STRÁNKA

1. projekt

První projekt - Pipeline merge sort

Pomocí knihovny Open MPI implementujte v jazyce C++ algoritmus Pipeline Merge Sort podle přednášek.

Aktualizace zadání

- 21.2.2024: přidán příklad výstupu
- 22.2.2024: doplněno upřesnění ohledně počtu procesorů
- 26.2.2024: upraven spouštěcí skript, nyní se počet procesorů zaokrouhluje vždy nahoru

Odevzdání

Projekt odevzdejte nejpozději **8.4.2024, 23:59:59** do StudISu. Odevzdává se pouze **dobře okomentovaný** zdrojový soubor **pms.cpp**.

Zadání

Implementuje algoritmus Pipeline Merge Sort tak, jak byl prezentován na přednáškách. Na vstupu předpokládejte neseřazenou posloupnost celých čísel. Vstupní posloupnost načítejte ze souboru *numbers.*

Soubor numbers

Soubor *numbers* obsahující čísla velikosti 1 byte, která jdou bez mezery za sebou. Pro příklad vytvoření tohoto souboru uvádíme kód níže. ve kterém je ukázáno vytvoření takovéto posloupnosti náhodných čísel a její uložení do souboru pomocí utility *dd*. Tato utilita generuje náhodná čísla v rozsahu určeném velikostí bloku. Při bloku 1B jsou hodnoty v rozsahu 0-255. Vygenerovaná čísla jsou pak přesměrována do souboru. Vznikne tedy další soubor s náhodnými znaky jdoucími bez mezery za sebou. Po otevření v libovolném textovém editoru se hodnoty tváří jako náhodné ascii znaky, které by však měly být chápany jako celá čísla. Soubor je v tomto případě chápan jako binární.

Lze ho vygenerovat pomocí následujícího skriptu:

```
#!/bin/bash

if [ $# -lt 1 ];then #pocet cisel bud zadam nebo 4 :)
    numbers=4;
else
    numbers=$1;
fi;

# pocet procesoru
calc=$(echo "(l($numbers)/l(2))+1" | bc -l)
proc=$(python3 -c "from math import ceil; print (ceil($calc))") # zaokrohleni nahoru
```

```
#pocet procesoru nastaven podle poctu cisel (lze i jinak)
#proc=$(echo "(l($numbers)/l(2))+1" | bc -l | xargs printf "%1.0f") # uprava 26.2.
```

```
#preklad zdrojoveho souboru
mpic++ --prefix /usr/local/share/OpenMPI -o pms pms.cpp
```

```
#vyrobeni souboru s nahodnymi cisly
dd if=/dev/random bs=1 count=$numbers of=numbers 2> /dev/null
```

```
#spusteni programu
mpirun --prefix /usr/local/share/OpenMPI  -np $proc pms
```

```
#uklid
rm -f pms numbers
```

Výstup

Program na standardní výstup (*stdout*) vypíše:

- posloupnost na vstupu,
- posloupnost seřazenou **vzestupně** (každé číslo na samostatném řádku).

Pokud se vyskytne chyba, vypište ji na standardní chybový výstup (stderr). Výstup může vypadat pro posloupnost 4 prvků např.:

```
54 53 70 25
25
53
54
70
```

Postup, doporučení

Procesor s rankem 0 načte vstupní posloupnost a rozešle její prvky. Další procesory spojují přijaté prvky do posloupností dle algoritmu. Výstupem je seřazená posloupnost. Zamyslete se nad funkčností jednotlivých procesorů a nad funkčností jednotlivých vstupních front. Pro implementaci fronty vyberte vhodný datový typ. Vztah pro počet procesorů z přednášek vrací počet procesorů bez vstupního procesoru (tj. procesoru, který pouze ze vstupní fronty rozděluje prvky do dvou vstupních front následujícího procesoru).

Hodnocení

V rámci hodnocení se zaměřím na

- dodržení zadání (tj. aby program dělal co má dle zadání, a vypisoval co má na stdout)
- funkčnost programu, dodržení předepsaného algoritmu
- kvalitu zdrojového kódu (komentáře, pojmenování proměnných a konstant, ...)

Diskuse, dotazy

S dotazy se obracejte na iveigend@fit.vut.cz , případně lze využít i diskusní fórum pro dotazy.

Naposledy změněno: pondělí, 26. února 2024, 17.13