

Multiprocesorski sistemi

Domaći zadatak 1

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.

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 ili MinGW alata. Za rešavanje domaćeg zadatka je potrebno imati **gcc/g++** prevodilac verzije 4.4.0 ili noviji koji podržava OpenMP standard 3.0.

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=8**. Preporučuje se testiranje zadataka sa 1, 2, 4 i 8 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, koja će biti korišćena kao referentna (*gold*) implementacija prilikom testiranja programa.

Svaki program treba da:

- Generiše ili koristi već obezbeđene ulazne test primere.
- Izvrši sekvencijalnu implementaciju nad zadatim test primerom.
- Izvrši paralelnu, OpenMP implementaciju nad zadatim test primerom.
- Ispiše 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_DZ1_OpenMP.zip** ili **MPS_DZ1_OpenMP.tar.bz2** koje se mogu preuzeti na adresi <http://mups.etf.rs/dz/2016-2017/>. Na **rtidev5.etf.rs** računaru arhiva se može dohvatiti i raspakovati sledećim komandama:

Dohvatanje: `wget http://mups.etf.rs/dz/2016-2017/MPS_DZ1_OpenMP.tar.bz2`

Raspakivanje: `tar xjvf MPS_DZ1_OpenMP.tar.bz2`

1. Paralelizovati program koji vrši izračunavanje određenog integrala korišćenjem kvadratnog pravila u dvostrukoj preciznosti. Program se nalazi u datoteci **quad.c** 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 datoteci **run**. [1, N]
2. Prethodni program paralelizovati korišćenjem direktiva za podelu posla (*worksharing* direktive). Program testirati sa parametrima koji su dati u datoteci **run**. [1, N]
3. Paralelizovati program koji vrši izračunavanje dvodimenzionalnog histograma za zadataku sliku. Kod koji treba paralelizovati se nalazi u datoteci **main.c** u arhivi koja je priložena uz ovaj dokument, dok ostale datoteke sadrže prateći kod. Obratiti pažnju na petlje koje je moguće paralelizovati i potrebu za sinhronizacijom. Ulazni test primeri se nalaze u direktorijumu **data**, a način pokretanja programa u datoteci **run**. [1, N]
4. Rešiti prethodni problem korišćenjem koncepta poslova (*tasks*). Obratiti pažnju na potrebu za sinhronizacijom i granularnost poslova. Ulazni test primeri se nalaze u direktorijumu **data**, a način pokretanja programa u datoteci **run**. [1, N]
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. Obratiti pažnju na efikasnost paralelizacije. Ulazni test primeri se nalaze u direktorijumu **data**, a način pokretanja programa u datoteci **run**. [1, N]