Univerzitet u Beogradu - Elektrotehnički fakultet

Multiprocesorki sistemi (13S114MUPS, 13E114MUPS)



Domaći zadatak 2 – MPI

Izveštaj o urađenom domaćem zadatku

|  |  |
| --- | --- |
| Predmetni saradnici: | Student: |
| doc. dr Marko Mišić  dipl. ing. Pavle Divović | Uros Jovanovic 2018/0412 |

Beograd, decembar 2021.

Sadržaj

[Sadržaj 2](#_Toc529188776)

[1. Problem 1 - SGEMM 3](#_Toc529188777)

[1.1. Tekst problema 3](#_Toc529188778)

[1.2. Delovi koje treba paralelizovati 3](#_Toc529188779)

[1.2.1. Diskusija 3](#_Toc529188780)

[1.2.2. Način paralelizacije 3](#_Toc529188781)

[1.3. Rezultati 3](#_Toc529188782)

[1.3.1. Logovi izvršavanja 3](#_Toc529188783)

[1.3.2. Grafici ubrzanja 4](#_Toc529188784)

1. Problem 1 - SIMPLEX

U okviru ovog poglavlja je dat kratak izveštaj u vezi rešenja zadatog problema 1. Rešenja svih problema dati u okviru **jednog** dokumenta. U zavisnosti od broja postavljenih problema, dodati nova poglavlja u dokument.

* 1. Tekst problema

Paralelizovati program koji računa integral funkcije F na osnovu unutrašnjosti simplex-a u 20 dimenzija korišćenjem Monte Carlo metode. Program se nalazi u datoteci simplex.c. U izvornom kodu data je matrica eksponenata jednačine i ivica simplexa. Ulazni parametar programa je broj iteracija aproksimacije. 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. Delovi koje treba paralelizovati
     1. Diskusija

While petlja u run funkciji ima veliki broj iteracija koje treba da izvrsi ali tacnost naseg rezultata raste sa porastom iteracija, tako da rasporedjivanje iteracija na veci broj procesa kod ove while petlje nema smisla posto svaka iteracija zavisi od prethodne. Ono sto moze da se paralelizuje je sama unutrasnjost while petlje odnosno dva ugnjezdena for-a u kojima se racuna rezultat.

* + 1. Način paralelizacije

Nacin na koji sam ja paralelizovao ova dva fora je tako sto sam svakom procesu odredio za koliko podataka od okupnog broja podataka krajnjeg rezultata je zaduzen. Taj broj podataka za koje je proces zaduzen zavisi od ukupnog broja podataka krajnjeg resenja i broja procesa (chunk = result.size / num\_processes). Master proces ce na kraju da pokupi i u svoj krajnji niz sa rezultatima da stavi rezultate od svakog procesa.

* 1. Rezultati

U okviru ove sekcije su izloženi rezultati paralelizacije problema 1.

* + 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.

Input: 50000

Result:

3.10708e+13 5.02405e+12 4.19575e+12 6.89731e+18 1.83995e+16 5.02776e+12 6.29503e+16 3.49699e+17 1.97214e+13 4.52376e+19 6.69407e+15 1.12567e+11 5.81623e+14 2.25285e+10 8.70572e+13 2.52511e+15 6.89669e+11 1.26806e+12 1.65051e+30 1.58334e+18 4.05003e+18 6.68409e+13 1.83776e+19 1.0274e+11 5.04517e+09 2.83783e+26 4.3338e+17 1.09827e+12 4.17324e+12 2.83983e+13 2.41437e+15 1.11156e+07 1.4798e+08 3.79249e+16 8.27702e+12 4.15981e+26 4.2958e+16 8.66601e+15 1.67664e+22 8.65089e+17

Elapsed time in ms:

1 process: 1083

4 processes: 540

Listing 1. Sekvencijalno izvršavanje SIMPLEX

Input: 100000

Result:

2.45987e+13 5.69933e+12 5.65776e+12 3.53226e+18 1.79875e+16 4.79862e+12 9.05699e+16 3.01167e+17 1.78658e+13 2.30181e+19 6.16165e+15 1.15649e+11 6.60153e+14 1.90618e+10 9.29861e+13 2.84049e+15 7.12649e+11 1.26341e+12 8.18149e+29 1.33903e+18 4.75301e+18 5.19337e+13 1.34611e+19 8.93407e+10 4.2083e+09 3.31353e+26 9.79033e+17 1.36716e+12 3.89285e+12 3.06654e+13 2.71441e+15 1.11992e+07 1.5775e+08 4.77223e+16 7.78478e+12 4.28168e+26 5.37112e+16 1.57385e+16 1.12387e+22 8.18056e+17

Elapsed time in ms:

1 process: 2294

4 processes: 912

Input: 1000000

Result:

3.83659e+13 5.07352e+12 8.70975e+12 4.09657e+18 1.78672e+16 4.47594e+12 8.18366e+16 3.05429e+17 1.67152e+13 2.35681e+19 6.7244e+15 1.10094e+11 6.38012e+14 1.70158e+10 8.80065e+13 2.17889e+15 7.1347e+11 1.08584e+12 1.60841e+30 1.22614e+18 5.41957e+18 5.38603e+13 1.93084e+19 9.26227e+10 4.42344e+09 3.40302e+26 5.76578e+17 7.99091e+11 3.91869e+12 3.09605e+13 2.40957e+15 1.06712e+07 1.49546e+08 4.25394e+16 8.33449e+12 5.22412e+26 6.93429e+16 1.01992e+16 2.09114e+22 6.91833e+17

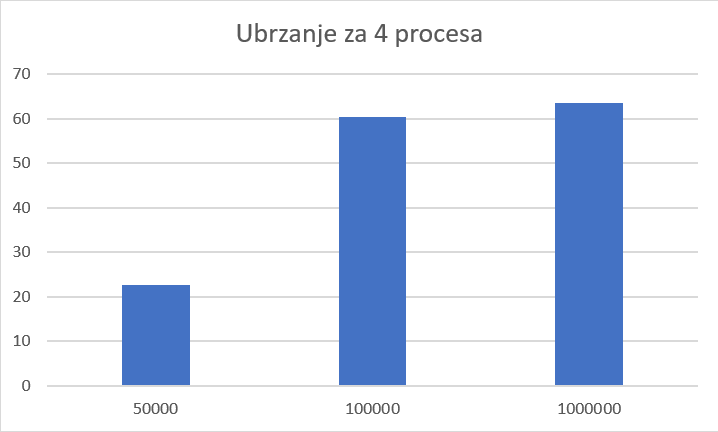
Elapsed time in ms:

1 process: 19840

4 processes: 7733

* + 1. Grafici ubrzanja

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



Slika 1. Grafik zavisnosti ubrzanja simplex algoritma od broja iteracija za N = 4 niti (primer)

* + 1. Diskusija dobijenih rezultata

Iz dobijenih rezultata vidimo da je sa povecanjem iteracija i ubrzanje naseg naseg programa vece.