Dokumentacija za projektni zadatak

Tema: Izračunavanje integral upotrebom TBB zadataka

Predmet: Paralelno programiranje

Profesor: prof dr. Miroslav Popović

Asistent: Stefan Pijetlović

Kandidat: Bojan Vlasonjić

Broj indeksa: SW76-2016

Sadržaj

[Uvod 2](#_Toc516480926)

[Analiza problema 3](#_Toc516480927)

[Koncept rešenja 3](#_Toc516480928)

[Programsko rešenje 3](#_Toc516480929)

[Glavni modul: 4](#_Toc516480930)

[Serijska verzija 4](#_Toc516480931)

[Funkcija za izračunavanje integrala kroz for petlju: 4](#_Toc516480932)

[Funkcija za rekurzivno izračunavanje integrala: 4](#_Toc516480933)

[Paralelna verzija: 4](#_Toc516480934)

[Funkcija za pravljenje zadatka: 5](#_Toc516480935)

[Preklopljena virtuelna metoda za sam račun: 5](#_Toc516480936)

[Ispitivanje 5](#_Toc516480937)

[Analiza rezultata 6](#_Toc516480938)

[Zaključak 9](#_Toc516480939)

# Uvod

Projektni zadatak se zasniva naizračunavanju integrala na intervalu (0, 2π), gde je veličina intervala definisana u zavisnosti od testnog slučaja. Funkcija nad kojom se računa integral je:

∫exp(-0.5\*x) \* cos(x) dx .

Paralelizacija programa se postiže upotrebom TBB biblioteke.U početku se formira jedan zadatak kom se prosledi dužina celog intervala. Ako je veličina intervala veća od neke zadate vrednosti (0.01) onda se formiraju dva zadatka koja polove interval i sabiraju svoje rezultate.

# Analiza problema

Serijska verzija programa ne predstavlja neki naročit problem s obzirom da se izračunavanje vrši po formuli:

\*cos(x)dx .

Usko grlo u ovom projektu predstavlja paralelna implementacija. Glavni problem je osmisliti na koji način ćemo formirati i podeliti zadatke kako bismo postigli ubrzanje, a da pri tome i izračunavanja budu relativno precizna. Problem bi trebalo izdeliti na dovoljno malih zadataka kako bi raspoređivač korektno dodelio zadatak niti u cilju uravnoteženja opterećenja.

# Koncept rešenja

Serijska verzija izračuvanja se svodi na primenu formule prolaskom kroz for petlju od indeksa 0 do 2\*3.14 na zadatom intervalu. Rezultat je inicijalizovan na 0 i kroz svaku iteraciju mu uvećavamo vrednost.

Paralelna verzija ima određeni „cutoff“. U prevodu ako je veličina intervala veća od neke zadate vrednosti formiraju se zadaci, u suprotnom je isplativije izračunavati rezultat serijski nad celim intervalom. Ovaj koncept je sličan primeru za izračunavanje n Fibonačijevih brojeva upotrebom TBB zadataka. Po uzoru na taj primer je implementirana paralelna verzija programa.U tu svrhu je napravljena posebna funkcija koja izračunava integral rekurzivno. TBB obrazac rekurzivnih zadataka omogućava stvaranje većeg broja zadataka, a to je uslov za postizanje ubrzanja.

# Programsko rešenje

## Glavni modul:

U main funkciji se učitavaju vrednosti intervala iz ulazne datoteke „Test.txt“. Za svaku vrednost iz testne datoteke se integral računa serijski i paralelno nad intervalom(0, 2π). Ispisuju se rezultati i vremena izvršenja, potom se odgovarajući rezultati upisuju u izlaznu datoteku „Rezultati.txt“.

## Serijska verzija

Serijska implementacija predstavlja implementaciju formule. Realizovana je kroz for petlju. Rezultat je inicijalizovan na 0, i uvećava se kroz svaku iteraciju.

### Funkcija za izračunavanje integrala kroz for petlju:

doubleserijskoRacunanjeIntegrala(double& pocetniInterval, double &krajnjiInterval);

Realizacija formule navedene u Analizi problema upotrebom for petlje, prosleđenih parametara i odgovarajućih konstanti.

Parametri: pocetniInterval - donja granica integrala

krajnjiInterval - gornja granica integrala

Povratna vrednost: rezultat izračunavanja integral

### Funkcija za rekurzivno izračunavanje integrala:

double rekurzivnoRacunanje(double pocetniInterval, double krajnjiInterval);

Funkcija izračunava integral rekurzivno, takođe na osnovu formule. Koristi se u paralelnoj verziji, kao i u serijskoj ukoliko je veličina intervala dovoljno mala. Bazni slučaj je dostignut ako je veličina intervala < zadate vrednosti(0.01). Nad tim malim intervalom se izračunava integral. U suprotnom se interval podeli na osnovu prosleđenih parametara i prilikom vraćanja iz funkcije se sabiraju rezultati dva rekurzivna poziva nad polovljenim intervalom.

Parametri: početniInterval - donja granica integrala,

krajnjiInterval - gornja granica integrala.

Povratna vrednost: rezultat izračunavanja integrala.

## Paralelna verzija:

Realizovana kroz klasu IntegralZadatak, koja nasleđuje klasu tbb::task. Sadrži atribute za donju granicu intervala, gornju granicu intervala, sredinu intervala, i pokazivač na rezultat. U konstruktoru se izračunava sredina intervala, a u preklopljenoj virtuelnoj metodi se izvršava sam račun.Sledi opis metoda i funkcije u sklopu paralelne implementacije.

### Funkcija za pravljenje zadatka:

double ParalelnoRacunanje(double dg, double gg);

Funkcija zauzima prostor za zadatak preklopljenim operatorom “new”, metodom task::allocate\_root() i pozivom konstruktora klase IntegralZadatak kom prosledjuje citav interval i rezultat inicijalizovan na 0. Zadatak se potom pravi i pokreće, i čeka se na njegovo izvršenje(“task::spawn\_root\_and\_wait” ).

Parametri: dg – donja granica intervala,

gg – gornja granica intervala.

Povratna vrednost: rezultat izračunavanja.

### Preklopljena virtuelna metoda za sam račun:

task\* execute();

Metoda proverava da li je veličina intervala dovoljno mala da bi se primenila serijska verzija za rekurzivno izračunavanje integrala nad celim intervalom. U suprotnom se prave dva zadatka potomka, koja polove interval i rekurzivno izračunavaju integral nad polovljenim intervalima. Tu se koristi metoda “allocate\_child()“ za zauzimanje prostora za zadatak jer zadatak pravi zadatak – potomak. Poziva se „set\_ref\_count(3)“ pre mrešćenja bilo kog potomka. 3 je zbog dva zadatka potomka + 1 zbog čekanja. Stvaraju se 2 zadatka potomka.Prvo stvaranje zadatka se završava odmah, bez ikakvog čekanja. Drugo stvaranje zadatka dovodi do čekanja roditelja da se završe svi zadaci potomci („spawn\_and\_wait\_for\_all“). Nakon završetka oba zadatka predak sabira njihove rezultate i smešta ih u promenljivu rezultat.

Povratna vrednost metode execute() je ili „NULL“ ili pokazivač na sledeći zadatak. U ovom slučaju je to „NULL“.

# Ispitivanje

Prilikom ispitivanja vrednosti u okviru različitih testova jedino što sam menjao je vrednost intervala. Sva testiranja su izvršena na intervalu (0, 2π), a vrednost na osnovu koje se određuju bazni slučajevi ( u programu definisana kao “ZADATA\_VR\_INTERVALA“) je bila postavljena na 0.01. Sva ispitivanja su izvršena na “Intel”-ovom “i5” procesoru sa 4 jezgra. Analiza rezultata za neke druge vrednosti intervala dovode do približno istih rezultata po pitanju ubrzanja i preciznosti prilikom izračunavanja rezultata.

# Analiza rezultata

Na osnovu vrednosti intervala ulazne datoteke “Test.txt”, nakon obavljenih računanja za serijsku i paralelnu verziju, u izlaznoj datoteci “Rezultati.txt” su prikazani sledeći rezultati.

|  |  |  |  |
| --- | --- | --- | --- |
| Serijska | | Paralelna | |
| Rezultat | Vreme izvrsenja | Rezultat | Vreme izvrsenja | Interval |
| 0.387797 | 0.0343598ms | 0.628514 | 1.73083ms | 0.01 |
| 0.383228 | 0.15141ms | 0.436684 | 0.520683ms | 0.001 |
| 0.382763 | 1.43632ms | 0.386728 | 0.847291ms | 0.0001 |
| 0.382719 | 15.4234ms | 0.382972 | 7.39378ms | 0.00001 |
| 0.382715 | 151.435ms | 0.38272 | 59.4025ms| | 0.000001 |
| 0.382714 | 1492.93ms | 0.382719 | 575.164ms | 0.0000001 |
| 0.382714 | 14894.8ms | 0.382715 | 6107.87ms | 0.00000001 |

Napomena: ms = milisekunde

Rezultati gotovo nikad ne budu isti. Uvek se makar za nijansu razlikuju nakon svakog pokretanja. U suštini, rezultate možemo analizirati sa dva aspekta:

* Na osnovu dobijenih rezultata izračunavanja
* Na osnovu vremena izvršenja

Radi lakše vizuelizacije i analize rezultata izračunavanja obratimo pažnju na sledeći grafik.

Sa grafika se vidi da je nad većim intervalom serijska verzija bolja od paralelne. Rezultat paralelne verzije divergira od tačnog rešenja na najvećem intervalu. Međutim, rezultat paralelne verzije biva precizniji sa smanjivanjem intervala. Isto važi i za serijsku, s tim da je ona od samog početka dosta bliža tačnom rezultatu za razliku od paralelne verzije.

Kad je u pitanju vreme izvršenja, bacimo prvo pogled na sledeće grafike. Nad većim intervalom je vreme izvršenja dosta kratko. Sa smanjivanjem intervala vreme trajanja se znatno poveća pa je radi lakšeg iščitavanja odgovarajućih vrednosti prikazano više grafika.

Sa grafika se vidi da nad većim vrednostima intervala serijska verzija radi brže od paralelne. Međutim, smanjivanjem intervala vreme izvršavanja paralelne verzije se približava serijskoj, da bi već od vrednosti intervala 0.0001 paralelna verzija postala brža. Kako se nastavi smanjivanje intervala paralelna verzija radi sve brže i brže od serijske, uglavnom oko 2-4 puta brže u zavisnosti od procesora na kom se izvršava program.

# Zaključak

Serijska verzija je efikasnija od paralelne nad većim vrednostima intervala. Paralelna verzija se poboljšava smanjivanjem vrednosti intervala, pogotovo u pogledu brzine izvršenja. Moja pretpostavka je da ćeubrzanje nakon određenegranice smanjivanja intervala stagnirati, i da će ostati na približno istim vrednostima. Presudan uticaj ima broj jezgara na procesoru. Obe verzije su efikasne na svoj način, serijsku treba koristiti za brza izračunavanja nad većim vrednostima intervala, dok je paralelna bolja za preciznija izračunavanja nad dovoljno malim vrednostima intervala.