Univerzitet u Beogradu - Elektrotehnički fakultet

Multiprocesorki sistemi (13S114MUPS, 13E114MUPS)



Domaći zadatak 1 – OpenMP

Izveštaj o urađenom domaćem zadatku

|  |  |
| --- | --- |
| Predmetni asistent: | Kandidati: |
| doc. dr Marko Mišić | Marija Kostić 2015/0096  Stefan Milanović 2015/0361 |

Beograd, novembar 2018.

Sadržaj

[Sadržaj 2](#_Toc529923697)

[1. Problem 1 - SGEMM 3](#_Toc529923698)

[1.1. Tekst problema 3](#_Toc529923699)

[1.2. Delovi koje treba paralelizovati 3](#_Toc529923700)

[1.2.1. Diskusija 3](#_Toc529923701)

[1.2.2. Način paralelizacije 4](#_Toc529923702)

[1.3. Rezultati 5](#_Toc529923703)

[1.3.1. Logovi izvršavanja 5](#_Toc529923704)

[1.3.2. Tabela trajanja programa 13](#_Toc529923705)

[1.3.3. Grafici ubrzanja 13](#_Toc529923706)

[1.3.4. Diskusija dobijenih rezultata 15](#_Toc529923707)

[2. Problem 2 - Jacobi 16](#_Toc529923708)

[2.1. Tekst problema 16](#_Toc529923709)

[2.2. Delovi koje treba paralelizovati 16](#_Toc529923710)

[2.2.1. Diskusija 16](#_Toc529923711)

[2.2.2. Način paralelizacije 16](#_Toc529923712)

[2.3. Rezultati 17](#_Toc529923713)

[2.3.1. Logovi izvršavanja 17](#_Toc529923714)

[2.3.2. Tabela trajanja programa 20](#_Toc529923715)

[2.3.3. Grafici ubrzanja 20](#_Toc529923716)

[2.3.4. Diskusija dobijenih rezultata 22](#_Toc529923717)

1. Problem 1 - SGEMM

U okviru ovog poglavlja je dat kratak izveštaj u vezi rešenja zadatog problema 1.

* 1. Tekst problema

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

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 datoteci *run*. [1, N]

* 1. Delovi koje treba paralelizovati
     1. Diskusija

Kod koji rešava ovaj problem se može razbiti na tri velike celine – najpre se vrši učitavanje ulaznih matrica iz dva tekstualna fajla, vrši se obrada ulaznih podataka u vidu kreiranja izlazne matrice koja predstavlja proizvod unetih matrica, i na samom kraju se vrši ispis izlazne matrice u tekstualni fajl.

Prvu i treću celinu nije moguće paralelizovati – ovi delovi koda predstavljaju rad sa I/O sistemom koji se mora obaviti isključivo unutar jedne niti. U teoriji bi bilo moguće uz pomoć više niti obraditi jedan ulazni fajl tako što bi se fajl najpre memorijski mapirao, a nakon toga bi se vršilo raspoređivanje delova mapiranog fajla različitim nitima u cilju paralelne obrade. Ovo rešenje daleko komplikuje kod, a zbog neophodnog *overhead*-a i zauzeća memorije može dovesti i do usporenja.

Jedini deo koda nad kojim se može vršiti paralelizacija jeste deo koda koji radi obradu, tj. funkcija *void basicSgemm(args)*. U skladu sa postavkom zadatka, paralelizacija će se vršiti na tri načina koji će zasebno biti diskutovati i prezentovani na graficima i tabelama.

* + 1. Način paralelizacije

**Zadatak 1**

Rešenje ovog zadatka nalazi se u datoteci *dz1z1.cpp*. U ovom zadatku neophodno je ručno rasporediti iteracije petlje koja se paralelizuje.

Ovo je urađeno tako što je glavna *for* petlja koja se nalazi u *basicSgemm()* funkciji podeljena na više delova, pri čemu svaki deo obrađuje jedna nit. Uvedena je promenljiva *chunk* koja predstavlja veličinu dela glavne *for* petlje i ova vrednost je ista za sve niti (osim eventualno za poslednju nit), pa se iz tog razloga ona može izračunati samo jednom. Ovo je omogućeno *single* regionom na početku *parallel* regiona programa koja obuhvata glavnu *for* petlju. Pošto se ovaj region nalazi na samom početku *parallel* regiona, obezbeđeno je da je vrednost *chunk* definisana za sve niti (pošto je promenljiva *shared*) u trenutku kada one dolaze do glavne *for* petlje koja se paralelizuje. Kod iz *single* regiona nije bilo moguće staviti van *parallel regiona*, jer u tom slučaju poziv funkcije *omp\_get\_num\_threads()* vraća 1, a ne predviđen broj niti.

Nakon ovoga, definišu se vrednosti *start* i *end* koje su različite za svaku nit i predstavljaju granice obrade glavne *for* petlje (jedna nit obrađuje vrednosti od 0 do *chunk – 1,* druga od *chunk* do (*2 \* chunk) – 1*, itd.).

Jedino što se sada menja jeste samo telo glavne *for* petlje, koje sad iterira od *mm = start* sve dokle važi *mm < end,* čime se postiže ručna paralelizacija petlje.

**Zadatak 2**

Rešenje ovog zadatka nalazi se u datoteci *dz1z2.cpp*. U ovom zadatku neophodno je paralelizovati program koristeći *worksharing* direktive.

U ovom zadatku je kod dosta jednostavniji i malo se menja u odnosu na sekvencijalnu implementaciju, zato što *omp* biblioteka vrši raspoređivanje petlje na više niti. Uočena je savršeno ugneždena pravougana petlja koja se sastoji od 2 petlje, pa je njih moguće paralelizovati *collapse* odredbom u okviru *parallel for* regiona. Ovo je ujedno sve što je neophodno odraditi kako bi se izvršila paralelizacija u ovom zadatku. Za razliku od prethodnog zadatka, ovde se sam kod petlji ne menja uopšte.

**Zadatak 3**

Rešenje ovog zadatka nalazi se u datoteci *dz1z3.cpp*. U ovom zadatku neophodno je paralelizovati program koristeći koncept zadataka (engl. *task-ova*).

Ovaj zadatak takođe ima jednostavniji kod u odnosu na prvi zadatak. Neophodno je staviti glavnu *for* petlju u jedan paralelni region, a zatim izdvojiti deo unutrašnjeg koda kao *task* koji će se izvršiti kasnije. Samim tim se ovaj zadatak može rešiti na 4 načina: Moguće je izvršiti *coarse grain* ili *fine grain* paralelizaciju tako što task obuhvata sve što se nalazi unutar glavne *for* petlje, ili tako što *task* obuhvata samo kod unutar druge *for* petlje. Na ovaj način se posao raspoređuje tako da jedan task predstavlja računanje vrednosti jednog reda, odnosno jedne ćelije krajnje matrice. Pored ovoga, moguće je postaviti samo jednu nit koja će kreirati *task*-ove koje će ostale niti izvršavati, dok je druga mogućnost da sve niti kreiraju *task*-ove, a onda ih i izvršavaju.

Svi ovi načini su isprobani i pokazalo se da najbolje performanse pruža varijanta *coarse grain* paralelizacije pri čemu sve *task*-ove pravi jedna nit. Nakon pokretanja test primera za sve varijante više puta pokazalo se da je jako mala razlika u performansama kada se obrađuje mali ulazni skup podataka. Za veći ulazni skup podataka dobijeno je da su *coarse grain* implementacije u svim merenjima bile brže od *fine* *grain* implementacija. Nakon posmatranja samo *coarse grain* implementacija i njihovih performansi nakon obrade velikog ulaznog skupa podataka, uočilo se da *coarse grain* implementacija kod koje samo jedna nit kreira *task-*ove konstantno daje bolje rezultate od *coarse grain* implementacije kod koje sve nitikreiraju *task-*ove.

Rezultati koji su dobijeni koristeći ovu varijantu predstavljeni su u sekciji **Rezultati** i predstavljeni su na graficima performansi.

* 1. Rezultati

U okviru ove sekcije su izloženi rezultati paralelizacije problema 1 koristeći rešenja sva tri zadatka.

* + 1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere i različit broj niti. Obavezno uključiti u ispis i vremena izvršavanja. Logove pojedinačno uokviriti i obeležiti.

Listing 1. Sekvencijalno izvršavanje SGEMM

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SEQ\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: 0.0245002.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SEQ\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: 12.891.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 2. Zadatak 1 – 1 nit

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 3. Zadatak 1 – 2 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 4. Zadatak 1 – 4 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 5. Zadatak 1 – 8 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 6. Zadatak 2 – 1 nit

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 7. Zadatak 2 – 2 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 8. Zadatak 2 – 4 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 9. Zadatak 2 – 8 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 10. Zadatak 3 – 1 nit

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 11. Zadatak 3 – 2 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 11. Zadatak 3 – 4 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 12. Zadatak 3 – 8 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* + 1. Tabela trajanja programa

|  |  |  |
| --- | --- | --- |
|  | **Small input** | **Medium input** |
| **Sekvencijalno izvršavanje** | 0.0245002 | 12.891 |
| **Zadatak 1 – 1 nit** | 0.0256683 | 12.8851 |
| **Zadatak 1 – 2 niti** | 0.0664114 | 7.05922 |
| **Zadatak 1 – 4 niti** | 0.0758083 | 4.08204 |
| **Zadatak 1 – 8 niti** | 0.0695984 | 4.03053 |
| **Zadatak 2 – 1 nit** | 0.0551437 | 12.9156 |
| **Zadatak 2 – 2 niti** | 0.0603416 | 7.09492 |
| **Zadatak 2 – 4 niti** | 0.0772603 | 4.0402 |
| **Zadatak 2 – 8 niti** | 0.0753353 | 4.00763 |
| **Zadatak 3 – 1 nit** |  |  |
| **Zadatak 3 – 2 niti** |  |  |
| **Zadatak 3 – 4 niti** |  |  |
| **Zadatak 3 – 8 niti** |  |  |

* + 1. Grafici ubrzanja

U okviru ove sekcije su dati grafici ubrzanja paralelnih implementacija u odnosu na sekvencijalnu implementaciju. Ubrzanje na svim narednim graficima predstavljeno je kao odnos trajanja sekvencijalne implementacije programa i trajanja paralelizovane implementacije.

Grafik 1. Zavisnost ubrzanja programa dz1z1 od veličine izlazne matrice i broja niti

Grafik 2. Zavisnost ubrzanja programa dz1z2 od veličine izlazne matrice i broja niti

Grafik 3. Zavisnost ubrzanja programa dz1z3 od veličine izlazne matrice i broja niti

* + 1. Diskusija dobijenih rezultata

Dobijene rezultate treba izdiskutovati i objasniti sa nekoliko rečenica u odnosu na dobijena ubrzanja (ili usporenja), način paralelizacije, karakteristike problema i sl.

zadatak 2 – 1 nit – brze paralelno jer smo komprimovali for petlje na x mestima i time se sprecile nepotrebne iteracije

paralelno izvrsavanje je mnog sporije kod malih fajlova (500 i 1000) - 8 niti

pokusani taskovi – ispali su dosta sporiji ( NAJMANJE DUPLO SPORIJI OD SEKVENCIJALNOG) – ZA MALI FAJL DESET PUTA SPORIJI!!!!

1. Problem 2 - Jacobi

U okviru ovog poglavlja je dat kratak izveštaj u vezi rešenja zadatog problema 2.

* 1. Tekst problema

Paralelizovati program koji vrši jednostavno generalizovano množenje matrica u jednostrukoj Paralelizovati program koji rešava sistem linearnih jednačina *A \* x = b* Jakobijevim metodom. Kod koji treba paralelizovati se nalazi u datoteci *jacobi.c* u arhivi koja je priložena uz ovaj dokument. Obratiti pažnju na raspodelu opterećenja po nitima i testirati program za različite načine raspoređivanja posla. Program testirati sa parametrima koji su dati u datoteci run. [1, N].

* 1. Delovi koje treba paralelizovati
     1. Diskusija

U ovom zadatku moguće je paralelizovati prve dve *for* petlje koje služe za inicijalizaciju, a nakon toga i sav kod koji se nalazi unutar glavne *for* petlje obrade koja inkrementira vrednost promenljive *it* od 0 do *m – 1*. Ovu petlju nije moguće paralelizovati zato što postoji zavisnost po podacima – u narednoj iteraciji ove petlje neophodno je u nizu *x* imati vrednosti koje se računaju unutar same petlje, a te izračunate vrednosti se i koriste u poslednjem delu koda kada se računa vrednost promenljive *r*.

Samim tim, moguće je inicijalizovati kod za inicijalizaciju i kod unutar glavne *for* petlje.

Kod koji rešava ovaj problem se može razbiti na tri velike celine – najpre se vrši učitavanje ulaznih matrica iz dva tekstualna fajla, vrši se obrada ulaznih podataka u vidu kreiranja izlazne matrice koja predstavlja proizvod unetih matrica, i na samom kraju se vrši ispis izlazne matrice u tekstualni fajl.

Kod na samom početku koji vrši alokaciju i dealokaciju prostora, kao i ispise, nema potrebe paralelizovati.

* + 1. Način paralelizacije

Rezultati koji su dobijeni koristeći ovu varijantu predstavljeni su u sekciji **Rezultati** i predstavljeni su na graficima performansi.

* 1. Rezultati

U okviru ove sekcije su izloženi rezultati paralelizacije problema 2 koristeći rešenja sva tri zadatka.

* + 1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere i različit broj niti. Obavezno uključiti u ispis i vremena izvršavanja. Logove pojedinačno uokviriti i obeležiti.

Listing 1. Sekvencijalno izvršavanje Jacobi

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SEQ\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: 0.0245002.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SEQ\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: 12.891.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 2. Zadatak 4 – 1 nit

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 3. Zadatak 4 – 2 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 4. Zadatak 4 – 4 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Listing 5. Zadatak 4 – 8 niti

Opening file:data/small/input/matrix1.txt

Matrix dimension: 128x96

Opening file:data/small/input/matrix2t.txt

Matrix dimension: 160x96

Opening file:result\_small.txt for write.

Matrix dimension: 128x160

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Opening file:data/medium/input/matrix1.txt

Matrix dimension: 1024x992

Opening file:data/medium/input/matrix2t.txt

Matrix dimension: 1056x992

Opening file:result\_medium.txt for write.

Matrix dimension: 1024x1056

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DZ1Z1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Elapsed time: .

Number of threads: 8

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* + 1. Tabela trajanja programa

|  |  |  |
| --- | --- | --- |
|  | **Small input** | **Medium input** |
| **Sekvencijalno izvršavanje** | 0.0245002 | 12.891 |
| **Zadatak 4 – 1 nit** | 0.0256683 | 12.8851 |
| **Zadatak 4 – 2 niti** | 0.0664114 | 7.05922 |
| **Zadatak 4 – 4 niti** | 0.0758083 | 4.08204 |
| **Zadatak 4 – 8 niti** | 0.0695984 | 4.03053 |

* + 1. Grafici ubrzanja

U okviru ove sekcije su dati grafici ubrzanja paralelnih implementacija u odnosu na sekvencijalnu implementaciju. Ubrzanje na svim narednim graficima predstavljeno je kao odnos trajanja sekvencijalne implementacije programa i trajanja paralelizovane implementacije.

Grafik 1. Zavisnost ubrzanja programa dz1z1 od veličine izlazne matrice i broja niti

Grafik 2. Zavisnost ubrzanja programa dz1z2 od veličine izlazne matrice i broja niti

Grafik 3. Zavisnost ubrzanja programa dz1z3 od veličine izlazne matrice i broja niti

* + 1. Diskusija dobijenih rezultata

Dobijene rezultate treba izdiskutovati i objasniti sa nekoliko rečenica u odnosu na dobijena ubrzanja (ili usporenja), način paralelizacije, karakteristike problema i sl.