Multiprocesorski sistemi

Domaći zadatak 2

MPI – komunikacija, izvedeni tipovi, grupe i komunikatori

(10 poena)

Uvod

Cilj zadatka je da studente obuči da samostalno podese MPI okruženje i razvijaju osnovne MPI programe korišćenjem rutina za pojedinačnu i kolektivnu komunikaciju, izvedenih tipova podataka, grupa i komunikatora.

Podešavanje okruženja

Podešavanja okruženja izvršiti prema uputstvima koja se nalaze u dokumentu za laboratorijsku vežbu 2 - MPI. Obratiti pažnju na razlike koje postoje kod podešavanja za prevođenje na 32-bitnim i 64-bitnim računarskim sistemima. Alternativno, koristiti OpenMPI na računaru **`rtidev4.etf.rs`**.

Zadaci

Svaki od programa treba napisati tako da može biti izvršen sa bilo kojim od broja procesa iz opsega navedenog iza postavke zadatka. **`N`** označava maksimalan mogući broj procesa u trenutno dostupnom MPI okruženju. Za programe koji će biti izvršavani na samo jednom računaru, pretpostaviti da važi **`N=4`**. Svaki program treba da vrši proveru da li je broj procesa tekućeg izvršavanja odgovarajući postavci zadatka. U slučaju da to nije zadovoljeno, prekinuti izvršavanje korišćenjem MPI poziva **`Abort`**.

Korisnik zadaje samo dimenzije problema/nizova/matrica, a sve potrebne ulazne podatke generisati u operativnoj memoriji uz pomoć generatora slučajnih brojeva iz biblioteke jezika C. Generisani brojevi treba da budu odgovarajućeg tipa u opsegu od **`-MAX`** do **`+MAX`**, gde **`MAX`** ima vrednost 1024. Za sve zadatke je potrebno napisati i sekvencijalnu implementaciju odgovarajućeg problema koja će biti korišćena kao referentna (*gold*) implementacija prilikom testiranja programa. Svaki program treba da:

- Generiše ulazne test primere.
- Izvrši MPI implementaciju nad zadatim test primerom.
- Izvrši sekvencijalnu implementaciju nad zadatim test primerom.
- Uporedi rezultat MPI i sekvencijalne implementacije problema.
- Ispiše "**`Test PASSED`**" ili "**`Test FAILED`**" u zavisnosti da li se rezultat izvršavanja MPI implementacije podudara sa rezultatom izvršavanja sekvencijalne implementacije.

Poređenje rezultata MPI i sekvencijalne implementacije problema izvršiti unutar procesa sa rangom 0. Kod zadataka koji koriste realne tipove (**`float`**, **`double`**) tolerisati maksimalno odsupanje od ±**`ACCURACY`** prilikom poređenja rezultata CPU i MPI implementacije. Smatrati da konstanta **`ACCURACY`** ima vrednost 0.01. **Prilikom rešavanja zadataka voditi računa da se postigne maksimalni mogući paralelizam.**

1. Sastaviti program koji određuje vrednost broja π. Proces sa rangom 0 treba da obavesti ostale procese o broju tačaka koje treba da obrade, prikupi podatke od ostalih procesa i treba da učestvuje u računanju. Za komunikaciju koristiti samo MPI pozive **`Bcast`** i **`Reduce`**. [1,N]
2. Sastaviti program koji računa vrednost broja π po grupama od po približno **`K`** članova, gde je **`K`** manje od broja procesa. Procesi treba da budu što ravnomernije raspoređeni po novim grupama i komunikatorima. Svaka grupa treba da izračuna vrednost broja π. Proces sa rangom 0 u svakoj novoj grupi treba da ispiše dobijenu vrednost i pošalje je procesu sa rangom 0 u MPI svetu, koji treba da usrednji dobijene vrednosti i ispiše konačni rezultat. [2,N]
3. Sastaviti program koji vrši cirkularno pomeranje niza celih brojeva za određeni broj mesta ulevo ili udesno. Proces sa rangom 0 u MPI svetu treba da obavlja svu komunikaciju sa korisnikom (unos podataka i ispis rezultata) i ravnopravno učestvuje u poslu sa ostalim procesima. Ako broj elemenata niza nije odgovarajući broju procesa u MPI svetu, prekinuti program. [1,N]

4. Sastaviti program koji pronalazi maksimalni element za svaku kolonu matrice celih brojeva. Dimenzije matrice nisu unapred poznate, pa je matricu potrebno alocirati kao niz dimenzija **`PxQ`**, gde je **`P`** broj vrsta, a **`Q`** broj kolona matrice. Zadatak rešiti korišćenjem *manager - worker* modela. Proces sa rangom 0 (gospodar) učitava podatke, deli posao ostalim procesima i ispisuje dobijene rezultate. Svaki proces radnik prima jednu kolonu matrice, vrši zahtevanu obradu, šalje procesu gospodaru rezultate i ponavlja opisani postupak dok ne dobije signal da prekine sa radom. Za prenos jedne kolone matrice koristiti odgovarajući izvedeni tip podataka. [2,N]
5. Sastaviti program koji vrši sortiranje niza realnih brojeva. Proces sa rangom 0 treba da obavlja svu komunikaciju sa korisnikom (unos podataka i ispis rezultata) i ravnopravno učestvuje u poslu sa ostalim procesima. Raspodela elemenata po procesima treba da bude što ravnomernija. Sortiranje podataka obaviti spajanjem lokalno sortiranih segmenata po principu *tree-based merge*. Za lokalno sortiranje dobijenog segmenta niza svaki proces može da koristi **`qsort`** funkciju iz **`stdlib`** biblioteke. [1,N]

Važno: Ukoliko u nekom zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku i da nastavi da izgrađuje svoje rešenje na temeljima uvedene pretpostavke.

Napomena: Vrednost broja π se može statistički odrediti na više načina uz pomoć generatora pseudoslučajnih brojeva uniformne raspodele. Jedan od načina je generisanje tačaka u ravni sa obema koordinatama u realnom opsegu [0,1]. Vrednost broja π tada može biti određena na osnovu odnosa broja tačaka koje se nalaze u delu kruga poluprečnika 1 sa centrom u koordinatnom početku i broja tačaka koje pripadaju kvadratu stranice 1 koji obuhvata sve generisane tačke.