Course Wiki

Login xdobis01, **Dobiš Lukáš, Ing.**, 1st year NMAL, full-time, FIT
Ac.Year 2020/2021 study no.3: regular enrolment

Zpět na termíny

[[PRL Home]]

Navigation

- Main Page
- Recent changes
- Help

Page

Go Search

Toolbox

- What links here
- Upload file
- File list
- Page index
- History

- Page
- Source
- Watchlist

Implementace algoritmu "Pipeline merge sort"

Contents

Pomocí knihovny Open MPI implementujte v jazyce C/C++ algoritmus Pipeline-Merge sort tak, jak byl uveden na přednášce PRL.

Deadline

9. 4. 2021

Vstup

Vstupem je posloupnost šestnácti náhodných čísel uložených v souboru.

Soubor numbers

Soubor *numbers* obsahující čísla velikosti 1 byte, která jdou bez mezery za sebou. Pro příklad vytvoření tohoto souboru prostudujte soubor *test* (sekce ukázkové zdrojové kódy), ve kterém je ukázáno vytvoření takovéto posloupnosti náhodných čísel a její uložení do souboru pomocí utility *dd*. V případě tohoto projektu nastavíte "numbers" napevno na 16. 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.

Výstup

Výstup na *stdout* se skládá ze dvou částí:

1. Jednotlivé načtené neseřazené hodnoty v jednom řádku oddělené mezerou (vypsat po načtení prvním procesorem).
2. Jednotlivé seřazené hodnoty oddělené novým řádkem (od nejmenšího po největší).

Příklad výstupu

```
4 68 1 54 ...
1
4
54
68
...
```

Postup

Vytvořte testovací skript *test*, který bude řídit testování. Tento skript bude mít tyto vlastnosti:

- Bude pojmenován *test* nebo *test.sh*.

Skript vytvoří soubor *numbers* s 16ti náhodnými čísly a následně spustí program se správným počtem procesorů. Skript nakonec smaže vytvořenou binárku a soubor *numbers*. Vzhledem ke strojové kontrole výsledků se v odevzdané verzi kódu **nebudou vyskytovat žádné jiné výstupy** než uvedené a ze stejných důvodů je třeba dodržet výše uvedené body týkající se testovacího skriptu. Za nedodržení těchto požadavků budou strhávány body.

Implementace

Algoritmus implementujte v jazyce C/C++ pomocí knihovny Open MPI.

Dokumentace

Součástí řešení je dokumentace, která bude o rozsahu **maximálně 3 strany** (rozumné a odůvodněné překročení limitu stran není důvod k bodové srážce) funkčního textu.

Do dokumentace **nedávejte**:

- Úvodní stranu, obsah, popis zadání.

V dokumentaci **popište**:

- Rozbor a analýzu algoritmu, odvoďte jeho teoretickou složitost (časovou a prostorovou složitost, celkovou cenu). Uvedené vzorce slovně popište, včetně jejich proměnných.
- Implementaci.
- Komunikační protokol, jak si "procesy" zasílají zprávy. Pro vizualizaci použijte sekvenční diagram (http://en.wikipedia.org/wiki/Sequence_diagram). Nezapomeňte, že protokol musí být obecný, tedy pro *n* procesů.
- Závěr, ve kterém zhodnotíte dosažené výsledky.
- Pozor, hodnotí se i to, jak dokumentace působí! (Zejména vzhled, správná čeština/slovenština/angličtina.)

Knihovna Open MPI je navržena na praktické použití, proto neobsahuje žádné implicitní metody pro měření složitosti algoritmů. Je tedy třeba vymyslet nějaký explicitní způsob jejího měření a ten pak co nejdetailněji popsat v dokumentaci!

Odevzdání

Do wisu se odevzdává jeden archiv xlogin00.{tar|tgz|zip}, který bude velký do 1MB, a který obsahuje:

- zdrojový kód- pms.{c|cpp},
- hlavička- pms.h (pokud ji využijete),
- testovací shellový skript- {test|test.sh},
- dokumentaci- xlogin00.pdf,
- nic jiného...

Platí, že kvalita každé z části vzhledem k požadavkům má vliv na bodové ohodnocení . Počítejte s tím, že veškerá uvedená jména souborů jsou *case sensitive*.

Hodnocení

Způsob možné reklamace projektu bude zveřejněn později.

Top

Page id: 270, shown: 18298, revision: 16558, modified: 2021-04-02 08:16:11 by zborilf