Pregled knjige ,,An Introduction to the Analysis of Algorithm“ R. Sedgewick-a i P. Flajolet-a

Bojan Marinković

Matematički institut SANU
bojanm@mi.sanu.ac.yu

Argo seminar
12.03.2008.

Pregled predavanja

1. Analiza algoritama
   - Uvod
   - Kompleksnost izračunavanja
   - Analiza algoritama
   - Analiza prosečnog slučaja
   - Asimptotska aproksimacija
   - Distribucije
   - Verovatnosni algoritmi
2. Matematički aparat
3. Strukture podataka
   - Drveće
   - Permutacije
   - Stringovi
   - Reči i mape

- Razlozi za analiziranje nekog algoritma

- Razlozi za analiziranje nekog algoritma
- Dva različita pristupa u analizi:
  1. Analiza najgoreg slučaja
     Kompleksnost izračunavanja
     (Aho, Hopcroft, Ullman)
  2. Odredjivanje najboljeg, najgoreg i prosečnog slučaja
     Analiza algoritama
     (Knuth)

Mergesort

```
procedure mergesort(l,r: integer);
    var i, j, k, m: integer;
    begin
    if r-l > 0 then
        begin
            m := (r+1) div 2;
            mergesort(l, m); mergesort(m+1, r);
            for i := 1 to m-l+1 do b[i] := a[l+i-1];
            for j := m+1 to r do c[j-m] := a[j];
            i := 1; j := 1; b[m+1] := max; c[r-m+1] := max;
            for k := 1 to r do
                if b[i] < c[j]
                    then begin a[k] := b[i]; i := i+1 end
                    else begin a[k] := c[j]; j := j+1 end;
        end;
    end;
```

Teorema (Mergesort)

Za niz od N elemenata, Mergesort algoritam koristi $N \log(N) + O(N)$ poredjenja.

Dokaz.

C_N - broj poredjenja

Važi: $C_N = C_{N/2} + C_{N/2} + N, N >= 2$ i $C_1 = 0$

Za $N = 2^n$ dobijamo: $C_{2^n} = 2C_{2^{n-1}} + 2^n, n >= 1, C_1 = 0$

što nam i daje:

$$C_N = N \log(N)$$

□

Kao posledice ove teoreme možemo zaključiti:

1. Postoji algoritam kome je za niz od N elemenata potrebno vreme reda $N\log(N)$
2. Broj poredjenja *Mergesort* algoritma ne zavisi od rasporeda elemenata u nizu, već samo od njihovog broja.

Kao posledice ove teoreme možemo zaključiti:

1. Postoji algoritam kome je za niz od N elemenata potrebno vreme reda $N \log(N)$
2. Broj poredjenja *Mergesort* algoritma ne zavisi od rasporeda elemenata u nizu, već samo od njihovog broja.

Teorema (Kompleksnost sortirnja)

Bilo koji program za sortiranje mora koristiti barem $\lceil \log(N!) \rceil > N \log(N) - N/(ln 2)$ *poredjenja za neki ulaz.*

Dokaz.

- Broj mogućih rasporeda članova niza dužine N upravo $N!$
- Jednim poredjem broj mogućih rasporeda smanjuje za faktor 2
- Nejednakost iz Stirlingove aproksimacije faktorijela

$\square$

Kao posledice ove teoreme možemo zaključiti:

1. Postoji algoritam kome je za niz od N elemenata potrebno vreme reda $N \log(N)$
2. Broj poredjenja *Mergesort* algoritma ne zavisi od rasporeda elemenata u nizu, već samo od njihovog broja.

Teorema (Kompleksnost sortirnja)

Bilo koji program za sortiranje mora koristiti barem $\lceil \log(N!) \rceil > N \log(N) - N/(ln2)$ poredjenja za neki ulaz.

Posledica

Mergesort je optimalan algoritam za sortiranje.

Predhodnom metodom nismo dobili odgovore na pitanja:

1. Koliko će trajati implementacija *Mergesort* algoritma na odredjenom kompjuteru?
2. Koliko će ta implementacija trajati u odnosu na druge algoritme reda $O(N \log(N))$?
3. Koliko će ta implementacija trajati u najboljem ili prosečnom slučaju?
4. U kakvom je odnosu ta implementacija u odnosu na implementacije algoritama koje ne zahtevaju poredjenje izmedju elemenata niza?

Predhodnom metodom nismo dobili odgovore na pitanja:

1. Koliko će trajati implementacija *Mergesort* algoritma na odredjenom kompjuteru?
2. Koliko će ta implementacija trajati u odnosu na druge algoritme reda $O(N \log(N))$?
3. Koliko će ta implementacija trajati u najboljem ili prosečnom slučaju?
4. U kakvom je odnosu ta implementacija u odnosu na implementacije algoritama koje ne zahtevaju poredjenje izmedju elemenata niza?

Kompletna analiza algoritma obuhvata:

- kompletnu implementaciju algoritma
- odredjivanje potrebnog vremena za izvršavanje svake operacije koja se koristi
- identifikacija nepoznatih koje se povezuju sa frekvencijom izvršavanja osnovnih operacija
- razvoj realnog modela za ulazne podatke
- analiza nepoznatih za modelovani ulaz
- izračunavanje ukupno potrošenog vremena

Analiza prosečnog slučaja

Različite metode za izračunavanje srednje vrednosti:

- distribuciona: Neka je Π_N broj mogućih ulaza veličine N, a Π_{Nk} broj onih ulaza veličine N koji su uzrokovali potrošnju k od algoritma. Tada je: $\Pi_N = \sum_k \Pi_{Nk}$
- kumulativna: Neka je Σ_N ukupna cena algoritma za ulaze veličine N. Tada je $\Sigma_N = \displaystyle\sum_k k\Pi_{Nk}$

Analiza prosečnog slučaja

Različite metode za izračunavanje srednje vrednosti:

- distribuciona: Neka je Π_N broj mogućih ulaza veličine N, a Π_{Nk} broj onih ulaza veličine N koji su uzrokovali potrošnju k od algoritma. Tada je: $\Pi_N = \sum_k \Pi_{Nk}$
- kumulativna: Neka je Σ_N ukupna cena algoritma za ulaze veličine N. Tada je $\Sigma_N = \displaystyle\sum_k k\Pi_{Nk}$

U većini slučajeva proizvoljni ulaz je dovoljno dobar model te onda:

- uporedjujemo različite algoritme za isti zadatak
- predvidjamo potrebe za vreme i količinu memorije
- uporedjujemo isti algoritam na različitim mašinama.

Quicksort

```
procedure quicksort(l,r: integer);
    var v, t, i, j: integer;
    begin
        if r >= l then
            begin
                v := a[r]; i := l-1; j := r;
                repeat
                    repeat i := i+1 until a[i] >= v;
                    repeat j := j-1 until a[j] <= v;
                    t := a[i]; a[i] := a[j]; a[j] := t;
                until j <= i;
                a[j] := a[i]; a[i] := a[r]; a[r] := t;
                quicksort(l, i-1);
                quicksort(i+1, r);
            end
    end;
```

Repeat petlja na nivou nižem od programskog jezika:

```
LOOP     INC i, 1
         CMP v, a[i]
         BL LOOP
```

Repeat petlja na nivou nižem od programskog jezika:

```
LOOP    INC i, 1
        CMP v, a[i]
        BL LOOP
```

A - broj podela B - broj razmena C - broj poredjenja
Na uobičajnom računaru ukupno vreme rada je reda: $4C + 11B + 35A$.

Repeat petlja na nivou nižem od programskog jezika:

```
LOOP    INC i, 1
        CMP v, a[i]
        BL LOOP
```

A - broj podela *B* - broj razmena *C* - broj poredjenja
Na uobičajnom računaru ukupno vreme rada je reda: $4C + 11B + 35A$.

Teorema

Neka je $H_N = \sum_{k=1}^{N} \frac{1}{k}$*.Prosečna cena Quicksort algoritma za na niz od N različitih brojeva slučajno rasporedjenih je: N particionisanja,*
$2(N+1)(H_{N+1} - 1) \approx 2N\ln(N) - 0.846N$ *poredjenja i*
$(N+1)(H_{N+1} - 5/2)/3 + 1/2 \approx 0.333N\ln(N) - 1.346N$ *razmena.*

Repeat petlja na nivou nižem od programskog jezika:

```
LOOP     INC i, 1
         CMP v, a[i]
         BL LOOP
```

A - broj podela B - broj razmena C - broj poredjenja
Na uobičajnom računaru ukupno vreme rada je reda: $4C + 11B + 35A$.

Teorema

Neka je $H_N = \sum_{k=1}^{N} \frac{1}{k}$.*Prosečna cena Quicksort algoritma za na niz od* N *različitih brojeva slučajno rasporedjenih je:* N *particionisanja,*
$2(N+1)(H_{N+1} - 1) \approx 2N\ln(N) - 0.846N$ *poredjenja i*
$(N+1)(H_{N+1} - 5/2)/3 + 1/2 \approx 0.333N\ln(N) - 1.346N$ *razmena.*

Cena *Quicksort* algoritma: $11.667N\ln(N) - 16.840N$,
tj. izvrši se oko $2N\ln(N)$ poredjenja i $0.667N\ln(N)$ razmena.

Asimptotska aproksimacija

Tačni rezultati nisu poznati, a daleko lakše doći do dovoljno bliskog rezultata

Primer (Asimptotska aproksimacija Ojler-Maklorenovom formulom)

$$H_N = \sum_{k=1}^{N} \frac{1}{k} \approx \int_1^N \frac{1}{x} dx = \ln(N);$$

preciznije:

$$H_N = \ln(N) + \gamma + \frac{1}{2N} + O(\frac{1}{N^2}),\ \textit{gde je}\ \gamma = 0.5572\dots\ \textit{Ojlerova konstanta}$$

Preciznost ove formule može se povećavati tako da ostatak bude reda k, tj. $O(N^{-k})$.

Distribucije

Teorija verovatnoće za razumevanje performansi algoritma sa dodatnim podacima o distribuciji cene Π_N.
Očekivana cena je:

$$\mu = \sum_k k\Pi_{Nk}/\Pi_N.$$

Disperzija se definiše sa:

$$\sigma^2 = \sum_k (k-\mu)^2 \Pi_{Nk}/\Pi_N = \sum_k k^2 \Pi_{Nk}/\Pi_N - \mu^2.$$

Standardno odstupanje σ je kvadratni koren disperzije.

Verovatnosni algoritmi

- Kako odabrati dovoljno formalan model za ulaz koji je moguć u praksi.

Verovatnosni algoritmi

- Kako odabrati dovoljno formalan model za ulaz koji je moguć u praksi.
- Za algoritame za sortiranje dovoljno dobar model je niz nasumično odabranih brojeva.

Verovatnosni algoritmi

- Kako odabrati dovoljno formalan model za ulaz koji je moguć u praksi.
- Za algoritame za sortiranje dovoljno dobar model je niz nasumično odabranih brojeva.
- Algoritmi koji prave izbor ,,nasumičnih" brojeva.

Matematički aparat

- Rekurentne jednačine
  - U tesnoj vezi sa iterativnim i rekurzivnim programima
  - Razumevanje algoritama može izjednačiti sa znanjem da se odredi rekurentna jednačina koja opisuje značajne karakteristike tog algoritma
  - Poznavanje rekurentne jednačine koja odgovara jednom algoritmu moguće je proceniti potrebne zahteve samih aplikacija

Matematički aparat

- Rekurentne jednačine
  - U tesnoj vezi sa iterativnim i rekurzivnim programima
  - Razumevanje algoritama može izjednačiti sa znanjem da se odredi rekurentna jednačina koja opisuje značajne karakteristike tog algoritma
  - Poznavanje rekurentne jednačine koja odgovara jednom algoritmu moguće je proceniti potrebne zahteve samih aplikacija
- Redovi
  - Redovi se koriste kao kombinatorno sredstvo prilikom odredjivanja veličina od interesa, ali i kao analitičko sredstvo za njihovo izračunavanje
  - Razvijen je veliki broj alata za njihovo rešavanje
  - Često je jednostavnije doći do relevantnog reda koji odredjuje veličine od interesa, nego do odgovarajuće rekurentne jednačine

Matematički aparat

- Rekurentne jednačine
  - U tesnoj vezi sa iterativnim i rekurzivnim programima
  - Razumevanje algoritama može izjednačiti sa znanjem da se odredi rekurentna jednačina koja opisuje značajne karakteristike tog algoritma
  - Poznavanje rekurentne jednačine koja odgovara jednom algoritmu moguće je proceniti potrebne zahteve samih aplikacija
- Redovi
  - Redovi se koriste kao kombinatorno sredstvo prilikom odredjivanja veličina od interesa, ali i kao analitičko sredstvo za njihovo izračunavanje
  - Razvijen je veliki broj alata za njihovo rešavanje
  - Često je jednostavnije doći do relevantnog reda koji odredjuje veličine od interesa, nego do odgovarajuće rekurentne jednačine
- Granične vrednosti
  - Rešavanje problema najčešće se svodi na prepoznavanje i kombinovanje dobro poznatih izraza
  - Snažne metode dolaze iz nekoliko osnovnih osobina, naročito iz singularnosti

Drveće

Definicija

Drvo je prazno ili je koreni čvor koje je povezan sa drugim razdvojenim drvećem, koje zovemo i šuma.
Binarno drvo je ili spoljni čvor ili unutrašnji čvor koje je povezan sa parom binarnih drveća koje se nazivaju levo, odnosno desno, poddrvo toga čvora.

Drveće

Definicija

Drvo je prazno ili je koreni čvor koje je povezan sa drugim razdvojenim drvećem, koje zovemo i šuma.
Binarno drvo je ili spoljni čvor ili unutrašnji čvor koje je povezan sa parom binarnih drveća koje se nazivaju levo, odnosno desno, poddrvo toga čvora.

Lema

Broj spoljnih čvorova binarnog drveta je za tačno jedan veći od broja unutrašnjih čvorova.

Dokaz.

Svaki unutrašnji čvor ima tačno dve veze koje izlaze ,,iz'' njega i svaki čvor, osim korena ima vezu ,,ka'' njemu. Sve ovo nam daje:
$2i = i + e - 1$, tj. $i = e - 1$. □

Definicija

Listovi se definišu kao čvorovi bez naslednika, dok se kod binarnog drveća listovi definišu kao unutrašnji čvorovi sa oba spoljašnja naslednika.

Definicija

Listovi se definišu kao čvorovi bez naslednika, dok se kod binarnog drveća listovi definišu kao unutrašnji čvorovi sa oba spoljašnja naslednika.

Definicija

Neka je dato drvo t. Nivo korena je 0, dok je nivo svakog sledbenika $k+1$, gde je k nivo trenutnog čvora.
Broj čvorova nekog drveta t obično se označava sa $|t|$.
Kod binarnog drveta definišu se unutrašnja dužina puta $\pi(t)$, spoljašnja dužina puta $\varepsilon(t)$, kao zbirovi nivoa odgovarajućih čvorova, dok se visina $\eta(t)$ definiše kao maksimalni nivo svih unutrašnjih čvorova.

Definicija

Listovi se definišu kao čvorovi bez naslednika, dok se kod binarnog drveća listovi definišu kao unutrašnji čvorovi sa oba spoljašnja naslednika.

Definicija

Neka je dato drvo t. Nivo korena je 0, dok je nivo svakog sledbenika $k+1$, gde je k nivo trenutnog čvora.
Broj čvorova nekog drveta t obično se označava sa $|t|$.
Kod binarnog drveta definišu se unutrašnja dužina puta $\pi(t)$, spoljašnja dužina puta $\varepsilon(t)$, kao zbirovi nivoa odgovarajućih čvorova, dok se visina $\eta(t)$ definiše kao maksimalni nivo svih unutrašnjih čvorova.

Lema

Za dužine puteva u proizvoljnom binarnom drvetu t važi:
$\varepsilon(t) = \pi(t) + 2|t|$.
Dužina puta i visina svakog nepraznog binarnog drveta zadovoljavaju:
$\frac{\pi(t)}{|t|} \leqslant \eta(t) \leqslant \sqrt{2\pi(t)} + 1$.

Algoritmi i primene

Osnovni algoritam koji koriste drveće je obilazak drveta. Ovo je moguće uraditi na neki od sledećih načina:

- **postorder**: prvo poddrveće, pa koren;
- **preorder**: prvo koren,pa poddrveće;
- kod binarnih **redom**: levo poddrvo, koren, desno poddrvo.

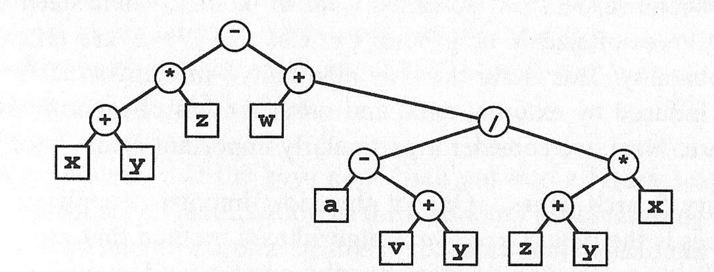
Algoritmi i primene

Osnovni algoritam koji koriste drveće je obilazak drveta. Ovo je moguće uraditi na neki od sledećih načina:

- **postorder**: prvo poddrveće, pa koren;
- **preorder**: prvo koren,pa poddrveće;
- kod binarnih **redom**: levo poddrvo, koren, desno poddrvo.

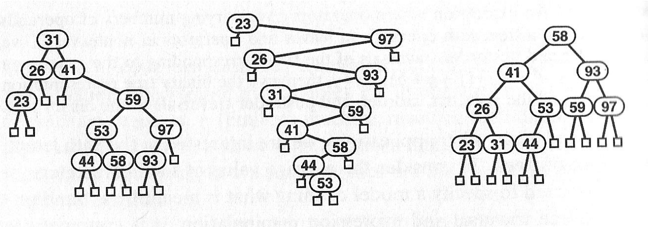
Strukturne prezentacije aritmetičkih izraza:

$$(x + y)z - (w + ((a - (v + y)/(z + y)x)))$$



Binarno drvo pretrage - BDP

Osnovno svojstvo BDP je da levo poddrvo ne sadrži vrednosti koje su veće od vrednosti u čvoru, dok vrednosti u desnom podrvetu nisu manje od vrednosti u čvoru.



Teorema (Troškovi konstrukcije BDP)

Prosečan broj poredjenja u procesu konstrukcije binarnog drveta za pretragu sa N nasumično odabranih elemenata od praznog drveta (tj. dužina prosečnog unutrašnjeg puta) je:
$2(N+1)(H_{N+1}-1)-2N \approx 1.386N\lg N - 2.846N$ *sa asimptotom* $(7-2\pi^2/3)N^2$.

Teorema (Troškovi konstrukcije BDP)

Prosečan broj poredjenja u procesu konstrukcije binarnog drveta za pretragu sa N nasumično odabranih elemenata od praznog drveta (tj. dužina prosečnog unutrašnjeg puta) je:
$2(N+1)(H_{N+1}-1)-2N \approx 1.386N\lg N - 2.846N$ *sa asimptotom* $(7-2\pi^2/3)N^2$.

Teorema (Troškovi pretrage BDP)

U proizvoljnom binarnom drvetu sa N čvorova prosečna cena uspešne pretrage je $2H_N - 3 - 2H_N/N$*, dok je prosečna cena neuspešne pretrage* $2H_{N+1}-2$ *i u oba slučaja asimptota je* $2H_N + O(1)$

Teorema (Troškovi konstrukcije BDP)

Prosečan broj poredjenja u procesu konstrukcije binarnog drveta za pretragu sa N nasumično odabranih elemenata od praznog drveta (tj. dužina prosečnog unutrašnjeg puta) je:
$2(N+1)(H_{N+1}-1)-2N \approx 1.386N\lg N - 2.846N$ *sa asimptotom* $(7-2\pi^2/3)N^2$.

Teorema (Troškovi pretrage BDP)

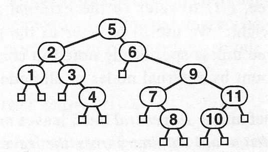
U proizvoljnom binarnom drvetu sa N čvorova prosečna cena uspešne pretrage je $2H_N - 3 - 2H_N/N$, *dok je prosečna cena neuspešne pretrage* $2H_{N+1}-2$ *i u oba slučaja asimptota je* $2H_N + O(1)$

Teorema (Prosečna i očekivana visina)

Prosečna visina proizvoljnog binarnog drveta sa N čvorova je $2\sqrt{\pi N} + O(N^{\frac{1}{4}}+\varepsilon), \varepsilon > 0$. *Očekivana visina BDP od N čvorova je reda* $c\log N$, *gde je* $c \approx 4.3407\ldots$ *tj,* $(c>2, c\ln(2e/c)=1)$.

Reprezentacije

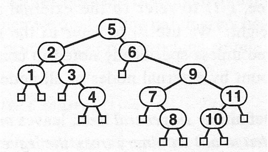
- **Sistem roditeljsva**:



  - *preorder*: (5 (2 (1)(3 (4))) (6 (9 (7 (8))(11 (10)))))
  - *postorder*:(((1) ((4) 3) 2) ((((8) 7) ((10) 11) 9) 6) 5)

Reprezentacije

- **Sistem roditeljsva**:



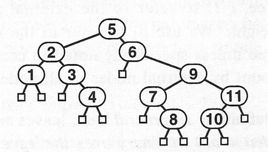
- *preorder*: (5 (2 (1)(3 (4))) (6 (9 (7 (8))(11 (10)))))
- *postorder*:(((1) ((4) 3) 2) ((((8) 7) ((10) 11) 9) 6) 5)

$$(x + y)z - (w + ((a - (v + y)/(z + y)x)))$$

- prefiksno: $- * +xyz + w/ - a + vy * +zyx$
- postfiksno:$xy + z * wavy + -zy + x * / + -$

Reprezentacije

- **Sistem roditeljsva**:



  - *preorder*: (5 (2 (1)(3 (4))) (6 (9 (7 (8))(11 (10)))))
  - *postorder*:(((1) ((4) 3) 2) ((((8) 7) ((10) 11) 9) 6) 5)

$$(x+y)z-(w+((a-(v+y)/(z+y)x)))$$

  - prefiksno: $-*+xyz+w/-a+vy*+zyx$
  - postfiksno:$xy+z*wavy+-zy+x*/+-$

- **,,Kockarov gubitnički" niz i lattice putanja**:

$$++\,+\,-\,-\,-\,+\,-\,+\,+\,-\,+\,-\,-\,+\,+\,-\,-\,-$$

Permutacije

,,Preuredjivanje" niza brojeva od 1 do N

Definicija

Inverzija je par $i < j$ sa $p_i > p_j$. Ukoliko je q_j broj onih $i < j$ sa $p_i > p_j$, tada se $q_1 \dots q_N$ zove tabela inverzije permutacije $p_1 p_2 \dots p_N$. Oznaka $\sigma(p)$ koristi se da se prikaže broj inverzja u permutaciji p, tj. sumu članova tabele inverzija.

Permutacije

,,Preuredjivanje" niza brojeva od 1 do N

Definicija

Inverzija je par $i < j$ *sa* $p_i > p_j$. *Ukoliko je* q_j *broj onih* $i < j$ *sa* $p_i > p_j$, *tada se* $q_1 \dots q_N$ *zove tabela inverzije permutacije* $p_1 p_2 \dots p_N$. *Oznaka* $\sigma(p)$ *koristi se da se prikaže broj inverzja u permutaciji* p, *tj. sumu članova tabele inverzija.*
Levo-na-desni maksimum je indeks i *za koji važi da je* $p_j < p_i$ *za sve* $j < i$, *u oznaci* $\lambda(p)$.

Permutacije

,,Preuredjivanje" niza brojeva od 1 do N

Definicija

Inverzija je par $i < j$ sa $p_i > p_j$. Ukoliko je q_j broj onih $i < j$ sa $p_i > p_j$, tada se $q_1 \dots q_N$ zove tabela inverzije permutacije $p_1 p_2 \dots p_N$. Oznaka $\sigma(p)$ koristi se da se prikaže broj inverzja u permutaciji p, tj. sumu članova tabele inverzija.
Levo-na-desni maksimum je indeks i za koji važi da je $p_j < p_i$ za sve $j < i$, u oznaci $\lambda(p)$.
Ciklus je niz indeksa $i_1 \dots i_t$ za koje važi: $p_{i_1} = i_2, p_{i_2} = i_3, \dots,$ $p_{i_t} = i_1$, u oznaci $(i_1 \dots i_t)$.

Permutacije

,,Preuredjivanje" niza brojeva od 1 do N

Definicija

Inverzija je par $i < j$ sa $p_i > p_j$. Ukoliko je q_j broj onih $i < j$ sa $p_i > p_j$, tada se $q_1 \dots q_N$ zove tabela inverzije permutacije $p_1 p_2 \dots p_N$. Oznaka $\sigma(p)$ koristi se da se prikaže broj inverzja u permutaciji p, tj. sumu članova tabele inverzija.

Levo-na-desni maksimum je indeks i za koji važi da je $p_j < p_i$ za sve $j < i$, u oznaci $\lambda(p)$.

Ciklus je niz indeksa $i_1 \dots i_t$ za koje važi: $p_{i_1} = i_2, p_{i_2} = i_3, \dots,$ $p_{i_t} = i_1$, u oznaci $(i_1 \dots i_t)$.

Kanonska reprezentacija permutacije je ona u kojoj se ciklusi redjaju prema rastućem redosledu najvećih elemenata svakog od ciklusa.

Permutacije

,,Preuredjivanje" niza brojeva od 1 do N

Definicija

Inverzija je par $i < j$ sa $p_i > p_j$. Ukoliko je q_j broj onih $i < j$ sa $p_i > p_j$, tada se $q_1 \dots q_N$ zove tabela inverzije permutacije $p_1 p_2 \dots p_N$. Oznaka $\sigma(p)$ koristi se da se prikaže broj inverzja u permutaciji p, tj. sumu članova tabele inverzija.

Levo-na-desni maksimum je indeks i za koji važi da je $p_j < p_i$ za sve $j < i$, u oznaci $\lambda(p)$.

Ciklus je niz indeksa $i_1 \dots i_t$ za koje važi: $p_{i_1} = i_2, p_{i_2} = i_3, \dots,$ $p_{i_t} = i_1$, u oznaci $(i_1 \dots i_t)$.

Kanonska reprezentacija permutacije je ona u kojoj se ciklusi redjaju prema rastućem redosledu najvećih elemenata svakog od ciklusa.

Inverzna premutacija $q_1 q_2 \dots q_N$ permutaciji $p_1 p_2 \dots p_N$ je permutacija za koju važi: $q_{p_i} = p_{q_i} = i$. Involucija je permutacija koja je inverzija samoj sebi, tj. $p_{p_i} = i$.

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Permutacija	9	14	4	1	12	2	10	13	5	6	11	3	8	15	7
Tab. inverzije	0	0	2	3	1	4	2	1	5	5	3	9	6	0	8
Ciklusi	(1	9	5	12	3	4)	(2	14	15	7	10	6)	(8	13)	(11)
Kanonska forma	11	12	3	4	1	9	5	13	8	15	7	10	6	2	14
Inv. perutacija	4	6	12	3	9	10	15	13	1	7	11	5	8	2	14
Involucija	9	2	12	4	7	10	5	13	1	6	11	3	8	15	14
Cikl. involucije	(1	9)	(2)	(3	12)	(4)	(5	7)	(6	10)	(8	13)	(11)	(14	15)

Indeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Permutacija	9	14	4	1	12	2	10	13	5	6	11	3	8	15	7
Tab. inverzije	0	0	2	3	1	4	2	1	5	5	3	9	6	0	8
Ciklusi	(1	9	5	12	3	4)	(2	14	15	7	10	6)	(8	13)	(11)
Kanonska forma	11	12	3	4	1	9	5	13	8	15	7	10	6	2	14
Inv. perutacija	4	6	12	3	9	10	15	13	1	7	11	5	8	2	14
Involucija	9	2	12	4	7	10	5	13	1	6	11	3	8	15	14
Cikl. involucije	(1	9)	(2)	(3	12)	(4)	(5	7)	(6	10)	(8	13)	(11)	(14	15)

Teorema

Prosečan broj umetanja pri sortiranju niza od N proizvoljnih elemenata je reda $\frac{N^2}{4}$ *sa prosečnim brojem premeštanja reda* $\frac{N^2}{4}$

Algoritmi za sortiranje nizova - Shellsort

```
for h := h[t], h[t-1], ... , h[1] do
    for i := h+1 to N do
        begin
        v := a[i]; j := i-1;
        while a[j] > v do
            begin a[j+1] := a[j]; j := j-1 end;
        a[j+1] := v;
        end
```

Algoritmi za sortiranje nizova - Shellsort

```
for h := h[t], h[t-1], ... , h[1] do
    for i := h+1 to N do
        begin
        v := a[i]; j := i-1;
        while a[j] > v do
            begin a[j+1] := a[j]; j := j-1 end;
        a[j+1] := v;
        end
```

Teorema

Broj poredjenja koji se koristi pri $(2,1)$ Shellsort-*u na fajlu veličine* N *je* $\frac{N^2}{8} + \sqrt{\frac{\pi}{128}} N^{3/2} + O(N)$.

Algoritmi za sortiranje nizova - Selectionsort

```
for j := 1 to N do
    begin
    v := infinity; k := j;
    for i := j to N do
        if a[i] < v then
           begin v := a[i]; k := i; end
    t := a[j]; a[j] := a[k]; a[k] := t;
    end
```

Algoritmi za sortiranje nizova - Selectionsort

```
for j := 1 to N do
    begin
    v := infinity; k := j;
    for i := j to N do
        if a[i] < v then
           begin v := a[i]; k := i; end
    t := a[j]; a[j] := a[k]; a[k] := t;
    end
```

Teorema

Prosečan broj poredjenja pri Selectionsort *algroritmu za prozivoljan niz dužine N je reda $\frac{N^2}{2}$, prosečan broj čuvanih vrednosti je reda $N \ln N$ i prosečan broj razmena je reda N.*

Algoritmi za sortiranje nizova - Bubblesort

```
i := N;
repeat
    t := a[1]; i := i-1;
    for j := 2 to i do
        if a[j-1] > a[j] then
           begin t := a[j-1]; a[j-1] := a[j]; a[j] := t end
until (t = a[1]);
```

Algoritmi za sortiranje nizova - Bubblesort

```
i := N;
repeat
    t := a[1]; i := i-1;
    for j := 2 to i do
        if a[j-1] > a[j] then
           begin t := a[j-1]; a[j-1] := a[j]; a[j] := t end
until (t = a[1]);
```

Teorema

Najveći element u tabeli inverzije proizvoljne permutacije ima vrednost reda $N - \sqrt{\pi N/2}$*, što za posledicu daje da je broj poredjenja kod* Bubblesort *algoritma reda* $N^2/2$*, dok je broj razmena reda* $N^2/2$ *u* $N - \sqrt{\pi N/2}$ *prolazaka.*

Stringovi

Definicija

Stringovi su niske karaktera ili slova iz poznatog alfabeta.

Stringovi

Definicija

Stringovi su niske karaktera ili slova iz poznatog alfabeta.

```
procedure stringsearch;
    var i, j: integer;
    begin
    for i := 1 to N do
        begin
        j :=0;
        repeat j := j+1 until a[i+j+1] <> p[j];
        if j = k+1 then <<match found>>
        end
    end
```

Osnovne osobine

Teorema

Očekivani broj poredjenja prilikom osnovnog algoritma pretrage svih pojavljivanja uzorka dužine k u bitstringu dužine N je $N(2 - 2^{-k}) + O(1)$.

Osnovne osobine

Teorema

Očekivani broj poredjenja prilikom osnovnog algoritma pretrage svih pojavljivanja uzorka dužine k u bitstringu dužine N je $N(2 - 2^{-k}) + O(1)$.

Teorema

Prosečan broj ispitanih bitova prilikom pretrage za najdužim poklapanjem uzorka dužine k u bitstringu dužine N je reda $2N$.

Osnovne osobine

Teorema

Broj bitstringova bez nizova sa k uzastopnih nula dat je redom:

$$S_k(z) = \frac{1 - z^k}{1 - 2z + z^{k+1}}.$$

Osnovne osobine

Teorema

Broj bitstringova bez nizova sa k uzastopnih nula dat je redom:

$$S_k(z) = \frac{1 - z^k}{1 - 2z + z^{k+1}}.$$

Posledica

Prosečna pozicija kraja prve pojave k uzastopnih nula i proizvoljnom bitstringu je $S_k(\frac{1}{2}) = 2^{k+1} - 2$. Najduži niz uzastopnih nula u proizvoljnom bitstringu dužine N je

$$\frac{1}{2^N}[z^N]\sum_{k \geqslant 0}\left(\frac{1}{1 - 2z} - \frac{1 - z^k}{1 - 2z + z^{k+1}}\right).$$

KMP algoritam

- Optimalni algoritam za pretragu stringova razvili su Knut, Moris i Prat 1977. godine

KMP algoritam

- Optimalni algoritam za pretragu stringova razvili su Knut, Moris i Prat 1977. godine
- Ideja je izgradnja konačnog determinisanog automata prema uzorku

KMP algoritam

- Optimalni algoritam za pretragu stringova razvili su Knut, Moris i Prat 1977. godine
- Ideja je izgradnja konačnog determinisanog automata prema uzorku

```
procedure fsasearch;
    begin
    state := 0;
    for i :=  1 to N do
        begin
        state := next (state, a [i]);
        if (state = k) then << match found >>
        end;
    end
```

KMP algoritam

- Optimalni algoritam za pretragu stringova razvili su Knut, Moris i Prat 1977. godine
- Ideja je izgradnja konačnog determinisanog automata prema uzorku

```
procedure fsasearch;
    begin
    state := 0;
    for i :=  1 to N do
        begin
        state := next (state, a [i]);
        if (state = k) then << match found >>
        end;
    end
```

Teorema

KMP algoritam ima N poredjenja prilikom pretrage uzorka veličine k u bitstringu dužine N.

Trie

Definicija

Neka je dat skup $\mathcal{B}$ prefiksno slobodnih bitstringova. Odgovarajući trie je binarno drvo dato sledećom rekurzivnom definicijom:

- *ukoliko je $\mathcal{B}$ prazan, trie je null i reprezentovan je void spoljašnjim čvorom*

Trie

Definicija

Neka je dat skup $\mathcal{B}$ prefiksno slobodnih bitstringova. Odgovarajući trie je binarno drvo dato sledećom rekurzivnom definicijom:

- *ukoliko je $\mathcal{B}$ prazan, trie je null i reprezentovan je void spoljašnjim čvorom*
- *ukoliko je $|\mathcal{B}| = 1$, trie se sastoji od jednog spoljašnjeg čvora koji odgovara tom bitstringu.*

Trie

Definicija

Neka je dat skup $\mathcal{B}$ prefiksno slobodnih bitstringova. Odgovarajući trie je binarno drvo dato sledećom rekurzivnom definicijom:

- *ukoliko je $\mathcal{B}$ prazan, trie je null i reprezentovan je void spoljašnjim čvorom*
- *ukoliko je $|\mathcal{B}| = 1$, trie se sastoji od jednog spoljašnjeg čvora koji odgovara tom bitstringu.*
- *$\mathcal{B}_0$ (tj. $\mathcal{B}_1$) definišemo kao skup bitstringova koji počinju sa 0, (tj. 1) i uklanjanjem početnog bita iz svakog od njih. Tada je trie za $\mathcal{B}$ unutrašnji čvor koji je povezan sa trie za $\mathcal{B}_0$ sleva i trie za $\mathcal{B}_1$ zdesna.*

Trie

Definicija

Neka je dat skup $\mathcal{B}$ prefiksno slobodnih bitstringova. Odgovarajući trie je binarno drvo dato sledećom rekurzivnom definicijom:

- *ukoliko je $\mathcal{B}$ prazan, trie je null i reprezentovan je void spoljašnjim čvorom*
- *ukoliko je $|\mathcal{B}| = 1$, trie se sastoji od jednog spoljašnjeg čvora koji odgovara tom bitstringu.*
- *$\mathcal{B}_0$ (tj. $\mathcal{B}_1$) definišemo kao skup bitstringova koji počinju sa 0, (tj. 1) i uklanjanjem početnog bita iz svakog od njih. Tada je trie za $\mathcal{B}$ unutrašnji čvor koji je povezan sa trie za $\mathcal{B}_0$ sleva i trie za $\mathcal{B}_1$ zdesna.*

Ova struktura nije jednoznačno odredjena - većem broju skupova može odgovarati jedno drvo.

```
function  search(v: integer; x: link): link;
    begin
    if x = NILL <<no match >> else
    if prefix(v, x^.key) << match found >> else
    if firstbit(v) = 0 then search(otherbits(v), x^.left)
                       else search(otherbits(v), x^.right)
    end
```

- `prefix`: vraća *true* ako je prvi argument prefiks drugog
- `firstbit`: prvi bit argumenta
- `otherbits`: uklanja prvi bit.

```
function  search(v: integer; x: link): link;
    begin
    if x = NILL <<no match >> else
    if prefix(v, x^.key) << match found >> else
    if firstbit(v) = 0 then search(otherbits(v), x^.left)
                       else search(otherbits(v), x^.right)
    end
```

- `prefix`: vraća *true* ako je prvi argument prefiks drugog
- `firstbit`: prvi bit argumenta
- `otherbits`: uklanja prvi bit.

Teorema

Broj poredjenja prilikom sortiranja niza od N proizvoljnih bitstringova radix-razmenom je reda $N \log N$.

Reči i mape

Reči i mape

```
procedure insert(v: integer);
    var x: link;
    begin
    new(x); x^.next = table[hash(v)];
    table[hash(v)] = x;
    end;
```

Reči i mape

```
procedure insert(v: integer);
    var x: link;
    begin
    new(x); x^.next = table[hash(v)];
    table[hash(v)] = x;
    end;

function search(v: integer): link;
    var t: link;
    begin
    t := table[hash(v)]; z^.key = v;
    while t^.next^.key <> v do t := t^.next;
    if t = z then t := NIL;
    search := t;
    end;
```

Definicija

M-reč dužine N je funkcija f mapiranja celih brojeva iz intervala $[1..N]$ u interval $[1..M]$.

Definicija

M-reč dužine N je funkcija f mapiranja celih brojeva iz intervala $[1..N]$ u interval $[1..M]$.

Model: N kuglica slučajno se rasporedjuje u M kutija

Definicija

M-reč dužine N je funkcija f mapiranja celih brojeva iz intervala $[1..N]$ u interval $[1..M]$.

Model: N kuglica slučajno se rasporedjuje u M kutija
Najvažniji problem: odrediti optimalno $\alpha \in [0,1]$, tako da je $M = \alpha N$
Pitanja od interesa:

- koja je verovatnoća da nijedna kutija ne bude prazna?
- koja je verovatnoća da nijedna kutija ne sadrži više od jedne kuglice?
- koliko je praznih kutija?
- koliko je max/min broj kuglica koji sadrži pojedinačna kutija?

Dva problema

Rodjendanski problem

Koliko se ljudi može okupiti u jednoj grupi tako da se može očekivati da nemamo dvoje koji su rodjeni istog datuma.

Dva problema

Rodjendanski problem

Koliko se ljudi može okupiti u jednoj grupi tako da se može očekivati da nemamo dvoje koji su rodjeni istog datuma.

Problem skupljača kupona

Ukoliko svaka kutija sadrži neki podskup skupa od M kupona, koliko je prosečno kutija potrebno kupiti da bi sigurno imali sve kupone.

Dva problema

Rodjendanski problem

Koliko se ljudi može okupiti u jednoj grupi tako da se može očekivati da nemamo dvoje koji su rodjeni istog datuma.

Problem skupljača kupona

Ukoliko svaka kutija sadrži neki podskup skupa od M kupona, koliko je prosečno kutija potrebno kupiti da bi sigurno imali sve kupone.

Rešenja

Očekivani broj kuglica ubačenih pre prve kolizije je reda $\sqrt{\frac{\pi M}{2}} + \frac{2}{3}$

Prosečan broj kuglica koje potrebno ubaciti u M kutija da se one popunile je reda $M^2\pi^2/6$

Svojstva

Teorema

Prosečan broj kutija sa k kuglica, pri slučajnoj distribuciji N kuglica u M kutija je $M\binom{N}{k}\left(\frac{1}{M}\right)^{k}\left(1-\frac{1}{M}\right)^{N-k}$.

Normalna aproksimacija - za fiksirano M i $k = N/M + x\sqrt{N/M}, x = O(1)$
Puasonova aproksimacija - za $N/M = \alpha > 0$ fiksirano i $k = O(1)$

Svojstva

Teorema

Prosečan broj kutija sa k kuglica, pri slučajnoj distribuciji N kuglica u M kutija je $M\binom{N}{k}\left(\frac{1}{M}\right)^k\left(1-\frac{1}{M}\right)^{N-k}$.

Normalna aproksimacija - za fiksirano M i $k = N/M + x\sqrt{N/M}, x = O(1)$
Puasonova aproksimacija - za $N/M = \alpha > 0$ fiksirano i $k = O(1)$

Posledica

Za konstantno $N/M = \alpha$, prosečan broj praznih kutija asimptotski teži $Me^{-\alpha}$.

Posledica

Prosečan broj kuglica po kutiji je N/M, sa standardnim odstupanjem $\sqrt{N/M - N/M^2}$.

Teorema

Korišćenjem tabele veličine M za N ključeva, heširanje sa razdvojenim lancima u prosečnom slučaju zahteva N/M pokušaja pri neuspešnoj pretrazi i $(N+1)/2M$ pokušaja pri uspešnoj pretrazi.

Teorema

Korišćenjem tabele veličine M od N ključeva, potreban broj uspešnih i neuspešnih pretraga u prosečnom slučaju uniformnim heširanjem su respektivno $\frac{M+1}{M-N+1}$ i $\frac{M+1}{N}(H_{M+1} - H_{M-N+1})$.

Mape

Definicija

Mapa f *je f-ja koja preslikava skup celih brojeva iz intervala* $[1..N]$ *na sebe.*

Mape

Definicija

Mapa f je f-ja koja preslikava skup celih brojeva iz intervala $[1..N]$ *na sebe. Dužina puta (*ρ *dužina) indeksa k u mapi f je broj medjusobno različitih brojeva dobijenim iteriranjem:* $f(k), f(f(k)), \ldots$ *Dužina ciklusa indeksa k u mapi f je dužina ciklusa koja se dostiže iteriranjem, dok je dužina repa indeksa k u mapi f razlika* ρ *dužine i dužine ciklusa, tj. broj koraka koliko je potrebno da bi se stiglo do punog ciklusa.* ρ *dužina mape f je suma* ρ *dužina za sve indekse k iz mape* f*. Dužina grana u mapi f je zbir dužina repa za sve indekse k iz mape* f*.*

Mape

Definicija

Mapa f je f-ja koja preslikava skup celih brojeva iz intervala $[1..N]$ na sebe. Dužina puta (ρ dužina) indeksa k u mapi f je broj medjusobno različitih brojeva dobijenim iteriranjem: $f(k), f(f(k)), \ldots$ Dužina ciklusa indeksa k u mapi f je dužina ciklusa koja se dostiže iteriranjem, dok je dužina repa indeksa k u mapi f razlika ρ dužine i dužine ciklusa, tj. broj koraka koliko je potrebno da bi se stiglo do punog ciklusa. ρ dužina mape f je suma ρ dužina za sve indekse k iz mape f. Dužina grana u mapi f je zbir dužina repa za sve indekse k iz mape f.

Teorema

*Prosečna ρ dužina proizvoljne tačke u proizvoljnoj mapi je reda $\sqrt{\pi N/2}$.
Prosečna ρ dužina proizvoljne mape je reda $N\sqrt{\pi N/2}$.
Slučajna N-mapa u prosečnom slučaju ima reda $\frac{1}{2} \ln N$ komponenti i $\sqrt{\pi N}$ čvorova u ciklusu.*

Hvala na pažnji
Pitanja?

Power by: LaTeX- GNU Free Documentation License