Vysoké učení technické v Brně
Fakulta informačních technologií

# Paralelní a distribuované algoritmy
# 1.Projekt – Odd-Even Merge Sort

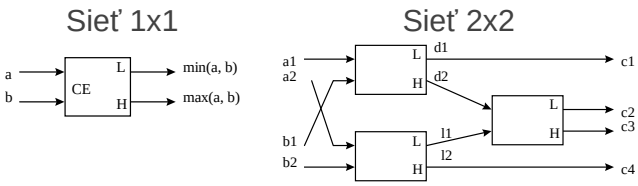
Dominik Boboš (xbobos00)
xbobos00@stud.fit.vutbr.cz
10. apríla 2022

## 1 Rozbor algoritmu

Zadaním projektu bolo implementovať algoritmus *Odd-Even Merge Sort* (OEMS) pre zoradenie ôsmich čísel. OEMS je paralelný triediaci algoritmus, ktorý predpokladá špeciálnu sieť procesorov.

Špeciálna sieť spočíva v tom, že každý procesor má dva vstupné a dva výstupné kanály. Každý procesor porovnáva hodnoty na svojich vstupoch, pričom menší vloží na výstup *L – low* a väčší na výstup *H – high*.

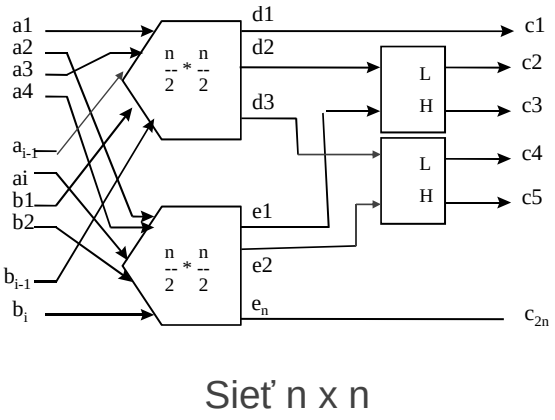
Takto nám vznikajú rôzne veľké siete v závislosti od počtu čísel k zoradeniu na vstupe (obr. 1)




Obr. 1: Architektúra siete pre OEMS pre zoradenie dvoch čísel (sieť 1x1) a štyroch čísel (sieť 2x2) (*Prevzaté z prednášok PRL doc. Hanáčka.*)

Napríklad zo vstupu čísel $a_1$, $a_2$, $b_1$, $b_2$ dostávame výslednú zoradenú postupnosť $c_1, c_2, c_3, c_4$. Všeobecná architektúra siete $n \times n$ je vyobrazená na obrázku 2

Počet procesov teda závisí na veľkosti vstupnej sekvencie. Pre sekvenciu $S = s_1, s_2, \ldots, s_n$, kde $n$ je druhá mocnina. Idea spočíva v tom, že najskôr prvých $n/2$ komparátorov vytvorí $n/2$ zoradených sekvencií, každá o dĺžke 2. V ďalšom kroku sa páry zlúčia do zoradených sekvencií o dĺžke 4 za použitia zlučovacích sietí $2 \times 2$. Podobne sieť narastá v ďalších krokoch, až pokiaľ sú 2 sekvencie o dĺžke $n/2$ zlúčené sieťou veľkosti $n/2 \times n/2$ a vytvorí sa výsledná zoradená sekvencia o dĺžke $n$.




Obr. 2: Architektúra OEMS ($n \times n$) siete. (*Prevzaté z prednášok PRL doc. Hanáčka.*)

Z uvedeného vyplýva, že pre zoradenie ôsmich čísel bude potrebné vybudovať sieť pozostávajúcu postupne zo sietí $1 \times 1$ až po sieť $4 \times 4$.

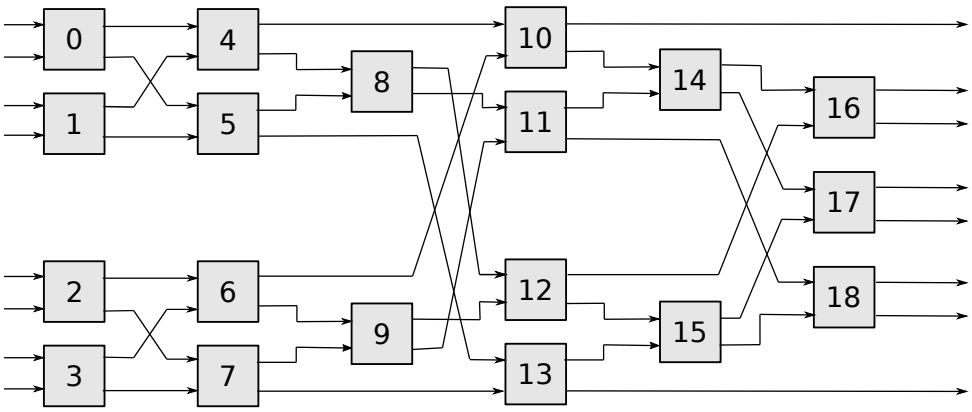
### 1.1 Analýza algoritmu

Keďže sa veľkosť zlučovaných sekvencií zdvojnásobuje každou etapou, vzniká nám celkovo $log\, n$ etáp.

#### Časová zložitosť

Čas potrebný k zlúčeniu sekvencie $k(2^i)$ v *i*-tej etape dvoch zoradených sekvencií o veľkosti $k(2^{i-1})$ čísel je podľa rovnice 1 $k(2^i) = 1$:

$$\begin{cases} k(2) = 1 & pre\, i = 1 \\ k(2^i) = k(2^{i-1}) + 1 & pre\, i > 1 \end{cases} \quad (1)$$




Obr. 3: Architektúra implementovanej OEMS siete pre radenie 8 čísel.

Pre zoradenie sekvencie $k$ pomocou OEMS je časová zložitosť o dĺžke $n$:

$$t(n) = \sum_{i=1}^{log\, n} k(2^1) = \mathcal{O}(log^2 n). \tag{2}$$

Sieť je veľmi rýchla. Umožňuje zoradiť sekvenciu o dĺžke $2^{20}$ za $20^2$ časových jednotiek.

### Priestorová zložitosť

Počet komparátorov potrebných k zlúčeniu sekvencie $l(2^i)$ v $i$-tej etape dvoch zoradených sekvencií s počtom čísel $l(2^{i-1})$ je podľa rovnice 3 $l(2^i) = (i-1)2^{i-1}+1$.

$$\begin{cases} l(2) = 1 & pre\, i = 1 \\ l(2^i) = 2l(2^{i-1}) + (2^{i-1}) - 1 & pre\, i > 1 \end{cases} \tag{3}$$

K zoradeniu sekvencie $l$ o dĺžke $n$ pomocou OEMS dostávame priestorovú zložitosť:

$$p(n) = \sum_{i=1}^{log\, n} 2^{(log\, n)-1} l(2^i) = \mathcal{O}(n\, log^2 n). \tag{4}$$

OEMS vyžaduje veľký počet komparátorov. Pre sekvenciu o počte $2^{20}$ potrebuje 400 miliónov komparátorov.

### Cena

Z rovnice 2 poznáme časovú zložitosť $t(n) = \mathcal{O}(log^2 n)$ a z rovnice 4 priestorovú zložitosť $p(n) = \mathcal{O}(n\, log^2 n)$. Celková cena zoradenia pomocou OEMS je:

$$c(n) = p(n) \times t(n) = \mathcal{O}(n\, log^4 n). \tag{5}$$

Z ceny vyplýva, že OEMS nie je optimálny, keďže potrebuje vyše operácií ako optimálne sekvenčný algoritmus ($\mathcal{O}(n\, log\, n)$). Architektúra OEMS je taktiež veľmi nepravidelná a prepojenia komparátorov sa líšia v závislosti na veľkosti $n$.

## 2 Implementácia

Algoritmus je implementovaný v programovacom jazyku *C++*. Pre posielanie správ medzi procesmi je použitá knižnica *Open MPI*. Implementácia sa nachádza v súbore `oems.cpp`. Pri spustení programu sa ihneď po inicializácií knižnice Open MPI zistí číslo procesu, ktoré mu bolo pridelené. Podľa tohto čísla `world_rank` sa potom rozhoduje ako sa bude daný proces správať a ako bude komunikovať. Číslo `world_rank` sú prirodzené čísla od 0 (vrátane) po celkový počet procesov (`world_size`). Proces s `world_rank` == 0 je tzv. `MASTER`, ktorý uskutočňuje vstupné a výstupné operácie a iniciuje a riadi komunikáciu s ostatnými procesmi.

Program načítava vstupnú sekvenciu zo súboru `numbers` vo funkcií `read_send_numbers`. Funkcia je riadená procesom `MASTER`, teda procesom s `world_rank` == 0. Vstupnú sekvenciu postupne vypisuje v žiadanom tvare na štandardný výstup. Zároveň sa zasielajú správy načítavaných čísel pomocou `MPI_Send` prvým 4 procesom (s rank 0 – 3). Teda master si posiela načítané čísla aj sám sebe. Zasielaním či prijímaním prvkov od procesov sa bude ďalej myslieť dvojica funkcií Open MPI – `MPI_Send`/`MPI_Recv`.

Nasledujúca časť spočíva v samotnom algoritme OEMS, ktorého architektúra je znázornená na obrázku 3, kde čísla znázorňujú `world_rank` jednotlivých procesov.

Radenie prebieha vo funkcií `oems`. Pre každý z

procesov sa podľa `rank` vyberie konkrétna vetva pre daný proces. Prijímanie a odosielanie správ je implementované na základe architektúry zobrazenej na obrázku 3. Teda prijíma správy s číslom pre radenie podľa zo `source` podľa toho, ktorý `rank` proces má dané číslo uložené vo svojom bufferi a svoj zoradený vstup zasiela na konkrétne vstupy procesov daného rank-u. Výnimkou je prvá štvorica, (teda ranky 0-3), kde master zaslal hodnoty na vstupy ešte vo funkcií `read_send_numbers`.

Procesy s rank-om *10, 13, 16, 17 a 18*[1] posielajú správy s výsledkom master-u, ktorý zoradenú postupnosť vypisuje na štandardný výstup v očakávanom formáte.

# 3 Záver

Vrámci tohto projektu bola úspešne vytvorená implementácia algoritmu Odd-Even Merge Sort pre zoraďovanie 8 čísel. Bola uvedená časová a priestorová zložitosť, ktorá vyhodnocuje výhody a nevýhody algoritmu, vzhľadom na rastúce sekvencie.

[1] Proces 10 zasiela master-u len svoj výstup **L**, naopak proces 13 len svoj výstup **H**, a zvyšné procesy 16, 17, 18 oboje svoje výstupy v správnom poradí.