Dokumentácia ku 1. projektu z predmetu PRL

René Rešetár (xreset00)

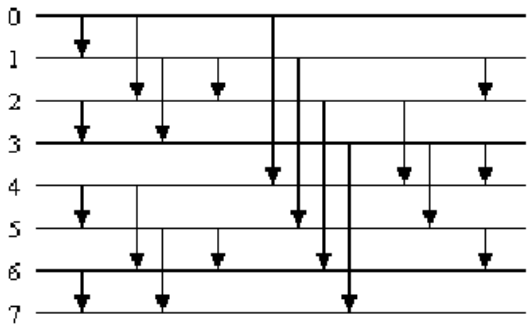
10. apríla 2022

1 Rozbor algoritmu

V tomto projekte sme mali za úlohu implementovať algoritmus *Odd-even mergesort*, ktorý bol vyvinutý pánom K. E. Batcherom. Algoritmus je založený na spojovacom (*merge*) algoritme, ktorý vždy spája dve zoradené polovice postupnosti do kompletne zoradenej postupnosti. Algoritmus nie je závislý na dátach, tj. prevádzajú sa rovnaké porovnania bez ohľadu na skutočné údaje. Preto môže byť tento algoritmus implementovaný ako triediaca sieť.

1.1 Algoritmus odd-even mergesort(n)

- **Vstup:** postupnosť $a_0, ..., a_n - 1$ dĺžky $n > 1$ ktorých dve polovice sú zoradené (n^2)
- **Výstup:** zoradená postupnosť
- **Metóda:** if n > 2 then
  1. Použi odd-even merge$(n/2)$ rekurzívne na dve polovice $a_0, ..., a_n/2 - 1$ a $a_n/2, ..., a_n - 1$ postupnosti
  2. odd-even mergesort(n)



Obr. 1: Obrázok z http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/networks/oemen.htm ukazuje odd-even mergesort pre n = 8

V tomto projekte sme však využili paralelizmu a preto sa tento algoritmus nevykonával rekurzívne, ale postupným spájaním porovnávaných dvojíc následovne. V prvej fáze sa na vytvorenie použije rad n/2 komparátorov n/2 triedených sekvencií, každá s dĺžkou 2. V druhej fáze sa ich páry zlúčia do triedených sekvencií dĺžky 4 pomocou radu (2,2)-zlučovacích sietí. Opäť v a V tretej fáze sa páry sekvencií dĺžky 4 spájajú pomocou (4,4)-zlučovacích sietí do sekvencií dĺžky 8. Proces pokračuje až do dvoch sekvencií dĺžky n/2 každý z nich je zlúčený pomocou (42, n/2) zlučovacej siete, aby sa vytvorila jediná triedená sekvencia dĺžky n. Výsledná architektúra je známa ako sieť triedenia nepárne-párne a je znázornené na obr. 4.1 pre S = {8,4,7,2, 1,5,6,3}.

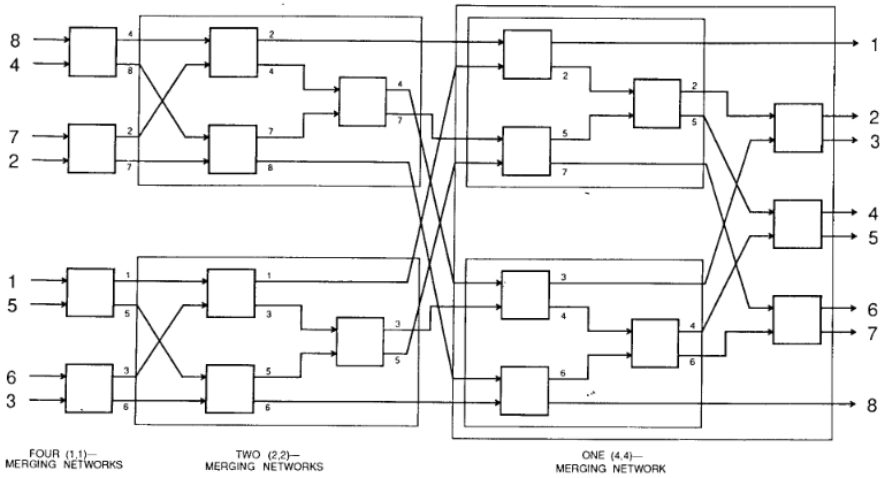


Figure 4.1 Odd–even sorting networks for sequence of eight elements.

Obr. 2: Obrázok aj popis z prezentácií k predmetu PRL

1.2 Časová zložitosť pre postupnosť o dĺžke n ($t(n)$)

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

Kde $s(2^i)$ je čas potrebný pre í-tu časť na spojenie dvoch zoradených postupnosti o $2^i - 1$ elementoch.

1.3 Počet procesorov pre postupnosť o dĺžke n ($p(n)$

$$\begin{aligned} p(n) &= \sum_{i=1}^{\log n} 2^{(\log n)-i} q(2^i) \\ &= O(n \log^2 n). \end{aligned}$$

Kde $q(2^i)$ je počet procesorov (komparátorov) potrebných v i-tej fáze pre zlúčenie dvoch zoradených sekvencií po $2^i - 1$ prvkoch.

1.4 Cena pre postupnosť o dĺžke n ($c(n)$)

Keďže $t(n) = P(log^2 n)$ a $p(n) = O(nlog^2 n)$ celkový počet prevedených porovnaní sieťou odd-even mergesort (čo je cena siete) je:

$$\begin{aligned} c(n) &= p(n) \times t(n) \\ &= O(n \log^4 n). \end{aligned}$$

2 Implementácia

Algoritmus sme implementovali v jazyku *C++* a pre podporu paralelizmu sme použili knižnicu *OpenMPI*.

Pre odosielanie správ medzi procesmi sme použili funkciu:
MPI_Send(void* data, int count, MPI_Datatype datatype, int destination, int tag, MPI_Comm communicator)

Pre príjmanie správ:
MPI_Recv(void* data, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm communicator, MPI_Status* status)

Po implementácií a otestovaní algoritmu sme prišli na to, že synchronizáciu nebolo treba v tomto prípade zabezpečovať. Akurát sme pre závisle **send** a **recv** volania nastavili rovnaký tag.

3 Záver

Kvôli zlému odhadu z mojej strany ohľadom časovej náročnosti projektu som si na neho vymedzil málo času. Preto som nestihol urobiť žiadne merania. Podarilo sa mi iba implementovať funkciu na meranie času ktorá vypíše čas priebehu algoritmu v tvare **sekundy:nanosekundy**. Jej výsledkom bolo **0:121349591** a teda priebeh algoritmu trval 0.121349591 sekundy.