Elektrotehnički fakultet   
Univerziteta u Beogradu



Diplomski rad

Uporedna analiza paralelnih algoritama   
za digitalnu obradu zvuka

Kandidat: Mentor:

Dušan Nikolov dr Milo Tomašević

vanr. prof

Beograd, septembar 2015.

SADRŽAJ

[1. Uvod 4](#_Toc426884258)

[2. O digitalnoj obradi signala / zvuka 5](#_Toc426884259)

[2.1. Digitalna obrada zvuka 6](#_Toc426884260)

[2.1.1. Diskretizacija (uzorkovanje) 7](#_Toc426884261)

[2.1.2. Kvantizacija 8](#_Toc426884262)

[2.2. Performanse algoritama... 9](#_Toc426884263)

[Vremenska složenost 9](#_Toc426884264)

[Složenost implementacije 9](#_Toc426884265)

[Predmet i ciljevi rada 9](#_Toc426884266)

[3. Pregled izabranih algoritama... 10](#_Toc426884267)

[3.1. Quicksort 10](#_Toc426884268)

[Vremenska složenost 11](#_Toc426884269)

[Mogućnosti za paralelizaciju 11](#_Toc426884270)

[4. Pregled korišćenih tehnologija 12](#_Toc426884271)

[4.1. Paralelni računarski sistemi... 12](#_Toc426884272)

[Memorijske arhitekture paralelnih računara 12](#_Toc426884273)

[Paralelni programski modeli 12](#_Toc426884274)

[Programski model deljene memorije / OpenMP 12](#_Toc426884275)

[5. Implementacije algoritama... 13](#_Toc426884276)

[5.1. Sekvencijalne verzije algoritama 13](#_Toc426884277)

[5.2. Paralelne verzije algoritama 13](#_Toc426884278)

[5.3. Implementaciona iskustva 13](#_Toc426884279)

[6. Metodologija analize 14](#_Toc426884280)

[6.1. Razvojna platforma 14](#_Toc426884281)

[6.2. Hardverska platforma 14](#_Toc426884282)

[6.3. Test okruženje 14](#_Toc426884283)

[6.3.1. Merenje vremena 14](#_Toc426884284)

[7. Rezultati analize 15](#_Toc426884285)

[7.1. Sekvencijalni algoritmi 15](#_Toc426884286)

[7.2. pThreads implementacije 16](#_Toc426884287)

[7.3. MPI implementacije 16](#_Toc426884288)

[7.4. CUDA implementacije 18](#_Toc426884289)

[8. Zaključak 20](#_Toc426884290)

[9. Literatura 21](#_Toc426884291)

[Prilozi 22](#_Toc426884292)

[TestSuite okruženje za testiranje 22](#_Toc426884293)

# Uvod

Sortiranje podataka je jedna od osnovnih i veoma važnih aktivnosti u algoritmima za obradu podataka. Sortiranje se može definisati kao proces preuređivanja skupa podataka po nekom utvrđenom poretku [1]. Najčešće se podaci uređuju u numeričkom ili leksikografskom poretku...

Paralelne arhitekture i programski modeli...

Cilj ovog rada je..

Rezultati su prikazani tabelarno i grafički, nakon čega su detaljno diskutovani u odnosu na rezultate iz otvorene literature...

U prvom poglavlju je dat uvod u problem sortiranja..., a sam problem je detaljnije obrađen u drugom poglavlju... U trećem poglavlju su date opšte ideje za izabrane algoritme...Četvrto poglavlje daje pregled...U petom poglavlju su dati opisi implementacija...Sedmo poglavlje sadrži prikaz relevantnih rezultata koje je dao ovaj rad. U osmom poglavlju je dat zaključak rada i naznačeni pravci daljeg istraživanja.

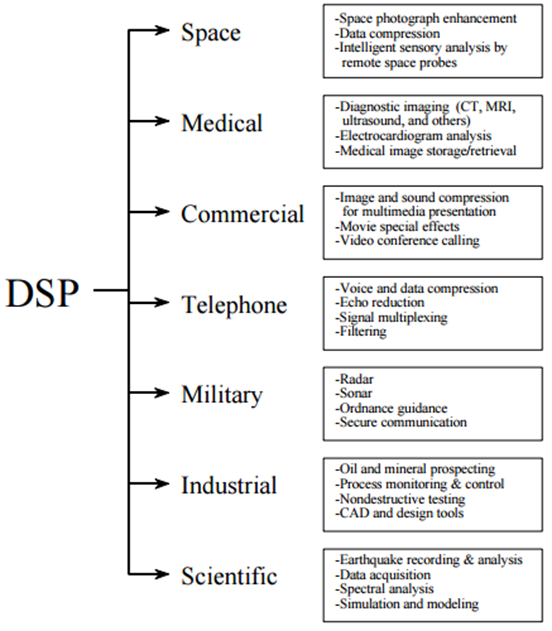
Deveto poglavlje sadrži pregled literature i izvora korišćenih prilikom izrade ovog rada. Literatura je navedena u IEEE formatu preporučenom u [25], a tamo gde je bio dostupan postavljen je i odgovarajući link na internetu. Deseto poglavlje sadrži priloge i u njemu su se našli svi oni delovi teksta za koje je autor smatrao da dopunjuju sadržaj ovog rada...

# O digitalnoj obradi signala / zvuka

Digitalna obrada signala je disciplina računarskih nauka koja predstavlja matematičku manipulaciju signala, koja dovodi do njegove modifikacije.

Za razliku od ostalih oblasti računarskih nauka, digitalna obrada signala radi sa specifičnim tipom podataka: signalima. U velikoj većini slučajeva ovi signali potiču iz realnog sveta – senzorni podaci. S druge strane postoje i čisto digitalni signali – sintetitovani signali. Cilj digitalne obrade signala je najčešće merenje nekih njihovih karakteristika, filtriranje ili kompresija signala. Da bi se analogni signal mogao modifikovati nekim od alata za digitalnu obradu signala, neophodno ga je prvo prevesti iz analognog u digitalni oblik, a u te svrhe se koriste AD (Analogno-digitalni) konvertori. Analogno, postoje i DA (Digitalno-analogni) konvertori koji se koriste da signal nakon izvršene obrade vrate nazad u analogni oblik.

Oblast digitalne obrade signala je celina sačinjena iz više manjih, konkretnijih oblasti, specijalizovanih za obradu specifičnijih vrsta signala. U zavisnosti od prirode, odnosno porekla signala, postoje: obrada zvuka, obrada signala govora, obrada radarskih i sonarnih signala, obrada slike, obrada seizmičkih podataka i dr…

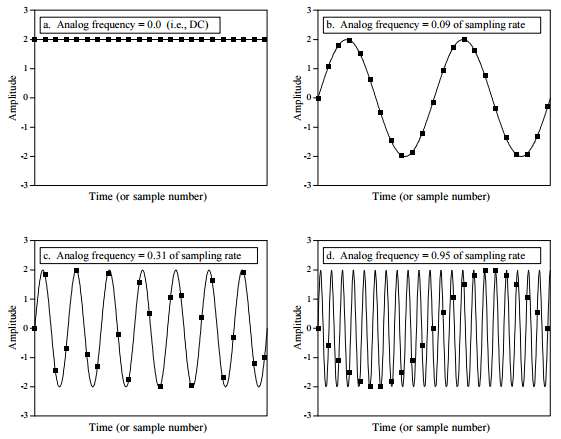


Digitalni signali se mogu posmatrati iz nekoliko domena: vremenskog (jednodimenzionalni signali), prostornog (višedimenzionalni signali), frekvencijskog ili vremensko-frekvencijskog (wavelet) domena. Prilikom digitalne obrade signala, neretko je neophodno transformisati signal iz domena u domen, jer se neke određene karakteristike tog signala najbolje ispoljavaju u jednom, dok se neke druge ispoljavaju u drugom domenu. Sve to zavisi od tipa obrade koja se obavlja nad signalom. Za prelazak između vremenskog i frekvencijskog domena se koristi Fourier-ova transformacija.

## Digitalna obrada zvuka

Zvuk je jedan od gorepomenutih tipova signala, i glavni predmet ovog rada je digitalna obrada zvuka. Značajne karakteristike signala zvuka su trajanje u vremenskom domenu i opseg frekvencija koje zauzima u frekvencijskom domenu.

Jedna od važnih karakteristika prirodnog zuka je kontinualnost. Međutim, obrada kontinualnih signala je jako intenzivan i vremenski zahtevan poduhvat, pa kao takav nije pogodan za računarsku obradu. Da bi se analogni kontinualni signal mogao obrađivati na računaru, neophodno je digitalizovati ga. Digitalizacija signala se u dve faze: diskretizacija (uzorkovanje) i kvantizacija.



### Diskretizacija (uzorkovanje)

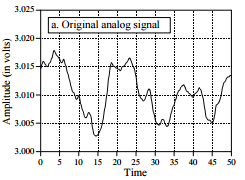
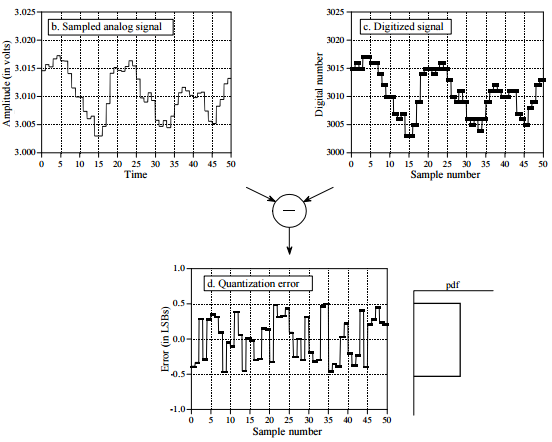
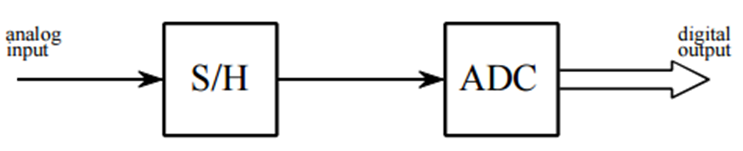
Diskretizacija je proces u odabiranja određenog broja vrednosti iz kontinualnog signala u tačno određenim vremenskim trenucima. Vremenski trenuci u kojima se odabiraju vrednosti signala su određeni frekvencijom odabiranja.

Teorema odabiranja govori o uslovima koji moraju da budu zadovoljeni da bi frekvencija odabiranja bila dobra. Frekvencija odabiranja je zadovoljavajuća ukoliko se analogni signal može verno rekonstruisati na osnovu dobijenih odbiraka. Na prethodnoj slici su prikazani primeri dobro i loše odabranih frekvencija odabiranja. Sa slike se vidi da je u poslednjem slučaju frekvencija odabiranja loše izabrana, jer dobijeni signal (diskretne vrednosti obeležene kvadratićima) nije ni nalik na originalni, dok je u prethodna tri slučaja diskretizacija dala vernu aproksimaciju originalnog signala.

Teorema o odabiranju (Shannon sampling theorem / Nyquist theorem) nalaže da kontinualni signal može biti pravilno diskretizovan, samo ako ne sadrži učestanosti koje su veće od polovine frekvencije odabiranja. Polovina frekvencije odabiranja se naziva Nikvistova frekvencija (Nyquist freq.).

### Kvantizacija

Kvantizacija je drugi deo AD konverzije signala. Odnosi se na proces u kome se kontinualne vrednosti signala diskretizuju. Kvantizacijom se kontinualni raspon amplitude signala deli u jednakih 2n diskretnih nivoa, a zatim se vrednosti signala koje su sačuvane nakon diskretizacije (uzorkovanja) pridružuju nivou koji im je najbliži. Ovakvo zaokruživanje na najbliži nivo dovodi do određene greške, koja se naziva greška kvantizacije. Greška kvantizacije predstavlja upravo tu razliku između stvarne, odabrane vrednosti i vrednosti na koju se ona zaokružuje. Ovo je između ostalog i izvor najvećih smetnji (šumova) u digitalnoj obradi signala.



## Performanse algoritama...

Uobičajeni indikatori performansi algoritama...

### Vremenska složenost

Vremenska složenost...

### Složenost implementacije

Složenost implementacije...

### Predmet i ciljevi rada

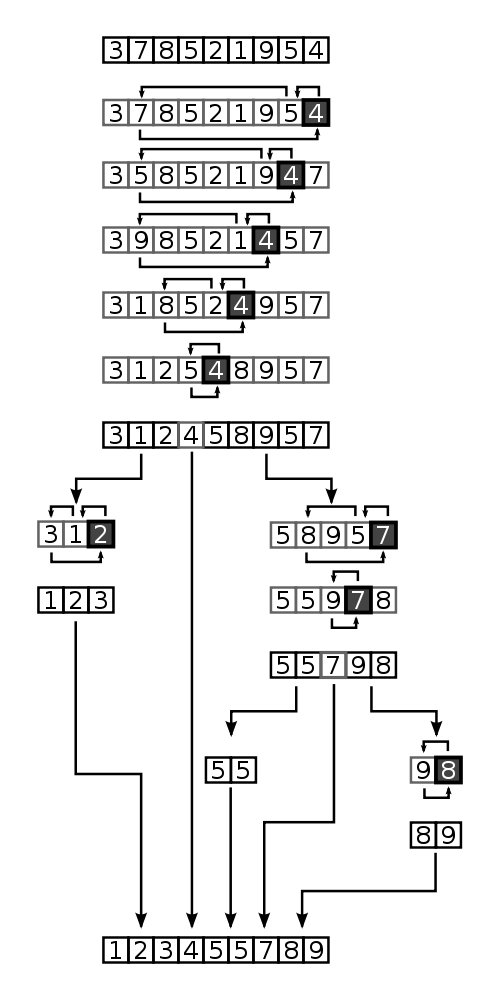
Kao što je prethodno izloženo, sortiranje...

# Pregled izabranih algoritama...

Ovo poglavlje opisuje izabrane algoritme za sortiranje i prikazuje njihove osobine...

## Quicksort

Quicksort ili partition-exchange sort je dobro poznat i popularan algoritam za sortiranje...



Slika 3.1 – Primer slike

Primer koda:

procedure quicksort(array, left, right)

if right > left

select a pivot index (e.g. pivotIndex := (left+right)/2)

pivotNewIndex := partition(array, left, right, pivotIndex)

quicksort(array, left, pivotNewIndex - 1)

quicksort(array, pivotNewIndex + 1, right)

### Vremenska složenost

Quicksort u prosečnom i najboljem slučaju ostvaruje vremensku složenost...

### Mogućnosti za paralelizaciju

Kako se algoritam zasniva na divide-and-conquer principu, postoji mogućnost za njegovu paralelizaciju...

# Pregled korišćenih tehnologija

U ovom poglavlju će biti napravljen osvrt na najviše korišćene paralelne programske modele i njihove dostupne i popularne implementacije...

## Paralelni računarski sistemi...

...

### Memorijske arhitekture paralelnih računara

Poznate memorijske arhitekture su deljena memorija (shared memory), distribuirana memorija (distributed memory) i hibridni pristup (distributed shared memory)...

### Paralelni programski modeli

Paralelni programski modeli...

### Programski model deljene memorije / OpenMP

Model deljene memorije (engl. shared memory) je

# Implementacije algoritama...

U ovom poglavlju je napravljen pregled implementacija za uporednu analizu

## Sekvencijalne verzije algoritama

Da bi se izvršila adekvatna poređenja, merenja i analiza ponašanja paralelnih implementacija algoritama za sortiranje, korišćeni se u odgovarajuće sekvencijalne implementacije ovih algoritama...

## Paralelne verzije algoritama

Ova sekcija ukratko opisuje implementacije izabranih paralelnih algoritama za sortiranje na odgovarajućim platformama...

Metod paralelizacije

Jedan od često korišćenih pristupa prilikom paralelnog rešavanja nekog problema je...

## Implementaciona iskustva

Tokom procesa implementacije, autor se susreo sa nekoliko izazova...

# Metodologija analize

U ovom poglavlju je opisana metodologija analize paralelnih algoritama. Do detalja je opisano test okruženje i navedena kako razvojna, tako i hardverska platforma na kojoj je vršena evaluacija algoritama...

## Razvojna platforma

Za implementaciju i testiranje algoritama za sortiranje korišćen je programski jezik C...

## Hardverska platforma

Generalnu hardversku platformu za implementaciju i testiranje algoritama je činio računar zasnovan...

## Test okruženje

Da bi bila sprovedena adekvatna analiza realizovanih algoritama realizovano je test okruženje...

### Merenje vremena

Merenje vremena u sistemu je realizovano pomoću odgovarajućih funkcija operativnog sistema domaćina...

# Rezultati analize

U ovom poglavlju su izneseni relevantni rezultati i data je diskusija rezultata u odnosu na očekivane vrednosti i donesene projektne odluke...

## Sekvencijalni algoritmi

U ovoj sekciji se mogu videti rezultati dobijeni testiranjem algoritama opisanih...

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteracija** | **Veličina particije** | **Proteklo vreme [ms]** | | |
| **quicksort** | **mergesort** | **radixsort** |
| **16bitna uniformna raspodela** | | | | |
| 0 | 100 | 0,010159 | 0,006801 | 0,006567 |
| 1 | 1K | 0,133311 | 0,08548 | 0,049365 |
| 2 | 10K | 1,700406 | 1,08488 | 0,497629 |
| 3 | 100K | 20,449937 | 13,23435 | 5,03891 |
| 4 | 1M | 183,665449 | 155,580612 | 89,880532 |
| 5 | 10M | 1784,88895 | 1696,053501 | 899,825924 |

Tabela 7.1 – Primer tabele



Slika 7.1 – Uporedni prikaz performansi sekvencijalnih algoritama   
za sortiranje za različite raspodele ulaznih podataka;  
1 – 16bit uniformna raspodela, 2 – 32bit uniformna raspodela, 3 – Gausova raspodela, 4 – Few unique keys,  
5 – Skoro sortirani nizovi, 6 – Sortirani nizovi, 7 – Obrnuto sortirani nizovi

## pThreads implementacije

U ovom odeljku su dati rezultati merenja za pThreads implementacije algoritama...

## MPI implementacije

Evaluacija MPI implementacija...



Slika 7.2 - Uporedni prikaz performansi sekvencijalnih i MPI algoritama za sortiranje   
za podatke 32-bitne uniformne raspodele za različite veličine ulaznih nizova

## CUDA implementacije

Rezultati dobijeni merenjima CUDA implementacija su slični onima iz literature [12] i [13]...



Slika 7.3 - Uporedni prikaz performansi CUDA implementacija algoritama   
za sortiranje za različite raspodele ulaznih podataka;  
1 – 16bit uniformna raspodela, 2 – 32bit uniformna raspodela, 3 – Gausova raspodela, 4 – Few unique keys,  
5 – Skoro sortirani nizovi, 6 – Sortirani nizovi, 7 – Obrnuto sortirani nizovi



Slika 7.4 – Uporedni prikaz performansi sekvencijalnih i CUDA algoritama za sortiranje   
za podatke 32-bitne uniformne raspodele za različite veličine ulaznih nizova

# Zaključak

Sortiranje je jedna od vrlo čestih aktivnosti u računarskim programima. Kao takvo, predmet je istraživanja već duži niz godina... Cilj rada je bio da se uporede i testiraju danas najpopularniji algoritmi za sortiranje sa potencijalom za paralelizaciju...

Obrađeni algoritmi su najpre posmatrani i opisani sa teorijske strane, uz komentarisanje njihovih najvažnijih karakteristika i ograničenja...

Može se smatrati da ne postoji apsolutni pobednik u ovoj analiz...

Dalji razvoj ovih algoritama bi se mogao koncentrisati na... Na kraju, automatizovano test okruženje bi se moglo proširiti...

# Literatura

Format je usklađen sa IEEE uputstvom za navođenje referenci, datim u [6].

1. Tomašević, M., Algoritmi i strukture podatka, Akademska misao Elektrotehnički fakultet, Beograd, 2008.
2. Smith W. Stewen, The Scientist and Engineering’s guide to Digital Signal Processing, 1997.
3. Culler, D., Singh, J.,P., Gupta, A., Parallel Computer Architecture, Morgan Kaufmann Publishers, 1999.
4. Kirk B. David/NVIDIA Corporation, Hwu W. Wen-mei, Programming Massively Parallel Processors – A Hands-on Approach, second edition, Elsevier Inc. 2013.
5. NVIDIA CUDA Programming Guide, Version 7.0, NVIDIA Corporaton, 2015.
6. Kraus, L., Programski jezik C sa rešenim zadacima, Akademska misao, Beograd, 2004.
7. IEEE, Preparation of Papers for IEEE Transactions and Journals,   
   https://www.ieee.org/documents/transactions\_journals.pdf, 2015.

# Prilozi

U ovom poglavlju se nalaze delovi teksta koji ili olakšavaju snalaženje unutar celog teksta ovog rada ili oni čiji je sadržaj potrebno navesti zbog celovitosti rada, ali bi njihovo stavljanje u neko od ranijih poglavlja nepotrebno opteretilo tekst na datom mestu.

## TestSuite okruženje za testiranje

Pored opisa iz sekcije 6.3, ovde je dat izvod iz datoteke...