Dokumentace k projektu č. 1 předmětu PRL

Enumeration sort

Jakub Stejskal

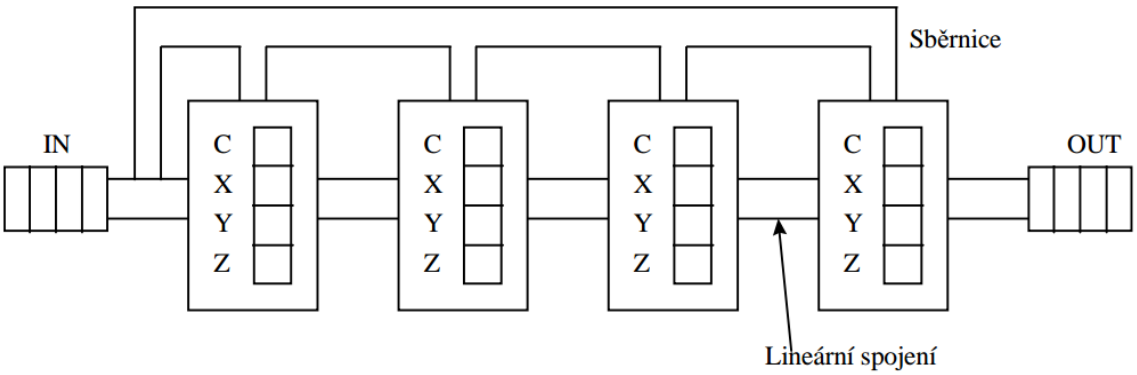
29. března 2017

Rozbor algoritmu

Implementovaná verze Enumeration sortu řadí prvky pomocí lineárního pole procesorů o počtu n, což znamená, že počet procesorů je stejný jako počet vstupních prvků. Všechny procesory jsou propojeny společnou sběrnicí, která je schopna v danou chvíli přenést jeden prvek z libovolného procesoru na jiný.

Každý procesor obsahuje čtyři registry ($i \in 1..n$ značí pořadí procesoru v lineárním poli):

- X_i - obsahuje vstupní prvek x_i
- Y_i - obsahuje postupně prvky $x_1 \ldots x_i$
- Z_i - obsahuje seřazený prvek Y_i
- C_i - počet prvků menších než x_i



Obrázek 1: Nákres sítě pro řazení pomocí Enumeration sortu.

Algoritmus pracuje ve třech krocích:

1. Všechny registry C se nastaví na hodnotu 1.
2. Následující činnosti se opakují 2n-krát ($1 \leq k \leq 2n$):
   - Pokud vstup není vyčerpán, vstupní prvek x_i se vloží do X_i pomocí sběrnice a do Y_i pomocí lineárního spojení. Obsah všech registrů Y_i se posune vpravo (lineárním spojením, procesor i pošle hodnotu Y_i procesoru $i + 1$.
   - Každý procesor, který má neprázdné registry X_i a Y_i porovná jejich hodnotu a pokud platí $X_i > Y_i$ inkrementuje hodnotu v registru C.
   - Je-li $k > n$ (vyčerpaný vstup) procesor P_{k-n} pošle sběrnicí obsah svého registru X procesoru P_{Ck-n}, který jej uloží do svého registru Z.
3. V následujících n cyklech procesory přesouvají obsah svých registrů Z doprava a procesor P_n produkuje seřazenou posloupnost.

Analýza:

- Krok 1: konstantní čas
- Krok 2: trvá $2n$ cyklů
- Krok 3: trvá n cyklů
- Časová složitost: $t(n) = O(n)$
- Cena: $c(n) = t(n) \cdot p(n) = O(n) \cdot n = O(n^2)$

Implementace

Implementace byla provedena v jazyce C++ s využitím knihovny *Message Passing Interface (MPI)* pro paralelní komunikaci jednotlivých procesorů.

Implementovaný algoritmus se nepatrně liší od Enumeration sortu probraného na přednáškách ve dvou bodech:

- Umožňuje řadit takové posloupnost, které obsahují stejná čísla.
- Hlavní část řazení netrvá $2n$ kroků, ale pouze n kroků. Hodnotu registru X je totiž možné poslat na správný procesor již po paralelním porovnání všech čísel po n krocích v konstantním čase po sběrnici.

Sběrnice i lineární propojení bylo realizováno funkcí `MPI_Send()`. Zatímco po sběrnici byla čísla odeslána zvlášť každému procesoru (n-krát), přes lineární spojení byl vždy proveden jen posun vpravo (na procesor s id od jedničku větším). Přijímání zpráv bylo realizováno pomocí funkce `MPI_Recv()`.

Obě výše zmíněné funkce nabízí možnost přijímat/posílat zprávy s určitým označením (TAG). Pro rozeznání, kam patří jednotlivé zaslané hodnoty byly navrženy čtyři TAGy, pro každý registr jeden.

Vyhodnocení stejných čísel

Základní verze Enumeration sortu neumožňuje řadit posloupnosti, které obsahují dvě a více stejných čísel. Proto bylo nutné algoritmus upravit pro tuto variantu. Úprava spočívá v zasílání ne pouze řazeného čísla, ale v zasílání struktury, která obsahuje řazené číslo a jeho původní index. Pokud se vyskytnou dvě stejná čísla přijde na řadu porovnání indexů a pokud je index čísla v registru X vyšší než v registru Y je inkrementována hodnota v registru C.

Výsledná posloupnost

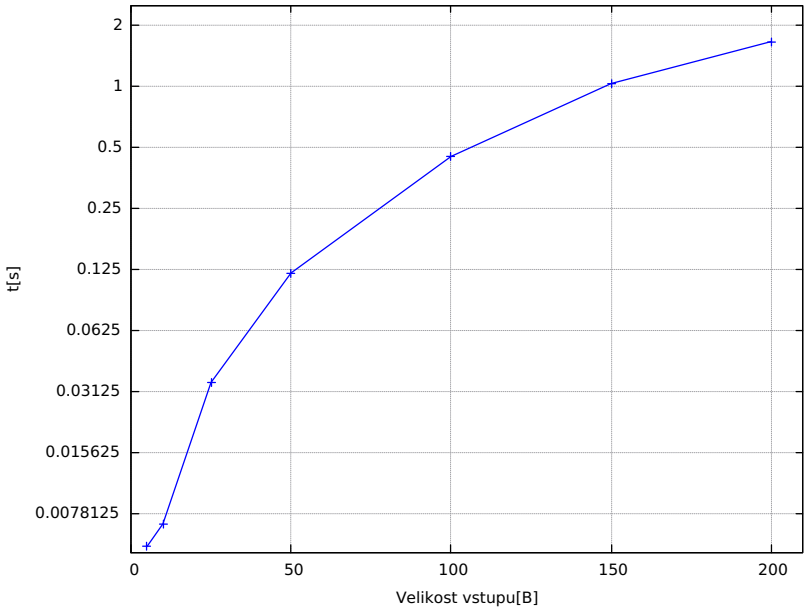
Výsledná posloupnost je tvořena v prvním, řídícím, procesoru. V cyklu je n-krát odchycena zaslaná zpráva od ostatních procesorů s TAGem pro seřazené číslo. Ze zaslané struktury je pak tvořeno pole o délce n, kde na zaslaný index je přiřazena zaslaná hodnota. Po skončení cyklu je v poli seřazená posloupnost vstupních čísel.

Experimenty s vytvořenou realizací

Pro měření času řazení vstupních posloupností byla využita funkce `MPI_Wtime()` z knihovny *MPI*. Pro co největší rychlost byly odstraněny všechny výpisy a části programu, které nejsou potřebné pro řazení. Testy pro jednotlivé velikosti vstupu byly provedeny vícekrát a výsledek byl zprůměrován.

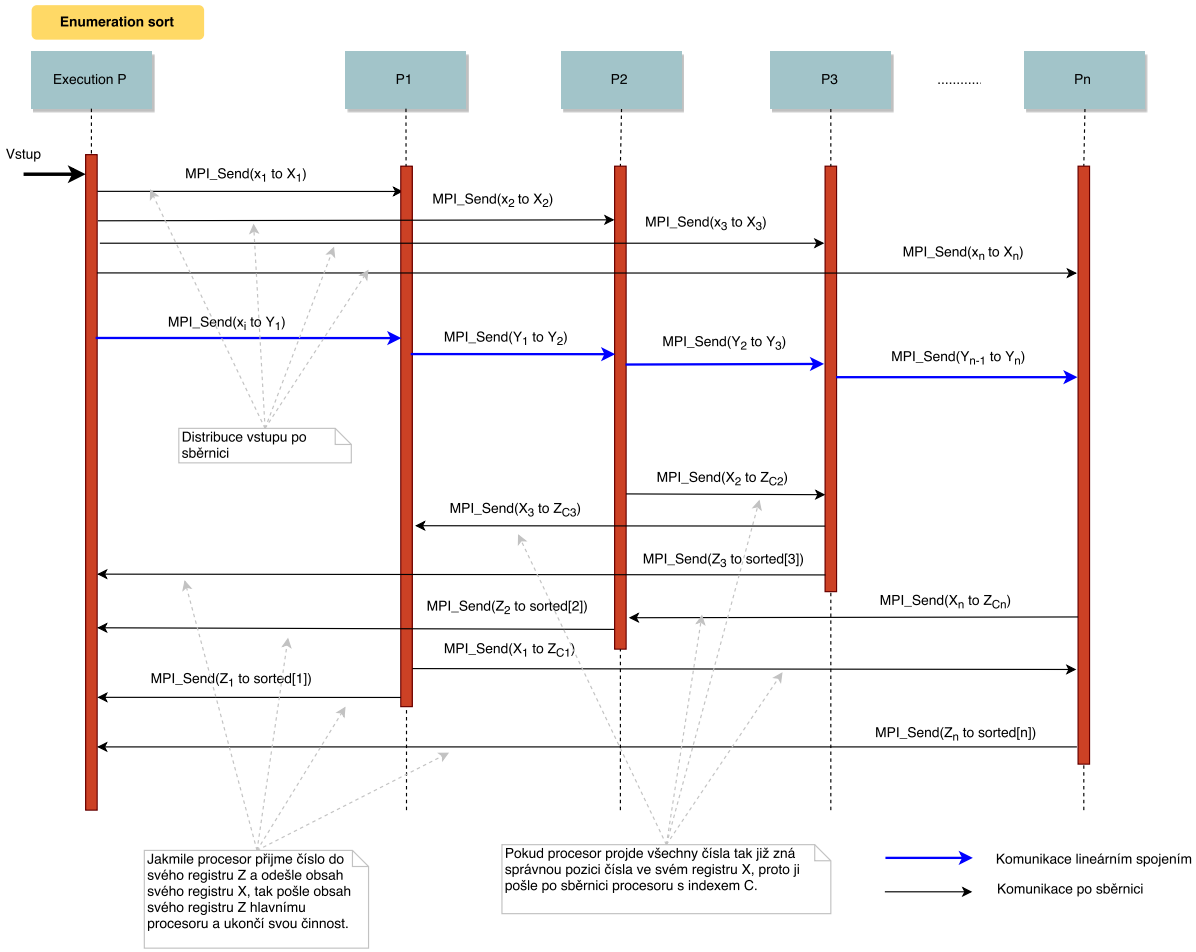
Počet prvků [B]	5	10	25	50	100	150	200
Naměřený čas [s]	0.00534	0.00691	0.03441	0.11931	0.44845	1.02665	1.65442

Tabulka 1: Naměřené hodnoty.



Obrázek 2: Graf znázorňující naměřené hodnoty v závislosti na velikosti vstupu.

Komunikační protokol



Obrázek 3: Sekvenční diagram pro komunikaci procesorů.

Závěr

TODO