Řadící algoritmy II.

Metodický koncept k efektivní podpoře klíčových odborných kompetencí s využitím cizího jazyka ATCZ62 - CLIL jako výuková strategie na vysoké škole

Interreg
EVROPSKÁ UNIE
Rakousko-Česká republika
Evropský fond pro regionální rozvoj

Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Merge sort

- metoda Rozděl a panuj
- Nejhorší i průměrná časová složitost *O(N*log*N)*
- Potřebuje navíc pole o velikosti *N*
- Stabilní, paralelizovatelný
- Postup:
  1. Rozdělí neseřazenou množinu dat na dvě podmnožiny o přibližně stejné velikosti.
  2. Seřadí obě podmnožiny.
  3. Spojí seřazené podmnožiny do jedné seřazené množiny.

Merge sort

- Algoritmus

```
mergesort(m)
        var list left, right
        if length(m) ≤ 1
                return m
        else
                middle = length(m) / 2
                for each x in m up to middle
                        add x to left
                for each x in m after middle
                        add x to right
                left = mergesort(left)
                right = mergesort(right)
                result = merge(left, right)
                return result
```

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Quicksort

- Jeden z nejrychlejších běžných algoritmů řazení založený na porovnávání prvků
- Časová složitost $O(N\log N) – O(N^2)$
- Metoda Rozděl a panuj
- Rekurzivní algoritmus
- Postup:
  - výběr pivot – rozdělení posloupnosti na dvě části – větší a menší než pivot
  - Seřaď obě části stejným způsobem

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Quicksort

- Volba pivotu – ideální medián
  - První prvek (jakákoli fixní pozice) – nevýhodné na částečně seřazených množinách
  - Náhodný prvek – ve skutečnosti pseudonáhodný
  - Medián tří (pěti…) – či jiného počtu prvků z fixních nebo náhodných pozic
- Při správné implementaci nepotřebuje paměť navíc
- Nestabilní algoritmus
- Způsob volby pivotu má vliv na řazení
- v průměru jde o nejrychlejší známý univerzální algoritmus pro řazení polí v operační paměti počítače

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Selection sort

- Jednoduchý algoritmus
- Časová složitost $O(N^2)$
- Vhodný pro malé množství dat
- Univerzální, lokální, nestabilní
- Postup:
  1. Rozdělíme si posloupnost na seřazenou a neseřazenou část
  2. Najdeme prvek s nejmenší hodnotou v neseřazené části posloupnosti
  3. Zaměníme ho s prvkem na první pozici neseřazené části
  4. První prvek neseřazené části zahrneme do seřazené části a zároveň neseřazenou část zmenšíme o 1 prvek zleva
  5. Zbytek posloupnosti se uspořádá opakováním kroků 2 až 5 pro zbylou neseřazenou část

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Porovnání řadících algoritmů

Název		Časová složitost			Dodatečná paměť	Stabilní	Přirozená	Metoda
Anglicky	Česky	Minimum	Průměrně	Maximum				
Bubble sort	Bublinkové řazení	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	ano	ano	záměna
Heapsort	Řazení hald ou	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(1)$	ne	ne	halda, záměna
Insertion sort	Řazení vkládáním	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	ano	ano	vkládání
Merge sort	Řazení slučováním	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(\log n)$	ano	ano	slučování
Quicksort	Rychlé řazení	$O(n \log n)$	$O(n \log n)$	$O(n^2)$	$O(\log n)$	ne	ne	záměna
Selection sort	Řazení výběrem	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	zprav. ne	ne	výběr

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Zdroj: www.wikipedia.cz

Bucket sort

- Rozděluje data do několika přihrádek
- Časová náročnost: *O(n*k)*, kde *k=n/m,* vstupní data *n,* počet přihrádek *m*.
- Předpoklady:
  - Vhodný pro rovnoměrně rozložené hodnoty vstupních dat.
  - Algoritmus pro seřazení přihrádek musí být stabilní
- Postup:
  - Vstupní data jsou rozdělena do předem definovaného počtu přihrádek.
  - Na každou přihrádku volán stabilní řadicí algoritmus.
  - Jednotlivé přihrádky postupně kopírovány do výstupního pole.
- Výhody: dobře paralelizovatelný, nemusí mít všechna data v paměti najednou

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Radix sort

- Řadí celá čísla procházením všech číslic
- LSD (Least Significant Digit) – řazení podle nejméně významných číslic (odzadu)
- MSD (Most Significant Digit) – nejvíce významné číslice (odpředu)
- Časová složitost: O((z+n)*$\log_z u$), kde z je základ zvolené číselné soustavy, n počet čísel na vstupu a u je maximální rozmezí čísel na vstupu
- Nehodí se pro neomezeně velké vstupy
- Příklad LSD radix: 170, 45, 75, 90, 802, 2, 24, 66 ⇒ 17<u>0</u>, 9<u>0</u>, 80<u>2</u>, <u>2</u>, 2<u>4</u>, 4<u>5</u>, 7<u>5</u>, 6<u>6</u> ⇒ 8<u>0</u>2, 2, <u>2</u>4, <u>4</u>5, <u>6</u>6, 1<u>7</u>0, <u>7</u>5, <u>9</u>0 ⇒2, 24, 45, 66, 75, 90, <u>1</u>70, <u>8</u>02

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Counting sort

- Vhodný pro velké soubory s malým množstvím diskrétních hodnot
- Stabilní
- Časová náročnost: *O(N+M)*
- Paměťová náročnost: *O(M)*
- Předpoklady:
  - Počet různých hodnot (*M*) významně menší než celkový počet prvků (*N*).
  - Pomocné pole – zápis a čtení v konstantním čase (pole indexované hodnotou nebo hashem hodnoty)

- Algoritmus:
  - zleva (či zprava) projde vstupní pole
  - pro každý prvek zvýší v pom. poli četnost výskytu tohoto prvku
  - ke každé položce přičte počet výskytů všech předchozích položek (získá přesnou pozici hranice)
  - začne zprava procházet neseřazené pole
  - pro každý prvek se podívá do pomocného pole na horní hranici pro umístění
  - na tuto hranici ho umístí a zároveň ji sníží o jedna
  - takto postupuje, dokud neprojde celé pole

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA