



Vnitřní třídění

Zadání: Uspořádejte pole délky N podle hodnot prvků

Měřítko efektivity:

- * počet porovnání
- * počet přesunů

Třídění přímým výběrem

Algoritmus:

1. Vyber prvek s nejmenší hodnotou (nejmenším klíčem)
2. Vyměň ho s prvkem na prvním místě pole
3. Totéž opakuj pro zbývající prvky pole

Třídění přímým vkládáním

Algoritmus:

Pole rozdělíme na dvě části - zdrojová a cílová (setříděná).

Na začátku cílová část je $a[1]$, zdrojová $a[2]..a[N]$

Krok - rozšíření cílové části $a[1]..a[i-1]$ o jeden prvek:

- zjistí, kam patří prvek $a[i]$
- zařad' ho tam

(tj. případně odsuň prvky cílové (setříděné) části,
které budou za ním)

Bublínkové třídění

Algoritmus:

1. Jsou-li v poli dva sousední prvky ve špatném pořadí, vyměň je.
2. Opakuj krok 1.
 - N-krát
 - (N-1)-krát
 - , dokud je co vyměňovat.

Třídění přetřásáním

Algoritmus:

Stejný jako u bublinkového třídění, jen jiný způsob vyhledávání chybných dvojic.

Další zlepšení Bublínkového třídění:

hranice setříděných okrajů...

Shellovo třídění

Nejdříve porovnávat a vyměňovat vzdálené prvky,
pak zmenšovat vzdálenost až k 1.

Stromové třídění

Idea: Získávat informace tak, aby se nevztahovaly všechny k jednomu prvku.

Algoritmus:

1. Sestrojit strom => nejmenší prvek
2. Vyloučit nejmenší prvek a obnovit strom.

Třídění haldou

Cíl: Nepotřebovat $2N-1$ paměťových jednotek

Definice HALDA:

!! POZOR! Zkouším !!

- binární strom
- všechny hladiny jsou zcela zaplněny, až na poslední, která je zaplněná zleva
- pro každý uzel: hodnota \leq hodnota v synovském uzlu

Třídění haldou:

- ze všech prvků vytvoř haldu
- dokud není halda prázdná, odeber prvek z kořene

Uložení haldy v poli...

Vytvoření haldy I.

```
procedure VytvorHaldu( L,R: index );
var i,j: index; x: Prvek;
begin
  i := L; j := 2*i; x := a[i]; { hodnota otce }
  while j <= R do { levý syn je ještě v haldě }
  begin
    if j+1 <= R then { pravý syn také... }
      if a[j] > a[j+1] then
        { ...a je menší! }
          j := j+1;
    if x <= a[j] then break; { OK }
    a[i] := a[j]; i := j; j := 2*i { vyměním }
  end;
  a[i] := x { až teď zapíšeme hodnotu otce }
end;
```

Vytvoření haldy II.

```
L := n div 2+1;  
while L > 1 do  
begin  
    Dec( L );  
    VytvorHaldu( L,n )  
end
```

Třídění sléváním

Idea:

1. Pomocí N porovnání dovedeme spojit dvě setříděné posloupnosti po $N/2$ do setříděné posloupnosti (délky N)
2. 1-prvková posloupnost JE setříděná.

Algoritmus:

Posloupnost rozdělíme na dvě části, ty setřídíme*)
a potom slijeme.

*) stejným algoritmem

Quicksort

Idea: Posloupnost rozdělíme na části, které setřídíme tímž algoritmem. Jednoprvková posloupnost je setříděná.

Algoritmus na setřídění úseku pole:

- menší prvky dát do levé části
- větší prvky dát do pravé části
- setříd' levou část (není-li prázdná)
- setříd' pravou část (není-li prázdná)

Co znamená „menší/větší prvky“?

menší/větší než určená hodnota (PIVOT)

metoda „Rozděl a panuj“ („Divide & Impera“)

Quicksort - kód

```
procedure QuickSort( zac,kon: index );
var  M,pom: Hodnota;
begin
  x := zac; y := kon; M := ....(pivot)
  repeat
    while A[x] < M do Inc( x );
    while A[y] > M do Dec( y );
    if x<=y then
      begin
        pom := A[x]; A[x] := A[y]; A[y] := pom;
        Inc( x ); Dec( y )
      end
  until x > y;
  if zac < y then QuickSort( zac,y );
  if x < kon then QuickSort( x, kon )
end;
```

QuickSort III. - složitost

Volba pivota

Složitost v nejhorším případě

Volba pivota prakticky

Rozděl a panuj (Divide & Impera)

Řešení úlohy tím,
že ji rozdělíme na stejné úlohy menšího rozměru
a ty zase dělíme... až k úlohám, které mají snadné řešení.

Příklady

Quicksort

Třídění sléváním

výpočet hodnoty výrazu rozdělením na podvýrazy

(rychlé) násobení matic

(rychlá) Fourierova transformace

Hledání k-tého nej...šího prvku z N

- postupným vybíráním minima/maxima
- pomocí haldy
- lineární algoritmus

. Složitost úlohy

- . je nejmenší složitost algoritmu ze všech algoritmů řešících tuto úlohu

Složitost úlohy vnitřního třídění založeného na porovnávání dvou prvků je $O(N \times \log N)$.

- Přihrádkové třídění (bucket sort)
- ...s více průchody (radix sort).

