UNIVERZITET U BEOGRADU - ELEKTROTEHNIČKI FAKULTET

MULTIPROCESORKI SISTEMI (13S114MUPS, 13E114MUPS)

DOMAĆI ZADATAK 2 – MPI

Izveštaj o urađenom domaćem zadatku

Predmetni saradnici:

doc. dr Marko Mišić

dipl. ing. Matija Dodović

Studenti:

Petar Marković 2020/0203

Lana Stamenković 2020/0206

Beograd, maj 2024.

SADRŽAJ

1. PROBLEM 1 – ARITMETIČKI BROJEVI

1.1. Tekst problema

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 **aritmetic.c** u arhivi koja je priložena uz ovaj dokument. Program testirati sa parametrima koji su dati u **run** skripti.

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.

1.2. Delovi koje treba paralelizovati

1.2.1. Diskusija

U priloženom sekvencijalnom kodu za izračunavanje aritmetičkih brojeva uočili smo dve celine koje je potencijalno moguće paralelizovati. Jedna celina je petlja u **main** funkciji programa koja iterira kroz sve prirodne brojeve, proverava da li je dati broj aritmetički i staje tek kad je nađen zadati broj aritmetičkih brojeva. Druga celina u kodu koja bi mogla biti paralelizovana jesu **for** petlje u funkciji **divisor_count_and_sum()** koje traže proste činioce zadatog broja.

1.2.2. Način paralelizacije

Za paralelizaciju je izabrana petlja u main funkciji programa jer se prilikom testiranja i istraživanja mogućih rešenja pokazala kao najpogodniji deo koda za raspodelu po nitima. Ideja za paralelizaciju programa se oslanja na činjenicu da je gustina aritmetičkih brojeva približno ½. Sa sigurnošću možemo da tvrdimo da će broj aritmetičkih brojeva u nekom intervalu biti manji ili jednak broju prirodnih brojeva u tom intervalu, odnosno veličini intervala. Na osnovu toga, u prvom koraku tražimo aritmetičke brojeve u intervalu od 1 do traženog broja aritmetičkih brojeva. Dati interval može jednostavno da se podeli na podintervale koji će biti dodeljeni pojedinačnim nitima na obradu. Kada sve niti završe obradu dodeljenih intervala, sumiramo koliko je ukupno nađeno aritmetičkih brojeva. U narednom koraku, računamo razliku traženog broja i broja pronađenih aritmetičkih brojeva. Dobijena razlika predstavlja veličinu narednog intervala za pretragu. Ovaj postupak se ponavlja do trenutka kada je potrebno naći samo sledeći aritmetički broj, što će veoma brzo moći jedna nit da odradi.

1.2.3. Zadatak 1

Osnovni koncept rešenja jeste da postoji jedna nit zadužena za raspodelu posla i ispis rezultata. Kako bi se postigli što bolji rezultati, pokušano je da se broj naredbi za komunikaciju i količina razmenjenoh podataka između niti svede na minimum. Zbog toga, glavna (master) nit šalje ostalim nitima samo ukupan broj pronađenih aritmetičkih brojeva, na osnovu čega ostale niti pojedinačno mogu da odrede koji deo posla je njihov za obradu. Nakon završetka svog dela posla, niti koriste MPI_Reduce direktivu pomoću koje se u master procesu dobija ukupan broj nađenih aritmetičkih brojeva. Vodi se računa o slučaju gde je broj procesa radnika veći od preostalog broja aritmetičkih brojeva koje treba naći. U tom slučaju, neki procesi ne preuzimaju zadatak traženja aritmetičkih brojeva, a čitav obim preostalog posla je ispravno raspodeljen na druge procese.

1.3. Rezultati – Zadatak 1

U okviru ove sekcije su izloženi rezultati paralelizacije koda za traženje aritmetičkih brojeva pomoću worksharing direktiva.

1.3.1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere i različit broj niti.

```
10001th arithmetic number is 12955
Number of composite arithmetic numbers <= 12955: 8459
```

```
real    0m0.005s
user    0m0.005s
sys     0m0.000s

100001th arithmetic number is 125589
Number of composite arithmetic numbers <= 125589: 88220

real    0m0.014s
user    0m0.014s
sys     0m0.000s

1000001th arithmetic number is 1228665
Number of composite arithmetic numbers <= 1228665: 905044

real    0m0.176s
user    0m0.176s
sys     0m0.000s

10000001th arithmetic number is 12088245
Number of composite arithmetic numbers <= 12088245: 9206548

real    0m4.189s
user    0m4.189s
sys     0m0.004s
```

Listing 1. Sekvencijalno izvršavanje

```
10001th arithmetic number is 12955
Number of composite arithmetic numbers <= 12955: 8459
```

```
real    0m0.299s
user    0m0.025s
sys     0m0.042s

100001th arithmetic number is 125589
Number of composite arithmetic numbers <= 125589: 88220

real    0m0.291s
user    0m0.044s
sys     0m0.033s

1000001th arithmetic number is 1228665
Number of composite arithmetic numbers <= 1228665: 905044

real    0m0.514s
user    0m0.269s
sys     0m0.049s

10000001th arithmetic number is 12088245
Number of composite arithmetic numbers <= 12088245: 9206548

real    0m5.135s
user    0m4.890s
sys     0m0.048s
```

Listing 2. Izvršavanje paralelizovanog koda za zadatak 1 za 1 nit

```
10001th arithmetic number is 12955
Number of composite arithmetic numbers <= 12955: 8459

real    0m0.301s
user    0m0.073s
sys     0m0.067s

100001th arithmetic number is 125589
Number of composite arithmetic numbers <= 125589: 88220

real    0m0.288s
user    0m0.082s
sys     0m0.070s

1000001th arithmetic number is 1228665
Number of composite arithmetic numbers <= 1228665: 905044

real    0m0.417s
user    0m0.305s
sys     0m0.083s

10000001th arithmetic number is 12088245
Number of composite arithmetic numbers <= 12088245: 9206548

real    0m3.243s
user    0m5.998s
sys     0m0.063s
```

Listing 3. Izvršavanje paralelizovanog koda za zadatak 1 za 2 niti

```
10001th arithmetic number is 12955
Number of composite arithmetic numbers <= 12955: 8459

real    0m0.289s
user    0m0.104s
sys     0m0.114s

100001th arithmetic number is 125589
Number of composite arithmetic numbers <= 125589: 88220

real    0m0.296s
user    0m0.104s
sys     0m0.122s

1000001th arithmetic number is 1228665
Number of composite arithmetic numbers <= 1228665: 905044

real    0m0.362s
user    0m0.447s
sys     0m0.127s

10000001th arithmetic number is 12088245
Number of composite arithmetic numbers <= 12088245: 9206548

real    0m1.809s
user    0m6.231s
sys     0m0.166s
```

Listing 4. Izvršavanje paralelizovanog koda za zadatak 1 za 4 niti

```
10001th arithmetic number is 12955
Number of composite arithmetic numbers <= 12955: 8459

real    0m0.333s
user    0m0.106s
sys     0m0.624s

100001th arithmetic number is 125589
Number of composite arithmetic numbers <= 125589: 88220

real    0m0.352s
user    0m0.153s
sys     0m0.836s

1000001th arithmetic number is 1228665
Number of composite arithmetic numbers <= 1228665: 905044

real    0m0.404s
user    0m0.599s
sys     0m0.839s

10000001th arithmetic number is 12088245
Number of composite arithmetic numbers <= 12088245: 9206548

real    0m1.204s
user    0m6.820s
sys     0m1.039s
```

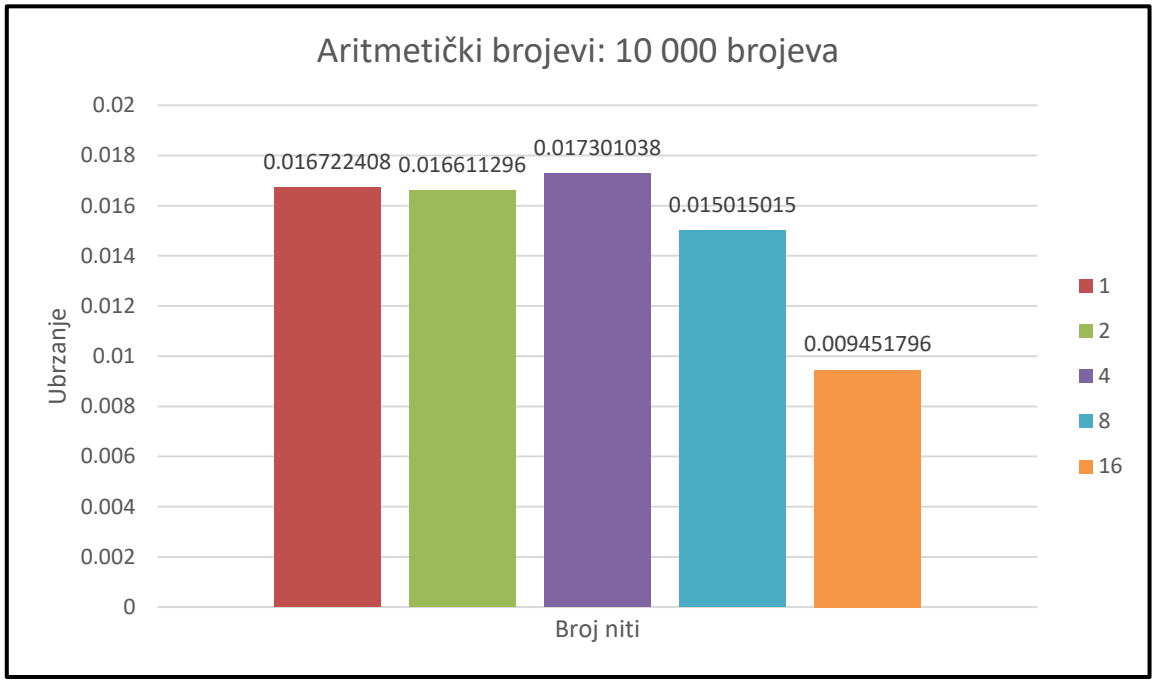
Listing 5. Izvršavanje paralelizovanog koda za zadatak 1 za 8 niti

```
10001th arithmetic number is 12955
Number of composite arithmetic numbers <= 12955: 8459

real    0m0.529s
user    0m0.602s
sys     0m3.388s

100001th arithmetic number is 125589
Number of composite arithmetic numbers <= 125589: 88220

real    0m0.536s
user    0m0.661s
sys     0m3.541s

1000001th arithmetic number is 1228665
Number of composite arithmetic numbers <= 1228665: 905044

real    0m0.564s
user    0m1.148s
sys     0m3.401s

10000001th arithmetic number is 12088245
Number of composite arithmetic numbers <= 12088245: 9206548

real    0m1.251s
user    0m12.231s
sys     0m3.367s
```

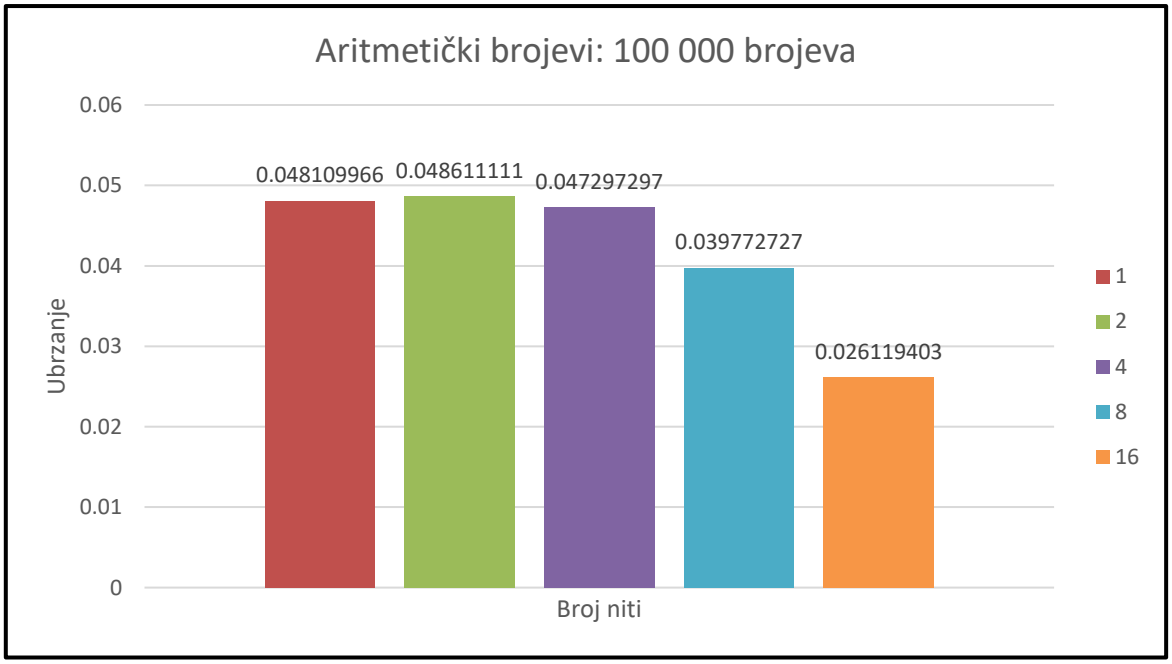
Listing 6. Izvršavanje paralelizovanog koda za zadatak 1 za 16 niti

1.3.2. Grafici ubrzanja

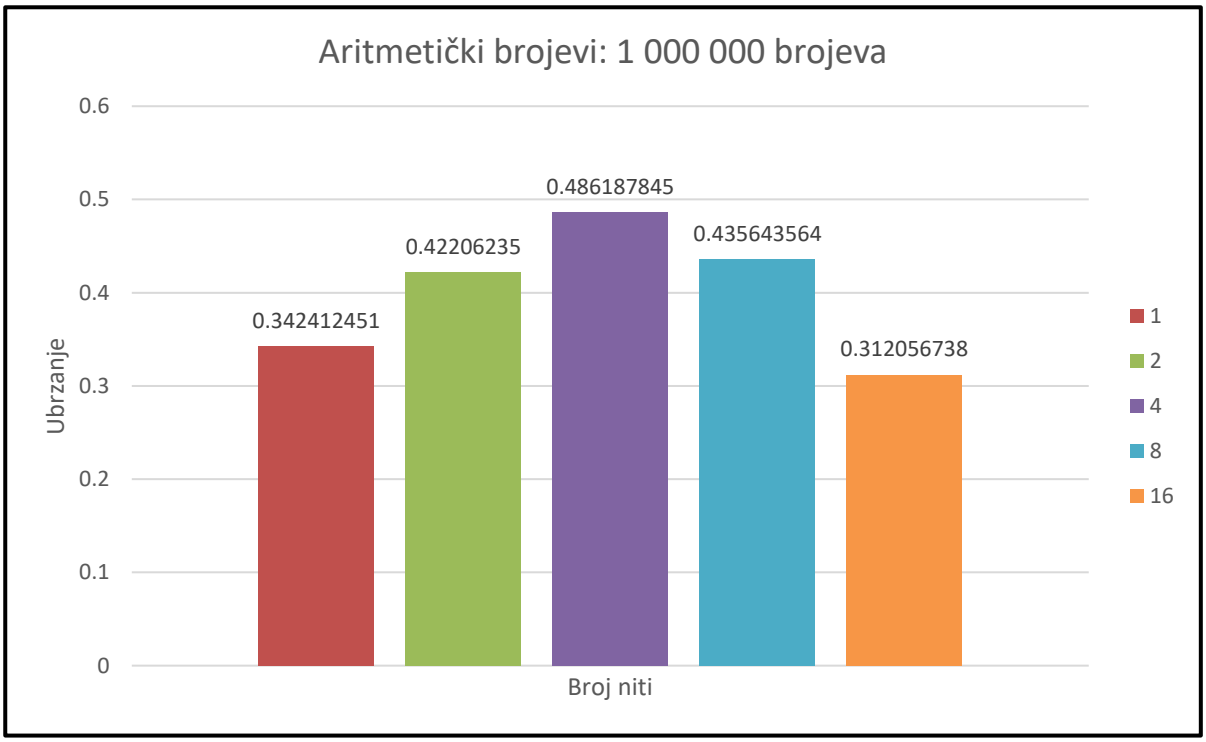
U okviru ove sekcije su dati grafici ubrzanja u odnosu na sekvencijalnu implementaciju.



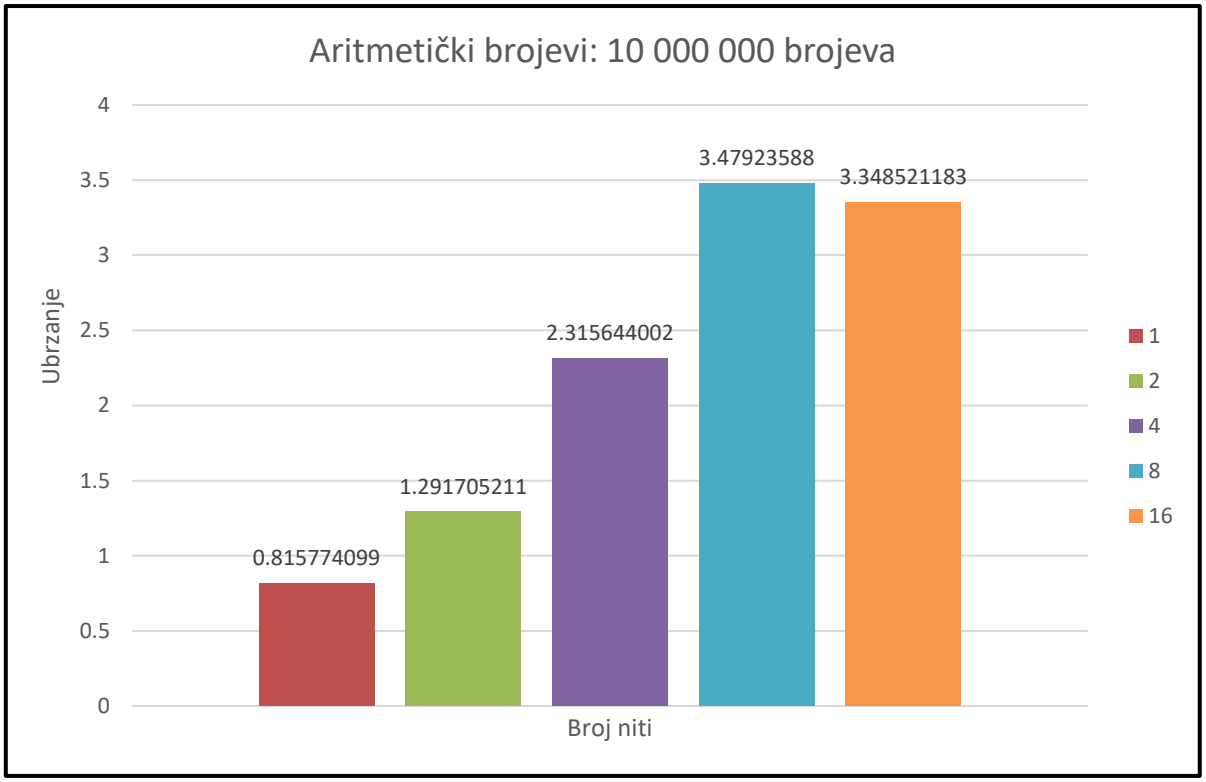
Slika 1. Aritmetički brojevi: 10 000 brojeva



Slika 2. Aritmetički brojevi: 100 000 brojeva



Slika 3. Aritmetički brojevi: 1 000 000 brojeva



Slika 4. Aritmetički brojevi: 10 000 000 brojeva

1.3.3. Diskusija dobijenih rezultata

Ubrzanje je postignuto samo za 10 000 000 aritmetičkih brojeva. Režijski trošak koji unose MPI naredbe za komunikaciju između procesa značajno nadjačava dobit paralelizma za manje veličine ovog problema.

2. PROBLEM 2 – HALTONQMC

2.1. Tekst problema

2.1.1. Zadatak 2

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. Program testirati sa parametrima koji su dati u run skripti.

Ukoliko je moguće, koristiti rutine za neblokirajuću komunikaciju za razmenu poruka.

2.2. Delovi koje treba paralelizovati

2.2.1. Diskusija

HaltonQMC je znatno lakši za paralelizaciju od problema izračunavanja aritmetičkih brojeva, pre svega zbog postojanja pravilnih pravougaonih petlji. Raspodela posla po iteracijama nije jednaka, međutim, u zavisnosti od ulaznih podataka, zahtevnije iteracije se mogu naći i na početku i na kraju petlje.

2.2.2. Zadatak 2 - Rešenje

Problem sračunavanja Halton sekvence se može dekomponovati na potpuno nezavisne celine, zbog čega svaki proces može da odredi svoj obim posla bez komunikacije sa ostalim procesima. Nakon što svaki worker proces završi svoj deo posla, pomoću direktive MPI_Gather za kolektivnu komunikaciju, delovi posla se uklapaju u traženu sekvencu brojeva u master niti i to predstavlja jedini oblik komunikacije i sinhronizacije u programu.

2.3. Rezultati – Zadatak 2

U okviru ove sekcije su izloženi rezultati paralelizacije koda za izračunavanja elemenata Haltonove sekvence ručnom paralelizacijom.

2.3.1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere i različit broj niti.

```
HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
    8:        0.434783        0.391304        0.347826        0.304348         0.26087
    9:        0.344828        0.310345        0.275862        0.241379        0.206897

  Normal end of execution.

  real    0m0.006s
  user    0m0.005s
  sys     0m0.001s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
   98:       0.0191205       0.0172084       0.0152964       0.0133843       0.0114723
   99:       0.0184843       0.0166359       0.0147874        0.012939       0.0110906

  Normal end of execution.

  real    0m0.026s
  user    0m0.023s
  sys     0m0.004s

HALTON_TEST:
```

```
HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
    ...
  998:       0.0012647     0.00113823     0.00101176    0.000885292    0.000758821
  999:      0.00126279     0.00113651     0.00101023     0.00088395    0.000757671

  Normal end of execution.

  real    0m1.995s
  user    0m1.991s
  sys     0m0.004s
```

Listing1. HaltonQMC: Sekvencijalno izvršavanje

```
HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
    8:        0.434783        0.391304        0.347826        0.304348         0.26087
    9:        0.344828        0.310345        0.275862        0.241379        0.206897

  Normal end of execution.

  real    0m0.300s
  user    0m0.038s
  sys     0m0.036s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
```

```
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
   98:       0.0191205       0.0172084       0.0152964       0.0133843       0.0114723
   99:       0.0184843       0.0166359       0.0147874        0.012939       0.0110906

  Normal end of execution.

  real    0m0.317s
  user    0m0.056s
  sys     0m0.067s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
    ...
  998:       0.0012647      0.00113823      0.00101176     0.000885292     0.000758821
  999:      0.00126279      0.00113651      0.00101023      0.00088395     0.000757671

  Normal end of execution.

  real    0m2.047s
  user    0m1.811s
  sys     0m0.043s
```

Listing2. HaltonQMC: Zadatak 2 – Izvršavanje za 1 nit

```
HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
    8:        0.434783        0.391304        0.347826        0.304348         0.26087
    9:        0.344828        0.310345        0.275862        0.241379        0.206897

  Normal end of execution.

  real    0m0.287s
  user    0m0.031s
  sys     0m0.071s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
   98:       0.0191205       0.0172084       0.0152964       0.0133843       0.0114723
   99:       0.0184843       0.0166359       0.0147874        0.012939       0.0110906

  Normal end of execution.

  real    0m0.318s
  user    0m0.116s
  sys     0m0.091s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
```

```
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
    ...
  998:       0.0012647      0.00113823      0.00101176     0.000885292     0.000758821
  999:      0.00126279      0.00113651      0.00101023      0.00088395     0.000757671

  Normal end of execution.

  real    0m1.876s
  user    0m3.245s
  sys     0m0.066s
```

Listing3. HaltonQMC: Zadatak 2 – Izvršavanje za 2 niti

```
HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
    8:        0.434783        0.391304        0.347826        0.304348         0.26087
    9:        0.344828        0.310345        0.275862        0.241379        0.206897

  Normal end of execution.

  real    0m0.286s
  user    0m0.090s
  sys     0m0.106s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
```

```
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
   98:       0.0191205       0.0172084       0.0152964       0.0133843       0.0114723
   99:       0.0184843       0.0166359       0.0147874        0.012939       0.0110906

  Normal end of execution.

  real    0m0.281s
  user    0m0.107s
  sys     0m0.130s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
    ...
  998:       0.0012647      0.00113823      0.00101176     0.000885292     0.000758821
  999:      0.00126279      0.00113651      0.00101023      0.00088395     0.000757671

  Normal end of execution.

  real    0m1.046s
  user    0m3.154s
  sys     0m0.203s
```

Listing4. HaltonQMC: Zadatak 2 – Izvršavanje za 4 niti

```
HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
```

```
...
    8:        0.434783        0.391304        0.347826        0.304348         0.26087
    9:        0.344828        0.310345        0.275862        0.241379        0.206897

  Normal end of execution.

  real    0m0.336s
  user    0m0.115s
  sys     0m0.683s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
   98:       0.0191205       0.0172084       0.0152964       0.0133843       0.0114723
   99:       0.0184843       0.0166359       0.0147874        0.012939       0.0110906

  Normal end of execution.

  real    0m0.354s
  user    0m0.184s
  sys     0m0.885s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
    ...
  998:       0.0012647      0.00113823      0.00101176     0.000885292     0.000758821
  999:      0.00126279      0.00113651      0.00101023      0.00088395     0.000757671
```

```
  Normal end of execution.

  real    0m0.897s
  user    0m4.322s
  sys     0m0.938s
```

Listing5. HaltonQMC: Zadatak 2 – Izvršavanje za 8 niti

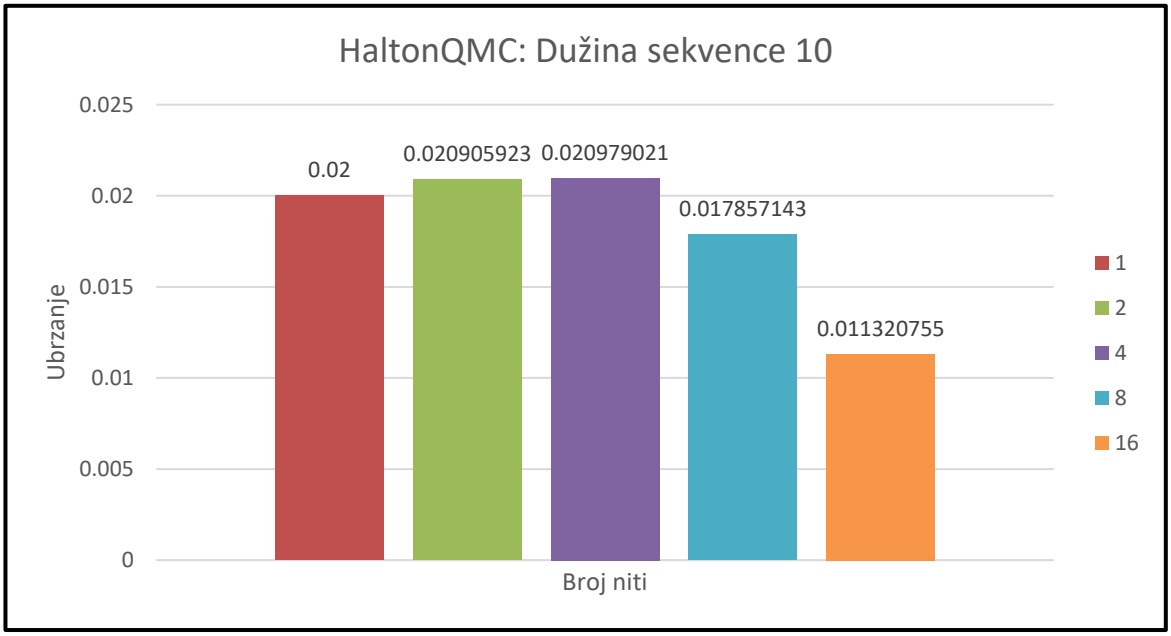
```
HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
    8:        0.434783        0.391304        0.347826        0.304348         0.26087
    9:        0.344828        0.310345        0.275862        0.241379        0.206897

  Normal end of execution.

  real    0m0.530s
  user    0m0.915s
  sys     0m2.933s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
...
   98:       0.0191205       0.0172084       0.0152964       0.0133843       0.0114723
   99:       0.0184843       0.0166359       0.0147874        0.012939       0.0110906

  Normal end of execution.

  real    0m0.552s
```

```
  user    0m0.857s
  sys     0m3.499s

HALTON_TEST:

HALTON_SEQUENCE_TEST
  HALTON_SEQUENCE returns the elements I1 through I2
  of an M-dimensional Halton sequence.

  R:
  Col:          0             1             2             3             4
  Row
    0:          0.3125          0.5625          0.0625          0.875           0.375
    ...
  998:       0.0012647      0.00113823      0.00101176     0.000885292     0.000758821
  999:      0.00126279      0.00113651      0.00101023      0.00088395     0.000757671

  Normal end of execution.

  real    0m1.223s
  user    0m12.004s
  sys     0m3.407s
```

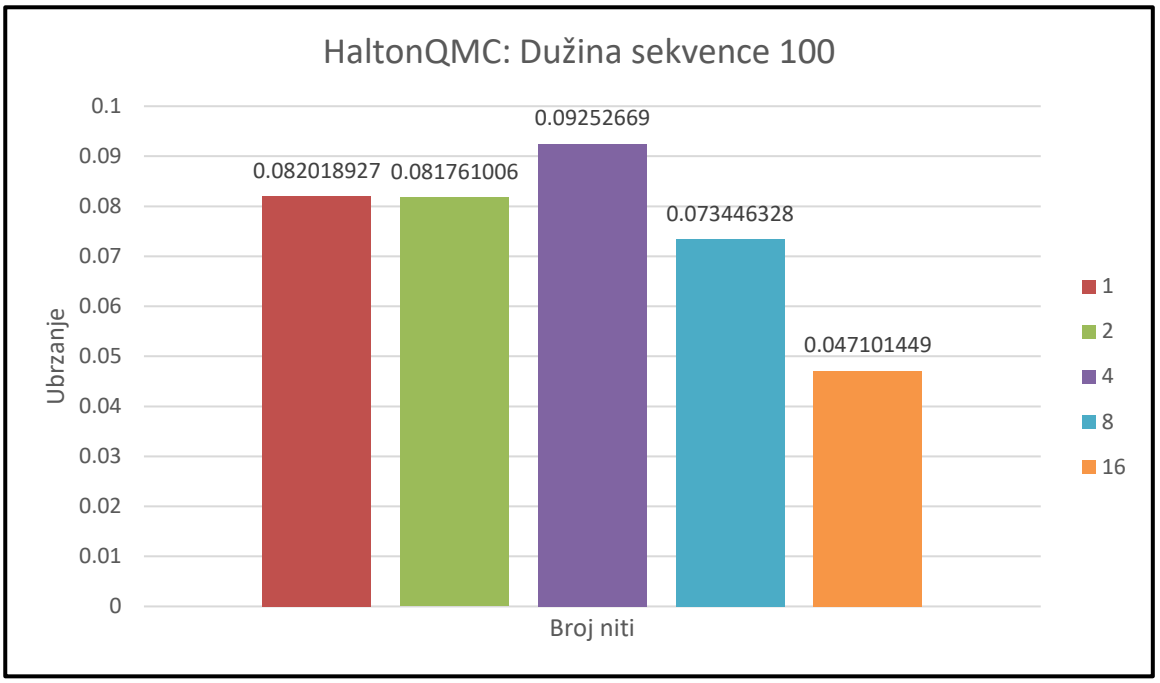
Listing6. HaltonQMC: Zadatak 2 – Izvršavanje za 16 niti

2.3.2. Grafici ubrzanja

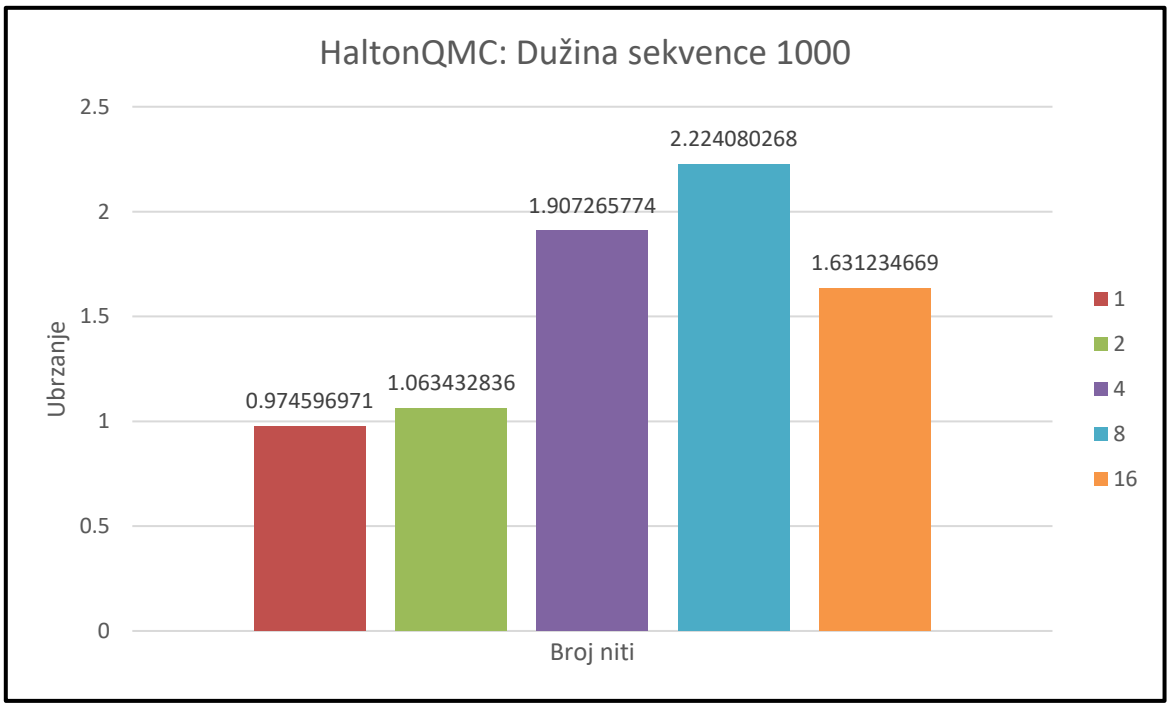
U okviru ove sekcije su dati grafici ubrzanja u odnosu na sekvencijalnu implementaciju.



Slika 1. HaltonQMC: Dužina sekvence 10



Slika 2. HaltonQMC: Dužina sekvence 100



Slika 3. HaltonQMC: Dužina sekvence 1000

2.3.3. Diskusija dobijenih rezultata

Kao i kod prethodnog zadatka, ubrzanje je ostvareno samo za najveće ulazne podatke, usled režijskih troškova koje unose MPI naredbe za komunikaciju. Problem koji se u ovom zadatku javlja, jeste način testiranja rada Halton sekvence. U funkciji halton_sequence_test postoji for petlja koja poziva generisanje sekvence brojeva za sve dužine sekvence počev od 1 pa do unetog ulaznog podatka. Zbog velikog broja generisanja Halton sekvence za manje dužine sekvence, režijski trošak MPI poziva višestruko negativno utiče na ubrzanje paralelne implementacije programa. Jedno od potencijalnih rešenja ovog problema bi bilo da generisanje sekvenci manjih dužina u pomenutoj petlji vrši samo jedna nit, dok bi se u paraleli odradilo poslednje generisanje Halton sekvence.

3. PROBLEM 3 – N BODY PROBLEM

3.1. Tekst problema

3.1.1. Zadatak 3

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 (vx, vy, vz). 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.

Skripta run pokreće simulaciju za različite parametre, i nakon toga, za određene simulacije poziva python kod koji kreira gifove.

3.1.2. Zadatak 4

Prethodni program paralelizovati korišćenjem manager - worker modela. Proces gospodar (master) treba da učita neophodne podatke, generiše poslove, deli posao ostalim procesima i ispiše na kraju dobijeni rezultat. U svakom koraku obrade, proces gospodar šalje procesu radniku na obradu jednu jedinicu posla čiji veličinu treba pažljivo odabrati. Proces radnik prima podatke, vrši obradu, vraća rezultat, signalizira gospodaru kada je spreman da primi sledeći posao i ponavlja opisani postupak dok ne dobije signal da prekine sa radom. Veličinu jedne jedinice posla prilagoditi karakteristikama programa. Ukoliko je moguće, koristiti rutine za neblokirajuću komunikaciju za razmenu poruka.

Skripta run pokreće simulaciju za različite parametre, i nakon toga, za određene simulacije poziva python kod koji kreira gifove.

3.2. Delovi koje treba paralelizovati

3.2.1. Diskusija

N Body Problem u svakoj iteraciji glavne petlje u main funkciji ažurira brzine tela na osnovu njihovog položaja u odnosu na druga tela. Zatim, položaji i brzine se sačuvaju u .csv fajl, nakon čega se ažuriraju položaji tela za narednu iteraciju. Zbog zavisnosti po podacima između iteracija koje je potrebno zabeležiti, nije moguće raspodeliti iteracije simulacije po nitima. Dodatan problem su upisi u fajlove u svakoj iteraciji simulacije, koje nije moguće paralelizovati. Sa druge strane, moguće je raspodeliti ažuriranje brzine i položaja tela između niti, tako da svaka nit dobije svoju grupu tela sa kojom radi.

3.2.2. Zadatak 3

Paralelizam je u ovom zadatku postignut raspodelom ažuriranja brzine i položaja tela između niti, tako da svaka nit dobije svoju grupu tela sa kojom radi. Komunikacija između niti je postignuta upotrebom naredbi za kolektivnu komunikaciju. Nakon što svaka nit izvrši ažuriranje brzine ili položaja grupe tela sa kojom radi, tada bi se rezultati razmenili između svih niti pomoću naredbe MPI_Allgather. Da bi se ispravno odredile brzine i položaji svih tela u simulaciji, potrebno je odrediti uticaj svih drugih na posmatrano telo, pa je zato neophodno razmeniti ažurne podatke između svih niti nakon svakog proračuna.

3.2.3. Zadatak 4 – manager – worker

Umesto upotrebe naredbi za kolektivnu komunikaciju kao u prethodnom zadatku, ovde je iskorišćena implementacija u kojoj postoji glavna nit koja služi isključivo za raspodelu posla, prikupljanje i slanje ažurnih podataka ostalim nitima i za čuvanje rezultata. Zbog toga što Master nit ne učestvuje u proračunima, nije moguće pokrenuti ovaj program samo za jednu nit.

3.3. Rezultati – Zadatak 3

U okviru ove sekcije su izloženi rezultati paralelizacije koda za izračunavanja problema n tela.

3.3.1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere i različit broj niti.

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.018s
user    0m0.013s
sys     0m0.005s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.086s
user    0m0.057s
sys     0m0.029s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m2.409s
user    0m2.380s
sys     0m0.029s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m24.677s
user    0m24.208s
sys     0m0.445s
```

Listing 1. NBodyMini: Zadatak 3– Sekvencijalno izvršavanje

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.348s
user    0m0.049s
sys     0m0.069s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.394s
```

```
user    0m0.058s
sys     0m0.107s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m2.750s
user    0m2.395s
sys     0m0.127s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m24.784s
user    0m24.025s
sys     0m0.491s
```

Listing 2. NBodyMini: Zadatak 3 – Izvršavanje za 1 nit

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.307s
user    0m0.076s
sys     0m0.085s

Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.338s
user    0m0.120s
sys     0m0.109s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m1.856s
user    0m3.135s
sys     0m0.109s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m14.457s
user    0m27.890s
sys     0m0.538s
```

Listing 3. NBodyMini: Zadatak 3 – Izvršavanje za 2 niti

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.320s
user    0m0.111s
sys     0m0.151s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.353s
user    0m0.301s
sys     0m0.120s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m1.295s
user    0m4.043s
sys     0m0.146s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m9.978s
user    0m38.318s
sys     0m0.631s
```

Listing 4. NBodyMini: Zadatak 3 – Izvršavanje za 4 niti

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.359s
user    0m0.241s
sys     0m0.591s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.439s
user    0m0.730s
sys     0m0.773s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m1.092s
```

```
user    0m6.040s
sys     0m0.795s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m8.027s
user    1m0.990s
sys     0m1.288s
```

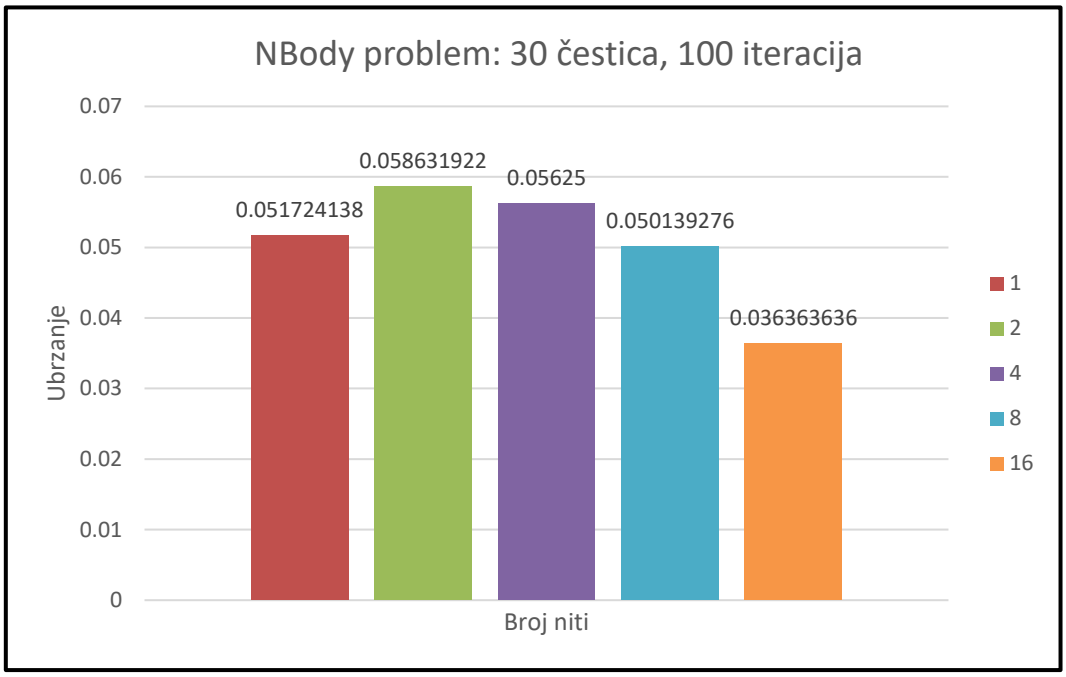
Listing 5. NBodyMini: Zadatak 3 – Izvršavanje za 8 niti

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.495s
user    0m0.740s
sys     0m2.557s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.641s
user    0m2.917s
sys     0m2.784s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m1.613s
user    0m18.252s
sys     0m2.728s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m14.283s
user    3m36.527s
sys     0m3.571s
```

Listing 6. NBodyMini: Zadatak 3 – Izvršavanje za 16 niti

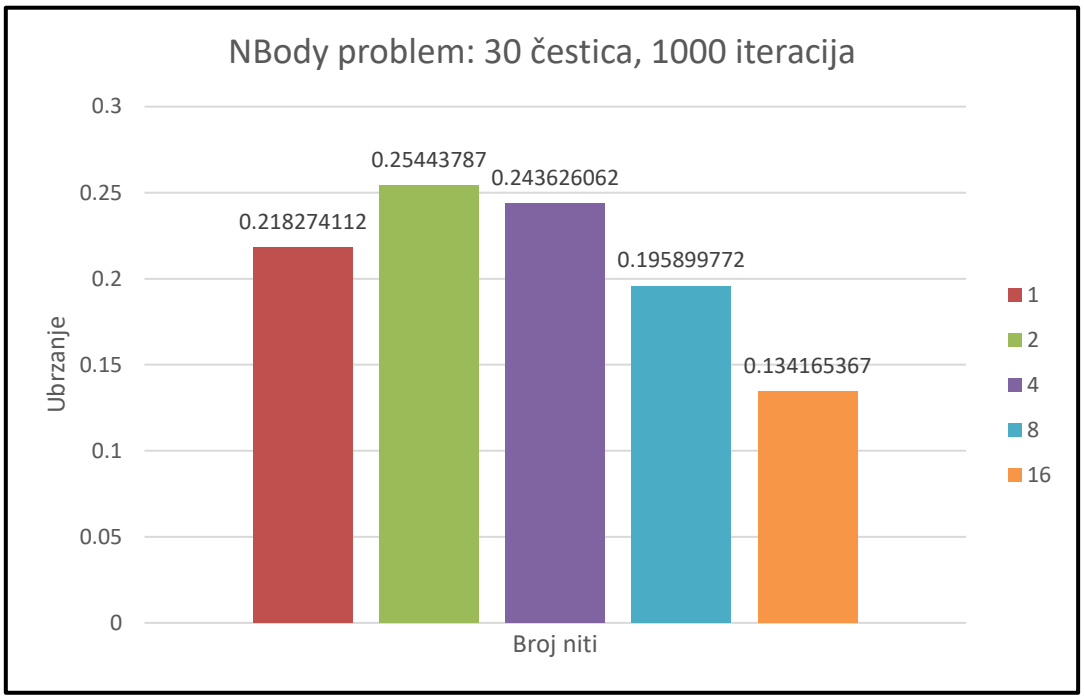
3.3.2. Grafici ubrzanja

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

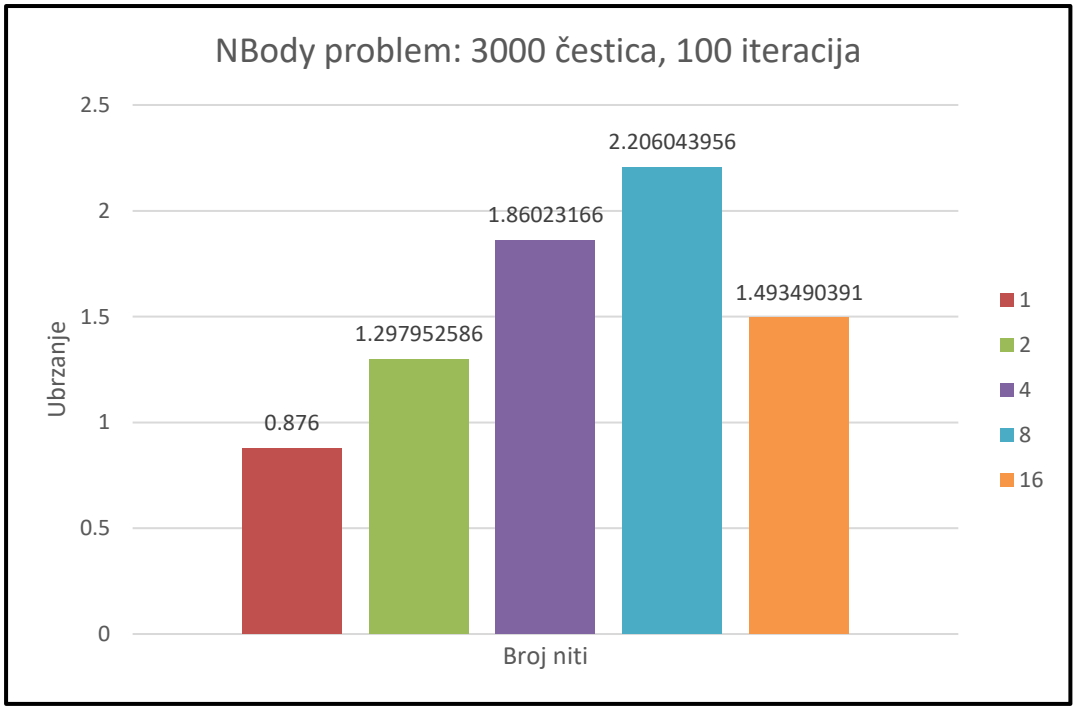


.

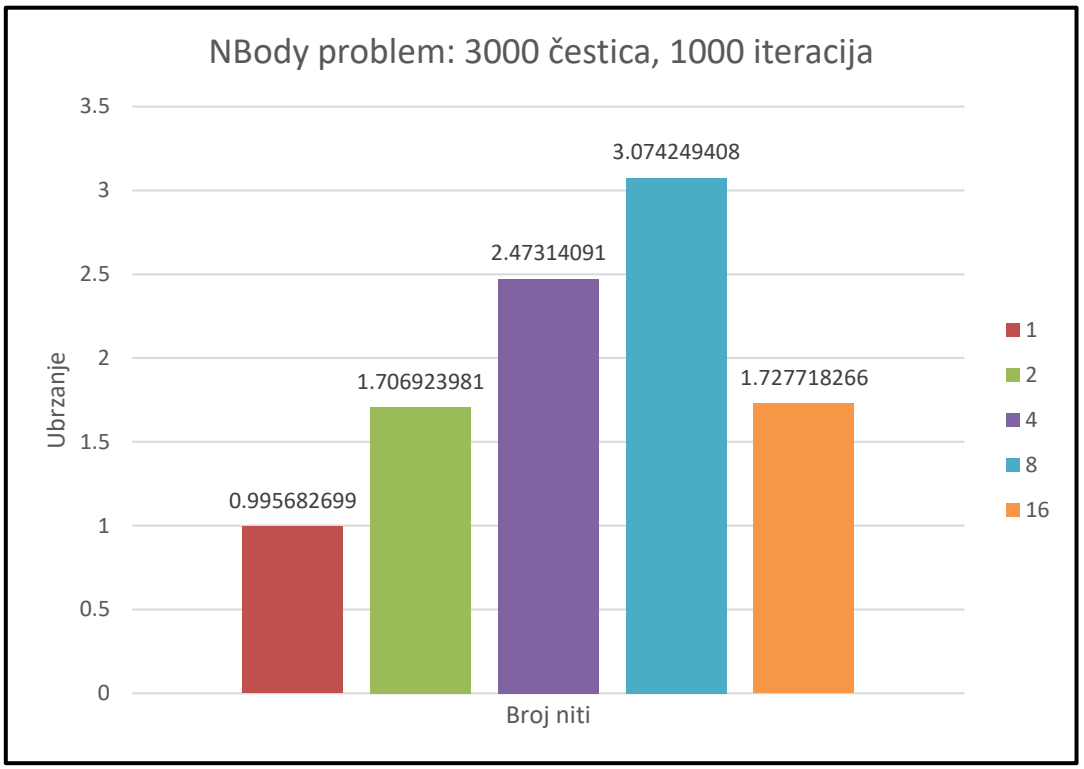
Slika 1. Nbody problem: 30 čestica, 100 iteracija



Slika 2. Nbody problem: 30 čestica, 1000 iteracija



Slika 3. Nbody problem: 3000 čestica, 100 iteracija



Slika 4. Nbody problem: 3000 čestica, 1000 iteracija

3.3.3. Diskusija dobijenih rezultata

Za razliku od OpenMP rešenja ovog problema, gde je na ubrzanje pozitivno uticalo isključivo povećanje broja tela, kod MPI implementacije možemo primetiti veće ubrzanje i za povećan broj iteracija. Glavni razlog tome jeste što kod MPI implementacije ne moraju sve niti da čekaju na kraj zapisivanja rezultata u fajlove, već dok glavna nit vrši upis, ostale niti vrše računanja. Kod OpenMP verzije rešenja, povećanje iteracija je povećavalo sekvencijalni deo koda, dok kod MPI varijante to ima posledicu samo na povećanje obima posla samo za jednu, glavnu, nit.

3.4. Rezultati – Zadatak 4

U okviru ove sekcije su izloženi rezultati paralelizacije koda za izračunavanja problema n tela.

3.4.1. Logovi izvršavanja

Ovde su dati logovi izvršavanja za definisane test primere i različit broj niti.

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.018s
user    0m0.013s
sys     0m0.005s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.086s
user    0m0.057s
sys     0m0.029s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m2.409s
user    0m2.380s
sys     0m0.029s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m24.677s
user    0m24.208s
sys     0m0.445s
```

Listing 1. NBodyMini: Zadatak 4 – Sekvencijalno izvršavanje

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.309s
user    0m0.059s
sys     0m0.107s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.383s
```

```
user    0m0.122s
sys     0m0.171s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m2.951s
user    0m5.315s
sys     0m0.137s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m25.730s
user    0m50.443s
sys     0m0.537s
```

Listing 2. NBodyMini: Zadatak 4 – Izvršavanje za 2 niti

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.298s
user    0m0.122s
sys     0m0.158s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.394s
user    0m0.384s
sys     0m0.171s

Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m1.535s
user    0m4.981s
sys     0m0.163s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m12.174s
user    0m47.236s
sys     0m0.547s
```

Listing 3. NBodyMini: Zadatak 4 – Izvršavanje za 4 niti

```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.358s
user    0m0.166s
sys     0m0.714s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.425s
user    0m0.751s
sys     0m0.656s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder

real    0m1.129s
user    0m6.308s
sys     0m0.694s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m8.510s
user    1m4.833s
sys     0m1.263s
```

Listing 4. NBodyMini: Zadatak 4 – Izvršavanje za 8 niti

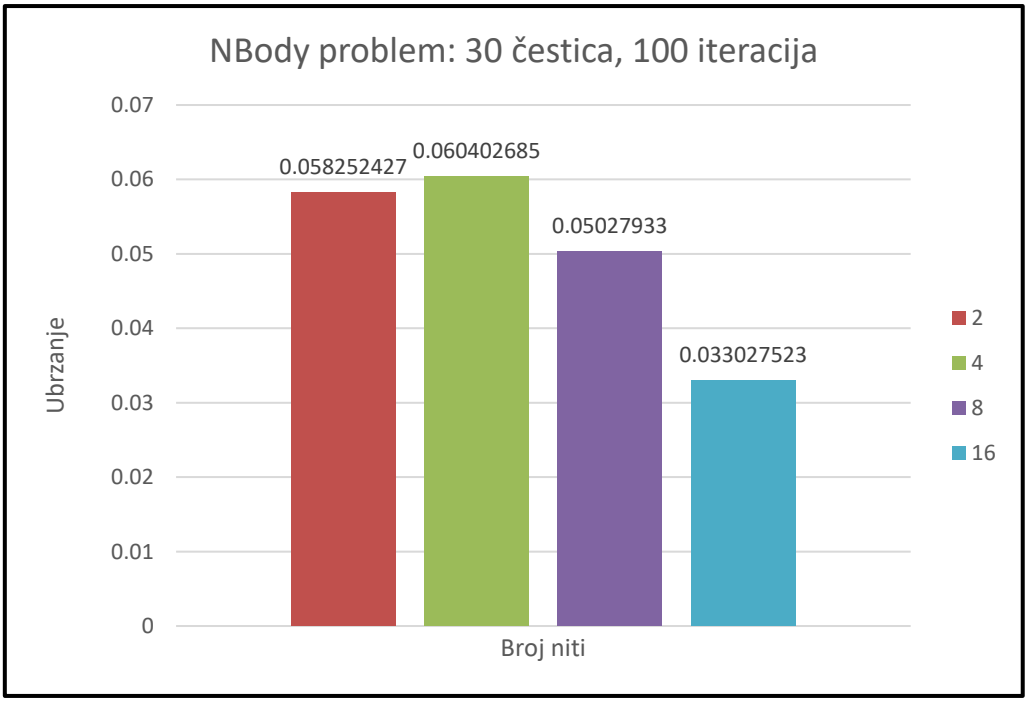
```
Running ./nbody with 30 particles, 100 iterations, and saving to simulation_1 folder

real    0m0.545s
user    0m1.526s
sys     0m2.806s


Running ./nbody with 30 particles, 1000 iterations, and saving to simulation_2 folder

real    0m0.669s
user    0m3.449s
sys     0m2.655s


Running ./nbody with 3000 particles, 100 iterations, and saving to simulation_3 folder
```

```
real    0m1.634s
user    0m18.493s
sys     0m2.861s


Running ./nbody with 3000 particles, 1000 iterations, and saving to simulation_4 folder

real    0m14.734s
user    3m43.637s
sys     0m3.710s
```

Listing 5. NBodyMini: Zadatak 4 – Izvršavanje za 16 niti

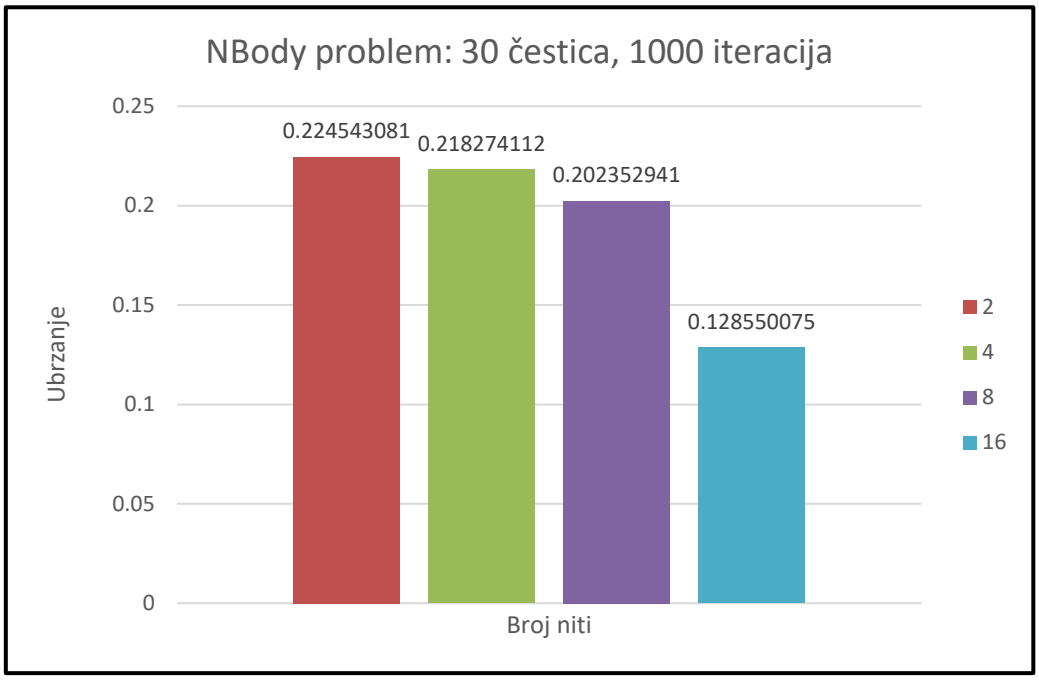
3.4.2. Grafici ubrzanja

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

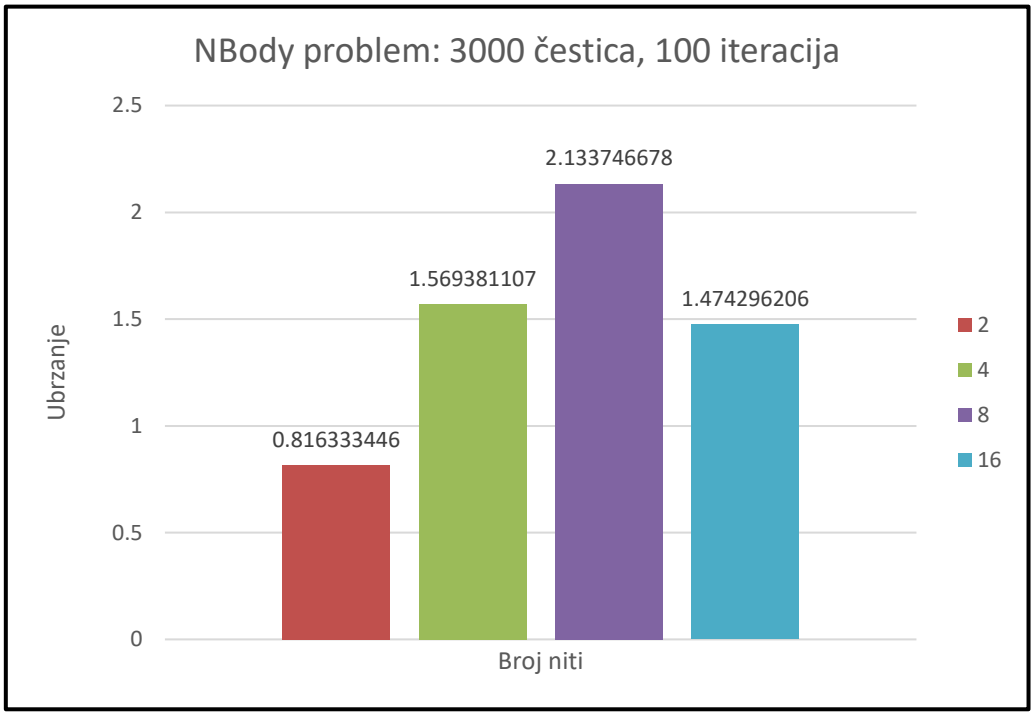


.

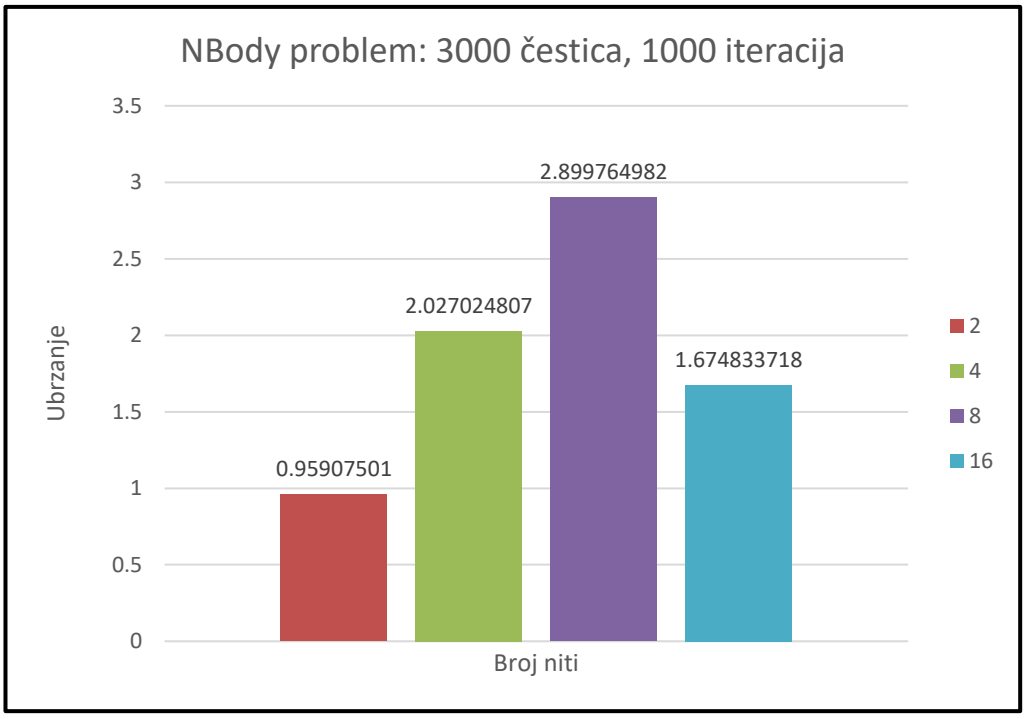
Slika 1. Nbody problem: 30 čestica, 100 iteracija



Slika 2. Nbody problem: 30 čestica, 1000 iteracija



Slika 3. Nbody problem: 3000 čestica, 100 iteracija



Slika 4. Nbody problem: 3000 čestica, 1000 iteracija

3.4.3. Diskusija dobijenih rezultata

Iz istih razloga kao kod implementacije sa kolektivnom komunikacijom, kod master-worker modela možemo zapaziti veće ubrzanje i prilikom povećanja broja iteracija i prilikom povećanja broja tela. Međutim, ukupno ostvareno ubrzanje je manje nego kod prethodne implementacije, zbog većeg broja naredbi za komunikaciju i zbog veće potrebe za sinhronizacijom između procesa.