

# 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. Domaći zadaci se rade **samostalno** ili **u paru**. Rešenja domaćih zadataka i izveštaj svaki student predaje samostalno.

## 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=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  $\pm$ **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/2023-2024/>. Na **rtidev5.etf.rs** računaru arhiva se može dohvatiti i raspakovati sledećim komandama:

Dohvatanje: `wget http://mups.etf.rs/dz/2023-2024/MPS_DZ.tar.gz`

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

1. Paralelizovati program koji vrši izračunavanje aritmetičkih brojeva. Pozitivan ceo broj je aritmetički ako je prosek njegovih pozitivnih delilaca takođe ceo broj. Program se nalazi u datoteci **arithmetic.c** u arhivi koja je priložena uz ovaj dokument. Obratiti pažnju na ispravno deklarisanje svih promenljivih prilikom paralelizacije. Program testirati sa parametrima koji su dati u **run** skripti.
2. Rešiti prethodni problem korišćenjem koncepta poslova (*tasks*). Obratiti pažnju na eventualnu potrebu za sinhronizacijom i granularnost poslova. Program testirati sa parametrima koji su dati u **run** skripti.
3. Paralelizovati program koji vrši generisanje elemenata **Halton** Quasi Monte Carlo (**QMC**) sekvence. Program se nalazi u datoteci **halton.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 **run** skripti.
4. Prethodni program paralelizovati korišćenjem direktiva za podelu posla (*worksharing* direktive). Program testirati sa parametrima koji su dati u **run** skripti.
5. Paralelizovati program koji se bavi problemom  $n$  tela (***n*-body problem**). Sva tela imaju jediničnu masu, trokomponentni vektor položaja ( $x, y, z$ ) i trokomponentni vektor brzine ( $v_x, v_y, v_z$ ). Simulaciju  $n$  tela se odvija u iteracijama, pri čemu se u svakoj iteraciji izračunava sila kojom sva tela deluju na sva ostala, a zatim se brzine i koordinate tela ažuriraju prema II Njutnovom zakonu. Brzine i položaji su slučajno generisani na početku simulacije. Zbog same prirode numeričke simulacije uveden je parametar **SOFTENING**, koji predstavlja korektivni faktor prilikom izračunavanja rastojanja između čestica (kako je gravitaciona sila obrnuto proporcionalna rastojanju između čestica, za nulta rastojanja i rastojanja bliska nuli, izračunata gravitaciona sila postaje izuzetno velika – teži beskonačnosti).

Program se nalazi u datoteci direktorijumu **nbodymini** u arhivi koja je priložena uz ovaj dokument. Program koji treba paralelizovati nalazi se u datoteci **nbody.c**. Pored samog izračunavanja, program čuva rezultate svake iteracije u zasebnim datotekama (za svako telo se čuvaju pozicije i brzine), dok kod **show\_nbody.py** kreira gif same simulacije.

Ukoliko je potrebno međusobno isključenje prilikom paralelizacije programa, koristiti kritične sekcije ili atomske operacije. Skripta **run** pokreće simulaciju za različite parametre, i nakon toga, za određene simulacije poziva python kod koji kreira gifove.