МЕЃУНАРОДЕН ЦЕНТАР ЗА СЛАВЈАНСКА ПРОСВЕТА - СВЕТИ НИКОЛЕ

«МЕЃУНАРОДЕН ДИЈАЛОГ: ИСТОК - ЗАПАД»
(ЕКОНОМИЈА, БЕЗБЕДНОСНО ИНЖЕНЕРСТВО, ИНФОРМАТИКА)

С П И С А Н И Е
на научни трудови

ДВАНАЕСЕТТА МЕЃУНАРОДНА НАУЧНА КОНФЕРЕНЦИЈА „МЕЃУНАРОДЕН ДИЈАЛОГ: ИСТОК - ЗАПАД“
МЕЃУНАРОДЕН СЛАВЈАНСКИ УНИВЕРЗИТЕТ „ГАВРИЛО РОМАНОВИЧ ДЕРЖАВИН“ СВЕТИ НИКОЛЕ - БИТОЛА

Година VIII ***Број 1*** ***Април 2021***

- СВЕТИ НИКОЛЕ, Р. СЕВЕРНА МАКЕДОНИЈА -
- 2021 -

Издавач: Меѓународен центар за славјанска просвета - Свети Николе

За издавачот: м-р Михаела Ѓорчева, директор

Наслов: «МЕЃУНАРОДЕН ДИЈАЛОГ: ИСТОК - ЗАПАД» (ЕКОНОМИЈА, БЕЗБЕДНОСНО ИНЖЕНЕРСТВО, ИНФОРМАТИКА)

Компјутерска обработка и дизајн: Адриано Панајотов, Маја Маријана Панајотова, Благој Митев

ISSN (принт) 1857-9299
ISSN (онлајн) 1857-9302

Адреса на комисијата: ул. Маршал Тито 77, Свети Николе, Р. Сверна Македонија

Контакт телефон: +389 (0)32 440 330

Уредувачкиот одбор им се заблагодарува на сите учесници за соработката!

Печати: Печатница и книжарница „Славјански“, Свети Николе

Тираж: 100

МЕЃУНАРОДЕН ДИЈАЛОГ

ИСТОК - ЗАПАД

ЕКОНОМИЈА, БЕЗБЕДНОСНО ИНЖЕНЕРСТВО,
ИНФОРМАТИКА

ОБЛАСТ

ИНФОРМАТИКА

004.434C++:004.633.2

Mahir Zajmović
Ibrahim Obhođaš
Milovan Dimić
Sveučilište/Univerzitet "VITEZ"
Bosna i Hercegovina

KOMPARATIVNA ANALIZA ALGORITAMA ZA SORTIRANJE PODATAKA U PROGRAMSKOM JEZIKU C++

APSTRAKT: U ovom radu prikazano je nekoliko rješenja sortiranja podataka u statičkim nizovima podataka. U praksi veoma često imamo potrebu za sortiranjem određenih podataka, pa je od velike pomoći izbor adekvatnog algoritma za sortiranje koji će raditi što brže i efikasnije. Prikazane su i praktične izvedbe algoritama za sortiranje koje su testirane na različitim nasumičnim skupovima podataka. Također, radi lakšeg shvaćanja funkcionalnosti određenog algoritma za sortiranje priložena je komparativna analiza brzine izvođenje svakog od testiranih algoritama za sortiranje. Testiranje algoritama za sortiranje realizovano je u programskom jeziku C++.

KLJUČNE RIJEČI: analiza, algoritam, sortiranje, podaci, programski jezik, C++

COMPARATIVE ANALYSIS OF ALGORITHMS FOR DATA SORTING IN C++ PROGRAMMING LANGUAGE

ABSTRACT: The problem, identified by the authors, is the lack of a unified model of general microprocessor that is similar to any of the real microprocessors from hardware and software aspects. In this paper we introduce an original 8-bit digital component as a unique model of general microprocessor, which is a combination of common features, but at the same time simplification of three real 8-bit microprocessors: Intel 8080, Intel 8085 and Zilog Z-80. More specifically, in this paper we focus on the hardware aspect (on the pin configuration), explaining the role of each pin of our model.

KEY WORDS: analysis, algorithm, sorting, data, programming language, C++

UVOD

Veoma često imamo potrebu da sortiramo određene podatke koji se nalaze unutar nizova ili samih datoteka. Iz više razloga prikladnije je analizirati jednostavnije metode nego neke kompleksne kao npr. što na jednostavan način možemo naučiti terminologiju sortiranja te nam to omogućava lakše proučavanje kompleksnijih algoritama. Moguće je za jedan problem imati više različitih algoritama koji rješavaju taj problem. Uobičajeno je da nisu svi algoritmi jednako dobri u smislu brzine izvođenja.

Algoritmi sortiranja koji su obrađeni u ovom radu:
a) Bubble sort
b) Insertion sort
c) Quick sort

Neke od tih metoda pripadaju osnovnoj vrsti algoritama, npr. Bubble sort i Insertion sort, dok drugi algoritmi pripadaju složenijoj vrsti algoritama poput Quick sort-a.

Analizirani su algoritmi, odnosno njihove složenosti kako bi odabrali najbolji mogući algoritam za rješavanje određenog zadatka.

1. ALGORITMI SORTIRANJA I NJIHOVA SLOŽENOST

U informacionim tehnologijama postoje različiti algoritmi za sortiranje koji rade na principu sortiranja slova i riječi (leksikografsko sortiranje) ili sortiranja brojeva (brojevno sortiranje). Sortiranje je moguće da bude uzlazno (od manjeg prema većem) i silazno (od većeg prema manjem).

Svaki put nam je lakše koristiti sortirane podatke i predmete u odnosu na nesortirane. Problem sortiranja možemo definirati na sljedeći način[1]:

» ULAZ: sekvenca $\{a1, a2, \dots, an\}$ brojeva
» IZLAZ: permutacija $\{a'1, a'2, \dots, a'n\}$ takva da vrijedi da je $a'1 \leq a'2 \leq a'n$ ili $a'1 \geq a'2 \geq a'n$

» Primjer:
  - ULAZ: 2, 5, 9, 13, 7, 6
  - IZLAZ: 2, 5, 6, 7, 9, 13 ili 13, 9, 7, 6, 5, 2

Sortiranje velike količine podataka i uz računare može dugo trajati. Neki od algoritama koji efikasno rade sortiranje su:

Algoritam	Grupa
Sortiranje izborom najmanjeg elementa (Selection sort)	Zamjena elemenata
Sortiranje zamjenom susjednih elemenata (Bubble sort)	
Jednostavno sortiranje umetanjem (Insertion sort)	Sortiranje umetanjem
Višestruko sortiranje umetanjem (Shell sort)	
Sažimanje sortiranih polja (Merge)	Rekurzivni
Sortiranje sažimanjem (Merge sort)	
Brzo sortiranje (Quick sort)	

Tabela 1. Prikaz algoritama za sortiranje[2]

1.1. SORTIRANJE ZAMJENOM SUSJEDNIH ELEMENATA (BUBBLE SORT)

Sortiranje zamjenom susjednih elemenata (engl. Bubble sort) je jedan od najjednostavnijih algoritama za sortiranje. Može se krenuti sortirati od početka ali isto tako i od kraja liste. Osnovna ideja algoritma je usporedba svaka dva susjedna elementa liste (polja) te im se mjenjaju mjesta ako poredak nije dobar. Nakon jednog prolaska kroz listu najveći element biva "izguran" na kraj liste odnosno ispliva kao mjehurić otkud dolazi i sam naziv algoritma. Takav postupak se ponavlja sve dok cijela lista nije sortirana. Lista se u svakom trenutku dijeli na dvije podliste, sortiranu i nesortiranu, odvojene „konceptualnim zidom".[3]

Pseudokôd ovog algoritma je sljedeći:

» Kreni od početka niza prema kraju;
» Zamijeni 2 elementa ako je prvi veći od drugog;
» Ako u prolasku kroz cijeli niz nije bilo zamjene, niz je sortiran.

1. Zajmović, M., Sortiranje i pretraživanje podataka, Studentska konferencija SKEI 2016, Sveučilište/Univerzitet „Vitez", zbornik radova, Vitez, 2016. godine, str. 327
2. Zajmović, M., Sortiranje i pretraživanje podataka, Studentska konferencija SKEI 2016, Sveučilište/Univerzitet „Vitez", zbornik radova, Vitez, 2016. godine, str. 327
3. Jakus, A., Algoritmi za sortiranje u programskom jeziku C++, Sveučilište u Rijeci, Filozofski fakultet, Odsjek za politehniku, završni rad, Rijeka, 2016. godine, str. 9

Ovaj algoritam ima najlošiju složenost koja dolazi iz činjenice da se za implementaciju koriste dvije for petlje od kojih jedna služi za usporedbu svaka dva susjedna elementa dok druga prolazi preko cijelog djela nesortiranog niza.

16	11	5	8	10	➔	Zamjena
11	16	5	8	10	➔	Zamjena
11	5	16	8	10	➔	Zamjena
11	5	8	16	10	➔	Zamjena
11	5	8	10	16	➔	Dio liste je sortiran (prolaz 1)
11	5	8	10	16	➔	Zamjena
5	11	8	10	16	➔	Zamjena
5	8	11	10	16	➔	Zamjena
5	8	10	11	16	➔	Dio liste je sortiran (prolaz 2)
5	8	10	11	16	➔	Nema zamjene
5	8	10	11	16	➔	Nema zamjene
5	8	10	11	16	➔	Dio liste je sortiran (prolaz 3)
5	8	10	11	16	➔	Nema zamjene
5	8	10	11	16	➔	Lista je sortirana (prolaz 4)

Tabela 2. Koraci sortiranja Bubble sort algoritma

Implementacija Bubble sort algoritma u programskom jeziku C++:

```
void bubble_sort(int niz[], int n)
{
        int temp;
        for (int i = n - 1; i > 0; i--)
        {
        for (int j = 0; j < i; j++)
        {
                if (niz[j] > niz[j + 1])
                {
                        temp = niz[j];
                        niz[j] = niz[j + 1];
                        niz[j + 1] = temp;
                        }
                }
        }
}
```

1.2. JEDNOSTAVNO SORTIRANJE UMETANJEM (INSERTION SORT)

Jednostavno sortiranje umetanjem (engl. "Insertion sort") se temelji na principu ubacivanja pojedinog elementa na odgovarajuće mjesto. Skup elemenata koji se sortiraju podjeli se na sortirani i nesortirani. Zatim se svakim korakom povećava sortirani dio na način, da se iz nesortiranog dijela uzima prvi element i ubacuje se na odgovarajuće mjesto u sortiranom dijelu.[4]

4. Zajmović, M., Sortiranje i pretraživanje podataka, Studentska konferencija SKEI 2016, Sveučilište/Univerzitet „Vi-

12	9	4	99	120	1	3	10
12	9	4	99	120	1	3	10
9	12	4	99	120	1	3	10
4	9	12	99	120	1	3	10
4	9	12	99	120	1	3	10
4	9	12	99	120	1	3	10
1	4	9	12	99	120	3	10
1	3	4	9	12	99	120	10
1	3	4	9	10	12	99	120

Tabela 3. Princip rada Insertion sort algoritma[5]

Ovaj algoritam je jednostavan agloritam i puno je brži i efikasniji od Bubble sort algortima. Kada imamo na ulazu već sortiran niz što je ujedno i najbolji slučaj, tada imamo linearnu složenost. Što se tiče vrlo malih nizova sortiranje umetanjem slovi kao jedno od najboljih rješenja, odnosno najbržih.

Implementacija Insertion sort algoritma u programskom jeziku C++:

```
void insertion_sort(int niz[], int n)
{
        int temp;
        int j;
        for (int i = 1; i < n; i++)
        {
                j = i;
                temp = niz[i];
                while (j>0 && temp < niz[j - 1])
                {
                        niz[j] = niz[j - 1];
                        j--;
                }
                niz[j] = temp;
        }
}
```

1.3. BRZO SORTIRANJE (QUICK SORT)

Brzo sortiranje (engl. "Quick sort") spada u klasu algoritama „podjeli pa vladaj" (engl. "devideand-conquer") i rekurzivan je. Ovaj algoritam izumio je C. A. R. Hoare 1962. godine. Primjene u praksi pokazuju kako je ovo jedan od najboljih algoritama za sortiranje uopće. Bolji algoritmi rade samo sa određenim specifičnim tipovima podataka (npr. int). Osnovna ideja ovog algoritma je da se odabere jedan element iz niza i proglasi ga se pivot elementom. Ostatak niza posloži se oko ovog pivot elementa tako da s lijeva budu elementi koji su <= od pivota, a s desna oni koji su >= od pivota. Time postižemo da je pivot "na pravom" mjestu. Lijevi i desni pod-niz se sortira rekurzivnim pozivom istog algoritma.[6]

tez", zbornik radova, Vitez, 2016. godine, str. 329

5. Jakus, A., Algoritmi za sortiranje u programskom jeziku C++, Sveučilište u Rijeci, Filozofski fakultet, Odsjek za politehniku, završni rad, Rijeka, 2016. godine, str. 11

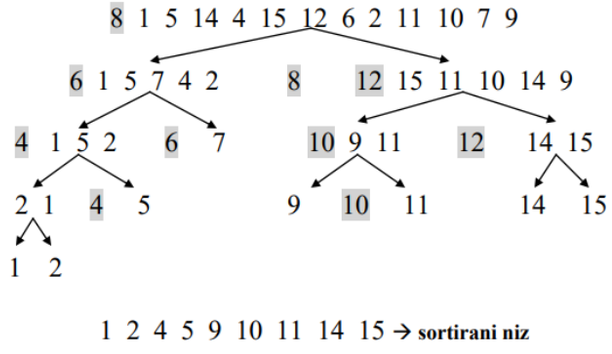
6. Zajmović, M., Sortiranje i pretraživanje podataka, Studentska konferencija SKEI 2016, Sveučilište/Univerzitet „Vi-

Glavni koraci ove metode su:

» Podijeli: ako je veličina problema (ulaza) prevelika da bi ga direktno rješavali, dijelimo problem na dva ili više disjunktnih potproblema .
» Savladaj: rekurzivno primjenjujemo metodu podijeli i savladaj za rješavanje potproblema.
» Kombiniraj: uzimamo dobivena rješenja potproblema i spajamo da bi dobili rješenje početnog problema.[7]

Kod implementacije Quick sort-a u jeziku C++ imamo dvije funkcije:

» Quicksort - poziva funkciju podijeliNiz koja dijeli niz na dva dijela ovisno o pivotu te rekurzivno poziva funkcije za prvi kao i za drugi dio niza .
» PodijeliNiz – funkcija prvo definira pivotni element, zatim pregledava sve elemente niza koje uspoređuje sa pivotom te po potrebi vrši zamjenu elemenata (engl. swap).



Slika 1. Princip rada Quick sort algoritma[8]

Implementacija Quick sort algoritma u programskom jeziku C++:

```
int podijeli(int niz[], int p, int q)
{
        int x = niz[p];
        int i = p, j;
        for (j = p + 1; j < q; j++)
        {
                if (niz[j] <= x)
                {
                        i = i + 1;
                        int temp = niz[i];
                        niz[i] = niz[j];
                        niz[j] = temp;
                }
        }
        int tempa = niz[i];
```

tez", zbornik radova, Vitez, 2016. godine, str. 330

7. Jakus, A., Algoritmi za sortiranje u programskom jeziku C++, Sveučilište u Rijeci, Filozofski fakultet, Odsjek za politehniku, završni rad, Rijeka, 2016. godine, str. 15

8. Jakus, A., Algoritmi za sortiranje u programskom jeziku C++, Sveučilište u Rijeci, Filozofski fakultet, Odsjek za politehniku, završni rad, Rijeka, 2016. godine, str. 15

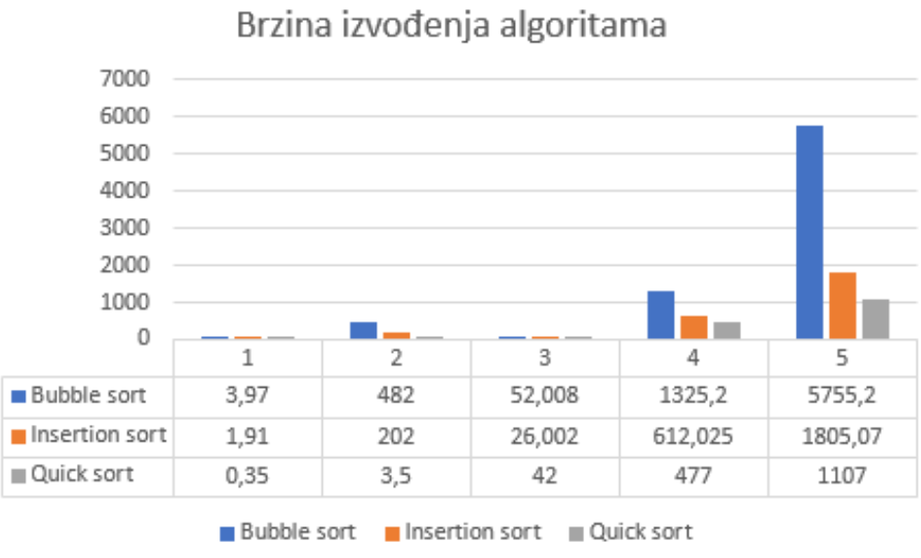
```
            niz[i] = niz[p];
            niz[p] = tempa;
            return i;
}

void quick_sort(int niz[], int p, int q)
{
            int r;
            if (p<q)
            {
                        r = podijeli(niz, p, q);
                        quick_sort(niz, p, r);
                        quick_sort(niz, r + 1, q);
            }
}
```

2. USPOREDBA ALGORITAMA ZA SORTIRANJE I ANALIZA REZULTATA

U ovom radu testirani su rezultati tri algoritma za sortiranje na nasumičnm nizovima različite veličine. Testiranje je relizovano u razvojnom okruženju Microsoft Visual Studio 2019. Korišten je računar sa 2.0 GHz Intel Core 7 procesorom, 8 GB RAM-a memorije, i operativnim sistemom Windows 10 Pro 64 bit.

Broj elemenata (n)	Bubble sort	Insertion sort	Quick sort
1000	3,97 ms	1,91 ms	0,35 ms
10000	482 ms	202 ms	3,5 ms
100000	52,008 sek	26,002 sek	42 ms
500000	1325,2 sek	612,025 sek	477 ms
1000000	5755,2 sek	1805,07 sek	1,107 sek

Tabela 4. Brzina izvođenja algoritama za sortiranje



Iz gore navedenog vidi se kako za kraće nizove (n=1000) razlika u brzini izvođenja i nije toliko velika koliko je kod dužih nizova (n>10000). Kako Bubble sort i Insertion sort imaju kvadratnu složenost tako se povećava vrijeme sortiranja. Drugim rječima, njihova se efikasnost drastično smanjuje na nizovima sa više podataka. Općenito, Insertion sort je bolji algoritam te je relativno učinkovit za male nizove te se koristi kao dio sofisticiranijih programa. Quick sort algoritam sortiranja pokazao se vrlo efikasnim posebno na velikim nizovima. Od svih testiranih algoritama za sortiranje, najbolja svojstva i najbrža vremena sortiranja pripadaju Quick sort-u.

ZAKLJUČAK

U programiranju algoritmi sortiranja su veoma važni najviše zbog toga što nam uveliko olakšavaju pronalazak prave informacije. Također, ovi algoritmi skraćuju naše dragocjeno vrijeme. Treba naglasiti da su važan dio upravljanja podataka koje možemo razlikovati po njihovoj složenosti. Svaki od algoritama ima svoje prednosti i nedostatke u odnosu na druge algoritme. Ova osobina se najviše odnosi na vremensku zahtjevnost algoritma.

Podaci mogu biti smješteni u nizove, vezane listom, datoteke, itd. u zavisnosti od toga kako ih želimo koristiti. Stoga trebamo znati odabrati odgovarajuće algoritme za sortiranje podataka kako bismo na najbrži mogući način mogli doći do željenje informacije koristeći određenu bazu podataka.

LITERATURA

1. Zajmović, M., Metodoška zbirka zadataka za učenje C++, Sveučilište/ Univerzitet „Vitez", Travnik, 2012. godine
2. Živković, D., Uvod u algoritme i strukture podataka, Univerzitet Singidunum, Beograd, 2010. godine
3. Motik, B., Šribar, J., Demistificirani C++ (Četvrto izdanje), Element d. o. o., Zagreb, 1997. godine
4. Zajmović, M., Sortiranje i pretraživanje podataka, Studentska konferencija SKEI 2016, Sveučilište/Univerzitet „Vitez", zbornik radova, Vitez, 2016. godine
5. Jakus, A., Algoritmi za sortiranje u programskom jeziku C++, Sveučilište u Rijeci, Filozofski fakultet, Odsjek za politehniku, završni rad, Rijeka, 2016. godine
6. http://alas.matf.bg.ac.rs/~mr09006/algoritmi-sortiranja/ (pristupano, 12.05.2021)
7. https://www.auckland.ac.nz/ (pristupano, 12.05.2021)
8. http://www.cadlab.fsb.hr/download/skripte/39.pdf (pristupano, 12.05.2021)
9. http://www.ijarcsse.com/ (pristupano, 12.05.2021)
10. https://cs.vt.edu/ (pristupano, 12.05.2021)