Univerzitet u Beogradu - Elektrotehnički fakultet

Multiprocesorki sistemi (13S114MUPS, 13E114MUPS)



Domaći zadatak 2 – MPI

Izveštaj o urađenom domaćem zadatku

|  |  |
| --- | --- |
| Predmetni asistent: | Studenti: |
| doc. dr Marko Mišić | Damjan Pavlović 2017/0312  Uroš Ugrinić 2017/0714 |

Beograd, novembar 2020.

Sadržaj

[1. Problem 1 - Pi 3](#_Toc59697321)

[1.1. Tekst problema 3](#_Toc59697322)

[1.1.1. Diskusija 3](#_Toc59697323)

[1.1.2. Način paralelizacije 3](#_Toc59697324)

[1.2. Rezultati 3](#_Toc59697325)

[1.2.1. Logovi izvršavanja 3](#_Toc59697326)

[1.2.2. Grafici ubrzanja 5](#_Toc59697327)

[1.2.3. Diskusija dobijenih rezultata 5](#_Toc59697328)

[2. Problem 2 – Needle - Wunsch 6](#_Toc59697329)

[2.1. Tekst problema 6](#_Toc59697330)

[2.2. Delovi koje treba paralelizovati 6](#_Toc59697331)

[2.2.1. Diskusija 6](#_Toc59697332)

[2.2.2. Način paralelizacije 6](#_Toc59697333)

[2.3. Rezultati 6](#_Toc59697334)

[2.3.1. Logovi izvršavanja 6](#_Toc59697335)

[2.3.2. Grafici ubrzanja 7](#_Toc59697336)

[2.3.3. Diskusija dobijenih rezultata 8](#_Toc59697337)

[3. Problem 5 – N body 9](#_Toc59697338)

[3.1. Tekst problema 9](#_Toc59697339)

[3.2. Delovi koje treba paralelizovati 9](#_Toc59697340)

[3.2.1. Diskusija 9](#_Toc59697341)

[3.2.2. Način paralelizacije 9](#_Toc59697342)

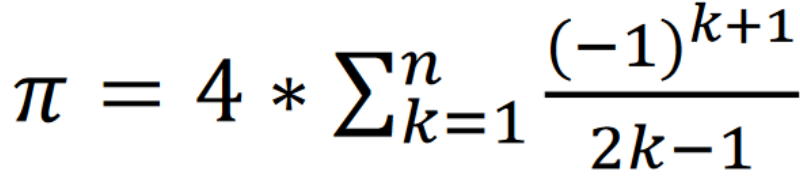
[3.3. Rezultati 10](#_Toc59697343)

[3.3.1. Logovi izvršavanja 10](#_Toc59697344)

[3.3.2. Grafici ubrzanja 11](#_Toc59697345)

[3.3.3. Diskusija dobijenih rezultata 11](#_Toc59697346)

1. Problem 1 - Pi
   1. Tekst problema

Paralelizovati program koji izračunava vrednost broja PI korišćenjem formule: 

Tačnost izračunavanja direktno zavisi od broja iteracija, a zbog malog radijusa konvergencije serija konvergira veoma sporo. Program se nalazi u datoteci piCalc.c u arhivi koja je priložena uz ovaj dokument. Proces sa rangom 0 treba da učita ulazne podatke, raspodeli posao ostalim procesima, na kraju prikupi dobijene rezultate i ravnopravno učestvuje u obradi. Za razmenu podataka, koristiti rutine za kolektivnu komunikaciju. Program testirati sa parametrima koji su dati u datoteci run. [1, N]

* + 1. Diskusija

Posao je podeljen na delove i dodeljen procesima da ih nezavisno rade. Na kraju se sve redukuje u jedno rešenje.

* + 1. Način paralelizacije

Preko master-slave šablona je posao podeljen i zatim preko funkcije MPI\_Reduce sabran u jedinstveno rešenje.

* 1. Rezultati
     1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere.

---------------sa 4 niti----------------

uu170714d@rtidev5:~$ ./test.exe 1000000

---------------Sequential---------------

Elapsed time = 3.188848e-03 seconds

Our estimate of pi = 3.14159165358977

----------------Parallel----------------

Elapsed time = 3.001142e-03 seconds

With n = 1000000 terms

Our estimate of pi = 3.14159165358956

----------------------------------------

TEST PASSED

----------------------------------------

uu170714d@rtidev5:~$ ./test.exe 10000000

---------------Sequential---------------

Elapsed time = 3.184986e-02 seconds

Our estimate of pi = 3.14159255358979

----------------Parallel----------------

Elapsed time = 9.970656e-03 seconds

With n = 10000000 terms

Our estimate of pi = 3.14159255358947

----------------------------------------

TEST PASSED

----------------------------------------

uu170714d@rtidev5:~$ ./test.exe 100000000

---------------Sequential---------------

Elapsed time = 3.192689e-01 seconds

Our estimate of pi = 3.14159264358933

----------------Parallel----------------

Elapsed time = 9.840291e-02 seconds

With n = 100000000 terms

Our estimate of pi = 3.14159264359058

----------------------------------------

TEST PASSED

----------------------------------------

uu170714d@rtidev5:~$ ./test.exe 1000000000

---------------Sequential---------------

Elapsed time = 3.256785e+00 seconds

Our estimate of pi = 3.14159265258805

----------------Parallel----------------

Elapsed time = 1.137900e+00 seconds

With n = 1000000000 terms

Our estimate of pi = 3.14159265259308

----------------------------------------

TEST PASSED

----------------------------------------

* + 1. Grafici ubrzanja

U okviru ove sekcije su dati grafici ubrzanja u odnosu na sekvencijalnu implementaciju.

* + 1. Diskusija dobijenih rezultata

Vidimo da vreme izvršavanja paralelnog programa mnogo sporije raste od sekvencijalnog. Drastična je razlika performansa između njih. Posao je lepo distribuiran između niti ali kada bi jedna brže radila od drugih onda ne bi bilo tako dobro ubrzanje.

1. Problem 2 – Needle - Wunsch

U okviru ovog poglavlja je dat kratak izveštaj u vezi rešenja zadatog problema 4.

* 1. Tekst problema

Paralelizovati program koji vrši poravnavanje bioloških sekvenci korišćenjem Needleman-Wunsch algoritma. Algoritam predstavlja primenu koncepta dinamičkog programiranja za globalno poravnavanje dve sekvence nukleotida ili aminokiselina (više o algoritmu na adresi: https://en.wikipedia.org/wiki/Needleman%E2%80%93Wunsch\_algorithm). Program se nalazi u datoteci needle.c u arhivi koja je priložena uz ovaj dokument. Paralelizaciju pokušati podelom sekvence za poravnavanje na podsekvence. Ukoliko je moguće, koristiti rutine za neblokirajuću komunikaciju za razmenu poruka. Program testirati sa parametrima koji su dati u datoteci run. [1, N]

* 1. Delovi koje treba paralelizovati
     1. Diskusija

Paralelizovano je obilazenje matrice po blokovima

* + 1. Način paralelizacije

Matrica je podeljena u manje podmatrice koje su paralelno obradjivane. Ovo je uradjeno da bi se smanjio broj poruka.

* 1. Rezultati

U okviru ove sekcije su izloženi rezultati paralelizacije problema 5.

* + 1. Logovi izvršavanja

|  |
| --- |
| ---------------Sequential---------------  Start Needleman-Wunsch  Processing whole matrix  ---------------------------------  Elapsed time = 2.056320e+01 seconds  ----------------Parallel----------------  Process 1 Processing whole matrix  Process 2 Processing whole matrix  Process 0 Processing whole matrix  Process 3 Processing whole matrix  Traceback...  ----------------------------------------  Elapsed time = 3.143740e+00 seconds  ----------------------------------------  TEST PASSED  ---------------------------------------- |

Listing 1. Izlaz koda za 16385 N=4

---------------Sequential---------------

Start Needleman-Wunsch

Processing whole matrix

----------------------------------------

Elapsed time = 9.896437e+000 seconds

----------------Parallel----------------

Start Needleman-Wunsch

Process 3 Processing whole matrix

Process 1 Processing whole matrix

Process 0 Processing whole matrix

Process 2 Processing whole matrix

Traceback...

----------------------------------------

Elapsed time = 5.105624e+00 seconds

----------------------------------------

TEST PASSED

----------------------------------------

Listing 2. Izlaz koda za 22528 N=4

* + 1. Grafici ubrzanja

U okviru ove sekcije su dati grafici ubrzanja u odnosu na sekvencijalnu implementaciju.

Za N = 4

Slika 1. Grafik zavisnosti ubrzanja (s) paralelnog algoritma u odnosu na sekvencijalni od veličine matrice

* + 1. Diskusija dobijenih rezultata

Dobijeno je skoro duplo ubrzanje. Ovo bi moglo da se optimizuje štelovanjem veličine blokova i naravno povećanjem broja procesa. Najsporiji delovi programa su slanje poruka, zbog toga su uneti blokovi u rešenje, oni povecavaju poruke i time smanjuju broj poruka.

Problem 3 – N body

U okviru ovog poglavlja je dat kratak izveštaj u vezi rešenja zadatog problema 5.

* 1. Tekst problema

Paralelizovati program kojisimulira problem interakcije čvrstih telau dvodimenzionalnom prostoru(n-body problem). Tela interaguju putem gravitacione silena osnovu sopstvene mase, pozicije u prostoru i trenutne brzine. Program se nalazi u direktorijumu nbodyu arhivi koja je priložena uz ovaj dokument. Program se sastoji od više datoteka, od kojih jeod interesa datotekanbody.c. Analizirati dati kod i obratiti pažnju na način izračunavanja sila i energija.Verifikaciju paralelizovanog rešenja vršiti nad dobijenim energijama i poslednjem stanju sistema.Način pokretanja programa se nalazi u datoteci run. [1, N]

* + 1. Diskusija

Compute\_force() je funkcija za koju se najvise ceka za izvrsavanje tako da je ona paralelizovana. Update\_part() nije paralelizovana jer razmena poruka vise oduzima vremena nego što skraćuje. Compute\_energy() je izbačena iz petlje jer je njegovo izvršavanje samo potrebno na kraju svega ostalog.

* + 1. Način paralelizacije

Compute\_force( ) je paralelizovan preko parcijalnih suma sila.

* 1. Rezultati

U okviru ove sekcije su izloženi rezultati paralelizacije problema 3.

* + 1. Logovi izvršavanja

|  |
| --- |
| ---------------Sequential---------------  PE = -7.035612e+36, KE = 1.304554e+36, Total Energy = -5.731058e+36  Elapsed time = 7.032204e-02 seconds  ----------------Parallel----------------  PE = -7.035612e+36, KE = 1.304554e+36, Total Energy = -5.731058e+36  Elapsed time = 2.275395e-02 seconds  ----------------------------------------  Speed-up = 4.756808e-02 seconds  ----------------------------------------  TEST PASSED  ----------------------------------------  uu170714d@rtidev5:~$ mpirun -n 4 ./nbody.exe 500 500 0.01 500 g  ---------------Sequential---------------  PE = -4.754056e+37, KE = 1.360414e+36, Total Energy = -4.618014e+37  Elapsed time = 3.028810e+00 seconds  ----------------Parallel----------------  PE = -4.754056e+37, KE = 1.360414e+36, Total Energy = -4.618014e+37  Elapsed time = 5.971911e-01 seconds  ----------------------------------------  Speed-up = 2.431619e+00 seconds  ----------------------------------------  TEST PASSED  ----------------------------------------  uu170714d@rtidev5:~$ mpirun -n 4 ./nbody.exe 5000 500 0.01 500 g  ---------------Sequential---------------  PE = -6.649074e+38, KE = 2.481116e+36, Total Energy = -6.624263e+38  Elapsed time = 2.360960e+02 seconds  ----------------Parallel----------------  PE = -6.649074e+38, KE = 2.481116e+36, Total Energy = -6.624263e+38  Elapsed time = 7.094961e+01 seconds  ----------------------------------------  Speed-up = 1.651464e+02 seconds  ----------------------------------------  TEST PASSED  ---------------------------------------- |

* + 1. Grafici ubrzanja

U okviru ove sekcije su dati grafici ubrzanja u odnosu na sekvencijalnu implementaciju.

Slika 1. Grafik zavisnosti ubrzanja (s) paralelnog algoritma u odnosu na sekvencijalni od broja čestica za N = 12 niti (primer)

* + 1. Diskusija dobijenih rezultata

Dobili smo neko ubrzanje paralelizacijom for petlji. Bitno je obratiti pažnju na podelu poslova zbog njihove neravrnopravnosti.