Multiprocesorski sistemi

Domaći zadatak 1
Školska godina 2024/2025
OpenMP – paralelizacija direktivama
(10 poena)

Uvod

Cilj prvog domaćeg zadatka je da studentima približi osnovne koncepte rada sa **`OpenMP`** tehnologijom koja omogućava paralelizaciju direktivama na sistemima sa deljenom memorijom. Domaći zadaci se rade **samostalno** ili **u paru**. Rešenja domaćih zadataka i izveštaj svaki student predaje **samostalno** u svoj svn repozitorijum.

Podešavanje okruženja

Za rad sa OpenMP tehnologijom koristiti **`gcc`**/**`g++`** na računaru **`rtidev5.etf.rs`** ili instalirati **`gcc`**/**`g++`** prevodilac na lokalnu Windows mašinu korišćenjem Cygwin, MinGW alata ili u okviru Linux subsystem for Windows. Za rešavanje domaćeg zadatka je potrebno imati **`gcc`**/**`g++`** prevodilac verzije 4.4.0 ili noviji koji podržava OpenMP standard 3.0.

Izveštaj

Uz predati domaći zadatak (izvorne kodove) treba napisati i priložiti kratak izveštaj o izvršenoj paralelizaciji i dobijenim ubrzanjima u odnosu na sekvencijalnu verziju koda. Za svaki rešeni zadatak treba kratko opisati uočena mesta koja je moguće paralelizovati i način paralelizacije. Takođe, potrebno je dati logove izvršenog koda za sve test primere koji se izvršavaju i nalaze se u **`run`** datoteci i nacrtati grafike ubrzanja u odnosu na sekvencijalnu verziju. Na graficima je potrebno dati i rezultate poređenja različitih načina paralelizacije za isti broj niti, ukoliko postoje takvi zahtevi u okviru teksta zadatka. Šablon za pisanje izveštaja se nalazi u okviru sekcije za domaće zadatke predmetnog sajta.

Zadaci

Svaki od programa treba napisati i paralelizovati tako da može biti izvršen sa bilo kojim brojem niti iz opsega navedenog iza postavke zadatka. **`N`** označava maksimalan mogući broj procesa u trenutno dostupnom OpenMP izvršnom okruženju. Za programe koji će biti izvršavani na samo jednom računaru, smatrati da **`N`** neće biti više od **`N=16`**. Preporučuje se testiranje zadataka sa 1, 2, 4, 8 i 16 niti.

Kod zadataka gde je to zahtevano, korisnik zadaje samo dimenzije problema/nizova/matrica, a sve potrebne ulazne podatke generisati u operativnoj memoriji uz pomoć generatora pseudoslučajnih 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 ili iskoristiti zadatu sekvencijalnu implementaciju odgovarajućeg problema. U cilju postizanja najboljih performansi, dozvoljeno je menjati izvorni sekvencijalni kod. Izvorni kod, ili njegova modifikacija koja postiže najbolje performanse, je potrebno koristiti kao referentnu (*gold*) implementaciju prilikom testiranja programa.

Svaki program treba da:

- Generiše ili koristi već obezbeđene ulazne test primere.
- Izvrši sekvencijalnu implementaciju nad zadatim test primerom.
- Izvrši paralelnu implementaciju nad zadatim test primerom.
- Ispište vreme izvršavanja sekvencijalne i paralelne implementacije problema.
- Uporedi rezultat sekvencijalne i OpenMP implementacije problema.
- Ispiše "**`Test PASSED`**" ili "**`Test FAILED`**" u zavisnosti da li se rezultat izvršavanja OpenMP implementacije podudara sa rezultatom izvršavanja sekvencijalne implementacije.

Poređenje rezultata OpenMP i sekvencijalne implementacije problema izvršiti na kraju sekvencijalnog dela programa. Kod zadataka koji koriste realne tipove (**`float`**, **`double`**) tolerisati maksimalno odsupanje od **`±ACCURACY`** prilikom poređenja rezultata sekvencijalne i OpenMP implementacije. Smatrati da konstanta

`ACCURACY` ima vrednost 0.01. **Prilikom rešavanja zadataka voditi računa da se postigne maksimalni mogući paralelizam.** Dozvoljeno je ograničeno preuređivanje dostupnih sekvencijalnih implementacija prilikom paralelizacije. **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.**

Dostupne sekvencijalne implementacije se nalaze u arhivi **`MPS_DZ.tar.gz`** koje se mogu preuzeti na adresi http://mups.etf.rs/dz/2024-2025/. Na **`rtidev5.etf.rs`** računaru arhiva se može dohvatiti i raspakovati sledećim komandama:

Dohvatanje: **`wget http://mups.etf.rs/dz/2024-2025/MPS_DZ.tar.gz`**

Raspakivanje: **`tar -xzvf MPS_DZ.tar.gz`**

1. Paralelizovati program koji vrši jednostavno generalizovano množenje matrica u jednostrukoj preciznosti *Single precision floating General Matrix Multiply* (SGEMM). SGEMM operacija je definisana sledećom formom:

   $$C \leftarrow \alpha \cdot A \cdot B + \beta \cdot C$$

   Program se nalazi u datoteci **sgemm.cc** u arhivi koja je priložena uz ovaj dokument.

   Prilikom paralelizacije nije dozvoljeno koristiti direktive za podelu posla (*worksharing direktive*), već je iteracije petlje koja se paralelizuje potrebno raspodeliti ručno. Obratiti pažnju na ispravno deklarisanje svih promenljivih prilikom paralelizacije. Program testirati sa parametrima koji su dati u **run** skripti.

2. Prethodni program paralelizovati korišćenjem direktiva za podelu posla (*worksharing direktive*). Program testirati sa parametrima koji su dati u **run** skripti.

3. Paralelizovati program koji izračunava aproksimaciju integrala koristeći Gauss-Laguerre kvadraturno pravilo. Po ovom pravilu, integral oblika:

   $$\int_a^{\infty} e^{-b(x-a)} f(x) dx$$

   se aproksimira sumom:

   $$\sum_{i=1}^{order} \omega(i) \cdot f(x(i))$$

   gde su $\omega(i)$ težinski koeficijenti, a $x(i)$ odgovarajuće čvorne tačke.

   Program se nalazi u direktorijumu **gauss_laguerre** u arhivi koja je priložena uz ovaj dokument. Program testirati sa parametrima koji su dati u **`run`** skripti.

4. Rešiti prethodni problem korišćenjem koncepta poslova (*tasks*). Obratiti pažnju na eventualnu potrebu za sinhronizacijom. Program testirati sa parametrima koji su dati u **`run`** skripti.

5. Paralelizovati program koji vrši statističku analizu prostorne distribucije posmatranih astronomskih tela korišćenjem tzv. *two point angular correlation function* (TPACF). Algoritam proračunava uglovne distance između svih parova tačaka (tela) sa ulaza i generiše histogram posmatranih distanci.

   Program se nalazi u direktorijumu **tpacf** u arhivi koja je priložena uz ovaj dokument. Program se sastoji od više datoteka, od kojih su od interesa datoteke **main.c** i **model_compute_cpu.c**. Analizirati dati kod i obratiti pažnju na algoritam za izračunavanje distanci unutar datoteke **model_compute_cpu.c**.

   Ukoliko je potrebno međusobno isključenje prilikom paralelizacije programa, koristiti dostupne OpenMP konstrukte. Program testirati sa parametrima koji su dati u **`run`** skripti.