Částicový systém – pomocí zasílání zpráv MPI
3. projekt z ARC 2011 / 2012

Vedoucí: Jiří Petrlík
Email ipetrlik@fit.vutbr.cz
Kancelář L327

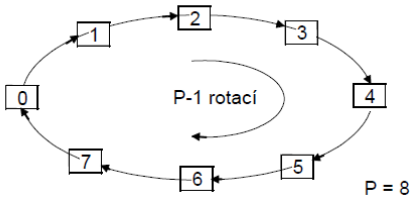
1. Zadání

Cílem tohoto projektu je vyzkoušet si paralelizaci částicového systému na svazku linuxových stanic, které mají nainstalovaný systém pro zasílání zpráv MPI (http://www.mpi-forum.org). Princip fyzikálního modelu je stejný jako u druhého projektu.

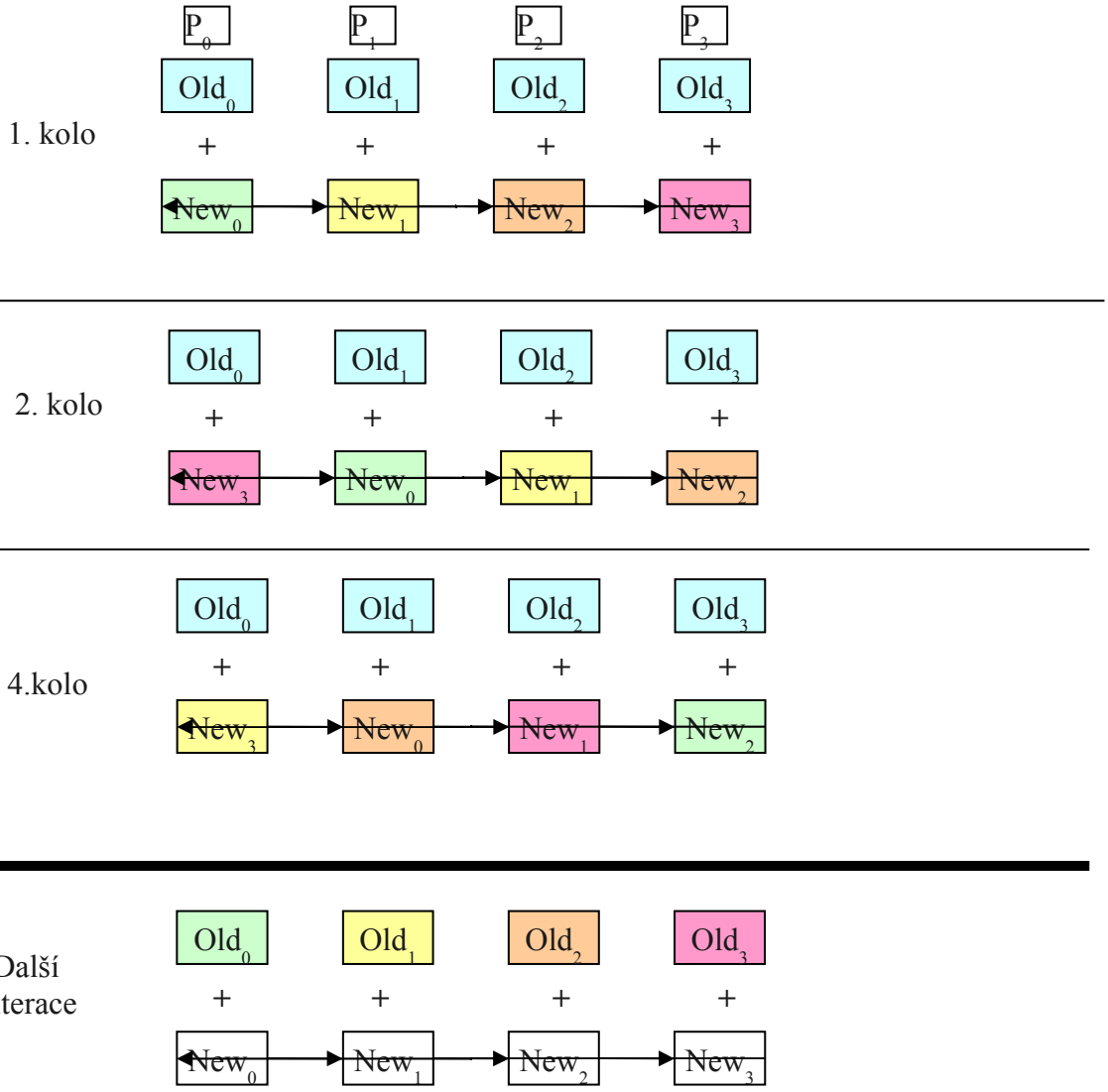
Implementujte 2 různé varianty částicového systému v jazyce C/C++.

1) **Komunikace na konci iterace**.
   a. Na začátku vypočtu obdrží každý procesor kompletní data o všech částicích (poloha, rychlost, hmotnost). Distribuci dat proveďte pomocí **komunikace** (ne současným čtením z disku).
   b. Každý procesor si určí vlastní porci (N/P) částic se kterými bude pracovat.
   c. Každý procesor aktualizuje rychlost svých částic a spočte jejich novou polohu
   d. Pomocí komunikace rozešle nové polohy svých částic ostatním uzlům

2) **Komunikace během iterace**
   a. Jednotlivé procesory jsou zapojeny do kruhové topologie a ke komunikaci požívají rotaci. V každém dílčím kroku komunikace (1 až *P*-1) se bude paralelně počítat s právě získanou dávkou dat (v prvním kroku s vlastní dávkou). V *P*-tém kroku se bude jen počítat s poslední získanou dávkou.V krocích 1,2, …*P*-1 jsou výpočet a komunikace překryty.



   b. Princip funkce je podobný jako u projektu 2 při paralelizaci vnitřní smyčky. Ke každému doručenému balíku dat, připočítám přírůstek rychlosti a polohy vzhledem ke vlastním částicím a data pošlu dále. Takto se během rotace spočtou všechny přírůstky. Po proběhnutí rotace bude mít každý procesor aktualizovanou polohu a rychlost. svých částic



Pro generování vstupních dat použijte program gen-data. Pro tyto testy si zvyšte konstantu NUM_OF_PARTICLES (500, 1000, 2000, 5000) a nastavte vhodné parametry délky kroku (1) a počet časových kroků (100) tak aby výpočet trval nejméně desítky sekund na nejrychlejším použitém stroji. Pro ladění použijte menší hodnoty dle uvážení.

Zaměřte se na efektivitu vámi navrženého algoritmu. Zaměřte se na redukci komunikační režie, zvolením vhodné velikosti zasílaných zpráv, použitých komunikačních rutin nebo překrýváním výpočtu s komunikací. Vámi navržené řešení vyzkoušejte na 2-8 pracovních stanicích (i lichý počet procesů, pokud to dává smysl).

Funkčnost programu (tj. i překlad bez jakýchkoliv warningů a chyb) ověřte na školních linuxech (server merlin, PCNxxx, PCOxxx). Ověřte především, zda váš program produkuje korektní výsledky pro různý počet uzlů

Výsledek projektu je doplněná kostra programu, odevzdává se tento jediný soubor - proj03.c. Tuto kostru lze libovolně modifikovat.
Odevzdávání se děje přes IS. Implementace obou metod vložte do jednoho souboru s názvem proj03.c. Jednotlivé varianty se budou volit okomentováním patřičné funkce nebo podmíněným překladem.

2. Postup řešení

Ve složce MPI_test je přiložena jednoduchá aplikace pro otestování MPI a pro pochopení základních principů. V tomto příkladu máme několik procesů (volte sudý počet), které jsou zapojeny do kruhové topologie. Každý uzel nejprve vypíše hlášku o tom kdo je a posléze pošle svému sousedovi s ID o jedničku vyšším jednoduchou zprávu. Tentýž uzel poté přijme zprávu od uzlu s ID o 1 nižším. Cílem tohoto příkladu je otestovat funkčnost MPI ve vašem prostředí a ukázat základní principy jednoduché komunikace.
Tento prográmek můžete využít pro testování základních principů MPI.
Zkompilovaní provedete příkazem `make`.
Ke spuštění dojde příkazem `./mpi Pocet_procesorů test_mpi`

3. Návod pro spuštění

Spuštění paralelního kodu:

1) nastavení MPI – provedeme pouze tehdy, pokud jsem MPI nikdy před tím nepoužili.
    a. Nejprve je dobré (ale ne nutné) natavit si RSA klíče, abychom při každém spuštění nemuseli zadávat znova hesla pro každý počítač na kterém budeme testovat.
    Spustě příkaz `ssh-keygen -t rsa`
    Nyní stistkněte 3x po sobě `Enter` – to znamená nechte default adresář pro zadání jména souboru, a prázdnou frázi pro ověření.
    Nyní se přepněte do adresáře `.ssh` a zkopírujte soubor `id_rsa.pub` do stejného adresáře a pojmenujte ho `authorized_keys`
    b. Nastavení dávky `mpi`, která slouží pro spuštění programu v prostředí MPI
    Změňte řádek
    `MPIHOSTS=${MPIHOSTS-"PCO103-00 PCO103-01 PCO103-02 PCO103-03 "}`
    Zde zapište na jeden řádek, jména počítačů na kterých budete ladit
    Dále nastavte dávku jako spustitelnou `chmod u+x mpi`
    c. Na každý z těchto počítačů se přihlaste pomocí `ssh` a přidejte si jejich klíč – stačí napsat `yes`
2) Spuštění samotného programu – program se spouští z příkazového řádku pomocí příkazu `./mpi POCET_PROCESORU spustitelny_souboru params..` Tedy například: `./mpi 4 proj03 particles.dat 1 1000` pokud chci pracovat se 4 počítači
3) Čištění po sobě – Pokud váš algoritmus havaruje – je **NUTNÉ** se přihlásit na jednotlivé stroje pomocí `ssh`. Poté pomocí příkazu `ps -ef | grep vas_login` si nechat vypsat vaše procesy spuštěné na jednotlivých stanicích a pokud tam zůstane něco z MPI tak je pomocí příkazu `kill -9 PID_procesu` všechny postřílet!!!
    Zabráníte tak zahlcení stanic, které by bez vašeho zásahu neustále prováděly zacyklený kód. Taková stanice by šla jiným člověkem efektivně použít pouze po restartu.

4. Užitečné poznámky a rady a upozornění!

Zde několik málo rad a poznámek, na které jsem v průběhu času přišel.

- Snažte se použít co nejméně komunikací – ale pozor, velikost zprávy je jistým způsobem omezena a záleží na implementaci MPI a na stavu počítačové sítě.
- MPI nemá rádo když spouštíte program s adresáře který má v názvu mezery.
- Inicializace MPI uvnitř programu (funkce `MPI_Init`) sice nemusí být na začátku algoritmu ale MUSÍ být před první funkcí pracující s parametry programu (`argc`, `argv`), protože tato funkce si též přebírá parametry `argc` a `argv` a pozmění je, tak aby všechny procesy dostaly správné vstupní parametry!!!!
- Dejte si pozor na to, které MPI funkce chtějí proměnou předanou odkazem a které hodnotou.
- Zasílání zpráv – v MPI je možné posílat celé struktury jazyka *C*. Pokud vám to nepůjde, nebo to nebudete chtít využít, je zde i možnost zabalit data například do pole typu char nebo double a u příjemce si je opět rozbalit do vlastních struktur.
- Ladění leze v podstatě provádět pouze tak – že si v do kódu vložíte vlastní výpisy. U každého výpisu si ovšem nezapomeňte připsat číslo procesu, jinak nepoznáte od koho výpis je. Snažte se dělat každou zprávu do svého logu pouze na jeden řádek – lépe se vám to bude hledat.
- Při zkoumání výpisů se může stát, že hlášky od jednotlivých procesů jsou zpřeházené, např. 1 posílá zprávu 2 se objeví později než 2 přijala zprávu od 1. Toto je dáno synchronizací při přístupu k `cout`.
- Pokud si nebudete vědět rady – napište mi email – ipetrlik@fit.vutbr.cz
- Pokud přijdete na další zajímavosti, klidně mi je napište na email – budou se hodit vašim kolegům za rok :-)