Algoritmy a dátové štruktúry

Triedenia (2)

26.10.2010

Quicksort -doplnenie

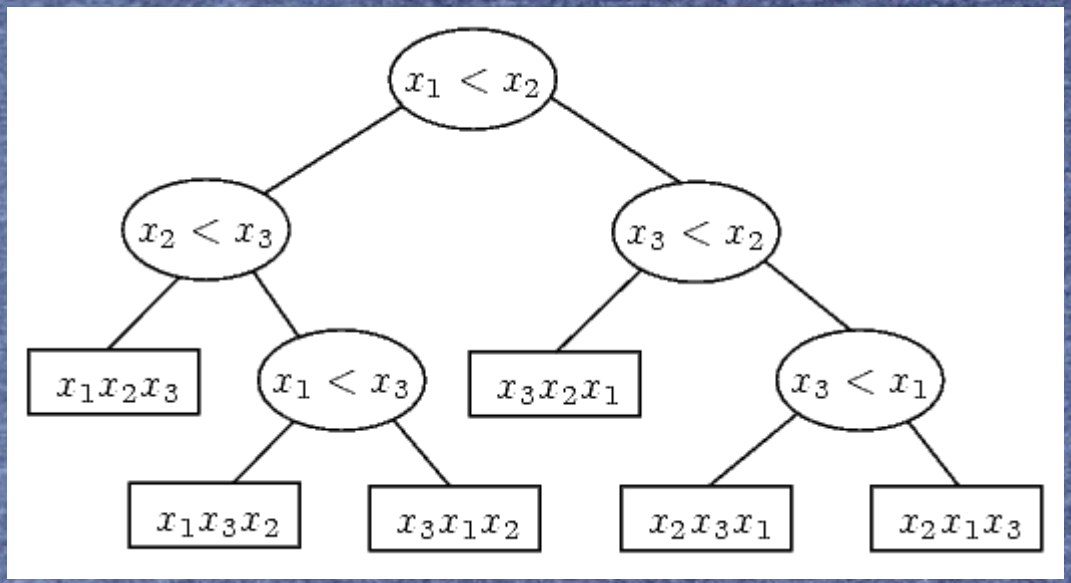
- Drozdeľovanie prvkov v Quicksorte je možné vykonávať aj nie priamo v poli – do š nových polí, ktoré budeme triediť
- To čo bolo minule sa nazýva občas In-place Quicksort
- Hľadanie vhodného pivota je možné v čase O(n) ale aj náhodný výber nebude tak zlý (asi polovica prvkov je vhodný pivot)

Quicksort -doplnenie

```
QuickSort(A[1..N]; l,r){
  i:=l; j:=r;  k:=A[(i+j) div 2];
  repeat
    while A[i]<k do i:=i+1;
    while A[j]>k do j:=j-1;
    if i<=j then { A[i]↔A[j]; i:=i++; j--; }
  until i >= j;
  if j>l then QuickSort(A, l, j);
  if i<r then QuickSort(A, i, r);
end;
```

Dolný odhad zložitosti

- Pre triedenia porovnaním
- Obmena triediaceho algoritmu – najprv všetky porovnania a z toho usúdi ako je usporiadané
- Reprezentácia rozhodovacím stromom



Dolný odhad zložitosti

- Počet listov stromu = počet možných poradí prvků = N!.
- Rôzným poradiam zodpovedajú rôzne listy
- Každé poradie prvkov určuje cestu do listu
- Na nultej úrovni je jediný vrchol, na každej ďalšej sa počet nanajvýš zdvojnásobí.
- Takže na i-tej úrovni je nejviac 2^i vrcholov a preto listov je najviac 2^h (niektoré listy môžu byť aj výššie, ale za každý iste chýba jeden vrchol na h-tej úrovni)

Dolný odhad zložitosti

- $2^h \geq$ počet listov $\geq N!$
- $h \geq \log_2 (N!)$
- $n! = n*(n-1)*\ldots *(n/2) *\ldots \geq (n/2)^{(n/2)}$
- $h \geq \log2 (N!) \geq \log2 ((N/2)N/2) =$
  $=N/2.\log_2(N/2) =1/2*N(\log_2 N-1) \geq 1/4*N \log_2 N.$

Lineárne triedenia

- Countsort
- Málo možných hodnôt
- Časová zložitosť lineárná od N
- Pozor! Pripočítať veľkosť intervalu

  tj. O(N+K) kde

  K=H-D+1

- D = 1; H = 10;
- CountSort(A);
- {int C[D..H],i,j,k;
- for (i=D; i<=H; i++) C[i]=0;
- for (i=0; i<N; i++) C[A[i]]++;
- k=0;
- for (i=D; i<=H; i++)
- for (j=0; j<C[i]; j++){
- A[k]:=i; k++;
- }
- }

Lineárne triedenia

- Bucketsort
- Prvky ukladáme do krabičiek podľa ich kľúča – prvok (k,x) uložíme do krabičky B[k]
- Časová zložitosť lineárná od N
- Pozor! Pripočítať počet krabičiek

  tj. O(N+M)

```
BucketSort(S){
for (i=0; i<M; i++) B[i]=null;
while not S.isEmpty(){
  (k,o)=S.removeFirst();
  B[k].insertLast((k,o));}
for (i=0; i<M; i++)
  while not B.isEmpty(){
    (k,o)=B[i].removeFirst();
    S.insertLast((k,o));}
}
```

Lineárne triedenia

- Radixsort
- Čísla v sústave desiatkovej/binárnej
- Číslo vyzerá $x_1 x_2 .. x_d$
- Využíva Buckesort
- Časová zložitosť lineárná od N
- ??? Kde je problém

```
RadixSort(A){
for (i=d; i>0; i--)
 Bucketsort(S,N,i);
}
```

Cvičenia

- Lexikografický sort
- Zlý príklad QuickSearch hľadania mediána
-
- Rozhodovací strom pre Bubblesort (N=3, N=4)