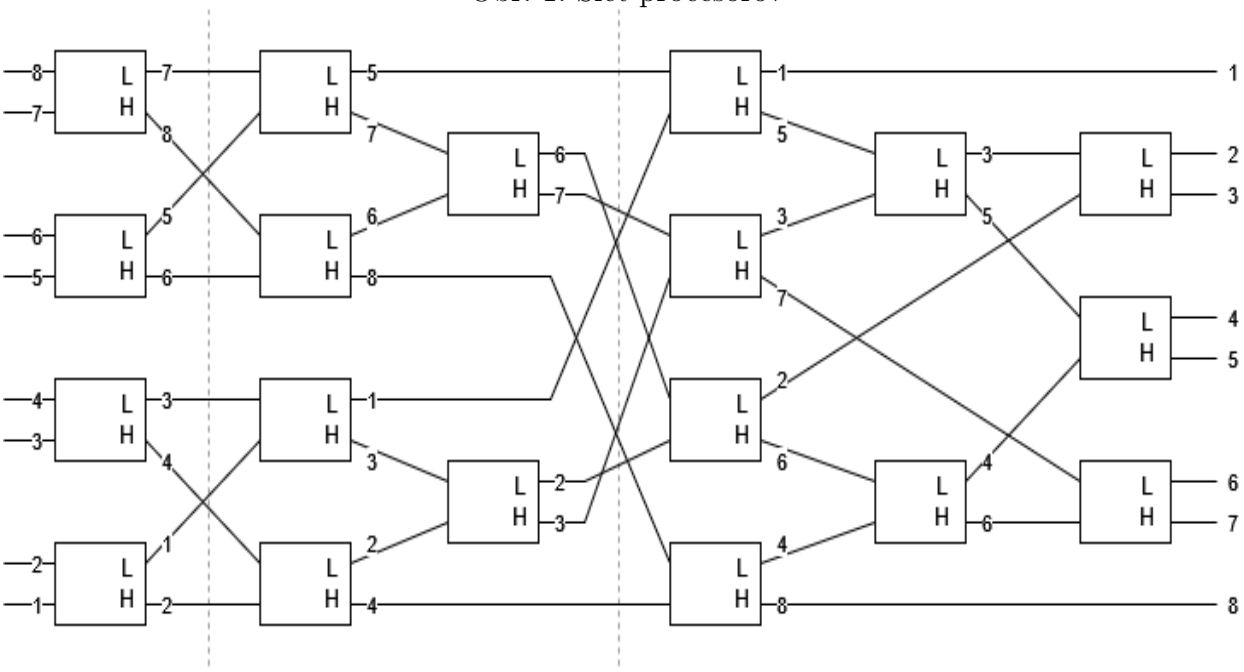
Paralelní a distribuované algoritmy - 2021/22
Projekt 1 - Odd-even merge sort
Sebastián Krajňák - xkrajn05

1 Implementácia

Algoritmus je implementovaný pre konštantný počet vstupných čísel, 8. Vstupom algoritmu je súbor numbers obsahujúci 8 náhodne generovaných čísel, veľkosti 1B v rozsahu 0 – 255. Generovanie a následné zmazanie súboru po skončení behu algoritmu, ako aj samotné spustenie celej implementácie zabezpečuje shell skript `test.sh`

Algoritmus je riadený špeciálnou sieťou, pozostávajúcou z 19 procesorov, kde každý procesor má dva vstupné a dva výstupné kanály. Každý procesor vykonáva porovnávaciu operáciu oboch vstupov, nájde minimum a maximum týchto vstupov a pošle nasledujúcim procesorom, podľa schémy siete viď Obr. 1.

Obr. 1: Sieť procesorov



Master procesor na začiatku implementácie načíta čísla zo súboru numbers a uloží ich do poľa `numbers` ako hodnoty typu `Int`. Kontrola správneho načítania sa vykonáva vo funkcii `checkNumbers`, ak sú všetky čísla načítané správne, vypíšu sa na `stdout` a master procesor ich rozpošle prvej vrstve procesorov. Jednotlivé prepojenia medzi procesormi sú uložené v dvojrozmernom poli `processors`, kde prvý index (0-18) predstavuje rank procesora a druhý index predstavuje dva nasledujúce procesory spojené výstupnými komunikačnými kanálmi.

Každý ďalší procesor pomocou funkcie `oddEvenMergeSort`, príjme čísla od predošlého procesora, následne si ich uloží do interého poľa a nájde maximum a minimum z týchto čísel, v závere funkcie ich pošle ďalším procesorom v sieti, podľa prepojení uložených v poli `processors`, ktoré dostane funkcia ako vstupný parameter spolu s rankom aktuálneho procesora.

V poslednej časti implementácie príjme master procesor zoradenú postupnosť čísel od jednotlivých procesorov a vypíše ju na `stdout`. Keďže je poradie posielania správ medzi procesormi asynchrónne, správna postupnosť prijímania správ od koncových procesorov je zabezpečená poľom `outputProcs`, ktorého hodnoty sa použijú ako `src` parameter funkcie `MPI_Recv`.

2 Analýza

Sieť procesorov je v skutočnosti tvorená kaskádou sietí typu 1x1 po 1 procesore, 2x2 po 3 procesoroch a 4x4 po 9 procesoroch. Označme dĺžku vstupnej postupnosti ako $n = 2^m$, teda dĺžka postupnosti bude nejaká mocnina 2, v našej implementácii je $m = 3$. Hodnota m zároveň predstavuje počet fáz[1] potrebných pre zotriedenie postupnosti. Analyzujeme časovú zložitosť implementácie, počet použitých procesorov a celkovú cenu algoritmu, v porovnaní s optimálnym sekvenčným algoritmom, ktorého $t(n) = c(n) = O(nlog(n)), p(n) = 1$. Keďže sa veľkosť zlúčených postupností zdvojnásobuje s každou fázou, celkovo bude $log(n)$ fáz.
V i-tej fáze radíme postupnosti dĺžky 2^{m-i} teda v 1. fáze potrebujeme na zotriedenie postupnosti 2^{m-1} CE[2], v 2. fáze potrebujeme 2^{m-2} sietí 2x2 s 3 CE a pod.

Časová zložitosť t(n)

Vďaka rekurzivite sietí je časová zložitosť $t(n) = O(m^2) = O(log^2(n))$.

Počet procesorov p(n)

Počet procesorov potrebných k zotriedení vstupnej postupnosti n je rovný $p(n) = O(nlog^2(n)$.

Celková cena c(n)

Predstavuje celkový počet porovnaní, ktoré sieť vykoná pre zotriedenie vstupnej postupnosti, ktorá je daná vzťahom $c(n) = t(n)p(n)$ teda výsledná cena implementácie je $c(n) = O(nlog^4(n))$.

3 Záver

Ako je z analýzy viditeľné algoritmus nie je v porovnaní s optimálnym sekvenčným algoritmom optimálny. Tento fakt je viditeľný hlavne z výpočtu celkovej ceny, z ktorej je zrejmé, že paralelný odd-even merge sort algoritmus vykonáva viac operácií ako $O(nlog(n))$ potrebných pre zotriedenie sekvenčným algoritmom. Týmto dochádzame k záveru, že odd-even merge sort nie je optimálny pri veľkých vstupoch, avšak pri menších vstupoch je jeho časová zložitosť lepšia v porovnaní s optimálnym sekvenčným algoritmom.

[1] počet kaskádových podsietí typu 1x1, 2x2, 4x4...

[2] procesorový element, sieť typu 1x1