Univerzitet u Beogradu - Elektrotehnički fakultet

Multiprocesorki sistemi (13S114MUPS, 13E114MUPS)



Domaći zadatak 4 – CUDA

Izveštaj o urađenom domaćem zadatku

|  |  |
| --- | --- |
| Predmetni asistent: | Studenti: |
| doc. dr Marko Mišić | Lazar Cvetković 2016/0127  Strahinja Stefanović 2016/0130 |

Beograd, maj 2020.

Sadržaj

[Sadržaj 2](#_Toc41316004)

[1. Problem 1 – Julia Set 3](#_Toc41316005)

[1.1. Tekst problema 3](#_Toc41316006)

[1.2. Delovi koje treba paralelizovati 3](#_Toc41316007)

[1.2.1. Diskusija 3](#_Toc41316008)

[1.2.2. Način paralelizacije 3](#_Toc41316009)

[1.3. Rezultati 3](#_Toc41316010)

[1.3.1. Logovi izvršavanja 3](#_Toc41316011)

[1.3.2. Grafici ubrzanja 5](#_Toc41316012)

[1.3.3. Diskusija dobijenih rezultata 6](#_Toc41316013)

[2. Problem 2 – Izoštravanje slike 7](#_Toc41316014)

[2.1. Tekst problema 7](#_Toc41316015)

[2.2. Delovi koje treba paralelizovati 7](#_Toc41316016)

[2.2.1. Diskusija 7](#_Toc41316017)

[2.2.2. Način paralelizacije 7](#_Toc41316018)

[2.3. Rezultati 7](#_Toc41316019)

[2.3.1. Logovi izvršavanja 7](#_Toc41316020)

[2.3.2. Grafici ubrzanja 9](#_Toc41316021)

[2.3.3. Diskusija dobijenih rezultata 9](#_Toc41316022)

[3. Problem 3 – MRI Gridding 10](#_Toc41316023)

[3.1. Tekst problema 10](#_Toc41316024)

[3.2. Delovi koje treba paralelizovati 10](#_Toc41316025)

[3.2.1. Diskusija 10](#_Toc41316026)

[3.2.2. Način paralelizacije 10](#_Toc41316027)

[3.3. Rezultati 10](#_Toc41316028)

[3.3.1. Logovi izvršavanja 10](#_Toc41316029)

[3.3.2. Grafici ubrzanja 11](#_Toc41316030)

[3.3.3. Diskusija dobijenih rezultata 11](#_Toc41316031)

1. Problem 1 – Julia Set
   1. Tekst problema

Paralelizovati program koji formira sliku tačaka koje pripadaju Julia skupu tačaka (https://en.wikipedia.org/wiki/Julia\_set). Neka se posmatra skup tačaka (x, y) u na pravougaonom domenu x, y ∈ [-1,5, 1.5] i neka važi z = x+yi. Julia skup je skup tačaka za koji iteracija z = z2 + c ne divergira za određene zadate početne uslove. U zadatom programu početni uslov odgovara c=- 0.8+0.156i. Ukoliko u bilo kom trenutku važi 1000 < |z|, smatra se da tačka z ne pripada Julia skupu. Program formira sliku u Targa (.tga) formatu koja se može otvoriti u nekom od namenskih pregledača slika. Program se nalazi u datoteci julia.c u arhivi koja je priložena uz ovaj dokument, dok se primeri izlaznih datoteka nalaze u direktorijumu output. Program testirati sa parametrima koji su dati u datoteci run.

* 1. Delovi koje treba paralelizovati
     1. Diskusija

Način paralelizacije je očigledan iz postavke problema i svodi se na podelu posla tako da svaka nit će biti zadužena da izračuna tri elementa izlaznog “rgb“ niza. Niti su grupisane u blokove, a to sve u grid. Blok veličine 16x16 je najbrže rezultate od ostalih kombinacija. Iteracije su nezavisne, pa je semantika očuvana.

* + 1. Način paralelizacije

Paralelizacija je izvršena jednostavko, alociranjem “rgb” niza na uređaju, podelom posla tako da svaki blok dobije 256 niti i onda pokretanjem kernela.

* 1. Rezultati
     1. Logovi izvršavanja

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

W: 500, H: 500, CNT: 200

Sequential execution time: 0.036501

Parallel execution time: 0.000202

Speedup: 180.655089

Test PASSED

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

W: 500, H: 500, CNT: 500

Sequential execution time: 0.046069

Parallel execution time: 0.000340

Speedup: 135.471563

Test PASSED

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

W: 500, H: 500, CNT: 1000

Sequential execution time: 0.049374

Parallel execution time: 0.000436

Speedup: 113.359599

Test PASSED

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

W: 1000, H: 1000, CNT: 200

Sequential execution time: 0.147453

Parallel execution time: 0.000503

Speedup: 293.366427

Test PASSED

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

W: 1000, H: 1000, CNT: 500

Sequential execution time: 0.181428

Parallel execution time: 0.000883

Speedup: 205.369113

Test PASSED

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

W: 1000, H: 1000, CNT: 1000

Sequential execution time: 0.188933

Parallel execution time: 0.001166

Speedup: 161.997367

Test PASSED

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

W: 1000, H: 2000, CNT: 200

Sequential execution time: 0.294411

Parallel execution time: 0.000863

Speedup: 341.208423

Test PASSED

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

W: 1000, H: 2000, CNT: 500

Sequential execution time: 0.365044

Parallel execution time: 0.001570

Speedup: 232.438675

Test PASSED

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

W: 1000, H: 2000, CNT: 1000

Sequential execution time: 0.379686

Parallel execution time: 0.002020

Speedup: 187.936572

Test PASSED

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

Listing 1. Izlaz programa

* + 1. Grafici ubrzanja

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

Slika 1. Grafik zavisnosti ubrzanja

* + 1. Diskusija dobijenih rezultata

Hardver GPU uređaja je dao ogromno ubrzanje, neporedivo sa OpenMP i MPI bibliotekama, tako da je ovaj problem daje vremensko ubrzanje koje je u rangu od 196 do 621 puta na osnovu rezultata merenja.

1. Problem 2 – Izoštravanje slike

U okviru ovog poglavlja je dat kratak izveštaj u vezi rešenja zadatog problema 3.

* 1. Tekst problema

Paralelizovati program koji izoštrava zadatu sliku u Portable Graymap Format (PGM) formatu. PGM format se može otvoriti u nekom od namenskih pregledača slika ili online na adresi http://paulcuth.me.uk/netpbm-viewer/. Program se nalazi u direktorijumu sharpen u arhivi koja je priložena uz ovaj dokument. Program se sastoji od više datoteka, od kojih su od interesa datoteke sharpen.c, dosharpen.c i filter.c. Obratiti pažnju na mogućnost korišćenja deljene memorije radi ubrzavanja pristupa podacima. Koristiti 2D organizaciju jezgra. Program testirati sa parametrima koji su dati u datoteci run.

* 1. Delovi koje treba paralelizovati
     1. Diskusija

Paralelizacija je izvršena na nivou pojedinačne tačke slike, tj. na nivou dve petlje sa brojačima “i“ i “j“ funkcije doSharpen. Sekvencijalni kod je na nekim mestima bio redundantan (convolution i convolutionPartial matrice su nepotrebne), te su nepotrebne stvari eliminisane.

* + 1. Način paralelizacije

Pri računanju puno vremena se trošilo na aritmetičkim operacijama nad filterMatrix, te je stoga ona implementirana kao LUT i smeštena u konstantnu memoriju (koja nije dala mnogo ubrzanja u našem slučaju). Eliminisanje matrice convolution je dalo još 20x ubrzanja.

* 1. Rezultati
     1. Logovi izvršavanja

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

Input file: data\_dz4z2/balloons\_noisy.pgm

Sequential execution time: 2.733503

Parallel execution time: 0.033964

Speedup: 80.483124

Test PASSED

Input file: data\_dz4z2/bone\_scint.pgm

Sequential execution time: 18.658480

Parallel execution time: 0.190916

Speedup: 97.731555

Test PASSED

Input file: data\_dz4z2/fuzzy.pgm

Sequential execution time: 3.993764

Parallel execution time: 0.047173

Speedup: 84.661460

Test PASSED

Input file: data\_dz4z2/lena512.pgm

Sequential execution time: 2.297920

Parallel execution time: 0.025296

Speedup: 90.842962

Test PASSED

Input file: data\_dz4z2/man.pgm

Sequential execution time: 9.213062

Parallel execution time: 0.099427

Speedup: 92.661204

Test PASSED

Input file: data\_dz4z2/Rainier\_blur.pgm

Sequential execution time: 18.552892

Parallel execution time: 0.185407

Speedup: 100.065920

Test PASSED

Listing 3. Ispis programa

* + 1. Grafici ubrzanja

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

Slika 3. Grafik zavisnosti ubrzanja

* + 1. Diskusija dobijenih rezultata

Primetno je značajno ubrzanje izvršavanja, u proseku do 100 puta.

1. Problem 3 – MRI Gridding

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

* 1. Tekst problema

Paralelizovati program koji vrši mapiranje neuniformnih podataka u 3D prostoru na regularnu mrežu u 3D prostoru. Svaka tačka iz neuniformnog 3D prostora doprinosi susednim tačkama u regularnoj mreži u skladu sa Kaiser-Bessel funkcijom za određivanje rastojanja. Program se nalazi u direktorijumu mri-gridding u arhivi koja je priložena uz ovaj dokument. Program se sastoji od više datoteka, od kojih su od interesa datoteke main.c i CPU\_kernels.c. Analizirati dati kod i obratiti pažnju na način generisanja vrednosti tačaka u regularnoj mreži. Obratiti pažnju na efikasnost paralelizacije, mogućnost upotrebe deljene memorije i potrebu za redukcijom. Ulazni test primeri se nalaze u direktorijumu data. Verifikaciju paralelizovanog rešenja vršiti nad nizovima gridData i sampleDensity iz glavnog programa. Način pokretanja programa se nalazi u datoteci run.

* 1. Delovi koje treba paralelizovati
     1. Diskusija

Paralelizacija je izvršena na nivou spoljne petlje funkcije gridding\_Gold tako da blok ima 1024 niti, a taj broj je dobijen na empirijski.

* + 1. Način paralelizacije

Poslovi su podeljeni na gore navedeni način, a kako različite niti pristupaju istim lokacijama nad nizovima gridData i sampleDensity, za sinhronizaciju je korišćena primitiva atomicAdd.

* 1. Rezultati
     1. Logovi izvršavanja

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

Reading parameters

Number of samples = 2655910

Grid Size = 256x256x256

Input Matrix Size = 60x60x60

Recon Matrix Size = 60x60x60

Kernel Width = 5.000000

KMax = 150.00 150.00 150.00

Oversampling = 5.000000

GPU Binsize = 32

Use LUT = Yes

Reading input data from files

Generating Look-Up Table

Sequential execution time: 2.615957

Parallel execution time: 0.083466

Speedup: 31.341607

TEST PASSED - gridData

TEST PASSED - sampleDensity

Listing 4. Ispis programa

* + 1. Grafici ubrzanja

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

Slika 5. Grafik zavisnosti ubrzanja

* + 1. Diskusija dobijenih rezultata

Primetno je osetno vremensko ubrzanje od 30-ak puta.